IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 January 2023, accepted 29 January 2023, date of publication 3 February 2023, date of current version 10 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3242327

RESEARCH ARTICLE

Mining Interesting Negative Sequential Patterns Based on Influence

FENGLING CUI, XIAOQIANG REN, AND XIANGJUN DONG
Department of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China
Corresponding author: Xiangjun Dong (d-xj@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 62076143, and in part by the Fundamental Research Promotion Plan of Qilu University of Technology (Shandong Academy of Sciences) under Grant 2021JC02009.

ABSTRACT Negative sequential pattern (NSP) mining can capture frequently occurring and non-occurring behavior information and can play an irreplaceable role in many applications. Most traditional NSP mining algorithms adopt a support measure to discover interesting patterns. However, the support measure does not truly reflect the interestingness of patterns in some cases. In particular, it ignores the effect of the support of every element and the order characteristics among these elements. Hence, an influence measure was proposed to truly reflect the interestingness of patterns. However, the current influence measure is used only in positive sequential pattern (PSP) mining and does not involve NSPs. To address these problems, this study proposes an algorithm, InfI-NSP, to mine interesting NSPs based on influence. First, we modify an existing NSP mining algorithm to efficiently mine NSPs. Second, we modify the influence measure and apply it to NSP mining to mine interesting NSPs. To the best of our knowledge, InfI-NSP is the first algorithm to mine interesting NSPs based on influence. Experiments on real-life and synthetic datasets show that InfI-NSP is effective.

INDEX TERMS Interestingness measure, negative sequential patterns (NSPs), negative item, negative element, sequential patterns.

I. INTRODUCTION

Behavior analysis plays an increasingly important role in many fields, such as in education [1], [2], [3], medical analyses [4], [5], [6], [7], [8], recommendation systems [9], and abnormal behavior detection [10], [11]. As an important means of behavior analysis [12], [13], [14], sequential pattern mining aims to find patterns in a set of sequences that satisfy a minimum interestingness threshold [15], and these patterns contain much valuable behavioral information. Sequential patterns that contain only occurring behavior are called positive sequential patterns (PSPs). Since 1995 [16], many effective PSP mining algorithms have been proposed, such as GSP [17], PrefixSpan [18], SPADE [19], and SPAM [20]. However, PSPs do not consider non-occurring behavior. Thus, negative sequential pattern (NSP) mining, which considers both occurring and non-occurring behaviors, was proposed. The analysis of non-occurring behavior can typically obtain more comprehensive information and can play an irreplaceable role in some aspects, such as in analyzing the association between treatment services and disease [21]. In addition, some efficient NSP mining algorithms have been proposed, such as Neg-GSP [22], e-NSP [23], NegI-NSP [21], and sc-NSP [24].

Most of these algorithms use only support as a measure of interestingness. However, the support measure does not truly reflect the interestingness of patterns in some cases. In particular, it has two problems in mining sequential patterns. One problem is that, when calculating the support of a pattern, it ignores the effect of the support of every element in the pattern. For example, the five sequential patterns with the highest support in the Adventures of Tom Sawyer (ATS) text sequence record set are <*and and*>, <*and to*>, <*to and*>, <*of and*>, and <*and of*> [25]. The five sequential patterns all comprise elements *and*, *to*, and *of* with the highest support. Even if *and*, *to*, and *of* are unrelated, the sequential patterns

The associate editor coordinating the review of this manuscript and approving it for publication was Dongxiao Yu.

obtained by their repetition and combination are likely to show high support. Therefore, determining whether the five sequential patterns are truly interesting is impossible based only on their supports. The second problem is that the support measure does not consider the order characteristics of the elements. For example, the support of patterns <*and to*> and <*to and*> is 9.8% and 9.1%, respectively. Their supports exhibit little difference, which indicates that the order of the elements *and* and *to* is irrelevant [25]. This contradicts the importance of order in a sequential pattern. Therefore, the two patterns should not be considered interesting patterns. In general, the support measure cannot truly reflect the interestingness of patterns in some cases. In particular, when handling large databases, the support measure may result in numerous uninteresting frequent patterns.

To solve the aforementioned problems, an influence measure has been proposed. The influence measure can consider the effect of the support of every element and the order characteristics among these elements [25]. Hence, the interestingness of the patterns can be truly reflected. Unfortunately, few studies discuss this measure. To the best of our knowledge, only Wu's study on the ISSPM algorithm [25] discusses it; Wu proposes the concept of an influence measure and related calculation methods. However, the influence measure has two limitations. One is that it is used only in PSP mining and does not involve NSPs. The other is that the influence measure targets element-based sequences (the elements only contain single item) and does not consider item-based sequences (the elements contain multiple items). Obviously, item-based NSP mining is more comprehensive. However, it has a new problem in that the number of uninteresting patterns in the mining results may be larger because of the larger number of frequent patterns mined. Therefore, methods for mining truly interesting NSPs is an urgent problem in NSP mining research.

In this study, we use the influence measure to mine item-based interesting NSPs. However, methods for using the influence measure in NSP mining is a new field with many challenges, which are summarized as follows.

1) *How to obtain an appropriate traditional NSP mining algorithm to use the influence measure?* As discussed, the influence measure proposed by Wu concerns only PSP mining as well as element-based sequences [25]. No related research on NSP mining has been conducted in this area. Therefore, we must analyze the applicability of the influence measure in traditional NSP mining and determine algorithms that can use the influence measure. Furthermore, we study the NSP mining of non-occurring items, which is item-based. For example, the sequence <(*keyboard*, *mouse*, ¬*flash disk*)> is the NSP of non-occurring items. Element (*keyboard*, *mouse*, ¬*flash disk*) indicates that the customer purchased a *keyboard* and *mouse* but not *flash disk* in one purchase. Therefore, the appropriate algorithm must also mine the NSPs of non-occurring items. The NSP mining of non-occurring items considers more situations. This research is more complicated.

2) *How to match the candidate sequence generation method of the influence measure?* The candidate sequence generation method of the influence measure is closely related to the candidate sequence generation method of the traditional NSP mining algorithm. Furthermore, the candidate sequence generation methods of different traditional NSP mining algorithms are mostly different. To match these two candidate sequence generation methods, we must modify the candidate sequence generation method of the influence measure. Thus, the generated influence candidate sequences (the sequences before using the minimum influence threshold (*min-inf*) constraint) are ensured to be consistent with the sequential patterns mined by the traditional NSP mining algorithm. That is, the influence candidate sequences can be found in the sequential pattern set mined based on support. In addition, some traditional NSP mining algorithms may be unable to use the influence measure. This is because a large difference may exist between the candidate sequence generation method of the influence measure and the candidate sequence generation method of the traditional NSP mining algorithm.

3) *Which step of traditional NSP mining can use the influence measure appropriately?* The influence measure can be used in different steps of traditional NSP mining, such as in generating candidate sequences or after mining the frequent patterns. The usefulness of the influence measure and algorithm efficiency may vary greatly depending on the steps of using the influence measure. We must analyze which step of traditional NSP mining is appropriate for using the influence measure. In particular, for different traditional NSP mining algorithms, the appropriate steps in which to use the influence measure may differ.

Based on these problems and challenges, this study proposes an algorithm, InfI-NSP, to mine interesting NSPs based on influence. To the best of our knowledge, InfI-NSP is the first NSP mining algorithm that considers influence measure. The main contributions of this study are summarized as follows.

First, we modified the traditional NSP mining algorithm sc-NSP [24] such that the influence measure can be used to mine the NSPs of non-occurring items. This is because the candidate sequence generation method of the sc-NSP algorithm has a high matching degree with the candidate sequence generation method of the influence measure. Furthermore, the speed of the sc-NSP algorithm is considerably fast. In addition, Wu's algorithm ISSPM [25] concerns element-based sequences; however, the sequences we study are item-based. Because the items in an element are unordered, that is, the order of items in an element is to be ignored, the method of influence measure can be extended to item-based NSP mining.

Second, we modified the influence candidate sequence generation method in the ISSPM algorithm [25]. In splicing the patterns to generate the influence candidate sequences, we modify it accordingly to match the candidate sequence generation method of the influence measure with the

candidate sequence generation method of the traditional NSP mining algorithm.

Third, we used the influence measure after capturing patterns that satisfy the minimum support threshold (*min-sup*) constraint. This ensures the usefulness of the influence measure and improves the efficiency of the algorithm compared with using it in other steps of traditional NSP mining.

Finally, a new algorithm InfI-NSP is proposed to mine interesting NSPs based on influence. This algorithm considers the effect of the support of every element and the order characteristics among these elements in the mining process, thereby truly reflecting the interestingness of the patterns. Experiments on real-life and synthetic datasets show that InfI-NSP is effective.

The remainder of this paper is organized as follows. Section II discusses related work. Section III introduces preliminaries. Section IV details the proposed algorithm InfI-NSP, and we discuss the experiment and results in Section V. Finally, Section VI presents conclusions and future work.

II. RELATED WORK

A. NSP MINING

In terms of research on NSP mining, Zheng et al. [22] proposed a similar NSP mining algorithm, NegGSP, based on the classical PSP mining algorithm GSP. Hsueh et al. [26] proposed an effective NSP mining algorithm, named PNSP. The algorithm converts frequent positive elements into negative elements and then connects the positive and negative elements to generate negative sequential candidates (NSCs) until the size of the NSC equals the maximum size of the data sequence. Zheng et al. [27] proposed a genetic algorithm (GA)-based method to find NSPs with novel crossover and mutation operations. Early algorithms as mentioned above calculate the support of the NSC by rescanning the database, and the time efficiency is low. The NSP mining algorithm e-NSP calculates the support of the NSC only by using the corresponding PSP information rather than rescanning the sequence database, which greatly improves the time efficiency of NSP mining [23]. e-RNSP extends e-NSP to effectively mine NSPs with repetitive properties in the sequence [28]. F-NSP+ uses a novel data structure bitmap to store the PSP information, obtain the support of the NSC only using bitwise operations, and further improves the time efficiency of NSP mining [29]. NegI-NSP [21] and VM-NSP [30] loosen the constraints in e-NSP to mine NSPs and can obtain more valuable information. NegPSpan uses the PrefixSpan depth-first method to extract NSPs with a maximum gap constraint enabled [31]. sc-NSP is an efficient NSP mining algorithm with an improved technique [24]. The algorithm utilizes the improved PrefixSpan algorithm of a bitmap storage structure to mine PSPs, loosens the frequent constraint, exploits the NSC generation method of PNSP, and uses the most efficient bitwise-based operation to calculate the support of the NSCs.

B. INTERESTINGNESS MEASURE

Support is the most basic measure of pattern interestingness. In [32], some frequent itemset mining algorithms were introduced, such as Apriori, Eclat, FP-Growth, H-Mine, and LCM. Gan et al. [33] described the related approaches of parallel sequential pattern mining (PSPM) in detail including partition-based algorithms for PSPM, apriori-based PSPM, pattern-growth-based PSPM, and hybrid algorithms for PSPM. However, only considering the support of patterns is sometimes insufficient for making predictions or suggestions. Based on support, some other measures are defined. In [34], confidence was defined, which can reflect the relationship between itemsets. The lift defined in [35] reflects the degree of independence between the pattern and the contained items. Leverage weakens the effect of items on pattern interestingness [36]. Fournier-Viger et al. [37] defined the standard deviation of periods and the sequence periodic ratio to discover periodic patterns common to multiple sequences. However, none of these interestingness measures can reflect the distribution differences of patterns in different types of record sets. To characterize such distribution differences, interestingness measures such as the growth rate [38] and odds ratio [39] have been proposed. The growth rate can reflect the proportion of absolute support of patterns in different types of record sets, and the odds ratio reflects the proportion of the relative support of patterns in different types of record sets. Utility reflects the importance of an item. Gan et al. [40] proposed a utility mining algorithm HUSP-ULL to discover high-utility sequential patterns. In addition, high-utility itemset mining algorithms FHN [41] and HUPNU [42] were proposed, which consider negative unit profits. In [43], an efficient approach EHMIN was proposed for mining high-utility patterns with negative unit profits. Generally, the number of patterns that satisfy the minimum interestingness constraint is large. To reduce the redundancy of the results, some algorithms design a global interestingness measure from the perspective of the pattern set. For example, Guns et al. [44] designed coverage, which reflects the dependence of support between patterns. Petitjean et al. [45] defined the maximum subleverage ratio, which considers the effect of the subpattern interestingness. To measure the interestingness of sequential patterns more truly, Wu et al. [25] defined influence. Influence considers the effect of the support of every element and the order characteristics among these elements; however, it is only for element-based PSP mining. In [46], a metric impact was proposed. Its general idea is to measure the effect of the removed item on the outcome by removing the last item from the sequential pattern; however, it is used in impact-oriented sequential rules.

III. PRELIMINARIES

For a sequence, the items that have occurred are called positive items, and, correspondingly, the items that have not occurred are called negative items. If an element contains

TABLE 1. Notation description.

Symbol	Description
I	A set of items, $I=\{x_1, x_2, ..., x_n\}$, consisting of n items $x_k(1\leq k\leq n)$
s	A sequence, $s =< s_1, ..., s_l >$, consisting of l elements $s_j(1\leq j\leq l)$
$min-sup$	The minimum support threshold
ns	A negative sequence
$length(s)$	The length of sequence s
$size(s)$	The size of sequence s
$sup(s)$	The support of s
$p(ns)$	ns's positive partner
$MPS(s)$	A maximum positive sub-sequence of s
$1-negMPSE$	A 1-neg-size maximum sub-element
$1-negMS_{ns}$	A 1-neg-length maximum sub-sequence of ns
$1-negMSS_{ns}$	A 1-neg-length maximum sub-sequence set of ns
$negsize(ns)$	The total number of negative elements in ns
$exsup(s)$	The expected value of the support of s
$adsup(s)$	The revised support of s
$mesup(s)$	The adjustment value of the revised support of s
$inf(s)$	The influence of s
$min-inf$	The minimum influence threshold

a negative item, it is called a negative element. Further, if a sequence contains a negative element, it is called a negative sequence. The sequences in source data are data sequences [27]. Next, we introduce some important PSP and NSP definitions. Table 1 lists some of the main notation used in this paper.

A. POSITIVE SEQUENTIAL PATTERNS

Let $I = \{x_1, x_2, \ldots, x_n\}$ be a set of items. An itemset is a subset of I. A sequence is an ordered list of itemsets. A sequence s is denoted by $s =< s_1 s_2, \ldots, s_l >$, where $s_j \subseteq I(1 \leq j \leq l)$. s_j is also called an element of the sequence and is denoted by $(x_1, x_2, \ldots, x_m)$, where x_k is an item, and $x_k \in I(1 \leq k \leq m)$. For simplicity, brackets are omitted if an element has only one item, that is, element (x) is coded x. To reduce complexity, we assume that an item can appear only once in an element but can appear many times in different elements of a sequence.

The length of sequence s, denoted by $length(s)$, is the total number of items in all elements in s. s is a k-length sequence if $length(s) = k$. The size of sequence s, denoted by $size(s)$, is the total number of elements in s. s is a k-size sequence if $size(s) = k$. For example, suppose there is a sequence $s =< a(bc)d >$, which has three elements a, (bc), and d and four items a, b, c, and d; thus, s is a *3*-size and *4*-length sequence.

Consider two sequences $s_\alpha =< \alpha_1\alpha_2 \ldots \alpha_i >$ and $s_\beta =< \beta_1\beta_2 \ldots \beta_k >$; if there exists $1 \leq j_1 < j_2 < \ldots < j_i \leq k$ such that $\alpha_1 \subseteq \beta_{j_1}, \alpha_2 \subseteq \beta_{j_2}, \ldots, \alpha_i \subseteq \beta_{j_i}$, we call sequence $s_\alpha =< \alpha_1\alpha_2 \ldots \alpha_i >$ a *subsequence* of sequence $s_\beta =< \beta_1\beta_2 \ldots \beta_k >$, which is expressed as $s_\alpha \subseteq s_\beta$, and s_β is called a *super-sequence* of s_α. For example, $s_1 =< a(bc)d >$ is a super-sequence of $s_2 =< (bc) >$ (and $s_2 =< (bc) >$ is a subsequence of $s_1 =< a(bc)d >$).

The number of tuples in sequence database D is expressed as $|D|$, where the tuples are $< sid\ (sequence-ID),\ ds\ (data\ sequence) >$. The set of tuples containing sequence s is denoted as $\{< s >\}$. The support of s, denoted by $sup(s)$, is the number of tuples contained in $\{< s >\}$. That is, $sup(s) = |\{< s >\}| = |\{< sid, ds >, < sid, ds > \in D \wedge (s \subseteq ds)\}|$. $min-sup$ is the minimum support threshold predefined by users. Sequence s is called a frequent sequential pattern if $sup(s) \geq min-sup$. Conversely, s is infrequent if $sup(s) < min-sup$.

B. NEGATIVE SEQUENTIAL PATTERNS

In real-life applications, the number of generated NSCs is sometimes large, but many may be meaningless [23]. Therefore, constraints are added to reduce the number of NSCs and discover the meaningful NSP efficiently. The key concepts and definitions of the negative constraints involved in this study are as follows:

Definition 1 (Negative Size): The total number of negative elements in sequence ns is called the negative size of sequence ns. ns is an n-neg-size sequence if $negsize(ns) = n$. For example, given $ns =< (ab)\neg cd\neg e >$, the number of negative elements is 2, and thus, ns is a 2-neg-size sequence.

Definition 2 (Positive Partner): The positive partner of a negative element $(\neg ab)$ is (ab), denoted by $p((\neg ab))$, that is, $p((\neg ab)) = (ab)$; the positive partner of positive element (ab) is (ab) itself, that is, $p((ab)) = (ab)$. The positive partner of a negative sequence $ns =< s_1 s_2, \ldots, s_k >$ changes all the negative elements in ns to their positive partners, expressed as $p(ns)$, that is, $p(ns) = \{< s_1 s_2, \ldots, s_k > | s_j = p(s_i), s_i \in ns\}$. For example, $p(< \neg a(b\neg c)d >) =< a(bc)d >$.

Definition 3 (Maximum Positive Subsequence): Assume that $ns =< s_1 s_2, \ldots, s_m >$ is an m-size and n-neg-size negative sequence $(m > n)$, and the subsequence s contains all positive elements. Then, it is called the maximum positive subsequence of ns and is expressed as $MPS(ns)$. For example, given $ns =< (a\neg b)c\neg d >$, we can obtain $MPS(ns) =< ac >$.

Definition 4 (1-Neg-Size Maximum Subsequence): For a negative sequence ns, its subsequences that include $MPS(ns)$ and one negative element e are called 1-neg-size maximum subsequences, denoted by $1 - negMS_i$. The subsequence set including all the 1-neg-size maximum subsequences of ns is called the 1-neg-size maximum subsequence set, denoted by $1-negMSS_{ns}$. For example, given $ns =< (a\neg b)c\neg d(ef) >$, $1\text{–}negMS_1 =< (a\neg b)c(ef) >$, $1\text{–}negMSS_{ns} = \{< (a\neg b)c(ef) >, < ac\neg d(ef) >\}$.

Constraint 1 (Element Frequency Constraint): A negative element e_n cannot appear in the NSCs unless its positive element partner $p(e_n)$ is frequent, that is, $sup(p(e_n)) \geq min-sup$. This is similar to the settings used in [30]. For example, if $sup(p((\neg ab))) = sup((ab)) \geq min-sup$, then element $(\neg ab)$ satisfies this constraint.

Note that similar to the NegI-NSP algorithm [21], this study does not consider cases such as $(\neg a\neg b)$ and $\neg(ab)$. This is because identifying the difference between negative elements $(\neg a\neg b)$ and $\neg(ab)$ is difficult in real life. Although sequence $< (\neg a\neg b) >$ is frequent, applying it to reality is also difficult.

Constraint 2 (1-length-neg Element Format Constraint): An NSC only does not allow continuous *1*-length negative elements. For example, $<\neg a(\neg bc)>$ satisfies this constraint, unlike $<\neg a\neg b>$.

A negative containment definition can convert a negative containment into a positive containment, and the support of the NSP can rely on the information of the corresponding PSP.

Definition 5 (Negative Containment): Let $ds =< d_1 d_2, \ldots, d_t >$ be a data sequence, $ns =< s_1 s_2, \ldots, s_m >$ be an m-size and n-neg-size negative sequence; (1) if $m > 2t+1$, then ds does not contain ns; (2) if $m \geq 1$ and $n = 1$, then ds contains ns when $p(1-negMS) \nsubseteq ds$; and (3) otherwise, ds contains ns if $MPS(ns) \nsubseteq ds \wedge \forall 1-negMS_i \in 1-negMSS_{ns}, p(1-negMS_i) \nsubseteq ds (1 < i \leq n)$.

For example, given $ds =< ab(cd)(ade)fc >$, 1) if $ns =< a(d\neg e)f\neg d >$, $1-negMSS_{ns} = \{< a(d\neg e)f >, < adf\neg d >\}$, then ds does not contain ns because $p(< a(d\neg e)f >) =< a(de)f > \subseteq ds$; 2) if $ns' =< a\neg c(\neg ab)f >$, $1-negMSS'_{ns} = \{< a\neg cbf >, < a(\neg ab)f >\}$, then ds contains ns because $MPS(ns) =< abf > \subseteq ds \wedge p(< a\neg cbf >) \nsubseteq ds \wedge p(< a(\neg ab)f >) \nsubseteq ds$.

IV. InfI-NSP ALGORITHM

The overall steps of the InfI-NSP algorithm are as follows.

First, all positive and negative patterns that satisfy the min-sup constraint were mined from the sequence database using the modified sc-NSP algorithm. Subsequently, the influence was calculated separately for the positive and negative patterns mined. Finally, all positive and negative patterns that satisfy the min-inf constraint were mined.

In this section, we first introduce related concepts and calculation methods for the influence. Second, we present an NSC generation method for the modified sc-NSP algorithm. Third, we introduce a support calculation method for the modified sc-NSP algorithm. Next, we present the steps and methods used to calculate the influence of the pattern. Finally, we present the mining steps of the InfI-NSP algorithm and corresponding pseudocode.

A. INFLUENCE MEASURE

The two problems of using only support in sequential pattern mining discussed in Section I can be effectively solved using influence. For convenience, we list the following two problems: 1) when calculating the support of a pattern, it ignores the effect of the support of every element in the pattern and 2) support does not consider the order characteristics of the elements.

The steps for calculating the influence are as follows.

Step 1. Given a sequential pattern $s =< e_1 e_2, \ldots, e_l >$, if the elements in s are independent of each other, the expected value of the support of s is

$$exsup(s) = \prod_{i=1}^{l} sup(e_i). \tag{1}$$

The meaning expressed by the expected value is that even if the elements of s do not have sequential connections, s has a high probability of appearing $exsup(s) \times |D|$ times in D. $exsup(s)$ quantifies the effect of the support for every element in the pattern to a certain extent.

Step 2. Next, the revised support of s is obtained by subtracting this effect, that is,

$$adsup(s) = sup(s) - exsup(s). \tag{2}$$

The first problem mentioned above can be solved effectively by revising the support for sequential patterns.

Step 3. Assuming $h(s)$ represents the first element of s, and $r(s)$ represents the elements of s except for the first element, s can be represented as $< h(s), r(s) >$. Given two patterns, s^a and s^b, the possible sequential pattern set $ger(s^a, s^b)$ can be calculated recursively as follows:

$$ger(s^a, s^b) = G^a \cup G^b, \tag{3}$$

$$G^a = \{< h(s^a), X > | X \in ger(r(s^a), s^b)\}, \tag{4}$$

$$G^b = \{< h(s^b), Y > | Y \in ger(s^a, r(s^b))\}, \tag{5}$$

where $ger(s^a, \varnothing) = \{s^a\}$, and $ger(\varnothing, s^b) = \{s^b\}$. For example, assuming that $s^a =< a(\neg bc) >$ and $s^b =< \neg de >$, the set of possible sequential patterns formed by s^a and s^b are $ger(s^a, s^b) = \{< a\neg b(\neg bc)e >, < a\neg be(\neg bc) >, < a(\neg bc)\neg be >, < \neg ba(\neg bc)e >, < \neg bae(\neg bc) >, < \neg bea(\neg bc) >\}$. If the supports of the sequential patterns in $ger(s^a, s^b)$ are not significantly different, then the order of elements in s^a and s^b is irrelevant. Hence, none of the sequential patterns in $ger(s^a, s^b)$ should be considered interesting sequential patterns.

Step 4. Next, the revised support of the sequential pattern is further adjusted to incorporate the order characteristics of the elements as follows:

$$mesup(s) = avg(\sum_{s^* \in G^{ger}} adsup(s^*)), \tag{6}$$

$$inf(s) = adsup(s) - mesup(s), \tag{7}$$

where G^{ger} denotes the result set of $ger(s^a, s^b)$, and $avg()$ denotes the average value of the support of the patterns in the set G^{ger}. This $inf()$ is called the *influence* of s, which considers the effect of the support of every element and the order characteristics among these elements.

B. NSC GENERATION OF MODIFIED SC-NSP

Because this study targets item-based NSP mining, we modified the NSC generation method of sc-NSP [24] by combining the NSC generation method of NegI-NSP [21]. Thus, this influence can be used to mine interesting item-based NSPs. The details are as follows.

First, the *1*-size NSC is generated by the *1*-size PSP. The basic idea behind generating an NSC is to change any item(s) in a *1*-size PSP into its (their) negative form(s). For example, the NSC based on $< (abc) >$ includes $< (\neg abc) >$, $< (a\neg bc) >$, $< (ab\neg c) >$, $< (\neg a\neg bc) >$, $< (\neg ab\neg c) >$,

and $< (a\neg b\neg c) >$. Evidently, this method can generate all possible *1*-size NSCs that satisfy the constraints.

Second, the generation of an *n*-size ($n \geq 2$) NSC is divided into two cases: 1) appending an (*n-1*)-size PSP with a *1*-size NSC. Note that the *1*-size NSC includes two cases in which the element in the sequence is the negative element of non-occurring items (such as $< (a\neg b) >$) and negative single-item element (such as $< \neg a >$); and 2) appending an (*n-1*)-size NSC with a *1*-size PSP and *1*-size NSC. Note that we use an (*n-1*)-size NSC instead of (*n-1*)-size NSP to generate the *n*-size NSC because the NSP does not satisfy the Apriori property [47].

Finally, this process is repeated until no NSC is generated or the size of NSC is greater than $2m+1$, where m is the maximum size of the sequence in the PSP. If the maximum size of the sequence in the PSP is m, then the maximum size of the generated NSP is $2m+1$.

Some *n*-size NSCs can be pruned before their support is calculated. According to Definition 4 (the definition of a 1-neg-size maximum subsequence) and Constraint 1 (the element frequency constraint), we use the following two pruning strategies: 1) if $\forall ns \in NSC$ and $MPS(ns) \notin PSP$, then sequence ns is pruned; and 2) if $\forall ns \in NSC$ and $\forall p(1-negMS) \notin PSP$, then sequence ns is pruned.

Algorithm 1 presents the pseudocode for the NSC generation of modified sc-NSP.

Algorithm 1 NSC Generation

Input: Sequence database D, PSP
Output: NSC

```
1: 1-size NSC is generated from 1-size PSP;
2: for n=2; n≤2m+1; n++do
3:    for each candidate sequence in (n-1)-size NSC do
4:       ns = candidate sequence append with 1-size PSP;
5:       if MPS(ns)∈PSP∧ ∀p(1−negMS)∈PSP then
6:          ns is stored in n-size NSC;
7:       end if
8:       ns = candidate sequence append with 1-size NSC;
9:       if MPS(ns)∈PSP∧ ∀p(1−negMS)∈PSP then
10:         ns is stored in n-size NSC;
11:      end if
12:   end for
13:   for each candidate sequence in (n-1)-size PSP do
14:      ns = candidate sequence append with 1-size NSC;
15:      if MPS(ns)∈PSP∧ ∀p(1−negMS)∈PSP then
16:         ns is stored in n-size NSC;
17:      end if
18:   end for
19: end for
20: return NSCs;
```

Generating NSC includes the following key steps. 1) Generate *1*-size NSC using *1*-size PSP (*line 1*). 2) (*n-1*)-size NSC generates the *n*-size sequence *ns* by appending with *1*-size PSP (*line 4*). 3) According to the pruning strategy, *ns* must satisfy $MPS(ns) \in PSP$ and $\forall p(1-negMS) \in PSP$ (*lines 5-7*). 4) (*n-1*)-size NSC generates the *n*-size sequence *ns* by appending with *1*-size NSC (*line 8*). 5) (*n-1*)-size PSP generates the *n*-size sequence *ns* by appending with *1*-size NSC (*line 14*).

C. CALCULATING THE SUPPORT OF THE NSC

The support calculation method used in the InfI-NSP algorithm, that is, the support calculation method of the modified sc-NSP algorithm, is the same as that of the NegI-NSP algorithm [21]. The support calculation equation in the NegI-NSP algorithm conforms to the set theory principle. The details are as follows.

Given an *m*-size and *n*-neg-size negative sequence *ns*, among the *n* negative elements, for $\forall 1-negMS_i \in 1-negMSS_{ns}(1 \leq i \leq n)$, the support of *ns* in sequence database *D* is

$$sup(ns) = sup(MPS(ns)) - |\cup_{i=1}^{n} \{p(1-negMS_i)\}|. \quad (8)$$

In particular, for negative sequences $< \neg e >$,

$$sup(< \neg e >) = |D| - sup(< e >). \quad (9)$$

D. CALCULATING THE INFLUENCE OF THE PATTERN

The calculation method and process of the influence in this study are based on the idea of the influence in the ISSPM algorithm [25]. Furthermore, some modifications were made to use the influence for NSP mining.

Because using influence in traditional NSP mining has the matching problem of candidate sequence generation methods, we modified the candidate sequence generation method of the influence in the ISSPM algorithm [25]. In splicing the patterns to generate the influence candidate sequences, we modified it to match the candidate sequence generation method of the influence with the candidate sequence generation method of the modified sc-NSP algorithm. Thus, the generated influence candidate sequences were ensured to be consistent with the sequential patterns mined by the modified sc-NSP algorithm. That is, the influence candidate sequences can be found in the sequential pattern set mined based on support. The details are as follows.

We used the appendant method to generate NSCs in the traditional NSP mining algorithm-modified sc-NSP. Section IV-B presents the details. Therefore, we combined this aspect to modify the influence candidate sequence generation method of the ISSPM algorithm [25]. In the ISSPM algorithm, for patterns mined using support, the patterns that satisfy specific conditions are spliced in pairs according to their length. Therefore, for s^a and s^b in Step 3 of Section IV-A, in the ISSPM algorithm, they are both from the same length of patterns mined based on support. Unlike in the ISSPM algorithm, we first mine all sequential patterns that satisfy the min-sup constraint and then calculate the influence of these patterns individually. To match the candidate sequence generation method of the influence with

the candidate sequence generation method of the modified sc-NSP algorithm, we modified the method of obtaining s^a and s^b in the calculation process of influence; that is, s^a and s^b are obtained from the same patterns. In particular, we assume that an n-size sequence $s =< x_1x_2, \ldots, x_n >$ and take the first $(n\text{-}1)$ elements of s as s^a and last element as s^b; that is, $s^a =< x_1x_2, \ldots, x_{n-1} >, s^b =< x_n >$. Then, the set of possible sequential patterns is obtained in Step 3 of Section IV-A, and some subsequent calculations are performed.

The pseudocode of the influence calculation is shown in Algorithm 2.

Algorithm 2 Influence Calculation

Input: Sequential pattern set Q (PSP or NSP), *min-inf*
Output: Sequential pattern set G satisfying the *min-inf* constraint (Inf-PSP or Inf-NSP)

1: **for** each sequence s in Q **do**
2: **if** $size(s) \geq 2$ **then**
3: Calculate $exsup(s)$ using equation (1);
4: Calculate $adsup(s)$ using equation (2);
5: s^b = the last element of s;
6: s^a = the elements of s except the last element;
7: Obtain the sequential pattern set P that s^a and s^b can form using equations (3), (4), and (5);
8: **for** each sequence s' in P **do**
9: Calculate $exsup(s')$ using equation (1);
10: Calculate $adsup(s')$ using equation (2);
11: $adsup(s')$ is stored in set M;
12: **end for**
13: Calculate $mesup(s)$ using equation (6);
14: Calculate $inf(s)$ using equation (7);
15: **if** $inf(s)$ satisfies the *min-inf* constraint **then**
16: s is stored in G;
17: **end if**
18: **end if**
19: **end for**
20: **return** G;

For the sequential pattern s mined based on support, when $size(s) \geq 2$, we calculate the influence of s. First, we calculate the expected value of the support of s $exsup(s)$ (*line 3*). Second, we calculate the revised support of s $adsup(s)$ (*line 4*). Subsequently, sequences s^a and s^b are obtained from s (*lines 5-6*), and the set P of the possible patterns formed by s^a and s^b is obtained according to the method in Step 3 of Section IV-A (*line 7*). Third, we calculate the revised support $adsup(s')$ for each sequence s' in set P and store it in set M (*lines 8-12*). Finally, the influence of s $inf(s)$ is calculated. If $inf(s)$ satisfies the min-inf constraint, it is stored in set G (*lines 13-17*).

E. MINING STEPS OF THE InfI-NSP ALGORITHM

Through analysis, the influence can be used when generating candidate sequences or after mining frequent patterns. However, numerous redundant patterns are in the generated candidate sequences. Hence, using influence in this step has the problem of invalid calculations, which wastes a significant amount of time and space. In the InfI-NSP algorithm, we use the influence after capturing patterns that satisfy the min-sup constraint. This ensures the usefulness of the influence and makes the algorithm more efficient than using it in other steps of traditional NSP mining.

Next, the InfI-NSP algorithm is described in detail, and Algorithm 3 presents the pseudocode.

Algorithm 3 InfI-NSP Algorithm

Input: Sequence database D, *min-sup*, *min-inf*
Output: Inf-PSP, Inf-NSP

1: PSPs are obtained using the modified sc-NSP algorithm;
2: Inf-PSPs are obtained using the method described in Algorithm 2;
3: NSCs are generated using the method described in Algorithm 1;
4: **for** each nsc in NSC **do**
5: **if** $size(nsc) = 1 \wedge length(nsc) = 1$ **then**
6: Calculate $sup(nsc)$ using equation (9);
7: **else**
8: Calculate $sup(nsc)$ using equation (8);
9: **end if**
10: **if** $sup(nsc)$ satisfies the *min-sup* constraint **then**
11: *nsc* is stored in NSP;
12: **end if**
13: **end for**
14: Inf-NSPs are obtained using the method described in Algorithm 2;
15: **return** *Inf-PSPs* and *Inf-NSPs*;

First, the InfI-NSP algorithm mines all PSPs from sequence database D using the PSP mining method of the modified sc-NSP algorithm (*line 1*). Second, all Inf-PSPs are obtained by using the influence calculation method described in Section IV-D (*line 2*). Third, we generate all NSCs using the NSC generation method described in Section IV-B (*line 3*). The support for each nsc in the NSC is calculated by using equations (8) and (9) (*lines 4-9*). An nsc is an NSP if its support satisfies the min-sup constraint (*lines 10-12*). Finally, all Inf-NSPs are obtained by using the influence calculation method described in Section IV-D (*line 14*).

V. EXPERIMENTS AND EVALUATION

We conducted experiments on two synthetic and two real-life datasets to compare the performance of InfI-NSP with that of modified sc-NSP. We compared the mined positive and negative patterns separately, including the number of patterns, the runtime of mining patterns, and the effect of different min-inf values on the number of patterns. All algorithms were coded in Java, implemented in Eclipse, and ran on a Windows 10 PC with an Intel Core i5 CPU 2.5 GHz and 4 GB of memory. In the experiments, unless specified, all supports (and minimum supports) were calculated in terms of the

percentage of the frequency $|<s>|$ of a pattern s compared with the number of sequences $|D|$ in the database.

A. DATASETS

Two real-life and two synthetic datasets are introduced in detail for the experiments. The two real-life datasets were collected from the SPMF website (www.philippe-fournier-viger.com), and the two synthetic datasets were generated using the IBM data generator [16].

Dataset 1 (*DS1*) is a conversion of the Bible as a sequence database. It contains 36,369 sequences and 13,905 distinct items. The average length of a sequence is 21.6 items. The average number of distinct items per sequence is 17.84.

Dataset 2 (*DS2*) is a conversion of the novel *Leviathan* by Thomas Hobbes as a sequence database. It contains 5,834 sequences and 9,025 distinct items. The average length of a sequence is 33.8 items. The average number of distinct items per sequence is 26.34.

Dataset 3 (*DS3*), C8_T6_S8_I8_DB10k_N100.

Dataset 4 (*DS4*), C10_T4_S6_I6_DB10k_N100.

For the synthetic datasets, the data factors describe the data characteristics from different aspects, and their general meanings are as follows [16]: C: average number of elements per sequence; T: average number of items per element; S: average size of maximal potentially large sequences; I: average size of items per element in maximal potentially large sequences; DB: number of sequences in a database; and N: number of items.

B. NUMBER COMPARISON OF POSITIVE AND NEGATIVE PATTERNS

In this section, we set min-inf to a fixed value. Subsequently, we analyze and compared the number of positive and negative patterns mined by InfI-NSP and modified sc-NSP under different min-sup values on *DS1*-*DS4*. To obtain sufficient patterns to observe the differences between the two algorithms more clearly, different datasets had different min-inf and min-sup ranges because of their inherent properties.

As shown in Fig 1 and 2, the number of patterns mined by InfI-NSP on DS1-DS4 is significantly reduced compared with that of the modified sc-NSP, regardless of the number of positive or negative patterns. This is because InfI-NSP considers the effect of the support of every element and the order characteristics among these elements, thereby removing the uninteresting frequent patterns.

For example, for a real-life dataset (*DS2*) and synthetic dataset (*DS3*), we list the partial patterns with the highest support in the removed 2 or 3-size positive and negative patterns (for ease of observation, we express the support as the number of sequences here), as shown in Tables 2 and 3. As shown in the two tables, the patterns are composed of repeated or combined elements with high support. Hence, the support of these patterns is primarily derived from the elements they contain and does not truly reflect their interestingness. In addition, these sequences, which are the exchange order of elements in the sequence, have little

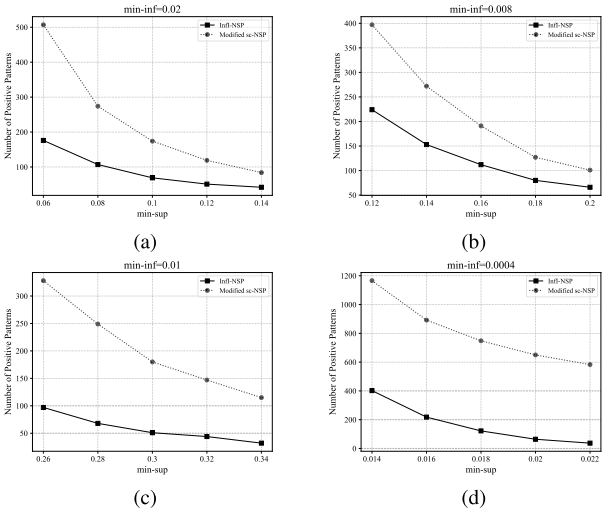

FIGURE 1. Number comparison of PSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

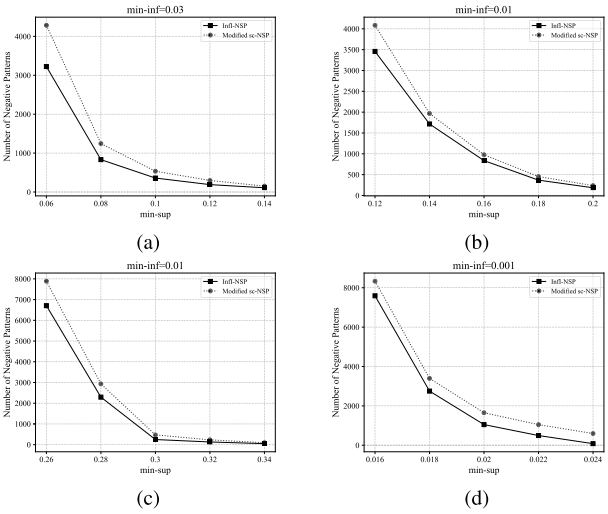

FIGURE 2. Number comparison of NSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

difference in terms of support; that is, the order characteristics of the elements in the sequence are not considered, such as $<\neg$*in the and*$>$ and $<\neg$*in and the*$>$ in Table 2. Therefore, these sequential patterns provide valueless information and should not be considered interesting. InfI-NSP removes these uninteresting frequent patterns using the influence, which reduces the number of mined patterns compared with the modified sc-NSP.

As shown in Fig 1 and 2, on DS1-DS4, as the value of min-sup gradually increases, the number of PSPs and NSPs mined by both algorithms gradually decreases. This is because the number of patterns that satisfy the min-sup constraint is smaller as the value of the min-sup becomes larger. Therefore, the number of patterns satisfying the min-inf constraint is also smaller. Simultaneously, as shown in Fig 1 and 2, we note that, as the value of min-sup becomes smaller, the difference

TABLE 2. Removed partial positive and negative patterns on DS2 (min-sup=0.12, min-inf=0.01).

Removed partial positive patterns		*Removed partial negative patterns*	
positive pattern	support count	negative pattern	support count
$<to\ the>$	2248	$<\neg in\ the\ and>$	1475
$<the\ to>$	2231	$<\neg of\ of\ the>$	1453
$<of\ and>$	2056	$<\neg is\ the\ and>$	1430
$<and\ of>$	1975	$<\neg and\ the\ to>$	1421
$<of\ to>$	1971	$<\neg in\ and\ the>$	1401
$<to\ of>$	1933	$<\neg of\ the\ of>$	1391
$<to\ and>$	1524	$<\neg that\ the\ and>$	1386
$<and\ to>$	1487	$<\neg that\ the\ to>$	1383
$<a\ the>$	1206	$<\neg and\ to\ the>$	1348
$<by\ the>$	1139	$<\neg is\ and\ the>$	1342
$<the\ a>$	1138	$<\neg that\ and\ the>$	1303
$<the\ by>$	1103	$<\neg that\ to\ the>$	1291

TABLE 3. Removed partial positive and negative patterns on DS3 (min-sup=0.26, min-inf=0.01).

Removed partial positive patterns		*Removed partial negative patterns*	
positive pattern	support count	negative pattern	support count
$<91\ 70>$	3623	$<(\neg 23, 91)\ \neg 23>$	3066
$<70\ 91>$	3522	$<(\neg 29, 91)\ \neg 29>$	3032
$<91\ 45>$	3252	$<\neg 23\ (\neg 23, 91)>$	3007
$<45\ 91>$	3211	$<\neg 29\ (\neg 29, 91)>$	2986
$<91\ 61>$	3006	$<(\neg 72, 91)\ \neg 72>$	2979
$<61\ 91>$	2929	$<(\neg 90, 91)\ \neg 90>$	2971
$<45\ 70>$	2909	$<\neg 72\ (\neg 72, 91)>$	2964
$<70\ 27>$	2902	$<\neg 90\ (\neg 90, 91)>$	2955
$<91\ 17>$	2880	$<\neg 52\ (\neg 52, 91)>$	2873
$<17\ 91>$	2878	$<(\neg 33, 91)\ \neg 33>$	2871
$<70\ 45>$	2877	$<(\neg 52, 91)\ \neg 52>$	2856
$<27\ 70>$	2868	$<\neg 33\ (\neg 33, 91)>$	2840

in the number of mining patterns between the two algorithms becomes larger. In addition, as shown in Fig 2, this feature is more obvious when mining negative patterns. This is because as the value of min-sup becomes smaller, the more patterns that are mined. Therefore, as more patterns provide valueless information, InfI-NSP removes more patterns during mining.

C. RUNTIME COMPARISON OF MINING POSITIVE AND NEGATIVE PATTERNS

In this section, we set min-inf to a fixed value. Then, on *DS1-DS4*, we analyze and compare the runtimes of InfI-NSP and modified sc-NSP for mining positive and negative patterns under different min-sup values.

As shown in Fig 3 (a), (b), and (c) and 4 (b), under different min-sup values, the runtime of InfI-NSP and modified sc-NSP are relatively close. Therefore, we can conclude that although InfI-NSP removes uninteresting frequent patterns, its runtime is insignificantly different from that of modified sc-NSP. This shows that InfI-NSP is highly efficient for the corresponding datasets. As shown in Fig 3 (d) and 4 (a), (c), and (d), when the value of min-sup is small, the runtime of InfI-NSP for mining patterns is relatively different from that

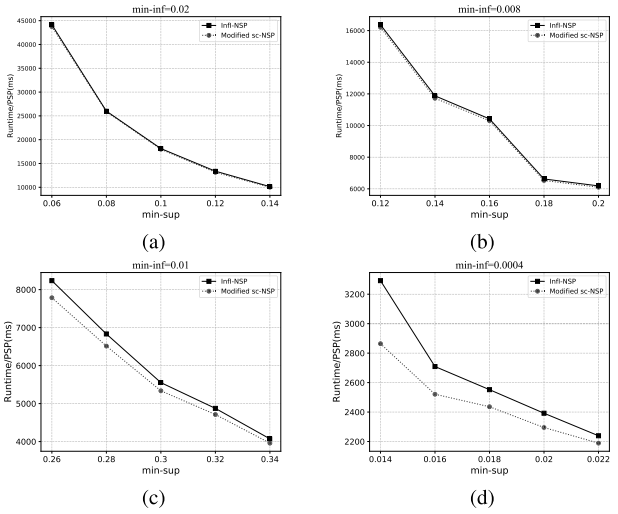

FIGURE 3. Runtime comparison of mining PSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

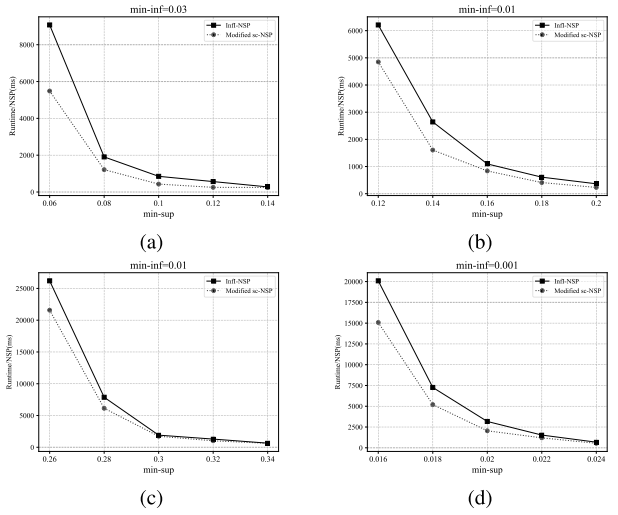

FIGURE 4. Runtime comparison of mining NSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

of the modified sc-NSP. When the value of min-sup is greater than a certain value, the runtime of the two algorithms for mining patterns is relatively close. This is because when the support decreases to a certain value, the number of patterns mined from the corresponding dataset increases significantly. Accordingly, the time required to calculate this influence increases significantly. Simultaneously, this is related to the characteristics of the dataset itself. As shown in Fig 3 and 4, the runtimes of both algorithms for mining PSPs and NSPs decrease gradually as the value of min-sup increases gradually on DS1-DS4.

D. EXPERIMENT TO ASSESS THE EFFECT OF MIN-INF

In this section, we set min-sup to a fixed value. Then, on *DS1-DS4*, we analyze and compare the effects of different min-inf values on the number of PSPs and NSPs. In particular, for mining PSPs and NSPs, the runtimes of InfI-NSP are

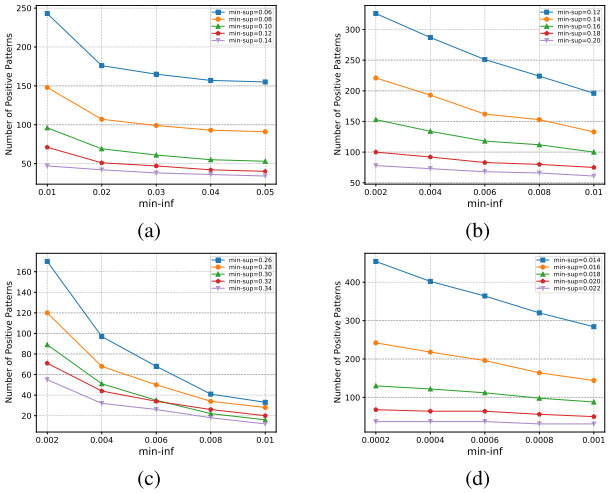

FIGURE 5. Effect of min-inf on the number of PSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

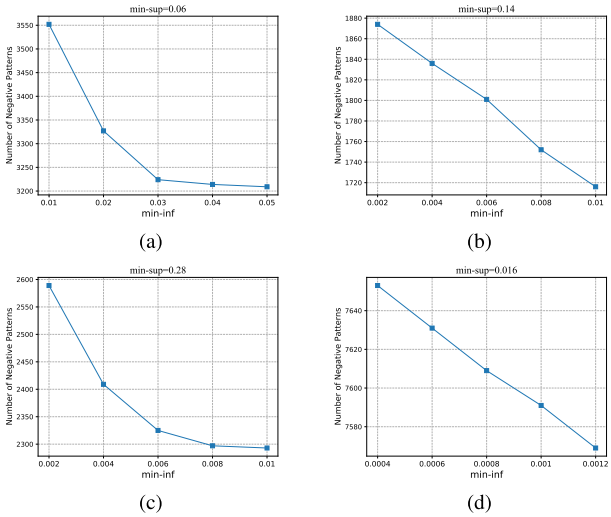

FIGURE 6. Effect of min-inf on the number of NSPs. (a) The experiment on DS1. (b) The experiment on DS2. (c) The experiment on DS3. (d) The experiment on DS4.

relatively close between different min-inf values under the same min-sup value, and the runtime does not change significantly with min-inf. Therefore, we no longer analyze this aspect.

For positive patterns, under different min-sup values, we analyze and compare the effect of min-inf on the number of PSPs on DS1-DS4. For negative patterns, the number of NSPs mined under different min-sup values varies greatly. To facilitate observation and analysis, we conducted only experiments to analyze and compare the effect of min-inf on the number of NSPs under a fixed min-sup on DS1-DS4.

As shown in Fig 5, on DS1-DS4, the number of positive patterns decreases as min-inf is increased under different min-sup values. This is because, when min-sup is a fixed value, with larger values of min-inf, more positive patterns do not satisfy the min-inf constraint. In addition, as shown in Fig 5(a) and (c), on DS1, when min-inf is 0.01-0.02, the reduction in the number of positive patterns is larger under different min-sup values. On DS3, when min-inf is 0.002-0.004, the decrease in the number of positive patterns is larger under different min-sup values. After that, the magnitude of the reduction in the number of positive patterns is almost the same. This is because many positive patterns that do not satisfy the min-inf constraint appear when min-inf is within the above ranges. As shown in Fig 5(b) and (d), we can see that, on DS2 and DS4, the decrease in the number of positive patterns is almost the same with the gradual increase of min-inf for different min-sup values. This is because the number of positive patterns that do not satisfy the min-inf constraint is relatively uniform when min-inf is within the corresponding range.

As shown in Fig 6, on DS1-DS4, when min-sup is a fixed value, the number of negative patterns decreases with the increase of min-inf values. This is because under the same min-sup, the larger the value of min-inf, the more negative patterns that do not satisfy the min-inf constraint. In addition, from Fig 6 (a), we note that, on DS1, when min-inf is 0.01-0.03, the number of negative patterns decreases greatly. However, this changes slightly after 0.03. This is because the appearances of negative patterns that do not satisfy the min-inf constraint are mainly concentrated when min-inf is 0.01-0.03. As shown in Fig 6 (c), on DS3, when min-inf is 0.002-0.006, the number of negative patterns decreases greatly. However, this changes slightly after 0.006. This is because the appearances of negative patterns that do not satisfy the min-inf constraint are mainly concentrated when min-inf is 0.002-0.006. From Fig 6 (b) and (d), we note that the decrease in the number of negative patterns is almost the same on DS2 and DS4 with a gradually increasing min-inf. This is because the number of negative patterns that do not satisfy the min-inf constraint is relatively uniform when min-inf is within the corresponding range.

E. SCALABILITY TEST ON InfI-NSP

A scalability test was conducted to evaluate the performance of the InfI-NSP on large datasets. We conducted experiments on the real dataset DS2 in terms of different data sizes: from 5 to 20 times of DS2. The performance of the InfI-NSP was evaluated by analyzing its runtime. Fig 7 (a) shows the runtime of the InfI-NSP on the datasets of different sizes

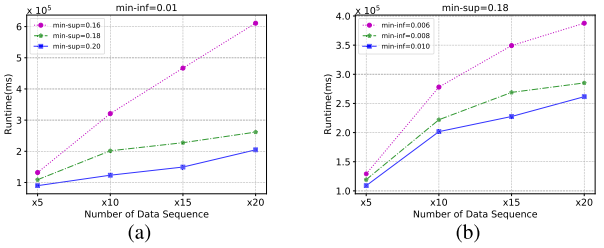

FIGURE 7. Scalability test of InfI-NSP on DS2. (a) The experiment on different *min-sup* values. (b) The experiment on different *min-inf* values.

when min-inf was a fixed value of 0.01 and the values of min-sup were 0.16, 0.18, and 0.20, respectively. Fig 7 (b) shows the runtime of the InfI-NSP on these datasets when min-sup was a fixed value of 0.18 and the values of min-inf were 0.006, 0.008, and 0.01, respectively. As shown in Fig 7 (a) and (b), the runtime has a roughly linear relationship as the data size increases under different min-sup and min-inf values. The results of the scalability test show that our algorithm InfI-NSP performs well on large datasets.

VI. CONCLUSION

NSP mining can help people analyze behavioral information more comprehensively and has attracted increasing attention in recent years. Traditional NSP mining algorithms use only support as a measure of interestingness. However, the support measure cannot truly reflect the interestingness of patterns, and thus, the mining results may contain some uninteresting frequent patterns. Moreover, the existing influence proposed for this problem does not involve NSP mining. Therefore, in this study, we propose a new NSP mining algorithm, InfI-NSP, to mine interesting NSPs based on influence. InfI-NSP uses the influence in traditional NSP mining, which considers the effect of the support of every element as well as the order characteristics among these elements. The interestingness of the patterns is truly reflected. Experiments show that our proposed InfI-NSP algorithm can effectively mine truly interesting NSPs.

In the future, we plan to focus on modifying the current influence calculation method and applying it to more traditional NSP mining algorithms. In addition, we will try to mine interesting NSPs based on influence on incremental databases or stream data. Furthermore, we intend to extend the influence to other research fields related to NSP mining, such as negative sequential rule mining, Top-k NSP mining, and high-utility NSP mining.

ACKNOWLEDGMENT

(Fengling Cui and Xiaoqiang Ren are co-first authors.)

REFERENCES

[1] X. Jiang, T. Xu, and X. Dong, "Campus data analysis based on positive and negative sequential patterns," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 33, no. 5, May 2019, Art. no. 1959016.

[2] E. M. Real, E. P. Pimentel, and J. C. Braga, "Analysis of learning behavior in a programming course using process mining and sequential pattern mining," in *Proc. IEEE Frontiers Educ. Conf. (FIE)*, Oct. 2021, pp. 1–9.

[3] M. I. Al-Twijri, J. M. Luna, F. Herrera, and S. Ventura, "Course recommendation based on sequences: An evolutionary search of emerging sequential patterns," *Cognit. Comput.*, vol. 14, no. 4, pp. 1–22, 2022.

[4] L. Cao, "Health and medical behavior informatics," in *Biomedical Information Technology*. Amsterdam, The Netherlands: Elsevier, 2020, pp. 735–761.

[5] S. Kang, "Personalized prediction of drug efficacy for diabetes treatment via patient-level sequential modeling with neural networks," *Artif. Intell. Med.*, vol. 85, pp. 1–6, Apr. 2018.

[6] B. Nelson, G. P. Amminger, H. P. Yuen, N. Wallis, M. J. Kerr, L. Dixon, C. Carter, R. Loewy, T. A. Niendam, M. Shumway, and S. Morris, "Staged treatment in early psychosis: A sequential multiple assignment randomised trial of interventions for ultra high risk of psychosis patients," *Early Intervent Psychiatry*, vol. 12, no. 3, pp. 292–306, 2018.

[7] H. H. Le, H. Edman, Y. Honda, M. Kushima, T. Yamazaki, K. Araki, and H. Yokota, "Fast generation of clinical pathways including time intervals in sequential pattern mining on electronic medical record systems," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2017, pp. 1726–1731.

[8] A. P. Wright, A. T. Wright, A. B. McCoy, and D. F. Sittig, "The use of sequential pattern mining to predict next prescribed medications," *J. Biomed. Inform.*, vol. 53, pp. 73–80, Feb. 2015.

[9] J. K. Tarus, Z. Niu, and D. Kalui, "A hybrid recommender system for e-learning based on context awareness and sequential pattern mining," *Soft Comput.*, vol. 22, no. 8, pp. 2449–2461, Apr. 2018.

[10] Y. Song, L. Cao, X. Wu, G. Wei, W. Ye, and W. Ding, "Coupled behavior analysis for capturing coupling relationships in group-based market manipulations," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2012, pp. 976–984.

[11] M. S. Nawaz, P. Fournier-Viger, M. Z. Nawaz, G. Chen, and Y. Wu, "MalSPM: Metamorphic malware behavior analysis and classification using sequential pattern mining," *Comput. Secur.*, vol. 118, Jul. 2022, Art. no. 102741.

[12] L. Cao and S. Y. Philip, *Behavior Computing: Modeling, Analysis, Mining and Decision*. London, U.K.: Springer, 2012.

[13] L. Cao, T. Joachims, C. Wang, E. Gaussier, J. Li, Y. Ou, D. Luo, R. Zafarani, H. Liu, G. Xu, Z. Wu, G. Pasi, Y. Zhang, X. Yang, H. Zha, E. Serra, and V. S. Subrahmanian, "Behavior informatics: A new perspective," *IEEE Intell. Syst.*, vol. 29, no. 4, pp. 62–80, Jul./Aug. 2014.

[14] L. Cao, "In-depth behavior understanding and use: The behavior informatics approach," *Inf. Sci.*, vol. 180, no. 17, pp. 3067–3085, Sep. 2010.

[15] J. Han, H. Cheng, D. Xin, and X. Yan, "Frequent pattern mining: Current status and future directions," *Data Mining Knowl. Discovery*, vol. 15, no. 1, pp. 55–86, Aug. 2007.

[16] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. 11th Int. Conf. data Eng.*, Mar. 1995, pp. 3–14.

[17] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. Int. Conf. Extending Database Technol.* Berlin, Germany: Springer, 1996, pp. 1–17.

[18] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proc. 17th Int. Conf. Data Eng.*, Apr. 2001, pp. 215–224.

[19] M. J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Mach. Learn.*, vol. 42, nos. 1–2, pp. 31–60, Jan. 2001.

[20] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu, "Sequential pattern mining using a bitmap representation," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 429–435.

[21] P. Qiu, L. Zhao, and X. Dong, "NegI-NSP: Negative sequential pattern mining based on loose constraints," in *Proc. 43rd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2017, pp. 3419–3425.

[22] Z. Zheng, Y. Zhao, Z. Zuo, and L. Cao, "Negative-GSP: An efficient method for mining negative sequential patterns," in *Proc. Conf. Res. Pract. Inf. Technol.*, 2009, pp. 1–5.

[23] L. Cao, X. Dong, and Z. Zheng, "e-NSP: Efficient negative sequential pattern mining," *Artif. Intell.*, vol. 235, pp. 156–182, Jun. 2016.

[24] X. Gao, Y. Gong, T. Xu, J. Lu, Y. Zhao, and X. Dong, "Toward better structure and constraint to mine negative sequential patterns," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 2, pp. 571–585, Feb. 2020.

[25] J. WU, A. OUYANG, and L. ZHANG, "Statistically significant sequential patterns mining algorithm under influence degree," *J. Comput. Appl.*, vol. 42, no. 9, p. 2713, 2022.

[26] S.-C. Hsueh, M.-Y. Lin, and C.-L. Chen, "Mining negative sequential patterns for E-commerce recommendations," in *Proc. IEEE Asia–Pacific Services Comput. Conf.*, Dec. 2008, pp. 1213–1218.

[27] Z. Zheng, Y. Zhao, Z. Zuo, and L. Cao, "An efficient ga-based algorithm for mining negative sequential patterns," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Berlin, Germany: Springer, 2010, pp. 262–273.

[28] X. Dong, Y. Gong, and L. Cao, "E-RNSP: An efficient method for mining repetition negative sequential patterns," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 2084–2096, May 2018.

[29] X. Dong, Y. Gong, and L. Cao, "F-NSP+: A fast negative sequential patterns mining method with self-adaptive data storage," *Pattern Recognit.*, vol. 84, pp. 13–27, Dec. 2018.

[30] W. Wang and L. Cao, "VM-NSP: Vertical negative sequential pattern mining with loose negative element constraints," *ACM Trans. Inf. Syst.*, vol. 39, no. 2, pp. 1–27, Apr. 2021.

[31] T. Guyet and R. Quiniou, "NegPSpan: Efficient extraction of negative sequential patterns with embedding constraints," *Data Mining Knowl. Discovery*, vol. 34, no. 2, pp. 563–609, Mar. 2020.
[32] P. Fournier-Viger, J. C. Lin, B. Vo, T. T. Chi, J. Zhang, and H. B. Le, "A survey of itemset mining," *WIREs Data Mining Knowl. Discovery*, vol. 7, no. 4, p. e1207, Jul. 2017.
[33] W. Gan, J. C. W. Lin, P. Fournier-Viger, H. C. Chao, and P. S. Yu, "A survey of parallel sequential pattern mining," *ACM Trans. Knowl. Discovery Data*, vol. 13, no. 3, pp. 1–34, 2019.
[34] F. Chiclana, R. Kumar, M. Mittal, M. Khari, J. M. Chatterjee, and S. W. Baik, "ARM–AMO: An efficient association rule mining algorithm based on animal migration optimization," *Knowl.-Based Syst.*, vol. 154, pp. 68–80, Aug. 2018.
[35] C.-S. Wang and J.-Y. Chang, "MISFP-growth: Hadoop-based frequent pattern mining with multiple item support," *Appl. Sci.*, vol. 9, no. 10, p. 2075, May 2019.
[36] Y. S. Koh and S. D. Ravana, "Unsupervised rare pattern mining: A survey," *ACM Trans. Knowl. Discovery Data*, vol. 10, no. 4, pp. 1–29, 2016.
[37] P. Fournier-Viger, Z. Li, J. C.-W. Lin, R. U. Kiran, and H. Fujita, "Efficient algorithms to identify periodic patterns in multiple sequences," *Inf. Sci.*, vol. 489, pp. 205–226, Jul. 2019.
[38] X. Liu, J. Wu, F. Gu, J. Wang, and Z. He, "Discriminative pattern mining and its applications in bioinformatics," *Briefings Bioinform.*, vol. 16, no. 5, pp. 884–900, 2015.
[39] H. H. Yu, C. H. Chen, and S. Tseng, "Mining emerging patterns from time series data with time gap constraint," *Int. J. Innov. Comput., Inf. Control*, vol. 7, no. 9, pp. 5515–5528, 2011.
[40] W. Gan, J. C.-W. Lin, J. Zhang, P. Fournier-Viger, H.-C. Chao, and P. S. Yu, "Fast utility mining on sequence data," *IEEE Trans. Cybern.*, vol. 51, no. 2, pp. 487–500, Feb. 2020.
[41] J. C.-W. Lin, P. Fournier-Viger, and W. Gan, "FHN: An efficient algorithm for mining high-utility itemsets with negative unit profits," *Knowl.-Based Syst.*, vol. 111, no. 1, pp. 283–298, 2016.
[42] W. Gan, J. C.-W. Lin, P. Fournier-Viger, H.-C. Chao, and V. S. Tseng, "Mining high-utility itemsets with both positive and negative unit profits from uncertain databases," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2017, pp. 434–446.
[43] H. Kim, T. Ryu, C. Lee, H. Kim, E. Yoon, B. Vo, J. C.-W. Lin, and U. Yun, "EHMIN: Efficient approach of list based high-utility pattern mining with negative unit profits," *Expert Syst. Appl.*, vol. 209, Dec. 2022, Art. no. 118214.
[44] T. Guns, S. Nijssen, and L. De Raedt, "K-pattern set mining under constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 2, pp. 402–418, Feb. 2011.
[45] F. Petitjean, T. Li, N. Tatti, and G. I. Webb, "Skopus: Mining top-K sequential patterns under leverage," *Data Mining Knowl. Discovery*, vol. 30, no. 5, pp. 1086–1111, Sep. 2016.
[46] Y. Zhao, H. Zhang, L. Cao, C. Zhang, and H. Bohlscheid, "Mining both positive and negative impact-oriented sequential rules from transactional data," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Berlin, Germany: Springer, 2009, pp. 656–663.
[47] X. Dong, Z. Zheng, L. Cao, Y. Zhao, C. Zhang, J. Li, W. Wei, and Y. Ou, "E-NSP: Efficient negative sequential pattern mining based on identified positive patterns without database rescanning," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, 2011, pp. 825–830.

FENGLING CUI received the bachelor's degree in software engineering. She is currently pursuing the master's degree in computer technology with the Qilu University of Technology (Shandong Academy of Sciences). Her research interests include sequential pattern mining, negative sequential pattern mining, sequential rule mining, and negative sequence analysis.

XIAOQIANG REN received the bachelor's degree from the School of Computer Science, Shandong Institute of Light Industry, in 2000, and the master's degree from the School of Computer Science and Engineering, Shandong University of Science and Technology, in 2008. He is currently an Assistant Professor at the Qilu University of Technology. His research interests include machine learning and data mining.

XIANGJUN DONG received the Ph.D. degree in computer applications from the Beijing Institute of Technology, China, in 2005. From 2007 to 2009, he worked as a Postdoctoral Fellow with the School of Management and Economics, Beijing Institute of Technology. From 2009 to 2010, he was a Visiting Scholar with the University of Technology Sydney, Australia. He is currently a Professor with the Department of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China. His research interests include data mining, artificial intelligence, and big data. He has published more than 100 journals/conference publications, including *Artificial Intelligence*, *IJCAI*, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *Pattern Recognition*, and CIKM.