Faculty of Engineering and Information Technology

University of Technology Sydney

Nonoccurring Sequential Behavior Analytics

A thesis submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

by

Wei Wang

July 2020

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Wei Wang declare that this thesis, is submitted in fulfilment of the requirements for the award of the degree: Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Production Note:

Signature: Signature removed prior to publication.

Date: 12 July 2020

Acknowledgments

Foremost, I would like to express the deepest appreciation to my supervisor, Professor Longbing Cao, for his professional guidance, persistent help, and continuous support throughout my Ph.D. studies and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me throughout the research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. studies.

In addition, I would like to thank all my colleagues in Advanced Analytics Institute: Liang Hu, Guansong Pang, Thac Do, Chengzhang Zhu, and Shoujin Wang for their stimulating discussions and scientific advice. Without their generous support, this dissertation would not have been possible.

Last but not least, I would like to thank my family for their unconditional support, both financially and emotionally throughout my Ph.D. studies. Without their encouragement, finishing this dissertation would have been impossible; without them, nothing would have any value.

Wei Wang
July 2020 @ UTS

Contents

List of Figures

List of Tables

List of Publications

Published Papers

1. **Wei Wang**, and Longbing Cao, 2019. "Negative Sequence Analysis: A Review," *ACM Computing Surveys*, 52(2), pp. 32:1-32:39.

Papers under Revision

1. **Wei Wang**, and Longbing Cao, "VM-NSP: An Efficient Vertical Negative Sequential Pattern Mining Framework," *ACM Transactions on Information Systems*, Under review (Minor revision, Submission ID: TOIS-2020-0018).
2. **Wei Wang**, and Longbing Cao, "Determinantal Point Process-based Relation Modeling in Negative Sequence Analysis," *IEEE Transactions on Knowledge and Data Engineering*, Under review (Major revision, Submission ID: TKDE-2020-03-0252).
3. **Wei Wang**, and Longbing Cao, "Sequential Basket Recommendation by Iteratively Learning Basket Relations and Interactive Feedbacks," *ACM Transactions on Information Systems*, Under review (Minor revision, Submission ID: TOIS-2019-0030.R1).

Abstract

Behavior analytics has attracted increasing attention in broad communities as a major research area in understanding and managing the dynamics of complex systems and problems such as series of medical treatments, interactions between customers and service providers, and online communications. Sequential behavior analytics aims to understand, analyze, detect, and predict existing or future behaviors and behavior sequences. Existing methods for sequential behavior analytics only focus on occurred or to-occur behaviors (also called positive behaviors), while ignoring nonoccurring behaviors (also called negative behaviors), which are often useful for understanding, managing and predicting hidden or unseen yet important behaviors that differ from and typically mix with occurred ones. Nonoccurring behaviors complement occurring ones for complete and deep behavior analytics, while very limited theoretical progress has been made.

This thesis studied the theory to comprehensively model the complex relations within and between behaviors and to effectively discover and predict interesting sequential occurring and nonocurring behaviors. Specifically, it focused on (1) forming a comprehensive and systematic representation, formalization, and theoretical system for defining and representing the concepts, problems, constraint settings, and negative containment of nonoccurring sequential behavior (NSB) analytics; (2) efficiently discovering the high-frequency negative sequential patterns (NSP) composed of both occurring/nonoccurring behaviors; (3) discovering the representative NSP subset by exploring the complicated explicit/implicit behavior relations; and (4) enabling the sequential basket recommendation system (SBRS) through learning behavior relations and interactive feedback. Accordingly, this thesis proposed (1) a vertical NSP mining framework and its instantiation for the efficient discovery of the complete set of NSP with the loose negative element constraint via the vertical representation of each sequence, which guarantees the coverage of flexible patterns with complicated behavior relations; (2) a determinantal point processes-based (DPP-based) representative NSP discovery approach for the selection of a representative subset of the high-quality and diverse patterns by jointly modeling explicit and implicit sequential element/pattern relations; and (3) a hierarchical attentive encoder-decoder model for interactive sequential basket recommendation, which jointly models both intra-/inter-basket relations in sequential user basket behaviors as well as incorporates positive/negative feedback to enable negative feedback-based refinement.

The extensive empirical analysis of the proposed methods demonstrated that our methods performed significantly better than the state-of-the-art methods in the same domain in terms of multiple evaluation metrics.

Chapter 1

Introduction

This chapter introduces the motivation, challenges, aims, and objectives of this thesis and presents the research contributions. In addition, the structure of this thesis is provided at the end of this chapter.

1.1 Research Motivation and Challenges

Behavior data exists widely in our daily life and work (Cao & Philip 2012), and nonoccurring sequential behaviors (NSBs), which are the ordered and behavioral sequences consisting of both occurring and nonoccurring behaviors (also called positive and negative behaviors, respectively), always act as a unique and critical tool for behavior informatics. Nonoccurring sequential behaviors are often useful and practical for capturing informative and actionable knowledge in comprehensively understanding, managing and predicting the complexities, dynamics, and impact of nonoccurring behaviors (NOBs) (Cao, Dong & Zheng 2016, Cao, Yu & Kumar 2015) and complicated behavior relations (Beg & Butt 2009, Song, Cao, Wu, Wei, Ye & Ding 2012, Cao 2014, Cao 2015), as they can disclose additional yet usually hidden information which otherwise cannot be replaced or informed by the pure occurring behaviors alone (Anwar, Petrounias, Morris & Kodogiannis 2010, Cao et al. 2016, Cao, Yu, Zhang & Zhang 2008). Here, the nonoccurring behaviors involved in an NSB stand for the absence of specific events or actions corresponding to important but undeclared or missing behaviors (Cao et al. 2015), often appearing in manipulated or undesirable behaviors (Cao & Yu 2012), concerned situations (Lin, Chen & Hao 2007), and business applications (Cao, Yu, Zhang & Zhang 2008), for example, hiding external income (thus undeclared in the transactions) to obtain a government low-income allowance or missing an important appointment such as a medical treatment (Hsueh, Lin & Chen 2008, Cao & Yu 2012). While nonoccurring events are usually overlooked, they can be very useful in areas including missing health/medical treatment detection (Gong, Liu & Dong 2015), fraudulent health insurance claim detection (Zheng, Zhao, Zuo & Cao 2009, Zheng, Zhao, Zuo & Cao 2010), debt detection in undeclared social welfare behaviors (Zhao, Zhang, Wu, Pei, Cao, Zhang & Bohlscheid 2009, Zhao, Zhang, Cao, Zhang & Bohlscheid 2008, Zhao, Zhang, Cao, Zhang & Bohlscheid 2009, Zhao, Zhang, Figueiredo, Cao & Zhang 2007, Cao, Zhao & Zhang 2008*a*) and undeclared taxpayer behaviors (Zheng, Wei, Liu, Cao, Cao & Bhatia 2016), manipulative trading behavior analysis (Song et al. 2012, Cao, Ou, Yu & Wei 2010, Cao, Ou & Yu 2012), pattern relation analysis (Cao 2012), academic performance detection (Jiang, Gao, Xu & Dong 2018), over-service detection and business decision-making analysis (Zheng 2012), counter-terrorism, security and risk management (Cao et al. 2016), etcetera. Accordingly, the analytics on NSB (i.e., nonoccurring sequential behavior analytics), is increasingly becoming significant and plays an irreplaceable role in many real-world applications.

For example, in social welfare debt detection, it is found that those allowance recipients who fail to claim some specific activities: PYR, RPR, REA and STM may receive overpayment paid by the government due to the misleading information unprovided (i.e., NSB: $\neg < PYR, RPR, REA, STM >$ $\rightarrow DEB$) (Zhao, Zhang, Wu, Pei, Cao, Zhang & Bohlscheid 2009). More generally, in detecting insurance claim fraud, suppose $S_{pos} =< a, b, c, d >$ is a claim sequence of a customer; if $S_{neg} =< a, b, \neg c, d >$ is a high-frequency NSB pattern (also called a *negative sequential pattern*, i.e., *NSP* in behavior and sequence analysis) and $sup(S_{pos})/sup(S_{neg}) < min\_ratio$, then S_{pos} is likely fraudulent, since code c should be claimed together with others but it fails to appear together in the claim (Zheng 2012). Here, each item a, b, c, and d stands for a claim item code, $sup(S_{pos})$ and $sup(S_{neg})$ are the corresponding supports, and min_ratio is a predefined frequency threshold. For the example of the e-commerce recommendation, given a *positive sequential pattern* (PSP) $S_1 =< MacBook, iPhone, iPod >$ and two NSP $S_2 =< MacBook, \neg iPhone, Xbox >$ and $S_3 =< MacBook, iPhone, \neg Xbox >$, we then know that: 1) for a customer who has purchased $MacBook$, $iPod$ should be recommended if $iPhone$ was bought after $MacBook$; otherwise, $Xbox$ can be recommended according to S_1 and S_2; and 2) the sales of $Xbox$ are highly dependent on the occurrences of $iPhone$, and recommending $Xbox$ can be made by validating the subsequent presence of $iPhone$ given that $MacBook$ is purchased ahead (Hsueh et al. 2008). For the example of system diagnosis, if some maintenance operations should be but were not conducted following certain alarms, a system fault or even disaster may occur; while if these operations were performed in time, then alarms would stop and no fault would occur, i.e., $S_\alpha =< a, b, o, X >$ and $S_\beta =< a, b, \neg o, Y >$. Here a and b stand for two alarms, o stands for the conduction of the maintenance operation, and X and Y stand for no-fault and fault. Given alarms a and b are received, S_α represents a pattern that no-fault X would occur if operation o were performed while S_β represents that otherwise fault Y would likely occur.

Although NSB is fundamental and significant, the analytics on NSB are still at an early stage, and only limited attention has been received in recent years (Cao et al. 2015, Liu, Dong, Li & Li 2015*a*, Dong, Gong & Zhao 2014, Kamepalli & Kurra 2014, Li, Algarni & Zhong 2010), which faces many significant challenges.

- First, the hidden nature of nonoccurring behavior makes the definition of the NSB concept and problem sophisticated, and thus significant inconsistencies exist between existing related work, with each only focusing on specific scenarios and settings (Cao et al. 2016). There are no uniform formal representations, problem statements, constraint settings, or negative containment definitions (Gong et al. 2015). This has limited the theoretical development and applications of NSB analytics compared to its widespread scenarios. To address the demand of and significant gaps in existing NSB research, a comprehensive and systematic formalization on NSB analytics is essential.

- Second, the intrinsic complexities of nonoccurring behaviors significantly increase the *computational complexity* and enlarge the *search space* of NSB analytics. It is especially the case for the pattern mining-based NSB analytics method (i.e., *negative sequential pattern (NSP) mining* method), which is one of few approaches available for understanding NOB (Cao et al. 2016). Negative sequential pattern mining aims to discover the set of high-frequency NSB patterns with both nonoccurring and occurring behaviors based on predefined constraints and containment setting (Wu, Zhang & Zhang 2004). However, suffering from a violation of the downward closure property of NOB, most state-of-the-art NSP mining methods are either too inefficient or too specific, and they cannot discover the complete set of high-frequency NSP efficiently because these methods tend to generate and test a large number of candidates even in a medium-sized dataset with a support threshold that is not too low. In addition, various strict constraints are incorporated to control the problem complexities (Liu et al. 2015*a*, Dong et al. 2014, Kamepalli & Kurra 2014, Li et al. 2010, Kazienko 2008), while they always violate the real-life scenarios, leading to limited coverage of resultant NSPs and missing valuable patterns. It is essential yet challenging to develop efficient mining frameworks and methods to discover NSP with loose constraints for complete pattern mining (Cao et al. 2016).

- Third, most of the existing NSP mining methods result in the extremely large-scale collection of high-frequency yet highly overlapping or redundant findings and missing informative yet relatively low-frequency behaviors, making the discovered NSP less actionable. Accordingly, the discovery of a small-scale but representative subset of high-quality and diverse NSP with low complexity and high efficiency is critical yet complicated (Zheng 2012). This *representative NSP discovery* aims to filter highly-similar and redundant patterns but retain those informative ones to represent the whole collection, making further analysis and applications of NSB patterns less complicated and time-consuming. However, little progress has been seen in representative NSP discovery (Liu et al. 2015*a*), and a significant gap exists between the limited research and its great application demand.

- Lastly, the pattern mining-based NSB analytics methods (e.g., NSP mining methods), only evaluate the significance of each NSB from the perspective of its global information, such as the frequency of entire NSBs. Thus, all the low-frequency NSBs are overlooked, even though they reflect valuable knowledge in certain domains. This makes the NSPs discovered less practical in some prediction-oriented applications (e.g., recommendation system), due to the fact that the given user's sequential behaviors may consist of arbitrary behaviors and thus may be too low-frequency to match any discovered NSP (Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017, Wang, Hu, Cao, Huang, Lian & Liu 2018). In addition, the NSPs discovered fail to capture the complicated behavior relations inside each NSB, especially for the behavior relations within one element (i.e., intra-element or intra-basket relations) as well as the behavior relations across different elements (i.e., inter-element or inter-basket relations), which is of great significance to distinguish the dominated behaviors or behavior combinations and make prediction for future behaviors as per the historical sequential behaviors. This inspires our research on the comprehensive exploration of the hierarchical behavior relations involved in sequential behaviors as well as the incorporation of the interactive occurring/nonoccurring feedback behaviors (i.e., positive/negative feedback) to enable sequential behavior prediction, which is of critical importance in analyzing and predicting NOB.

There still exists a significant gap between the wide NOB applications and limited research on NSB analytics. In this thesis, we introduce several NSB analytics methods to discern the knowledge in understanding, managing, and predicting the nonoccurring behaviors and sophisticated behavior relations. To address the first challenge, we provide a comprehensive overview of NSB analytics and propose a systematic formalization of the concepts, problems, constraint settings, and negative containment definitions for NSB analytics. In addition, for the second challenge, we incorporate a *loose negative element constraint* (LNEC) into *NSP mining* to resolve the issue of valuable patterns missing by discovering the NSP containing partial negative elements and propose a novel and efficient vertical mining framework, VM-NSP, by introducing a vertical representation of NSB to efficiently discover the complete set of NSPs. A bitmap-based NSP mining method, bM-NSP, is further proposed based on the VM-NSP framework to optimize discovery performance. Moreover, to solve the third challenge, we make the first attempt by inventing a determinantal point processes-based (DPP-based) method - explicit and implicit element/pattern relations-based representative NSP discovery (EINSP) - to select a representative subset of high-quality and diverse NSP by comprehensively involving both cooccurrence-based explicit relations and nonoccurrence-based implicit relations between NSP elements and patterns. Finally, for the last challenge, we first formalize the problem definition of interactive sequential basket recommender systems (SBRSs), which targets making continuous predictions for a sequence of future behaviors (in the form of baskets of items) based on the user's historical sequential behaviors and interactive feedback, and then propose a hierarchical attentive encoder-decoder model (HAEM) to address this intricate SBRS problem, which jointly models the intra-/inter-basket behavior relations in sequential user purchase behaviors and incorporates the occurring/nonoccurring feedbacks (i.e., positive/negative feedback) to enable negative feedback-based refinement.

The aims, objective, and contributions of this thesis are described in Section 1.2.

1.2 Research Aims and Objectives

This thesis aims to comprehensively model the complex relations within and between behaviors and effectively discover and predict interesting sequential occurring and nonocurring behaviors, with special attention paid to:

- consolidate and form a comprehensive and systematic representation, formalization, and theoretical system for defining and representing the concepts, problems, constraint settings, and negative containment for NSB analytics;
- conduct studies of more effective and efficient method that can discover a large-coverage complete set of high-frequency and flexible NSPs in health insurance claim detection, e-commerce analysis or click-stream behavior analysis;
- conduct studies of more actionable method that can select a small-scale but representative subset of high-quality and diverse NSP to represent the discriminative information of the whole pattern collection;
- conduct studies of more practical method that can enable continuous prediction for a sequence of future behaviors based on the user's historical sequential behaviors by comprehensively exploring the complicated behavior relations and incorporating interactive feedback, which shows great potential in handling sequential recommendation applications such as e-commerce recommendation or news/music prediction.

The objectives of this thesis are to:

- formalize a general description of NSB concepts and a generic problem statement of NSB analytics and then investigate and consolidate the definitions of constraints and negative containment;

- propose a loose constraint for NSP mining to include partial negative elements containing both positive and negative items, enabling the discovery of more flexible patterns with complicated behavior relations, and then present a novel framework and its instantiation method to efficiently discover the complete set of NSPs with the loose constraint;
- propose a novel representative NSP discovery approach to jointly explore the complicated explicit and implicit behavior relations between NSB elements and patterns for selecting a representative subset of high-quality and diverse NSPs; and
- formalize a general problem of *interactive sequential basket recommendation* for continuously sequential behavior prediction and propose a method to explore the compound behavior relations involved in NSB and incorporate interactive positive/negative feedback to address the intricate SBRS problem.

1.3 Thesis Contributions

This thesis makes the following contributions:

- Nonoccurring Sequential Behavior Analytics Formalization (*Chapter 3*)
  - Proposing a systematic formalization of the NSB analytics problem, in which the formulas of sequential behaviors are specified on top of formal concepts: *item*, *element*, and *sequence*;
  - Providing a comprehensive formalization of constraint settings and defining the negative containment in terms of various levels such as on element, sub-sequence and super-sequence.

- Efficient Negative Sequential Pattern Mining (*Chapter 4*)
    - Formulating the problem of loose negative element constraint-enabled (LNEC-enabled) NSP mining and incorporating the necessary constraint settings and negative containment to make the problem resolvable;
    - Proposing a novel vertical mining framework, VM-NSP, to efficiently discover the complete set of NSPs with partial negative elements by incorporating a vertical representation for each NSB;
    - Proposing an efficient bitmap-based NSP mining method, bM-NSP, by introducing a prefix-based *negative sequential candidate* (NSC) generation strategy to further optimize the efficiency of NSP mining, of which the performance superiority on datasets with different data characteristics can be confirmed by theoretical analysis.
- Representative Negative Sequential Pattern Discovery (*Chapter 5*)
    - Converting the problem of *representative NSP discovery* to a probabilistic subset selection problem in a DPP graph, which takes advantage of the probabilistic DPP theoretical foundation and strength in subset selection with diversity.
    - Modeling the cooccurring and nonoccurring behaviors and pattern relations in NSPs in terms of the direct and indirect DPP-based node/edge dependencies, which captures the quality and diversity of each pattern in the NSP collection in terms of both explicit element/pattern cooccurrences and implicit nonoccurrences conditional on third parties. This captures rich element interactions

and pattern relations in NSB analytics, rarely explored in existing related work.

  - Proposing the explicit and implicit element/pattern relations-based representative NSP discovery (EINSP) approach by integrating both the above explicit and implicit element/pattern relations in the DPP-based NSP graph and effectively sampling those highly explicitly and implicitly related NSPs as a representative subset of the high-quality and diverse NSB patterns.

- Interactive Sequential Basket Recommendation (*Chapter 6*)

  - Presenting and formalizing the problem of *sequential basket recommendation* that enables the prediction of a sequence of next-baskets by exploring the user's sequential behaviors and iteratively feeding positive/negative feedback.

  - Making the first effort to jointly model the intra-/inter-basket behavior relations in sequential behaviors and incorporating the both positive/negative feedback to enable negative feedback-based refinement for interactive sequential basket recommendation.

  - Proposing a hierarchical attentive encoder-decoder model (HAEM) to continuously predict next baskets one after another by analyzing behavior relations both within a basket and between adjacent sequential baskets and incorporating the selection/non-selection feedback for recommendation refinement.

The above contributions achieve the research objectives and establish a series of methods for NSB analytics.

1.4 Thesis Organization

The structure of this thesis is presented in Figure 1.1. The thesis begins with an introductory chapter that briefly introduces our research motivation, challenges, aims, objectives, and contributions, followed by a chapter that presents a comprehensive literature review of research related to our work on NSB analytics. This is followed by the main body of this thesis, which is organized into four chapters according to the research objectives: (1) Chapter 3 formalizes the problem of nonoccurring sequential behavior analytics and introduces fundamental concepts and definitions including constraint settings and negative containment throughout this thesis; (2) Chapter 4 proposes an effective and efficient framework and its method for the complete negative sequential pattern mining with loose constraints; (3) Chapter 5 offers a DPP-based subset selection method for the representative negative sequential pattern discovery; and (4) Chapter 6 presents a hierarchical attentive encoder-decoder model for the interactive sequential basket recommendation. The last chapter concludes the thesis and then presents possible future research directions. The summary of each chapter is as follows:

- *Chapter 1:* This chapter briefly introduces the research background, challenges, aims, and objectives of the issue on NSB analytics, together with the thesis contributions and organization.

- *Chapter 2:* This chapter presents an overview of the preliminaries and related work for our proposed methods. Specifically, we present the four typical categorizations of existing NSP mining methods, review and discuss some NSP selection and representative subset selection methods, and finally investigate and compare some classic basket-enabled sequential recommendation methods including two paradigms of next-basket recommender systems (NBRSs) methods, session-based recommendation, interactive recommendation, and hybrid methods.

- *Chapter 3:* Nonoccurring sequential behaviors always capture more

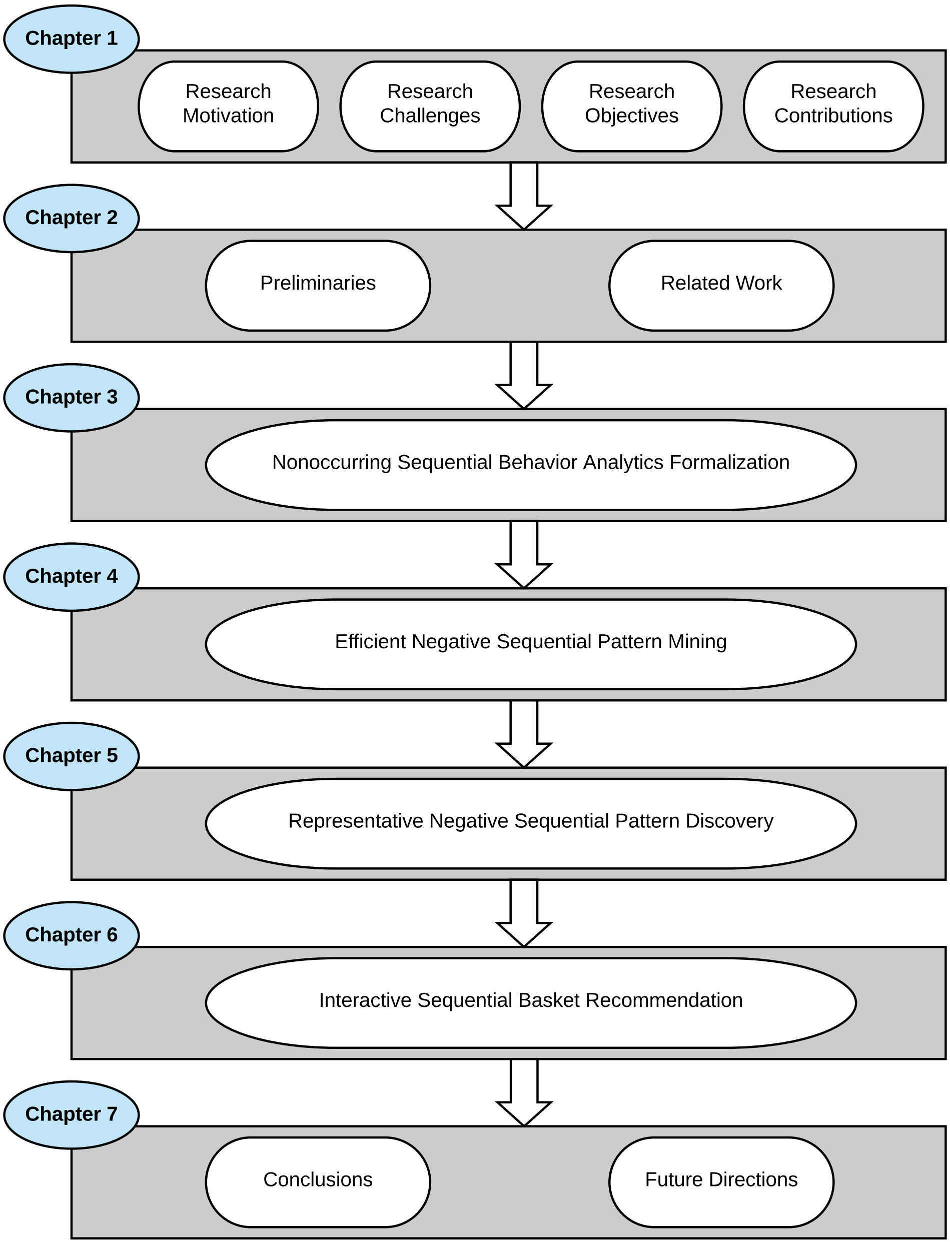

Chapter 1
Research Motivation
Research Challenges
Research Objectives
Research Contributions
Chapter 2
Preliminaries
Related Work
Chapter 3
Nonoccurring Sequential Behavior Analytics Formalization
Chapter 4
Efficient Negative Sequential Pattern Mining
Chapter 5
Representative Negative Sequential Pattern Discovery
Chapter 6
Interactive Sequential Basket Recommendation
Chapter 7
Conclusions
Future Directions

Figure 1.1: Thesis Structure

informative and actionable knowledge than classic occurring sequential behaviors due to involving both occurring and nonoccurring behaviors, which appear in many real-life applications. However, the research on NSB analytics is still at an early stage, and related methods involve high computational complexity and a large search space; there is no widely-accepted problem formalization on NSB analytics, and different settings on constraints and negative containment have been proposed in the existing work. This chapter formalizes a generic problem statement of NSB analytics and investigates and consolidates the definitions of constraints and negative containment.

- *Chapter 4:* Negative sequential pattern mining is challenging as it involves fundamental challenges that require its own theoretical foundation and cannot be directly addressed by traditional positive sequential pattern mining. In the limited research reported on NSP mining, negative element constraint (NEC) is incorporated to only consider the NSPs composed of specific forms of elements (containing either positive or negative items), which results in limited coverage of resultant patterns and the lack of many valuable NSPs. This chapter loosens the NEC (called loose NEC, i.e., LNEC) to include partial negative elements containing both positive and negative items, enabling the discovery of more flexible patterns. Accordingly, the LNEC-enabled NSP mining problem is formalized, and a novel vertical NSP mining framework, VM-NSP, is proposed to efficiently mine the complete set of NSP by a vertical representation of each NSB. An efficient bitmap-based NSP mining method, bM-NSP, introduces a bitmap hash table-based vertical representation and a prefix-based NSC generation strategy to optimize the discovery performance. VM-NSP and its implementation bM-NSP form the first vertical representation-based approach for complete NSP mining with LNEC. Theoretical analysis and experiments confirm the performance superiority of bM-NSP on synthetic and real-life datasets with respect to diverse data factors, which substantially expands existing NSP mining methods towards flexible NSP discovery.

- *Chapter 5:* Sequence analysis becomes increasingly valuable for behavior and event analysis, while limited research is available on NSB analytics which analyzes nonoccurring yet important sequential behaviors. In NSB analytics, NSP discovery aims to discover nonoccurring and occurring elements and patterns but faces significant challenges due to the hidden nature of element and pattern nonoccurrences and their intricate combinations with occurring ones. As one of the only few available NSB analytics approaches, NSP mining methods often result in high computational cost and large-scale and highly-redundant NSPs but miss relatively low-frequency yet informative patterns. This inspires a critical yet rarely explored task - representative NSP discovery: discovering a small-scale subset of high-quality but diverse NSP with low complexity and high efficiency to represent the whole collection. However, significant challenges exist in such representative NSP discovery: 1) no existing pattern selection criteria can judge the representativeness of an NSP subset to avoid repetition and low coverage; and 2) it is essential to explore complicated explicit and implicit behavior relations between NSP elements and patterns. This chapter addresses these challenges by proposing the novel representative NSP discovery approach EINSP, which represents the NSP discovery as a DPP and jointly models explicit and implicit sequential element/pattern relations for selecting representative high-quality and diverse NSPs. Extensive empirical analysis on real-life and synthetic datasets with different data characteristics verifies that the NSP subset selected by EINSP not only achieves higher and more balanced coverage but also more implicitly-related and informative NSPs. In addition, EINSP demonstrates strong performance robustness to diverse data factors.

- *Chapter 6:* While the above pattern mining-based NSB analytics methods offer potential for the discovery of NSB patterns with discriminating global significance in understanding NOBs, they face significant challenges in making predictions for a sequence of future behaviors based on the known sequential behaviors, which is of great practicality in many applications such as e-commerce recommendation or next-news/music prediction. Sequential recommendation such as NBRS emerges as a recent focus for making next recommendation by analyzing sequential user behaviors and the relevant context. However, such techniques only involve inter-basket relations while ignoring the hybridization with the behavior relations among items within a basket (also called intra-basket relation), often producing irrelevant or similar items in the next basket. Further, positive feedback on those items selected by users has been involved in sequential recommendation, while negative feedback on those non-selected ones is ignored, which reflects the user's underlying preference that can be an informative hint for refining the next-basket prediction. This paper addresses the significance and modeling of both intra-/inter-basket behavior relations and positive/negative feedback for interactive sequential basket recommendation. A hierarchical attentive encoder-decoder model (HAEM) is proposed in this chapter to continuously predict next baskets one after another during the interactions with users by analyzing the behavior relations among items both within a basket and between adjacent sequential baskets and incorporating the selection and non-selection feedback on recommended items for recommendation refinement. HAEM comprises a basket encoder and a sequence decoder to model intra-/inter-basket behavior relations and a prediction decoder to sequentially predict next-baskets through interactive feedback-based refinement. Empirical analysis demonstrates that HAEM significantly outperforms the state-of-the-art baselines for NBRS and session-based recommenders in terms of recommendation accuracy and novelty. We also test the effect of continuously refining sequential basket recommendations by involving non-selection feedback during interactions.

- *Chapter 7:* This chapter summarizes the content and contributions of this thesis. It further discusses the possible future directions of the research that may be undertaken in addition to this thesis work.

Overall, the research overview of this thesis regarding the research objectives, targeted challenges, and contributions of each chapter is summarized in Table 1.1.

Table 1.1: Thesis Research Overview (Main Body)

Objective	**Targeted Challenges**	**Contributions**	**Chapter**
NSB Analytics Formalization	Formalization of the concepts, problems, constraint settings, and negative containment in NSB analytics	Specify the formulas of sequential behaviors on top of basic concepts, formalize the problem statement, and consolidate the definitions of constraints and negative containment	3
Efficient Negative Sequential Pattern Mining	Efficient discovery of the complete set of high-frequency and flexible NSP with loose constraints	Formulate the LNEC-enabled NSP mining to discover flexible patterns with partial negative elements	4
		Propose framework and method for efficient LNEC-enabled NSP mining	
Representative NSP Discovery	Discovery of a small-scale yet representative subset of high-quality and diverse NSPs	Convert the problem of representative NSP discovery to a DPP graph-based subset selection problem	5
		Capture the quality and diversity of each NSP in terms of both explicit element/pattern cooccurrences and implicit nonoccurrences conditional on third parties	
		Propose a method to select the high-quality and diverse patterns as a representative subset	
Interactive Sequential Basket Recommendation	Continuous prediction of a sequence of future behaviors based on user's sequential behaviors and interactive feedback	Formalize the problem of sequential basket recommendation to enable the sequential behavior prediction	6
		Jointly model the intra-/inter-basket behavior relations in sequential behaviors and incorporate both positive/negative feedback to enable the negative feedback-based refinement	
		Propose a method for accurate sequential behavior prediction by analyzing behavior relations and incorporating selection/non-selection feedback	

Chapter 2

Preliminaries and Literature Review

In this chapter, we briefly review the work that is closely related to our proposed methods, which are categorized into three groups of methods as per the discussion in Chapter 1: negative sequential pattern mining methods, representative subset selection methods including pattern selection and DPP-based selection methods, and sequential recommendation methods including next-basket and session-based recommendation. First, we present the NSP mining methods, which aim to discover the high-frequency NSB patterns based on predefined constraint settings and negative containment. Next, we provide the preliminaries and related work on the existing subset selection methods, with specific attention on the DPP-based subset selection methods including fixed-size determinantal point processes (k-DPP) and structured determinantal point processes (SDPP), which form the core foundations of our work on representative NSP discovery. Finally, the basket-enabled sequential recommendation methods are introduced, including collaborative filtering (CF), sequential recommendation (SR), session-based recommendation, interactive recommendation and hybrid methods, which make predictions for next-basket behaviors based on historical sequential behaviors and interactive feedback.

2.1 Negative Sequential Pattern Mining

The NSP mining-related research is still at an early stage, and only limited research outcomes have been available in recent years (Gong et al. 2015, Cao et al. 2016). The following NSP mining methods have so far been reported in the current literature: NSPM (Lin, Chen & Hao 2007), two extended NSPM methods namely NFSPM (Lin, Hao, Chen, Chang & Chueh 2007) and PN-SPM (Lin, Chen, Hao, Chueh & Chang 2008), MSIS (Ouyang & Huang 2007), three MSIS extensions including MBFIFS (Ouyang, Huang & Luo 2008), CP-NFMLSP (Ouyang & Huang 2009), and CPNFSP (Ouyang & Huang 2010), Incremental CPNFSP (Khare & Rastogi 2013), SpamNeg (Zhao et al. 2008), PNSP (Hsueh et al. 2008), Negative-GSP (Zheng et al. 2009), GA-NSP (Zheng et al. 2010), e-NSP (Cao et al. 2016), and three extended e-NSP methods namely SAPNSP (Liu et al. 2015*a*), e-msNSP (Xu, Dong, Xu & Gong 2017), and e-NSPFI (Gong, Xu, Dong & Lv 2017). Below, we conduct a preliminary overview of these methods, summarize the technical challenges in NSP mining and analyze the gaps in existing NSP mining research.

2.1.1 Categorization of Existing NSP Mining Methods

As per analytical objectives and settings, the above NSP mining methods can be roughly categorized into four groups: *format-specific NSP mining*, *complete NSP mining*, *stochastic NSP mining*, and *PSP-based NSP mining.*

Format-specific NSPs refer to particular types of NSPs that impose specific constraints on pattern structures and formats. They thus generate NSCs in a smaller search space and reduce the number of NSCs to be verified. Typical methods include NSPM, NFSPM, PNSPM, MSIS, MBFIFS, CP-NFMLSP, and CPNFSP. Among these methods, NSPM, NFSPM, and PN-SPM introduce the concept of location-format constraint to generate NSCs in the format of $< e_1, e_2, \ldots, \neg e_s >$, where a negative element can only appear at the end of an NSC. However, since NSPM-based methods adopt such a strict format constraint, they fail to discover NSPs with multiple negative elements in the other locations, and no patterns discovered can involve the elements composed of both positive and negative items, which means that some informative NSPs can easily be neglected by NSPM. Furthermore, NSPM-based methods calculate the support count of each NSC by re-scanning the whole dataset, and thus significant runtime is inevitable with the generation of numerous NSCs. Moreover, MSIS, MBFIFS, CPNFMLSP and CPNFSP introduce another location-format constraint to define NSC in the form of $< e_1, \neg e_2 >$, $< \neg e_1, e_2 >$ or $< \neg e_1, \neg e_2 >$, namely an NSC can have only two elements, and there is at most one positive element that contains all positive items, which resembles the mining of negative association rules (Jiang, Luan & Dong 2012, Zhao, Zhang, Cao, Zhang & Bohlscheid 2009, Zhao, Zhang, Wu, Pei, Cao, Zhang & Bohlscheid 2009, Dong, Sun, Han & Hou 2006, Wu et al. 2004). This category of NSP methods explores NSC via a simplified NSC generation strategy with low computational complexity, but these methods are only applicable for specific applications since the NSPs discovered are quite constrained.

The second category, *complete NSP mining*, aims to discover the complete set of NSPs, which adjusts the existing PSP method to discover all the NSPs satisfying a given threshold. Typical methods include PNSP and Negative-

GSP (NegGSP), which are both extended from the GSP method (Srikant & Agrawal 1996). Both PNSP and NegGSP adopt the NSC generation-and-testing strategy. This strategy first adapts classic PSP methods to generate long-length or long-size NSCs based on mined PSPs and NSCs and then tests whether they are high-frequency patterns by calculating their negative supports in a pass over the whole dataset. Here, PNSP adopts an appending-based NSC generation strategy to discover the NSP containing no contiguous negative elements, which generates an s-size NSC by appending an (s-1)-size NSC or PSP with a high-frequency positive or negative itemset. PNSP maintains a relatively low performance method because it is a size-based NSP mining method and generates a larger number of invalid NSCs, requiring greater execution time in the appending and support calculating process (Zheng et al. 2009). Comparatively speaking, NegGSP adopts a joining-based strategy which generates an l-length NSC by joining an (l-1)-length seed or PSP with another (l-1)-length seed. This category of NSP mining methods demonstrate potential in discovering a larger number of NSPs, especially for long-size or long-length NSPs with wider item distribution. However, these methods always require greater runtime in generating and testing enormous NSCs, and much larger memory space is required to save the generated NSCs. In addition, many of the discovered NSPs may not be actionable (Cao, Yu, Zhao & Zhang 2010, Wang & Cao 2012) since their interestingness may be too low to attract business interest. Due to the large scale of the resultant pattern collection, it is challenging to distinguish and apply the actionable knowledge in practice, making the further analysis and application of NSP complicated and time-consuming (Zheng 2012, Liu et al. 2015*a*).

The third category, *stochastic NSP mining*, targets the discovery of the high-frequency NSPs by incorporating a stochastic strategy to NSP mining and only generating potentially high-frequency NSCs by specific operations between the selected optimal NSPs in a shrunken search space. GA-NSP is a typical stochastic NSP mining method built on a genetic algorithm. GA-NSP calculates the dynamic fitness for each NSC and NSP. It selects NSPs with high dynamic fitness and conducts crossover and mutation operations on these chosen NSPs to generate NSCs, thus requiring less runtime and memory during the mining process. Suffering from the characteristics of stochastic processing, these methods cannot guarantee the coverage of the NSPs discovered, leading to the drawbacks that sometimes only a small number of NSPs are discovered, and many informative but relatively low-frequency patterns are missed. In addition, no relevant research is available to optimize the setting of method parameters based on the dataset characteristics so far. For instance, several method parameters need to be set in GA-NSP, such as crossover rate, mutation rate and decay rate, and their parameter settings may greatly influence the performance of GA-NSP.

Lastly, *PSP-based NSP mining* invents a new NSB analytics theory based on the PSP-to-NSC conversion. This category of methods derives NSCs and calculates the negative supports by only using the corresponding information of the PSPs discovered and converts negative containment to positive containment. e-NSP-based methods are the only methods proposed which are set theory-based methods that apply sequence-frequency constraint and strictly-negative containment. They generate NSCs through a negative conversion strategy and calculate the strictly-negative-support of each NSC by using the support of its maximum positive sub-sequence and 1-negative-size maximum sub-sequences. Benefiting from not testing the generated NSC by repeatedly scanning the dataset, e-NSP is highly efficient and scalable with a short runtime and small memory usage. However, e-NSP performs efficiently at the cost of maintaining a strict frequency constraint and thus only discovers a small number of NSPs; long-size or long-length patterns may be lost. Built on e-NSP, SAPNSP mines patterns through e-NSP and proposes an interestingness measure to judge whether a mined pattern is actionable (Cao, Zhao, Zhang, Luo, Zhang & Park 2010). In addition, e-msNSP (Xu, Dong, Xu & Gong 2017) extends e-NSP to discover the NSP with multiple minimum supports, where each item is associated with a minimum item support (MIS) and the minimum support threshold for a negative sequence is computed by the MIS value of items within this sequence. In e-msNSP, a negative sequence is an NSP if its support is greater than its minimum support threshold rather than a predefined global threshold as in other methods. Moreover, e-NSPFI (Gong et al. 2017) is another extension of e-NSP to discover NSP from both high-frequency and some constrained low-frequency PSP. Finally, HUSP-NIV (Xu, Dong, Xu & Dong 2017) discovers the high utility sequential patterns (HUSP) from sequential utility-based databases, also built on e-NSP.

The above categorization of existing NSP mining methods is summarized in Table 2.1 according to the above discussion and their main research ideas, advantages, and disadvantages.

2.1.2 Technical Challenges

The above analysis of related work on NSP mining methods shows: (1) pattern mining-based NSB analytics is attracting increasing interest, with more advanced theories and methods progressively being reported; (2) NSP mining is much more difficult and complex than classic PSP mining; and (3) existing work only discovers a partial coverage of high-frequency NSPs with no universal and systematic definitions and mechanisms accepted in the area. Despite

Table 2.1: Categorization Overview of Existing NSP Mining Methods

Research Category	**Main Idea**	**Advantage**	**Disadvantage**
Format-specific Mining Methods	Adopt specific constraints to define interesting NSP in special formats	Smaller space is required to search, and fewer NSCs are generated	NSP against predefined formula are missed and applications are limited
Complete Mining Methods	Adopt the NSC generation-and-testing strategy and adapt PSP mining methods to generate long NSC based on mined PSP and NSC	Maintain the maximum coverage of the discovered NSP	Resource consumption is large, and many discovered NSPs may be less to non actionable
Stochastic Mining Methods	Adopt a stochastic strategy and only generate potentially high-frequency NSCs by performing stochastic operations on the selected optimal NSP	Search space is reduced, and the average resource consumption is small	Coverage of the discovered NSPs is not guaranteed
PSP-based Mining Methods	Convert negative containment to positive containment and generate and test NSPs by only using the information of the discovered PSP	Maintain high efficiency and scalability in runtime and memory usage by avoiding rescanning dataset	The problem statement is much stricter, and only a small number of NSPs are discovered

the difficulty in understanding nonoccurring behavioral data, the main technical challenges facing pattern mining-based NSB analytics research include: inconsistencies in NSB problem formalization, violation of the downward closure property, large search space, and high computational complexity.

First, there exist serious inconsistencies in the NSB definition and formalization. In contrast to the analytics on occurring behaviors, no widely accepted problem statement, constraint settings, or formal definitions about negative containment exist in the current research on the pattern mining-based NSB analytics. This reflects the much more challenging nature of NSB. Accordingly, different methods aim to discover specific forms of NSP in respective search spaces by adopting diverse constraints. One of the reasons different methods result in divergent pattern coverage is that they adopt different definitions of negative containment. For example, PNSP adopts a stricter definition of negative containment than NegGSP. Hence, PNSP maintains a smaller pattern coverage than NegGSP, even though they mine PSPs in the same search space. In addition, NSPM defines the length of sequences as the number of elements in a sequence, while NegGSP, GA-NSP and e-NSP define the length as the total number of items in all elements of a sequence. The definition of sequence length in NSPM is actually the definition of sequence size in the latter methods. The inconsistencies and confusion require a more systematic design of NSB analytics.

Second, NSPs do not satisfy the downward closure property as PSPs do. A super-sequence of an infrequent negative sequence may be a frequent NSP, and a sub-sequence of a frequent NSP may also be an infrequent negative sequence. For example, $S_\alpha =< a, \neg b, c >$ is a super-sequence of $S_\beta =< \neg b, c >$, but S_α is contained in $S_\gamma =< b, a, c >$ while S_β is not contained in S_γ. The fact that NSP does not satisfy the downward closure property further enlarges the search space and increases computational complexity, so PSP methods cannot be directly applied or adjusted for NSP mining since downward property forms the foundation of PSP mining.

Third, pattern mining-based NSB analytics faces a large search space and high computational complexity due to the lack of downward closure property. Despite the existence of diverse definitions of negative containment, a data sequence can contain many more NSCs compared with positive candidates (Zheng 2012). The search space of NSP mining is much larger than that of PSP mining. A large number of NSCs can be generated, and computational complexity can be much higher if an NSP method aims to maintain complete pattern coverage, such as PNSP and NegGSP. For example, for a set of items

$I = \{a, b, c\}$, data sequence $S =< a, b, c >$ can contain at most $\sum_{k=1}^{3} C_3^k = 7$ positive candidates, but it can contain at most $\sum_{k=1}^{3}(C_3^k 3^{k+1}) = 2916$ NSCs. This requires strategies such as adopting reasonable constraints and effective pruning strategies to reduce the research space and improve the performance of NSP mining methods. Pruning strategies can filter some invalid NSCs and reduce the computational complexity of NSP mining. However, few efficient pruning strategies are currently available. Applying constraints leads to a trade-off between method performance and pattern coverage but also reduces the number of NSCs generated. In practice, a well-designed constraint may ensure that a sufficient number of informative NSPs can be identified in a smaller search space, while the theoretical analysis should also be provided.

As discussed above and illustrated in Table 2.1, to address the demand of and significant technical challenges in existing research, a comprehensive and systematic formalization for NSB analytics problem is required to form a general theoretical system for NSB analytics research. In addition, existing methods on pattern mining-based NSB analytics suffer from either inadequate coverage or low performance, which is discussed in detailed in Section 2.1.3, and an efficient method of complete NSP mining with a well-designed constraint which discovers the complete set of high-frequency and informative NSPs with high efficiency is of great significance.

2.1.3 Gap Analysis in NSP Mining

While nonoccurring behaviors (Cao et al. 2015) can be found in many applications, limited yet increasing research on NSP mining has been conducted recently (Cao et al. 2016, Dong, Gong & Cao 2018*b*, Dong, Gong & Cao 2018*a*). Existing NSP methods are all based on *negative element constraint (NEC)* and either are highly inefficient due to the high computational cost or only detect a small NSP coverage owing to strong constraints on NSP settings (Cao et al. 2016), making them impractical for complete NSP mining with the *loose negative element constraint* (LNEC).

The existing NSP mining methods, overviewed in Section 2.1.1, include two types of methodologies: search-based methods and set theory-based methods. Search-based methods start with an initial seed set, iteratively generate long-length or long-size NSC based on shorter ones and calculate the support of each NSC through repeatedly scanning the dataset. Typical methods include PNSP (Hsueh et al. 2008) and NegGSP (Zheng et al. 2009), which aim to discover the complete set of NSPs but are usually inefficient because they have to handle a large number of candidates and need to rescan the entire dataset multiple times. In contrast, set theory-based methods convert the negative containment problem to a positive containment problem, derive NSCs from the discovered PSPs and calculate the support of NSC by only using the information of corresponding PSPs. Typical set theory-based methods include e-NSP (Cao et al. 2016), e-msNSP (Xu, Dong, Xu & Gong 2017), e-NSPFI (Gong et al. 2017), F-NSP+ (Dong et al. 2018*b*), and e-RNSP (Dong et al. 2018*a*), and they are efficient in terms of runtime because they avoid repeatedly rescanning the dataset. However, the PSP-to-NSC generation strategy covers a small search space and leads to a limited coverage of mined patterns, plus its strict settings of constraints and negative containment prevent it from discovering NSPs with the LNEC.

In PSP mining, the vertical database format strategy demonstrates its superior performance as demonstrated in (Fournier-Viger, Gomariz, Campos & Thomas 2014, Aseervatham, Osmani & Viennet 2006, Chiu, Wu & Chen 2004, Zaki 2001). SPAM (Ayres, Flannick, Gehrke & Yiu 2002) is one of the most efficient PSP mining methods, adopting bit-wise operations upon vertical bitmap representations of candidates for efficient support calculation. However, NSPs cannot be discovered directly by SPAM (Cao et al. 2016), and its traversal-based sequence enumeration strategy generates a large number of invalid candidates, which makes SPAM-like PSP methods inapplicable for NSP mining with the LNEC.

Accordingly, a better solution for complete NSP mining with the LNEC is to combine the advantage of search-based and set theory-based methods, which generates the NSC iteratively to ensure the complete coverage of resultant NSPs and calculate the support of NSC without any dataset re-scan to achieve high efficiency. Inspired by the tremendous success of the vertical database format strategy in PSP mining, we incorporate the vertical representation and propose the VM-NSP framework in Chapter 4 to efficiently discover the complete set of NSPs with the LNEC.

2.2 Representative Subset Selection

Different from the NSP mining methods which target the discovery of all patterns with high frequency, representative NSP selection methods aim to evaluate each NSB from the perspective of its quality and diversity and then select a small-scale, discriminating yet non-redundant subset to present the whole collection of the NSPs discovered.

Compared with the limited yet increasing attention paid to NSP mining (e.g., (Lin, Chen & Hao 2007, Zheng et al. 2009, Zheng et al. 2010, Cao et al. 2016)), little research has been conducted on representative NSP selection to enhance the actionability of NSP collection. In the current literature, SAPNSP (Liu et al. 2015*a*) is the only one reported that aims to extract actionable patterns from the whole NSP collection. However, approaches like SAPNSP face great challenges in the discovery of representative NSP due to three-fold shortcomings. 1) A single contribution metric is adopted to select the patterns with high frequency and high correlation between its prefix and the last element, resulting in a highly repetitive resultant subset, which is discussed in detail in Section 2.2.4. 2) The effectiveness of the proposed metric is doubtful as it tends to overestimate the quality of short-size patterns due to the fact that the metric itself is of downward closure, while some filtered long-size patterns with high-frequency elements may likely be more informative. 3) The implicit nonoccurrences between NSPs that filter those relatively low-frequency but highly informative patterns are overlooked.

Determinantal point processes, which are elegant probabilistic models of global and negative relations to offer efficient and accurate methods for subset selection (Kulesza, Taskar et al. 2012, Kulesza & Taskar 2010), form a solid foundation for the task of representative NSP discovery. Determinantal point processes have demonstrated great success in subset selection by modeling the probability over all subsets in terms of their quality and diversity. The variants of DPP-based subset selection methods have been proposed for different applications. A natural feature-based conditional DPP is proposed in (Kulesza & Taskar 2012) for extractive summarization to choose a small subset of sentences conveying the most important information from a set of documents. In addition, a fixed-size DPP method (k-DPP) is proposed to only select a significant and diverse subset with cardinality k (Kulesza & Taskar 2011), and a refined multigraph DPP method (MDPP) is proposed for high-dimensional hyperspectral band selection to select an optimal fixed-size subset by modeling multiple relations among the entities in the collection, which is based on k-DPP to capture the underlying relations between different spectral bands (Yuan, Zheng & Lu 2016). However, neither k-DPP nor MDPP can select structural patterns. Structured DPP (SDPP) is the only DPP-based method for distributions over sets of structures (Kulesza & Taskar 2010) and is extended to discover important and diverse threads for documentation summarization (Gillenwater, Kulesza & Taskar 2012). However, SDPP can only model the quality and diversity of structures in terms of a single relation but fails to handle multiple complex relations between entities, which may cause unacceptable information loss (Yuan et al. 2016).

Preliminary to our proposed EINSP method, we introduce DPP as well as its two state-of-the-art variants, k-DPP and SDPP, and then analyze the gaps in existing representative subset selection research.

2.2.1 Determinantal Point Processes

A DPP is a distribution over the subsets of a ground set, for example, subsets of the NSP discovered from a sequence dataset. A DPP assumes the items within this ground set are negatively related, and the negative relation strengths can be derived from a kernel matrix K that quantifies the similarity between item pairs. In this way, more diverse items are likely to cooccur in the selected subset. Accordingly, DPPs assign a higher probability to the subsets composed of diverse items. For example, a DPP prefers NSP subsets that are supported by different partitions of the sequence dataset, instead of only selecting the patterns with the highest frequency.

Suppose that a discrete ground set is $\mathbb{S} = \{S_1, S_2, \ldots, S_N\}$, and a point process P on set $\mathbb{S}$ is a probability measure on $2^{\mathbb{S}}$, which is the set of all subsets of $\mathbb{S}$. The point process P is called a *DPP* if, when $\mathcal{S}$ is a subset drawn according to P, then for each $A \subseteq \mathbb{S}$ we have:

$$P(A \subseteq \mathcal{S}) = det(K_A) \tag{2.1}$$

for some positive, symmetric and semi-definite matrix $K \preceq I$ indexed by the elements of $\mathbb{S}$. Here, $K_A = [K_{ij}]_{i,j \in A}$ denotes the restriction of matrix K to the entries indexed by elements of A, and $det(K_\varnothing) = 1$. K is always referred to as the marginal kernel, as it involves all the information required to calculate the probability of any subset A being included in $\mathcal{S}$. A few simple observations follow from Eq. (2.1) as follows:

$$\begin{aligned} P(i \in \mathcal{S}) &= K_{ii} && (2.2) \\ P(i,j \in \mathcal{S}) &= K_{ii}K_{jj} - K_{ij}K_{ji} \\ &= P(i \in \mathcal{S})P(j \in \mathcal{S}) - K_{i,j}^2. && (2.3) \end{aligned}$$

We note that the diagonal of matrix K provides the marginal probabilities for individual elements of $\mathbb{S}$, and off-diagonal elements determine the negative relations between element pairs: large values of K_{ij} indicate items i and j are less likely to cooccur. To model real data, however, the most relevant construction of DPPs is not through K but via L-ensembles (Borodin & Rains 2005). An L-ensemble defines a DPP via a positive semi-definite matrix L indexed by the elements of $\mathbb{S}$ as follows:

$$P_L(\mathcal{S}) = \frac{det(L_\mathcal{S})}{det(L+I)} \tag{2.4}$$

where I is the $N \times N$ identity matrix and $P_L(\mathcal{S})$ is normalized due to the fact that

$$\sum\nolimits_{\mathcal{S} \subseteq \mathbb{S}} det(L_\mathcal{S}) = det(L+I). \tag{2.5}$$

Matrix K and L enable alternative representations for the DPP, which can be easily translated between each other via $K = (L+I)^{-1}L$ and $L = K(I-K)^{-1}$. For both K and L representations, subsets which are of higher diversity receive the higher likelihood, as measured by the respective kernel. Compared with matrix K, which gives rise to marginal probabilities of inclusion for subset $A \subseteq \mathbb{S}$, L-ensembles directly model the atomic probabilities of exactly observing each possible subset of $\mathbb{S}$, offering a convenient target for optimization. In addition, eigen-values of matrix K need to be bounded, while matrix L is only required to be positive and semi-definite. Accordingly, L-ensemble representation is more widely used in recent related work.

2.2.2 Fixed-Size Determinantal Point Processes

In some realistic subset selection issues, the size of the target subset is known or restricted in advance or adjusted on-the-fly when tested. Accordingly, the fixed-size determinantal point processes (k-DPPs), which is a distribution over all the subsets $\mathcal{S} \subseteq \mathbb{S}$ with a fixed cardinality k, is proposed to address the problem in above situations (Kulesza & Taskar 2011). Different from the standard DPP that models both content and size of $mathcalS$, which is described in Section 2.2.1, k-DPP only models the content of the selected k-size subset. Thus, it receives significant flexibility for the content modeling of subsets at the cost of failing to distinguish subsets with various sizes, which can be a practical trade-off for real-life applications.

A k-DPP is defined by conditioning a standard DPP in the event that subset $\mathcal{S}$ is of cardinality k, and the k-DPP P_L^k gives probabilities as follows:

$$P_L^k(\mathcal{S}) = \frac{det(L_{\mathcal{S}})}{\sum_{|\mathcal{S}'|=k} det(L_{\mathcal{S}'})}. \tag{2.6}$$

Here, $\mathcal{S}$ is a k-size subset, $|\mathcal{S}| = k$, and the modifications of k-DPP from the standard DPP only exist in the size restriction on subset $\mathcal{S}$ as well as the adopted normalization constant.

2.2.3 Structured Determinantal Point Processes

Structured determinantal point processes (SDPPs) are effective probabilistic models proposed for distributions over the sets of structures (Kulesza & Taskar 2010). In this setting, each element $S \in \mathbb{S}$ is a structure represented as a sequence of R parts $< S[1], S[2], \ldots, S[R] >$, and each of them takes a value from a finite set of M possibilities. Thus, the ground set is denoted as $\mathbb{S} = \{S_1, S_2, \ldots, S_N\}$, where $S_i \in \mathbb{S}$ is the i-th structured element in $\mathbb{S}$. In SDPPs, the entries of kernel L can be factorized as follows:

$$L_{ij} = q(S_i)\phi(S_i)^T\phi(S_j)q(S_j) \tag{2.7}$$

where $q(S_i)$ is a non-negative measure of the quality of the structure S_i, and $\phi(S_i)$ is a multi-dimensional diversity feature vector, where $\phi(S_i)^T\phi(S_j)$ acts as a signed measure of the similarity between structures S_i and S_j.

To enable efficient normalization and sampling, SDPP assumes a factorization of the quality score $q(S_i)$ and the similarity score $\phi(S_i)^T\phi(S_j)$ into parts over a set of factors F, and each factor $\alpha \in F$ is a small subset of the parts of a structure. If S_i^α refers to the part collection of the structure S_i included in factor α, then the factorization assumption is that the quality score can be decomposed multiplicatively while the diversity feature vector can be decomposed additively over parts:

$$\begin{aligned} q(S_i) &= \prod\nolimits_{\alpha \in F} q(S_i^\alpha) && (2.8) \\ \phi(S_i) &= \sum\nolimits_{\alpha \in F} \phi(S_i^\alpha). && (2.9) \end{aligned}$$

Accordingly, the probability of a subset $\mathcal{S} \in \mathbb{S}$, $P_L(\mathcal{S})$, can be calculated by applying the second-order message passing algorithm (Li & Eisner 2009).

2.2.4 Gap Analysis in Representative Subset Selection

To the best of our knowledge, SAPNSP (Liu et al. 2015*a*) appears to be the only work that provides a contribution metric to evaluate the quality of each pattern and selects the high-quality ones as the representatives, which, however, completely ignores the complex behavior relationships between elements and patterns (Song et al. 2012, Cao 2013, Cao 2015) and thus leads to many redundant and similar patterns being selected.

On one hand, since elements often fall into an imbalanced distribution, a group of patterns which satisfy high-metric values but are similar are likely to be discovered, leading to less informative knowledge, low diversity and coverage of the often long-tailed subsets especially for low-frequency items. Following the frequency-based selection criteria, the patterns with rarely observed items only constitute a small part of the NSP cohort, and their score is typically low, thus usually filtered out in the existing methods. However, they may represent some rare but vital behaviors and be critical in specific situations, such as suspicious health claims in fraud detection, fault maintenance in system diagnosis, and missing treatments in medical service (Cao et al. 2016). On the other hand, the NSPs selected by a specific metric tend to be short in size, making the selected subset less capable of disclosing long-range behaviors. This indicates the importance of balancing NSP quality and diversity (Kulesza & Taskar 2012, Gillenwater, Kulesza, Fox & Taskar 2014) to serve multiple purposes. One is to select high-quality patterns that carry important information consistent with the entire dataset from one to multiple perspectives; the other is to select the diverse patterns as a group that ensure non-repetitive but informative subset representatives per the size. Intuitively, pattern diversity implies a repulsive interaction and negative dependency between NSPs so that similar ones are less likely cooccur (Pemantle 2000, Borcea, Brändén & Liggett 2009). In this way, the NSPs with low frequencies but specific information are more likely to be retained.

Determinantal point processes as an efficient probabilistic model captures negative correlations for subset selection with diversity in the subset (Kulesza et al. 2012, Borodin 2009, Affandi, Fox, Adams & Taskar 2014, Błaszczyszyn & Keeler 2018, Mariet & Sra 2015). Determinantal point processes demonstrates promise in areas including video and documentation summarization (Kulesza & Taskar 2012, Gong, Chao, Grauman & Sha 2014, Hong & Nenkova 2014, Mahasseni, Lam & Todorovic 2017), information retrieval (Gillenwater et al. 2012), recommendation systems (Chen, Zhang & Zhou 2018), sequence classification (Li, Hong & Chen 2019), and image processing (Yuan et al. 2016, Kulesza & Taskar 2011). However, DPP cannot be directly applied to *representative NSP discovery* due to two-fold challenges. On one hand, NSPs are embedded with sequential structures which cannot be modeled by the item-oriented DPP methods. So far, SDPP is the only model for distributions over sets of structures (Kulesza & Taskar 2010), but it cannot be used to model the pattern quality and diversity within an NSP subset. On the other hand, the relationships between NSPs are more complicated than the dependency modeled by the existing DPP-based methods. The method in (Gillenwater et al. 2012) incorporates the cosine similarity into SDPP to model the quality of each entity and quantify the diversity between two entities (an entity can be a structure or a component within the structure), which only reveals the cooccurrence-based relations. Inspired by (Wang & Cao 2017), nonoccurrence-based implicit behavior relations between these entities capture the correlations conditioned on a third-party entity, which can be even more useful to select informative but unexpected hidden knowledge (Beg & Butt 2009, Cao 2013). For example, in healthcare analysis, assume $S = < a, \neg b, c, X >$ is an NSP, where a, b and c stand for the codes of the medical services undertaken by a patient, and X is the final disease status of this treatment solution. Pattern S indicates that a patient who undertakes medical services a and c but misses treatment b has a high probability of having disease status X, highlighting the impact of the absence of b on status X. Here, pattern S may correspond to a special treatment of a serious but low-chance disease, such as a rare cancer, where services a, b, and c are less likely to cooccur frequently, and pattern S may not be selected by existing methods due to its relatively low frequency, even though S may be extremely insightful for this specific problem. This illustrates the importance of taking the implicit behavior relations per the third-parties into account and the possibility of being more likely to select the informative yet relatively low-frequency patterns related to specific scenarios (Wang & Cao 2017). However, no existing DPP-based subset selection methods can model the pattern quality and diversity by integrating both positive (occurrence) and negative (nonoccurrence) element/pattern relations.

In summary, NSP quality enhancement and representative subset selection are open issues, and no research exists to effectively select the representative NSP subset by jointly modeling occurrences and nonoccurrences-based explicit and implicit pattern relations (Cao 2013). Thus, our EINSP method proposed in Chapter 5 is significant to address the above need and gaps.

2.3 Sequential Basket Recommendation

In contrast to the above pattern mining-based NSB analytics methods which concentrate on the discovery of the set/subset of entire NSB patterns that are of highly-global significance, the sequential basket recommendation system

(SBRS) aims to continuously make predictions for a sequence of next-basket behaviors to habitual users based on their historical sequential behaviors and interactive positive/negative feedback. Many recommender methods have been reported in recent years; however, to the best of our knowledge, most existing related methods make predictions only based on the occurring behaviors, that is, the positive observations which are sequential data collected by recording users' historical explicit behaviors (Wang, Guo, Lan, Xu, Wan & Cheng 2015), but few of them considers the hidden and implicit nonoccurring behaviors to make and refine the recommendations. While recommendation research has emerged into a big family (Adomavicius & Tuzhilin 2005, Quadrana, Cremonesi & Jannach 2018, Cao 2016), the techniques related to our research on SBRS can be broadly categorized into collaborative filtering-based recommendation, sequential recommendation such as next-basket RS (NBRS), session-based recommendation, interactive recommendation, and hybrid methods. Another relevant topic concerns involving feedback in recommendation. Below, we review them in terms of recommending next-basket behaviors, which is the focus of our research in Chapter 6, and then analyze the gaps in existing SBRS research.

2.3.1 Collaborative Filtering Methods

Collaborative filtering-based methods predict the next-basket behaviors by capturing general user preferences based on overall purchase history (Wang, Guo, Lan, Xu, Wan & Cheng 2015) but ignore any sequential behavior relations and thus fail to handle sequential behaviors. Such methods make recommendations by finding the top-k similar users or items based on either particular measures or factoring the user-item matrix (Wang, Guo, Lan, Xu, Wan & Cheng 2015), which can be either memory-based methods or model-based methods (Su & Khoshgoftaar 2009). The former methods try to discover the k nearest neighbours of items or users based on certain similarity measures to make recommendations, while the latter make predictions by factorizing the constructed user-item correlation matrix (Linden, Smith & York 2003). For example, the work in (Lee, Jun, Lee & Kim 2005) constructs a binary user-item matrix based on the user's sequential behaviors, and makes predictions by applying a PCA-based logistic regression model. The work in (Hu, Koren & Volinsky 2008) applies the least-square optimization to factorize the user-item pair and then control the significance of observations based on the pair confidence. The work in (Pan & Scholz 2009) introduces weights to missing ratings of the user-item matrix and optimizes the factorization objective to address the sparsity issue with hinge-loss and least-square criteria. Bayesian personalized ranking (BPR) is proposed as a criterion to optimize the user preference ranking over item pairs rather than the user preference score on a single item (Rendle, Freudenthaler, Gantner & Schmidt-Thieme 2009). Collaborative filtering methods can obtain the general preference of each user; however, they discard item couplings within and between baskets (Cao 2016) and the couplings between user behaviors (Cao et al. 2012) and cannot handle sequential behaviors. As a result, CF methods tend to produce similar or duplicated recommendations owing to the similarity-based rationale but cannot capture the intrinsic nature of SBRS scenarios, which makes them unable to satisfy user dynamic expectations and business benefits (Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017). Finally, CF methods can suffer from sparsity-caused problems because of the power-law distributed items in data (Wang et al. 2018, Hu, Cao, Cao, Gu, Xu & Wang 2017, Hu, Cao, Cao, Gu, Xu & Yang 2016).

2.3.2 Sequential Recommendation Methods

Sequential recommendation emerges as a recent focus in recommender systems (RSs). Existing work involves the Markov chain (MC) mechanism to explore sequential user behaviors and predict the next purchase behavior only based on the last basket (Gu, Dong & Zeng 2014, Chand, Thakkar & Ganatra 2012). For example, a Markov chain-based sequential recommendation method is proposed in (Zimdars, Chickering & Meek 2001), which applies probabilistic decision-tree models to make predictions by extracting sequential patterns. The work in (Mobasher, Dai, Luo & Nakagawa 2002) proposes applying contiguous sequential patterns instead of general sequential patterns to generate recommendations for sequential prediction tasks. In addition, a personalized sequential pattern mining-based recommender is suggested to learn user sequence importance based on competence score for personalized next-item recommendations (Yap, Li & Philip 2012). A Markov decision processes-based prediction method is proposed in (Shani, Heckerman & Brafman 2005), which is verified to be effective for next-basket recommendation. The hidden Markov model is applied in (Gu et al. 2014) for the prediction of purchase behavior sequence based on purchase intervals. Logistic Markov embedding (LME) is proposed to treat playlists as Markov chains and learn to represent the songs in the latent space for music recommendation (Chen, Moore, Turnbull & Joachims 2012). With LME, personalized Markov embedding (PME) is proposed for next-song recommendation by modeling sequential user singing behaviors in the Euclidean space (Wu, Liu, Chen, He, Lv, Cao & Hu 2013), and personalized ranking metric embedding (PRME) is proposed for the next-POI recommendation by jointly modeling user check-in sequences and individual preference (Feng, Li, Zeng, Cong, Chee & Yuan 2015). However, the SR methods only model the transitions between adjacent behaviors but fail to model the intra-basket item couplings and coupled sequential behaviors (Wang, She & Cao 2013, Wang, Guo, Lan, Xu, Wan & Cheng 2015, Cao 2016).

2.3.3 Session-based and Interactive Recommendation

Session-based recommendation, where a session can refer to a transaction with clear boundaries or a period within a designated time window, has attracted significant attention in recommendation research. Typically, neural models and attention mechanisms are applied to represent the context or session of an item or item sequence for next-item or session-based recommendations. Examples include GRU4Rec (Hidasi, Karatzoglou, Baltrunas & Tikk 2015), SWIWO (Hu, Cao, Wang, Xu, Cao & Gu 2017), NTEM (Wang, Hu & Cao 2017), NARM (Li, Ren, Chen, Ren, Lian & Ma 2017), HCA-GRU (Cui, Wu, Huang & Wang 2017), and ATEM (Wang et al. 2018). However, they cannot capture the intra-/inter-basket relations (they only consider the items within one session) or coupled sequential behaviors. In addition, none of the above methods incorporate the NOB into the representation modeling of the session or the prediction of purchase behaviors. Interactive recommendation, where the recommender interacts with users over time and updates the recommendation model for further item delivery based on the feedback collected, has attracted some attention in recent years, such as ICTRTS (Wang, Zeng, Zhou, Li, Shwartz & Grabarnik 2017), MAB (Hariri, Mobasher & Burke 2015) and ICF (Zhao, Zhang & Wang 2013). Existing interactive recommenders do not address the SBRS problem and cannot model the intra-/inter-basket behavior relations; negative feedback is also ignored.

2.3.4 Hybrid Recommendation Methods

More recent hybrid methods incorporate both general user preference and sequential behaviors into better recommendations (Quadrana et al. 2018). For example, factorized personalized Markov chains (FPMC) is proposed to bring together both the advantages of MC and matrix factorization for NBRS, which models sequential user behaviors by capturing behavior relations between the last and next baskets and general user preferences by capturing the interactions between users and items (Rendle, Freudenthaler & Schmidt-Thieme 2010). This hybrid method can achieve better performance than either the CF or SR methods. However, FPMC assumes a linear relation between items within a basket (i.e., all the items are linearly combined and independently affect a user's next purchase behavior). Such strong assumptions are inconsistent with the sophisticated interactions in real-life scenarios (Cao 2016), since multiple interacting factors may affect the next purchase behavior of a user (Yu, Liu, Wu, Wang & Tan 2016).

In recent years, several network-based models have been proposed for NBRS. Hierarchical representation model (HRM) is proposed to involve the representations of both last basket and users (Wang, Guo, Lan, Xu, Wan & Cheng 2015). However, both FPMC and HRM only consider the local sequential behaviors between the last and target baskets but discard the impact of previous baskets on prediction; they thus cannot model the item relations among baskets. Dynamic recurrent basket model (DREAM) is proposed to learn the dynamic representation of users and apply RNN to capture global sequential relations among baskets (Yu et al. 2016). Nevertheless, HRM and DREAM hold the same deficiency in that they summarize the item relevance by simple pooling operations, which neither pay attention to dominant items nor capture compound intra-basket behavior relations among items. Furthermore, the recurrent model, on which DREAM is based, suffers from the temporal dependency assumption and thus is not sufficient to capture the inter-basket item relations within a local range. It pays greater attention to recent behaviors but fails to distinguish significant behaviors (Cui et al. 2017, Liu, Wu & Wang 2017). In other words, both HRM and DREAM fail to model inter-basket behavior relations among items across different baskets.

In addition, inspired by the factorization machine (FM) mechanism which has proven its superiority in modeling sophisticated relevance, some FM-based models have been proposed in recommendation. Based on the FM mechanism (Rendle 2012), the work in (Chou, Yang, Jang & Lin 2016) proposes a pairwise factorization model for next-song prediction by learning latent vectors of user, last-time item, and item. DeepFM is proposed to combine FM with deep networks to model low-order relations by FM and high-order relations by DNN (Guo, Tang, Ye, Li & He 2017). In addition, neural factorization machine (NFM) is proposed to enhance FM by modeling higher-order and non-linear relations for sparse prediction (He & Chua 2017). Attentional factorization machine (AFM) is proposed to learn attentive relations for better prediction (Xiao, Ye, He, Zhang, Wu & Chua 2017). These methods cannot jointly model the intra-/inter-basket behavior relations among items or embed negative feedback to refine recommendations.

Lastly, user feedback has been valued in recommendation. However, most existing work related to ours typically only involves explicit feedback. For example, the work in (Wang, Guo, Lan, Xu, Wan & Cheng 2015) records user purchase behavior-based observations as sequential data. Instead, a deep reinforcement learning-based model, DEERS, is proposed in (Zhao, Zhang, Ding, Xia, Tang & Yin 2018) to automatically learn the optimal recommendation strategies by incorporating both positive and negative feedback that can continuously improve its strategies during the interactions with users. DEERS validates the importance of negative feedback in accurate recommendations. However, it only recommends a single item to a user each time (i.e., assuming there is only one item in each basket). Though it may be extendable by adjusting the Markov Decision Process for multiple items, it fails to model the intra-basket behavior relations among items in each basket.

In contrast, our HAEM model proposed in Chapter 6 is motivated to learn multi-aspect and hierarchical item and behavior relations both within and between baskets for SBRS recommendation and to integrate the FM mechanism for such relation learning. In addition, HAEM also incorporates both positive and negative feedback to refine sequential basket recommendations in an interactive manner.

2.3.5 Gap Analysis in Sequential Basket Recommendation

The aspects and characteristics of SBRS, which is discussed in detail in Section 6.1.1, challenge the existing research on RSs including collaborative filtering-based recommendation (Liu et al. 2017), context-based recommendation (Liu, Wu, Wang, Li & Wang 2016), sequential recommendation (Gu et al. 2014) such as session-based recommendation for next-item recommendation (Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017, Wang et al. 2018) and NBRSs (Rendle et al. 2010, Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016), interactive recommendation (Hariri et al. 2015, Wang, Zeng, Zhou, Li, Shwartz & Grabarnik 2017, Zhao et al. 2013), and feedback-based recommendation (Zhao, Zhang, Ding, Xia, Tang & Yin 2018). The CF-based recommendation does not fit the SBRS settings and thus cannot handle the SBRS problem in addition to facing many other issues (Cao 2016, Hu, Cao, Cao, Gu, Xu & Wang 2017, Hu et al. 2016). Context-based recommendation models the context of a recommendation target but does not necessarily fit sequential recommendation. Next-item/basket sequential recommendation approaches (e.g., by Markov chain) make predictions for the next purchase behavior only based on a first-order transition from the last basket to the current one but ignore previous baskets (Wang et al. 2018, Hu, Cao, Wang, Xu, Cao & Gu 2017). Deep network-based context-based recommendation such as HRM (Wang, Guo, Lan, Xu, Wan & Cheng 2015) and DREAM (Yu et al. 2016) only handle one specific assumption (Wang et al. 2018, Hu, Cao, Wang, Xu, Cao & Gu 2017). Session-based and next-item recommendation such as those methods in (Hidasi et al. 2015, Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017, Li et al. 2017, Cui et al. 2017) can model the session (or context) related to the sequential behaviors and be extended to NBRSs by regarding a user session as the only known purchase behavior. However, these methods suggest the next target but do not involve both intra- and inter-basket item relations, and feedback may not be involved either. Interactive recommendation, where behavior predictions are made during the interactions between an RS and users, is rarely studied (Hariri et al. 2015). Existing interactive RSs do not address the above SBRS problem but recommend only one target item instead. Feedback-based recommendation typically focuses on positive feedback namely those recommended items selected by users. Items recommended but non-selected by users (i.e., negative feedback in this research) are usually ignored in recommendations since such non-selection behaviors are less explicit than selections, and people often only pay attention to what occurs in baskets but ignore those nonoccurring items (Cao et al. 2015).

Overall, while inter-basket relations and positive feedback have been considered in existing recommendation methods (Lian, Zheng, Ge, Cao, Chen & Xie 2018, Hariri et al. 2015), both intra-basket item relations and negative feedback are often overlooked. No work has comprehensively involved both intra-/inter-basket relations and positive/negative feedback for sequential basket recommendation in an interactive manner, which motivates our research on SBRS.

2.4 Summary

Even though research on NSB analytics is attracting greater attention and several methods have already been proposed, existing work cannot address the wide NOB applications. There still exists a significant gap between the wide applications and limited research on NSB analytics, and further efforts in this field are needed.

Chapter 3

Nonoccurring Sequential Behavior Analytics Formalization

In this chapter, we first introduce the concepts of basic entities required in NSB analytics, including *item*, *element*, and *sequence* and then formalize the problem statement of NSB analytics from the perspectives of *NSP mining* and *sequential basket recommendation*. Lastly, we investigate and consolidate the definitions of constraints and negative containment, which forms the foundation of the work in Chapters 4 and 5.

3.1 Basic Concepts

The main notations in this thesis are described in Tables 3.1, 3.2 and 3.3.

Table 3.1: Main Notations in NSB Analytics (Part I)

Notation	**Description**	**Example**
I	A set of items, i.e., $I = \{i_1, i_2, \ldots, i_n\}$	$I = \{a, b, c, d, \neg a, \neg b, \neg c, \neg d\}$

Table 3.2: Main Notations in NSB Analytics (Part II)

Notation	**Description**	**Example**
i_k	The k-th item of set I, $1 \leqslant k \leqslant \|I\|$	a, $\neg a$
$i_k.state$	The state of item i_k	$a.state$ is positive, $(\neg a).state$ is negative
$RI(i_k)$	The reverse item of item i_k	$RI(a) = \neg a$, $RI(\neg a) = a$
$PI(i_k)$	The positive item partner of item i_k	$PI(a) = a$, $PI(\neg a) = a$
e	An element, i.e., a non-empty subset of I	$e = (\neg a, b, \neg c)$
E^+	The set of all positive items in I	$E^+ = \{a, b, c, d\}$
E^-	The set of all negative items in I	$E^- = \{\neg a, \neg b, \neg c, \neg d\}$
$size(e)$	The size of element e, i.e., the number of items in element e	Given $e = (\neg a, b, \neg c)$, then $size(e) = 3$
$neg\text{-}size(e)$	The negative size of element e, i.e., the number of negative items in element e	Given $e = (\neg a, b, \neg c)$, then $neg\text{-}size(e) = 2$
$RE(e)$	The reverse element of element e	Given $e = (\neg a, b, \neg c)$, then $RE(e) = (a, \neg b, c)$
$PE(e)$	The positive element partner of negative element e	Given $e = (\neg a, b, \neg c)$, then $PE(e) = (a, b, c)$
$MPE(e)$	The maximum positive element of negative element e	Given $e = (\neg a, b, \neg c)$, then $MPE(e) = (b)$
e_{ce}	A conjunction element	$e_{ce} = (\neg a \wedge b \wedge \neg c)$
e_{de}	A disjunction element	$e_{de} = (\neg a \vee b \vee \neg c)$
$e_{ce} \subseteq_{ce} e_{pos}$	Conjunction element e_{ce} is a sub-element of e_{pos}, and e_{pos} is a super-element of e_{ce}	Given $e_{ce} = (\neg a \wedge b \wedge \neg c), e_{pos} = (b, d)$, then $e_{ce} \subseteq_{ce} e_{pos}$
$e_{de} \subseteq_{de} e_{pos}$	Disjunction element e_{de} is a sub-element of e_{pos}, and e_{pos} is a super-element of e_{de}	Given $e_{de} = (\neg a \vee b \vee \neg c), e_{pos} = (a, b, d)$, then $e_{de} \subseteq_{de} e_{pos}$
S	A sequence, i.e., an ordered list of elements	$S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$
$length(S)$	The length of sequence S, i.e., the total number of items in all elements in S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $length(S) = 9$
$size(S)$	The size of sequence S, i.e., the total number of elements in S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $size(S) = 5$
$neg\text{-}size(S)$	The negative size of sequence S, i.e., the number of negative elements in S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $neg\text{-}size(S) = 3$
$S[k]$	The k-th element of sequence S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $S[3] = \neg(b, d)$
$width(S)$	The width of sequence S, i.e., the maximum size of any element in S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $width(S) = 3)$
$PS(S)$	The positive sequence partner of negative sequence S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $PS(S) =< (a, b, c), (a, c), (b, d), a, c >$

Table 3.3: Main Notations in NSB Analytics (Part III)

Notation	**Description**	**Example**
$MPS(S)$	The maximum positive sub-sequence of negative sequence S	Given $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then $MPS(S) =< b, (a, c), a >$
$prefix(S, l)$	The l-prefix of sequence S	Given $S =< a, (b, c), d >$, then $prefix(S, 2) =< a, b >$
$[P_l]_e$	The equivalent class under l-prefix P_l	$[< a, b >]_e = \{< a, (b, c), d >, < a, b, d > \}$
$S_\alpha \subseteq_{pos} S_\beta$	Positive sequence S_α is a sub-sequence of positive sequence S_β, and S_β is a super-sequence of S_β	Given $S_\alpha =< (a, c), c >, S_\beta =< (a, b, c), (a, c), (b, d), a, c >$, then $S_\alpha \subseteq_{pos} S_\beta$
D	Sequence dataset, i.e., a set of binary tuples of data sequences and their identifiers	$D = \{(S_\alpha, < (a, b), a, d, c >), (S_\beta, < (a, b, c), (a, c), (b, d), a, c >)\}$
$\|D\|$	The size of sequence dataset D, i.e., the number of tuples in D	Given $D = \{(S_\alpha, < (a, b), a, d, c >), (S_\beta, < (a, b, c), (a, c), (b, d), a, c >)\}$, then $\|D\| = 2$
$\{< S >\}$	The set of binary tuples which contain sequence S	Given $S =< (b, c), (a, c) >$, then $\{S\} = \{(S_\beta, < (a, b, c), (a, c), (b, d), a, c >)\}$
$SC(S)$	The support count of sequence S in dataset D, i.e., the size of $\{< S >\}$	Given $S =< (b, c), (a, c) >$, then $SC(S) = 2$
$sup(S)$	The support (in percentage) of sequence S in dataset D	Given $S =< (b, c), (a, c) >$, then $sup(S) = 0.5$
min_sup	The minimum support threshold	$min\_sup = 0.45$
$\mathbb{S}$	The collection of the discovered NSP	$\mathbb{S} = \{S \| sup(S) \geqslant min\_sup\}$

3.1.1 Item and Its Properties

The concept *item* is the lowest level of entity and the smallest unit in NSB analytics. Let I be a non-empty set of behavior items (i.e., $I \equiv \{i_1, i_2, \ldots, i_n\}$), where each item i_k $(1 \leqslant k \leqslant n)$ is an atomic behavior entity in a sequence, which stands for a specific event or action, and is associated with a set of attributes, such as the state of an item. The value of item i on attribute A is denoted by $i.A$. The *state* of an item i, denoted as $i.state$, can be either *positive* or *negative*. A *negative item* is represented by the symbol $\neg$ in front of its corresponding positive item. An item in a positive state is called a *positive item*, which represents the *occurrence* (or *appearance*) of the specific event or action (i.e., the behavior item); while an item in a negative state is called a *negative item*, which represents the *nonoccurrence* (or *absence*) of its corresponding positive item; for example, the negative item $\neg a$ means that its positive item a does not appear or is absent.

The *reverse item (RI)* of a behavior item i_k is defined as the corresponding item with the opposite state, denoted as $RI(i_k)$; for example, $RI(a) = \neg a$ and $RI(\neg a) = a$. The *positive item partner (PI)* of a positive item is itself, while the *positive item partner* of a negative item is defined as its reverse item, denoted as $PI(i_k)$; for example, $PI(a) = a$ and $PI(\neg a) = a$.

3.1.2 Element and Its Properties

The concept *element* is the second lowest level in the conceptual system of NSB analytics. A behavior *element* is a non-empty subset of I, denoted as $e = (x_1, x_2, \ldots, x_s)$, where $x_k \in I$ $(1 \leqslant k \leqslant s)$. For example, in the NBRS task discussed in Chapter 6, an element represents a basket of purchase items (or services) that are consumed by a user at each time point (or period). An element is a compound entity of items and contains two attributes: *state* and *size*. The *state* of an element can be also either positive or negative. A *positive element* is the element that contains positive items only, and a *negative element* is the element that includes at least one negative item, of which the negative elements only composed of negative items are called *full negative elements* while the others are called *partial negative elements*. A full negative element $(\neg x_1, \neg x_2, \ldots, \neg x_s)$ can also be represented as $\neg(x_1, x_2, \ldots, x_s)$ for short, where x_k $(1 \leqslant k \leqslant s)$ is a positive item. For instance, a full negative element $(\neg a, \neg b, \neg c)$ can be represented as $\neg(a, b, c)$. *Complete positive element* is defined as the set of all positive items in I, denoted as $E^+ = (i_1, i_2, \ldots, i_n)$, and *complete negative element* is defined as the set of all negative items in I, denoted as $E^- = (\neg i_i, \neg i_2, \ldots, \neg i_n)$ and $E^- = \neg(i_1, i_2, \ldots, i_n)$, respectively. It is self-evident that $I = E^+ \cup E^-$.

Intuitively, a partial negative element, which consists of both positive and negative items, can only contain at most one of a behavior item and its reverse item but cannot contain both of them. For example, $(\neg a, b, \neg c)$ is allowed while $(\neg a, a, \neg c)$ is invalid in an NSB. Items in an element are on the same level, and their orders are not differentiated. Without loss of generality, items in the same element are sorted in lexicographic order with positive items before negative ones in this thesis.

The *size* of an element e is the number of items in element e, denoted as $size(e)$. An element e is called an *s-size* element if $size(e) = s$. Similarly, the *negative size* of element e is the number of negative items in element e, denoted as *neg-size*(e). An element e is called an *s-neg-size* element if *neg-size*$(e) = s$. For example, given element $e = (\neg a, b, \neg c)$, then $size(e) = 3$ and *neg-size*$(e) = 2$.

The *reverse element* of element e is defined as the element which consists of the corresponding *reverse items* of all items in element e, denoted as $RE(e) \equiv \{RI(x_k)|x_k \in e, 1 \leqslant k \leqslant size(e)\}$. For example, given element $e = (\neg a, b, \neg c)$, then its reverse element is $RE(e) = (a, \neg b, c)$.

Similarly, the *positive element partner* of negative element e is defined as the positive element which consists of the corresponding *positive item partners* of all items in element e, denoted as $PE(e) \equiv \{PI(x_k)|x_k \in e, 1 \leqslant k \leqslant size(e)\}$. For example, given negative element $e = (\neg a, b, \neg c)$, then its positive element partner is $PE(e) = (a, b, c)$.

The *maximum positive element* of negative element e is defined as the positive element which consists of all positive items in element e, denoted as

$MPE(e) \equiv \{x_k | x_k \in e \wedge x_k \in E^+, 1 \leqslant k \leqslant size(e)\} = e \cap E^+$. For example, given negative element $e = (\neg a, b, \neg c)$, then its maximum positive element partner is $MPE(e) = (b)$.

The negative items in an element share certain logical relationships, which can be categorized in terms of two inferential coupling relationships between items: *conjunction coupling* and *disjunction coupling* (Cao 2012, Cao et al. 2012, Wang et al. 2013, Wang, Cao & Chi 2015). We define them below.

Coupling 1 (Conjunction Coupling) *If any one reverse item of the negative items in element e_{ce} is not allowed to occur, these negative items are coupled with a conjunction relationship. Element e_{ce} is called a conjunction element, represented as $e_{ce} = (x_1 \wedge x_2 \wedge \ldots \wedge x_s)$.*

A conjunction element e_{ce} is called a *sub-element* of positive element e_{pos} and e_{pos} is called a *super-element* of e_{ce}, denoted as $e_{ce} \subseteq_{ce} e_{pos}$, if all positive items of e_{ce} appear in e_{pos} and no reverse item of negative items in e_{ce} appears in e_{pos}.

Example 1 *A conjunction element $(\neg a \wedge b \wedge \neg c)$ is a sub-element of positive element (b, d) but not a sub-element of positive element (b, c, d) since item c occurs in (b, c, d).*

Coupling 2 (Disjunction Coupling) *If the reverse items of the negative items in element e_{de} are required not to cooccur, in other words, at least one reverse item of these negative items does not occur, these negative items are coupled by a disjunction relationship. Element e_{de} is called a disjunction element, represented as $e_{de} = (x_1 \vee x_2 \vee \ldots \vee x_s)$.*

A disjunction element e_{de} is called a *sub-element* of positive element e_{pos}, and e_{pos} is called a *super-element* of e_{de}, denoted as $e_{de} \subseteq_{de} e_{pos}$, if all positive items of e_{de} appear in e_{pos} and at least one reverse item of the negative items in e_{de} does not appear in e_{pos}.

Example 2 *A disjunction element* $(\neg a \vee b \vee \neg c)$ *is a sub-element of positive element* (b, c, d)*, since items* a *and* c *do not cooccur in* (b, c, d)*, even though item* a *occurs alone.*

The conjunction couplings between negative items in an element specify the "AND" logical relationships, and all negative conditions must be satisfied simultaneously. For example, if positive element e_{pos} is a super-element of conjunction element $e_{ce} = (\neg a \wedge b \wedge \neg c)$, it means that "$e_{pos}$ contains $\neg a$" AND "e_{pos} contains b" AND "e_{pos} contains $\neg c$", which can be also denoted in another representation as "$\neg$(e_{pos} contains a)" AND "e_{pos} contains b" AND "$\neg$(e_{pos} contains c)". As long as any one of these containment conditions is not met in positive element e'_{pos} (i.e., e'_{pos} contains a or c), the negative element $(\neg a \wedge b \wedge \neg c)$ is not a sub-element of e'_{pos}. By contrast, the disjunction couplings in an element specify the "OR" logical relationship between negative items, and at least one of these negative conditions must be satisfied. For example, if positive element e_{pos} is a super-element of disjunction element $(\neg a \vee b \vee \neg c)$, it means that "($e_{pos}$ contains b)" AND "(e_{pos} contains $\neg a$ OR e_{pos} contains $\neg c$)", which can be also denoted as "(e_{pos} contains b)" AND "$\neg$(e_{pos} contains a AND e_{pos} contains c)". Only when both of these containment conditions are not met in positive element e'_{pos} (i.e., e'_{pos} contains both a and c), the negative element $(\neg a \vee b \vee \neg c)$ is not a sub-element of e'_{pos}. If positive element e_{pos} is a super-element of negative element e_{neg}, e_{pos} is also called *containing* or *supporting* e_{neg}, denoted as $e_{neg} \subseteq e_{pos}$; otherwise $e_{neg} \nsubseteq e_{pos}$.

Because of the page limitation, we only consider the *conjunction coupling* between negative items in elements in this thesis.

3.1.3 Sequence and Its Properties

A *sequence* S is an ordered list of elements, denoted as $S =< e_1, e_2, \ldots, e_s >$, where e_j $(1 \leqslant j \leqslant s)$ is an element. An element e in a sequence is also called an *element of a sequence*, denoted as $e = (x_1, x_2, \ldots, x_m) \in S$ $(x_k \in I, 1 \leqslant k \leqslant m)$. For simplicity, if only one item is contained in an element, the brackets around it can be omitted (i.e., element $e_k = (x)$ can be also represented as $e_k = x$). In general, in NSB analytics, an item is only allowed to appear at most one time in an element, but it can occur multiple times in several divergent elements of a sequence.

A sequence contains five attributes: *state*, *length*, *size*, *width*, and *frequency*. The *state* of a sequence can be either positive or negative. A *positive sequence* is the sequence composed of only positive elements, and a *negative sequence* is the sequence containing at least one negative element. The *length* of sequence S is the total number of items in all elements in S, denoted as $length(S) = \sum_{k=1}^{s} size(e_k)$, and a sequence S is called an *l-length* or *l-item* sequence if $length(S) = l$. The *size* of sequence S is the total number of elements in S, denoted as $size(S)$, and a sequence S is called an *s-size* or *s-element* sequence if $size(S) = s$. In addition, the *negative size* of sequence S is the number of negative elements in S, denoted as $neg\text{-}size(S) = |\{e'|e' \in S \wedge e' \cap E^- \neq \varnothing\}|$. Moreover, the k-th element of sequence S is denoted as $S[k]$, and the *width* of sequence S is the maximum size of any element in S, denoted as $width(S) = \max\limits_{1 \leqslant k \leqslant size(S)} size(S[k])$. A sequence S is called a *w-width* sequence if $width(S) = w$. For example, negative sequence $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$ consists of nine items, five elements, and three negative elements, and its maximum-size element is $S[1] = (\neg a, b, \neg c)$, which is a 3-size element. Accordingly, S is a 9-length, 5-size, 3-neg-size and 3-width sequence, and $S[3] = \neg(b, d)$.

The *positive sequence partner* of negative sequence S is defined as the positive sequence which transforms all the elements in sequence S to their corresponding *positive element partners*, denoted as $PS(S) \equiv < PE(S[k])|S[k] \in S, 1 \leqslant k \leqslant size(S) >$. The *maximum positive sub-sequence* of negative sequence S is defined as the positive sequence which transforms all the elements in sequence S to their corresponding *maximum positive elements*, denoted as $MPS(S) \equiv < MPE(S[k])|S[k] \in S, 1 \leqslant k \leqslant size(S) >$. For example, given negative sequence $S =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$, then its positive sequence partner is $PS(S) =< (a, b, c), (a, c), (b, d), a, c >$ and its maximum positive sub-sequence is $MPS(S) =< b, (a, c), a >$.

In addition, given two sequences $S_\alpha =< e_1, \ldots, e_n >$ and $S_\beta =< e'_1, \ldots, e'_m >$, where $n > m$ and $length(s_\beta) = l$, then S_β is called an l-prefix of S_α, denoted as $S_\beta = prefix(S_\alpha, l)$, if $\forall i \leqslant m - 1, e_i = e'_i$, $e'_m \subseteq e_m$, and all the items $i \in (e_m - e'_m)$ are listed after those in e'_m. For example, given $S_\alpha =< a, (b, c), d >$ and $S_\beta =< a, b >$, then $S_\beta = prefix(S_\alpha, 2)$. In addition, the set of sequences sharing the same l-prefix P_l is called an equivalent class under P_l, denoted as $[P_l]_e$, that is, $\forall S_\alpha, S_\beta \in [P_l]_e$, $prefix(S_\alpha, l) = prefix(S_\beta, l) = P_l$.

Finally, positive sequence $S_\alpha =< e_{\alpha_1}, e_{\alpha_2}, \ldots, e_{\alpha_m} >$ is called a *sub-sequence* of another positive sequence $S_\beta =< e_{\beta_1}, e_{\beta_2}, \ldots, e_{\beta_n} >$, and S_β is called a *super-sequence* of S_α, denoted as $S_\alpha \subseteq_{pos} S_\beta$. If $\forall e_{\alpha_k} \in S_\alpha, 1 \leqslant k \leqslant size(S_\alpha)$; there exists $j_k, 1 \leqslant j_k \leqslant size(S_\beta)$ such that $e_{\alpha_k} \subseteq e_{\beta_{j_k}}$ and $1 \leqslant j_1 < j_2 < \ldots < j_{size(S_\alpha)} \leqslant size(S_\beta)$, which also means that S_β *contains* or *supports* S_α. For example, given positive sequences $S_\alpha =< (a, c), c >$ and $S_\beta =< (a, b, c), (a,\ c),\ (b, d), a, c >$, then $S_\alpha \subseteq_{pos} S_\beta$.

3.2 Problem Statement of NSB Analytics

Here, we formalize the problems of NSB analytics from the perspectives of *NSP mining* and *sequential basket recommendation*, which are based on the entities introduced in Section 3.1.

3.2.1 Negative Sequential Pattern Mining

A *sequence dataset* is given as a set of binary tuples, denoted as $D = \{< Sid, S >\}$, where S is a positive sequence and Sid is the *sequence identifier* of sequence S. The positive sequence in dataset D is also called a *data sequence*, and the positive element in a data sequence is called a *data element*. The *size* of sequence dataset D is the number of binary tuples in D, denoted as $|D|$. The set of binary tuples which contain sequence S is denoted as $\{< S >\}$. The *support count* of sequence S in dataset D is the size of $\{< S >\}$ (i.e., the number of data sequences in D that contain S), denoted as $SC(S) = |\{< S >\}| = |\{< Sid', S' > | < Sid', S' > \in D \land S \subseteq_{pos} S'\}|$. Moreover, the *support* of sequence S in dataset D is the percentage of its support count with respect to the size of sequence dataset, denoted as $sup(S) = \frac{SC(S)}{|D|}$.

Definition 3.1 (Positive & Negative Sequential Pattern) *Given a predefined minimum support threshold min_sup, a sequence S is called a frequent sequence or a sequential pattern if the support of S is not less than min_sup (i.e., $sup(S) \geqslant min\_sup$). S is called an infrequent sequence if its support is less than min_sup (i.e., $sup(S) < min\_sup$). If a sequential pattern is a positive sequence, it is called a positive sequential pattern (PSP); if this pattern is a negative sequence, it is called a negative sequential pattern (NSP).*

Definition 3.2 (NSP Mining) *Given a sequence dataset D and a mini-*

mum support threshold min_sup predefined by users, sequential pattern mining is a process of discovering the collection of all the frequent sequential patterns with supports not less than min_sup, denoted as $\mathbb{S} = \{S|sup(S) \geqslant min\_sup\}$*. If this mining process only focuses on the discovery of all PSP, it is called PSP mining; if the mining process aims to discover all the sequential patterns including both PSP and NSP, it is called NSP mining. Accordingly, PSP mining is a subtask of NSP mining.*

Intrinsically, NSP has different semantics from PSP. A PSP means that its elements highly likely occur sequentially, while an NSP emphasizes that its negative elements would not occur in certain situations. For example, given a PSP $S_{psp} =< a, b, d >$ and a NSP $S_{nsp} =< a, \neg c, d >$, for a data sequence $S_{data} =< e_1, e_2, \ldots, e_m >$, we can see that $S_{psp} \subseteq S_{data}$ if $\exists 1 \leqslant i_1 \leqslant i_2 \leqslant i_3 \leqslant m$ such that $a \subseteq e_{i_1}, b \subseteq e_{i_2}, d \subseteq e_{i_3}$, while $S_{nsp} \subseteq S_{data}$ only if $\exists 1 \leqslant i_1 \leqslant i_2 \leqslant m$ such that $a \subseteq e_{i_1}, d \subseteq e_{i_2}$ and $\forall i_n, (i_1 < i_n < i_2)$ such that $c \nsubseteq e_{i_n}$. For instance, suppose $S_{data} =< (a, c), (b, e), (c, e), (d, f) >$, we can see that $S_{psp} \subseteq S_{data}$ but $S_{nsp} \nsubseteq S_{data}$, because $a \subseteq (a, c)$, $d \subseteq (d, f)$ but $c \subseteq (c, e)$. The formal definition of negative containment is provided in Section 3.3.2, but as can be seen from the above example, NSP mining cannot be simply considered as a sub-field of traditional PSP mining, and the existing PSP mining methods cannot be directly applied to discover NSP due to the intricate nature of negative entities.

3.2.2 Sequential Basket Recommendation

Here, we formulate the problem of sequential basket recommendation and define the relevant concepts derived from the basic entities described in Section 3.1. We assume that there exist a number of users, each user is associated with a sequence of purchase baskets, and each basket consists of a set of items, where each item stands for a purchased item or a received service. Let $U = \{u_1, u_2, \dots, u_{|U|}\}$ be the set of users, and $I = \{i_1, i_2, \dots, i_{|I|}\}$ be the set of items, where $|U|$ and $|I|$ denote the number of users and items respectively. The sequential baskets consumed by user u are denoted as $S^u :=< B^u_1, B^u_2, \dots, B^u_{t_u} >$, where each basket B^u_t denotes the basket of items consumed by user u at time t ($t \in [1, t_u]$) and B^u_t is a non-empty subset of I (i.e., $B^u_t \subseteq I$). We further associate each item $i \in B^u_t$ with a preference state which represents the user's feedback on this item in basket B^u_t, denoted as $i.pre$, which can be *positive* if user u consumes item i; otherwise it is negative. An item i with a negative state can be also denoted as $\neg i$. For example, given an item set $I = \{tomato, pumpkin, candy, chocolate, cookie\}$ and a basket $B^u_t = (pumpkin, candy, chocolate)$, B^u_t can be rewritten as $(pumpkin, candy, chocolate, \neg tomato, \neg cookie)$ to involve the preference states of all items. It reflects that user u selects items *pumpkin*, *candy* and *chocolate* but non-selects *tomato* and *cookie* at time t.

With all consumed items and baskets by all users, denoted as $D := \{S^{u_1}, S^{u_2}, \dots, S^{u_{|U|}}\}$, the objective of *sequential basket recommendation* is to suggest the top-k items to user u in each of the next p sequential target baskets $\{B^u_{t_u+1}, \dots, B^u_{t_u+p}\}$ based on predicting a personalized ranking $<_{u,t} \subset I^2$ for user u at time t. For items $i_\alpha, i_\beta \in B^u_t$, the ranking $i_\alpha >_{u,t} i_\beta$ represents that item i_α is higher ranked than item i_β, that is, user u prefers i_α over i_β at time t.

A detailed example of interactive sequential basket recommendation will be illustrated in Section 6.1 to explore the intrinsic characteristics and complexities of the SBRS problem, which greatly challenges existing RS methods.

3.3 Constraints and Negative Containment

3.3.1 Constraint Settings

As discussed in Chapter 1, NSB analytics methods, especially the pattern mining-based approaches, involve much higher computational complexity and a much larger search space than the classic analysis of positive behaviors, and some of the discovered patterns may be meaningless; thus constraints have been set from different aspects to make a trade-off between computational efficiency and pattern coverage, making pattern mining-based NSB analytics methods less costly and more feasible (Zheng 2012). In this chapter, a *constraint* C for NSB analytics is considered as a Boolean function $C(S)$ defined on an NSB pattern S (Pei, Han & Wang 2007, Pei, Han & Wang 2002).

Element Frequency Constraint (EFC), as defined in Constraint 1, is set on the element level and is widely adopted in NSP mining methods such as PNSP, NSPM, NFSPM, and PNSPM. Element frequency constraint is introduced to remove trivial situations that occur coincidentally because sometimes only the nonoccurrence of valuable item combinations of sufficient frequency is of business interest, which reduces the search space by limiting the scale of possible elements that constitute the NSC generated.

Constraint 1 (Element Frequency Constraint (EFC)) *Negative element e cannot appear in NSC unless its positive element partner $PE(e)$ is a frequent positive element, that is, $C_{EFC}(S) \equiv (\forall e \in S, sup(PE(e)) \geqslant min\_sup)$.*

Example 3 *Negative element $(\urcorner a, b, \urcorner c)$ cannot appear in any NSC unless (a, b, c) has a support greater than the user-given threshold, even if all three items a, b, and c are frequent items.*

In addition, the *negative element constraint (NEC)*, as defined in Constraint 2, is set on the element level and prunes invalid NSCs by formulating the composition of a negative element. The NEC is widely introduced into NSP mining methods such as NegGSP, GA-NSP, e-NSP and SAPNSP, and it is adopted to reduce the search space and lower the computational complexity by avoiding handling NSCs with partial negative elements, especially for NSP mining in a dense dataset. However, such NSPs with partial negative elements may be quite informative for some applications. In Example 4, if S_α and S_β are both NSPs, then whether item a cooccurs with b results in different consequences. Existing methods with the NEC cannot capture this knowledge, and accordingly, NSP mining methods enabling a loose NEC can deliver more informative results, allowing more flexible and informative NSB patterns with complicated behavior relations to be discovered, as discussed in Chapter 4.

Constraint 2 (Negative Element Constraint (NEC)) *The smallest negative unit in an NSC is required to be an element; if an element consists of more than one item, either all or none of these items are allowed to be negative, that is,* $C_{NEC}(S) \equiv (\forall k : 1 \leqslant k \leqslant size(S),\ if\ S[k] \cap E^* \neq \varnothing,\ then\ S[k] \cap (E - E^*) = \varnothing),\ where\ E^* \in \{E^+, E^-\}.$

Example 4 *Negative sequence* $< (a, b), \neg(c, d) >$ *satisfies Constraint 2 while* $S_\alpha =< (\neg a, b), (c, \neg d) >$ *and* $S_\beta =< (a, b), (\neg c, d) >$ *does not.*

Lastly, the *Continuity Format Constraint (CFC)*, as defined in Constraint 3, specifies that adjacent negative elements are not permitted in an NSC; this is a practical format constraint introduced by most existing NSP mining methods, such as PNSP, NegGSP, GA-NSP, and e-NSP. The reasons why the CFC is widely adopted include the following: if multiple adjacent negative elements are allowed in an NSB, a potentially infinite number of NSC will be generated and tested even in a small dataset, which will lead to an extremely high computational complexity. For example, data sequence $<(a,b,c),(a)>$ can support NSC in the form of $<(a,b,c),\neg(a,c),\neg(b,d),a,\neg c>$, $<(a,b,c),\neg(a,c),\neg(a,c),\neg(b,d),a,\neg c>$ and $<(a,b,c),\neg(a,c),\ldots,\neg(a,c),\neg(b,d),a,\neg c>$. On the other hand, if adjacent negative elements exist in an NSC, their ordering is sophisticated for many applications, and it would be difficult to distinguish the correct order of those negative elements if no positive elements exist between them. For example, the NSB S_β in Example 5 specifies that neither behavior combination (a,c) nor (b,d) appears between elements (a,b,c) and a.

Constraint 3 (Continuity Format Constraint (CFC)) *Two or more continuous negative elements in an NSC are not allowed, that is,* $C_{CFC}(S) \equiv (\forall k : 1 \leqslant k \leqslant size(S)-1,\ if\ \ S[k] \cap E^- \neq \varnothing,\ then\ S[k+1] \cap E^- = \varnothing)$.

Example 5 *Negative sequence* $S_\alpha =<(a,b,c),(a,c),\neg(b,d),a,\neg c>$ *satisfies constraint 3 while* $S_\beta =<(a,b,c),\neg(a,c),\neg(b,d),a,\neg c>$ *does not.*

In Example 5, it is unclear for sequence S_β to judge whether element (a,c) does not occur before (b,d) or after (b,d). In fact, S_β may be also represented as $<(a,b,c),\neg(a,b,c,d),\ a,\neg c>$, which also means that none of a,b,c or d occurs between (a,b,c) or a. Therefore, S_β may be meaningless in some applications, and the CFC can avoid generating such invalid NSCs by limiting the NSC formulas to reduce the search space.

3.3.2 Negative Containment

Negative containment determines whether an NSB can be contained and supported by a given data sequence, and we formalize the definition of negative containment in alignment with the systematic settings adopted in this thesis.

Containment 1 (Negative Coverage) *Negative sequence* $S_{neg} =< ne_1, ne_2, \ldots, ne_{ns} >$ *is covered by data sequence* $S_{data} =< de_1, de_2, \ldots, de_{ds} >$ *if the following two conditions are satisfied:*

(1) The maximum positive sub-sequence of S_{neg} *is contained by* S_{data}*, that is,* $MPS(S_{neg}) \subseteq_{pos} S_{data}$*;*

(2) $\forall ne_k \in S_{neg}$*, where* $ne_k \subseteq E^-, 1 \leqslant k \leqslant ns$*, there exist integers p, q and r,* $(p < q < r)$ *such that* $\exists ne_{k-1} \subseteq de_p$ *and* $\exists ne_{k+1} \subseteq de_r$*, and* $\exists de_q$ *such that* $PE(ne_k) \nsubseteq de_q$*.*

Example 6 *Given negative sequence* $S_{neg} =< (\neg a, b, \neg c), (a, c), \neg(b, d), a, \neg c >$*,* S_{neg} *is covered by data sequences* $S_\alpha =< b, (a, c), c, a >$ *and* $S_\beta =< b, (a, c), c, (b, d), a >$ *but not covered by* $S_\gamma =< b, (a, c), (b, d), a >$*, since element c, which* $PE(\neg(b, d)) \nsubseteq c$*, exists between elements* (a, c) *and a in* S_α *and* S_β*.*

The concept *negative coverage* was first introduced in (Zheng et al. 2009) and also adopted in (Zheng et al. 2010) to reduce the number of NSCs generated. If data sequence S_{data} covers negative sequence S_{neg}, it is also called *base-support* S_{neg} (Zheng 2012), which is denoted as $S_{neg} \subseteq_{base} S_{data}$. The *base-support count* of negative sequence S_{neg} in sequence dataset D is the number of data sequences in which *base-support* S_{neg} in D, denoted as $SC_{base}(S_{neg}) = |\{< Sid, S > | < Sid, S > \in D \wedge S_{neg} \subseteq_{base} S\}|$. The *base-support* of S_{neg} in D is the percentage of its base-support count with respect to the size of the sequence dataset, denoted as $sup_{base}(S_{neg}) = \frac{SC_{base}(S_{neg})}{|D|}$.

As per Containment 1, it is easy to derive Corollary 3.1: MPS Corollary.

Corollary 3.1 (MPS Corollary) *Given negative sequence* S_{neg}*, if* $MPS(S_{neg})$ *is not a PSP, then* $sup_{base}(S_{neg}) < min\_sup$*.*

Proof 3.2 (Proof of Corollary 3.1) *As per Containment 1, if* $MPS(S_{neg}) \nsubseteq_{pos} S_{data}$*, then* $S_{neg} \nsubseteq_{base} S_{data}$*, therefore* $SC_{base}(S) \leqslant SC(MPS(S_{neg}))$*. Since*

$MPS(S_{neg})$ is not a PSP, $SC(MPS(S_{neg})) \leqslant min\_sup \times |D|$, hence we have $sup_{base}(S_{neg}) \leqslant \frac{SC(MPS(S_{neg}))}{|D|} < min\_sup$.

The concept *negative containment* is defined in Containment 2 to judge whether a data sequence supports (or called contains) a negative sequence.

Containment 2 (Negative Containment) *Negative sequence $S_{neg} = < ne_1, ne_2, \ldots, ne_{ns} >$ is contained by data sequence $S_{data} =< de_1, de_2, \ldots, de_{ds} >$, denoted as $S_{neg} \subseteq_{con} S_{data}$, if the following two conditions are satisfied:*

(1) The maximum positive sub-sequence of S_{neg} is contained by S_{data}, that is, $MPS(S_{neg}) \subseteq_{pos} S_{data}$;

(2) $\forall ne_k \in S_{neg}$, where $ne_k \subseteq E^-, 1 \leqslant k \leqslant ns$, there exist integers p, q and $r, (p < q < r)$ such that $\exists ne_{k-1} \subseteq de_p$ and $\exists ne_{k+1} \subseteq de_r$, and $\forall de_q$ such that $PE(ne_k) \nsubseteq de_q$.

Example 7 *In Example 6, S_{neg} is only contained by S_α but not contained by S_β and S_γ, since element (b, d) can be found between elements (a, c) and a in both S_β and S_γ.*

It is clear that if $S_{neg} \subseteq_{con} S_{data}$, then $S_{neg} \subseteq_{base} S_{data}$ and $sup_{con}(S_{neg}) \leqslant sup_{base}(S_{neg})$. In addition, Corollary 3.3 can be drawn.

Corollary 3.3 (Negative Cover Corollary) *Given negative sequence S_{neg} $=< ne_1, ne_2, \ldots, ne_{ns} >$, where $ne_1 \nsubseteq E^-$ and $ne_{ns} \nsubseteq E^-$, if $sup_{base}(S_{neg}) < min\_sup$, then for any super-sequence S'_{neg} of sequence S_{neg} (i.e., $S_{neg} \subseteq S'_{neg}$), we have $sup_{con}(S'_{neg}) < min\_sup$.*

Finally, since NSBs do not hold the downward closure property, pruning strategies are required to reduce the search space and computational complexity of pattern mining-based methods. As discussed above, if negative sequence S_{neg} has a base-support smaller than threshold *min_sup*, then none of its negative super-sequences can be an NSP. Therefore, the base-support of a negative sequence can be used to judge whether its negative super-sequences need to be further generated and tested as NSC. Accordingly, the *MPS pruning strategy* and *cover pruning strategy* are designed as follows.

Pruning Strategy 1 (MPS Pruning Strategy) *Given negative sequence S_{neg}, if $MPS(S_{neg})$ is not a PSP (i.e., $sup(MPS(S_{neg})) < min\_sup$), then S_{neg} and its negative super-sequences cannot be NSPs and thus can be pruned.*

Pruning Strategy 2 (Cover Pruning Strategy) *Given negative sequence S_{neg}, if $sup_{base}(S_{neg}) < min\_sup$, then S_{neg} and its negative super-sequences cannot be NSPs and can be pruned.*

The above pruning strategies are applied in Chapter 4 to shrink the search space of the LNEC-enabled NSP mining task and optimize the efficiency of the proposed bM-NSP method.

3.4 Summary

In this chapter, we form a comprehensive and systematic representation, formalization, and theoretical system for defining and representing the basic concepts, problems, constraints and negative containment for NSB analytics, which provide a solid foundation for our research in the subsequent chapters.

Chapter 4

Efficient Negative Sequential Pattern Mining

4.1 Introduction

4.1.1 Problem Statement

Sequential pattern mining has been widely explored in many domains (Wang, Sheng & Wu 2017, Wang, Kam, Xiao, Bowen & Chaovalitwongse 2016). By contrast, NSPs are the patterns consisting of both occurring and nonoccurring elements in sequential data. Negative sequential pattern mining discovers high-frequency positive and negative sequences or sub-sequences in a sequential dataset. As discussed in Chapter 1, NSP play an irreplaceable role in comprehensively understanding and analyzing the problems with missing or nonoccurring behaviors (Cao et al. 2015, Cao, Zhao & Zhang 2008*b*), compared with a traditional focus on pure PSP mining (Anwar et al. 2010, Cao et al. 2016). Typical NSP problems include undeclared activities in tax and social welfare claims (Zhao, Zhang, Wu, Pei, Cao, Zhang & Bohlscheid 2009), health insurance claims (Cao et al. 2016), missing medical treatments (Gong et al. 2015), and business analysis (Zheng 2012), by considering both occurring and nonoccurring activities (Anwar et al. 2010). Negative sequential pattern is more actionable in many applications such as medical treatment detection, over-service detection (Zheng 2012), fraud detection, debt detection, business decision-making analysis, counter-terrorism, security, and risk management (Cao et al. 2016, Zhao, Zhang, Wu, Pei, Cao, Zhang & Bohlscheid 2009).

However, NSP mining is fundamentally different from PSP mining due to its inclusion of nonoccurring items and elements, leading to significant challenges including problem formalization, computational costs, and various settings (Cao et al. 2016, Dong et al. 2018*b*). Owing to the intrinsic complexities of NSP mining, including the *hidden nature of nonoccurring behaviors*, *high computational complexity*, and *large NSC search space*, PSP mining methods cannot be directly applied or adjusted to discover NSPs (Cao et al. 2016). Because of the hidden nature of negative items, the downward property, which forms the foundation of PSP mining, does not hold in NSP mining. In addition, it is far more challenging to discover NSPs even in a medium-sized dataset with a not-too-low support threshold because its search space is much larger and its computational complexity is significantly higher than PSP mining. Finally, because NSC does not satisfy the Apriori principle (Zheng et al. 2009), few pruning strategies are available to accelerate NSP mining. As a result, few methods have been proposed in NSP mining, and there is a significant gap between the wide applications and limited research on NSP mining.

To tackle the above challenges, NSP mining conventionally incorporates various constraints to control the problem complexities (Liu, Dong, Li &

Li 2015*b*, Dong et al. 2014, Kamepalli & Kurra 2014, Li et al. 2010). However, this results in limited coverage of the resultant NSPs and missing valuable patterns. It is essential yet challenging to develop innovative and efficient mining frameworks and methods to discover NSPs with loose constraints for complete NSP mining especially in large-scale data (Cao et al. 2016).

One fundamental constraint incorporated in all existing NSP mining methods is the *negative element constraint (NEC)*, which requires either all or none of the items in an element to be negative if the element consists of multiple items (i.e., the smallest negative unit in an NSP is required to be an element). This substantially reduces the search space and computational complexity by reducing the number of NSCs generated. For instance, NSB $< (a, b), \neg(c, d) >$ satisfies the NEC while $< (\neg a, b), (c, \neg d) >$ does not. The NEC brings a much smaller search coverage and thus downgrades the computational challenges in NSP mining; however, it fails to discover those patterns containing negative elements that consist of both positive and negative items (i.e., the NSP with *partial negative elements*). However, in some cases, NSPs with *partial negative elements* are important or even more valuable to capture actionable behavior knowledge for addressing specific business problems. By taking the *partial negative elements* into account, NSPs are also more informative in disclosing the associated relations between nonoccurring and occurring behavior and capture more comprehensive information to support business decision-making.

Let us illustrate the NSP and NEC problem with some examples. In e-commerce recommendation, given two NSPs $S_1 =< (a, b), (c, \neg d) >$ and $S_2 =< (b,\ \neg a),\ (d, \neg c) >$, we can get to know that: 1) for a customer who has purchased b, c should be recommended if a was bought together with b; otherwise d can be recommended; and 2) d or c may not be usually purchased together given that b is bought ahead, and the recommendation can be made by validating the cooccurrence of a with b. Such knowledge can not only contribute to a higher profit by providing accurate recommendation, but also improve user satisfaction by avoiding irrelevant items. Moreover, in insurance claim analysis, suppose a, b, c and d stand for claim item codes, and $S_{pos} =< a, (b, c), d >$ is a claim sequence; if $S_{neg} =< a, (b, \neg c), d >$ is an NSP and $sup(S_{pos})/sup(S_{neg}) < min\_ratio$, then S_{pos} is likely fraudulent, since code c should not be claimed with code b when it appears between a and d in a claim. Here, $sup(S_{pos})$ and $sup(S_{neg})$ are the corresponding supports, and min_ratio is a predefined threshold. We cannot conclude that S_{pos} is fraudulent only because it is not a high-frequency PSP. It may be a rare or new but reasonable treatment, and its low frequency does not necessarily indicate that it is fraudulent. However, when we consider the NSP S_{neg}, it is clear that most of the customers will not simultaneously conduct treatments b and c between a and d, which is likely to be fraudulent.

Similar examples can be also found in other applications, and the nonoccurring items included in *partial negative elements* of an NSB, which stand for the absence of specific actions or events, can help better explain the occurrences of user-concerned or undesirable situations. While the NSP with such actions are usually overlooked by existing NSP mining methods, they could be important.

4.1.2 Design and Contributions

In this chapter, we incorporate a *loose negative element constraint (LNEC)* into NSP mining to resolve the above issues by discovering the flexible NSP containing *partial negative elements*. The *LNEC* specifies that elements in an NSB are allowed to contain both positive and negative items but cannot contain both an item and its reverse item. For example, $< (b, \neg a), (c, \neg d) >$ satisfies the LNEC while $< (a, b, \neg a), (c, d, \neg d) >$ fails. The LNEC substantially addresses an existing NSP limit, which is important for NSPs to handle real-life scenarios such as fraud detection, debt detection and risk management, which cannot be resolved with the NEC.

However, NSP mining with the LNEC is significantly more challenging since it triggers a larger search space and generates higher computational cost. It generates a much larger number of NSCs, and the runtime consumed in processing the expanded NSC surges rapidly. Considering a dataset containing n items, the number of potential elements with NEC is $2^{n+1} - 2$ while those with LNEC is 3^n. For example, when $n = 3$, there are 14 potential elements with the NEC but 27 elements with the LNEC; if the maximum size of the NSC is 3, then the number of NSCs with the NEC is only $2,954$ while those with the LNEC are up to $20,439$. Existing methods cannot be simply extended to address this issue because they are either too inefficient or too specific to afford such a heavy computational burden.

To the best of our knowledge, this is the first work to address this fundamental yet challenging problem in complete NSP mining with the LNEC, which targets the discovery of all the high-frequency patterns with the LNEC. A novel and efficient vertical mining framework, VM-NSP, introduces a vertical representation (VR) for each NSB to efficiently mine the complete set of NSPs. An efficient bitmap-based NSP mining method, bM-NSP, further implements the VM-NSP framework, which adopts a bitmap hash table (BHT) with respect to a vertical representation of each NSB to enable efficient NSC support calculation without a dataset re-scan and adopts a prefix-based NSC generation strategy to reduce the number of candidates.

The main contributions of this work are as follows:

- The existing strict NEC-constrained NSP mining methods are substantially expanded with the LNEC to reduce the NEC limitations. We formulate the problem of LNEC-enabled NSP mining and incorporate the necessary constraint settings and negative containment to make the problem resolvable.

- VM-NSP is a vertical mining framework to enable the efficient discovery of the complete set of NSPs with partial negative elements, which cannot be addressed by existing methods.

- Accordingly, an efficient bitmap-based NSP mining method, bM-NSP, has a prefix-based NSC generation method to further optimize the efficiency of NSP mining. Theoretical analysis confirms its performance superiority as expected on datasets with different data characteristics.

Experimental analysis demonstrate that compared with the two existing NSP methods modified to support LNEC, bM-NSP has significantly better efficiency and scalability on both synthetic and real-life datasets in terms of various data characteristics.

4.2 Loose Constraint-enabled Constraints and Containment

Considering the large search space and high computational complexity of NSP mining, similar to (Hsueh et al. 2008, Zheng 2012, Cao et al. 2016), LNEC-enabled constraints are incorporated as follows for a trade-off between pattern coverage and computational efficiency to discover significant NSPs at an affordable scale, which is extended from those formalized in Section 3.3 to make the LNEC-enabled NSP mining problem solvable.

The *loose negative element constraint (LNEC)*, as defined in Constraint 4, is a relaxed version of the *negative element constraint (NEC)* defined in Constraint 2. The NEC has been widely introduced into existing NSP mining methods (Zheng et al. 2009, Zheng et al. 2010, Cao et al. 2016, Liu et al. 2015*a*) to reduce the search space and lower computational complexity by avoiding handling the NSC with partial negative elements. However, as discussed in Section 4.1, the patterns pruned by the NEC may be informative in realty, and hence the LNEC is incorporated to relax the NEC. The LNEC is reasonable because an element containing both an item and its reverse item would indicate that it both occurs as well as does not occur in the element simultaneously, such as e_γ in Example 8, which does not make sense. As a result, the LNEC is the loosest constraint empowered on the format of elements which guarantees sufficient coverage of the NSPs discovered.

Constraint 4 (Loose Negative Element Constraint (LNEC)) *If a negative element consists of both positive and negative items, it can contain an item or its reverse item but not both, that is,* $C_{LNEC}(S) \equiv (\forall k : 1 \leqslant k \leqslant size(S),\ if\ (S[k] \cap E^+ \neq \varnothing) \wedge (S[k] \cap E^- \neq \varnothing),\ then\ size(S[k]) = size(PE(S[k])))$*, where* $E^* \in \{E^+, E^-\}$*.*

Example 8 *Negative elements* $(\neg a, b, \neg c)$ *and* $\neg(c, d)$ *satisfy the LNEC while* $(\neg a, a, \neg c)$ *violates it.*

The *full continuity format constraint (FCFC)*, as defined in Constraint 5, is introduced as an LNEC-enabled variant of the *continuity format constraint (CFC)* defined in Constraint 3. Both the *LNEC* and *FCFC* help shrink the search space by avoiding the exploration of meaningless candidates.

Constraint 5 (Full Continuity Format Constraint (FCFC)) *A full negative element cannot be adjacent to other negative elements, that is,* $C_{FCFC}(S) \equiv$

$(\forall k : 1 \leqslant k \leqslant size(S) - 1,\ if\ S[k] \subseteq E^-,\ then\ S[k+1] \cap E^+ \neq \varnothing)$.

Example 9 *Negative sequence* $<(a, \neg b), (e, f), \neg(b,\ d)>$ *satisfies FCFC while* $<(a, \neg b), \neg(e, f), (b, d)>$ *denies.*

Accordingly, *LNEC-enabled negative containment* is defined below to determine whether a negative sequence is supported by a data sequence.

Containment 3 (LNEC-enabled Negative Containment) *Negative sequence* $S_{neg} =< ne_1, ne_2, \ldots, ne_{ns} >$ *is contained (or supported) by data sequence* $S_{data} =< de_1, de_2, \ldots, de_{ds} >$*, denoted as* $sup(S_{neg}, S_{data}) = 1$*, if the following conditions are satisfied:*

(1) The maximum positive sub-sequence of S_{neg} *is contained by* S_{data}*, that is,* $MPS(S_{neg}) \subseteq_{pos} S_{data}$*;*

(2) $\forall ne_k \in S_{neg}$*, where* $ne_k \cap E^- \neq \varnothing$*, there exist integers p, q and* $r, (p < q < r)$ *such that* $\exists ne_{k-1} \subseteq de_p$*,* $\exists ne_{k+1} \subseteq de_r$ *and* $\exists MPE(ne_k) \subseteq de_q$*, and* $\forall de_q$ *such that* $ne_k \nsubseteq de_q$*.*

Example 10 *Given negative sequence* $S_{neg} =< b,\ (a, c, \neg b), e >$ *and three data sequences* $S_{da} =< b, (a, c, d), e >$*,* $S_{db} =< b,\ (a, d), e >$ *and* $S_{dc} =< b, (a, c), f, (a, b, c), e >$*, then* $S_{neg} \subseteq S_{da}$*,* $S_{neg} \nsubseteq S_{db}$ *and* $S_{neg} \nsubseteq S_{dc}$*.*

The adopted LNEC is a loose version of the NEC in which all the negative elements in any NSB are full negative elements consisting of only negative items. As the *LNEC* and *FCFC* degenerate to the *NEC* and *CFC* respectively, *LNEC-enabled negative containment* degenerates to Containment 2 defined in Section 3.3, which is adopted by PNSP, NegGSP and GA-NSP. In other words, the existing NSP mining problem is a special and simpler case of the *LNEC-enabled NSP mining* addressed in this chapter, and accordingly, their pattern coverage is just a subset of that of our LNEC-enabled NSP.

In order to shrink the search space and improve efficiency, the *LNEC-enabled negative coverage* is incorporated as follows.

Containment 4 (LNEC-enabled Negative Coverage) *Negative sequence $S_{neg} =< ne_1, ne_2, \ldots, ne_{ns} >$ is covered by data sequence $S_{data} =< de_1, de_2, \ldots, de_{ds} >$, denoted as $sup_{base}(S_{neg}, S_{data}) = 1$, if the following two conditions are satisfied:*

(1) The maximum positive sub-sequence of S_{neg} is contained by S_{data}, that is, $MPS(S_{neg}) \subseteq_{pos} S_{data}$;

(2) $\forall ne_k \in S_{neg}$, where $ne_k \cap E^- \neq \varnothing$, there exist integers p, q and $r, (p < q < r)$ such that $\exists ne_{k-1} \subseteq de_p$ and $\exists ne_{k+1} \subseteq de_r$, and $\exists de_q$ such that $MPE(ne_k) \subseteq de_q$ and $ne_k \subseteq de_q$.

The *base-support* of negative sequence S_{neg} in dataset D is the number of data sequences covering S_{neg} in D with respect to the size of D, denoted as $sup_{base}(S_{neg})$. If S_{neg} has a base-support lower than min_sup, then all its super-sequences S'_{neg} have a support lower than min_sup and thus are not NSPs (i.e., $sup(S'_{neg}) \leqslant sup_{base}(S_{neg}) < min\_sup$), which works as an efficient pruning strategy to reduce the number of NSCs generated. It is noted that as the *LNEC* and *FCFC* degenerate to the *NEC* and *CFC*, *LNEC-enabled negative coverage* degenerates to Containment 1 defined in Section 3.3.

4.3 The VM-NSP Framework and bM-NSP Method

In this section, we present the VM-NSP framework and the details of the proposed bM-NSP method as an instantiation of VM-NSP.

As discussed in Section 4.1, the LNEC enlarges the search space of com-

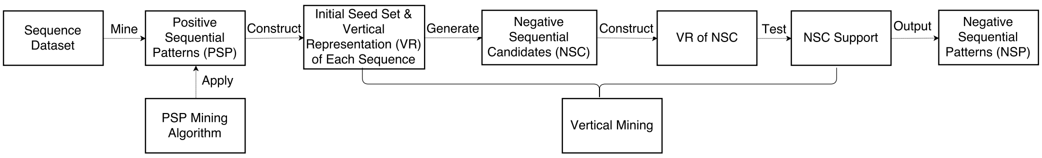

Sequence Dataset
Mine
Positive Sequential Patterns (PSP)
Construct
Initial Seed Set & Vertical Representation (VR) of Each Sequence
Generate
Negative Sequential Candidates (NSC)
Construct
VR of NSC
Test
NSC Support
Output
Negative Sequential Patterns (NSP)
Apply
PSP Mining Algorithm
Vertical Mining

Figure 4.1: Framework of Vertical Mining of Negative Sequential Patterns: VM-NSP

plete NSP mining and increases computational complexity rapidly. As demonstrated by existing research (Cao et al. 2016), the execution time of NSP mining is mainly consumed by the calculation of the support count of NSCs generated, and thus an NSC testing strategy without dataset re-scan can optimize the efficiency of NSP discovery. Inspired by the vertical representation-based strategy in PSP methods such as SPADE and SPAM, below we propose the VM-NSP framework. As illustrated in Figure 4.1: VM-NSP first applies a classic PSP mining algorithm to discover the set of high-frequency PSP on the sequence dataset and then derives an initial seed set and constructs a vertical representation (VR) for each sequence. Next, it generates long NSCs based on the current seed set and constructs their VR, from which NSC supports are tested and NSPs are discovered. As illustrated in Figure 4.1, to instantiate the VM-NSP framework, the following components are required:

(1) *Initial Seed Set Construction* to determine how to derive the initial NSC from the PSPs discovered and how to select negative seeds and suitable PSPs to construct an initial seed set, which is discussed in Section 4.3.1.

(2) *NSC generation* to determine how to generate a set of longer NSCs based on the PSPs and negative seeds of the current seed set, which is discussed in Section 4.3.2.

(3) *Vertical representation construction* to define how to construct the VR

for each sequence based on a dataset scan or the VR of relevant sequences, which will be discussed in Part 4.3.3.

(4) *NSC testing* to specify how to calculate the support and base-support of each NSC based on its VR and then determine whether this NSC is an NSP or a negative seed, which is discussed in Section 4.3.4.

Based on the working mechanism illustrated in Figure 4.1, the proposed bM-NSP method is introduced below.

4.3.1 Initial Seed Set Construction

bM-NSP adopts an item-based *initial seed set construction* which transforms all the discovered 1-length PSPs to their corresponding 1-length NSCs and selects the NSCs with high base-support and all the 1-length PSPs to construct the initial seed set. The initial seed set constructed is in an equivalent class $[\varnothing]_e$ since all the sequences in the initial seed set share a prefix $< \varnothing >$.

4.3.2 NSC Generation

bM-NSP adopts a prefix-based length-by-length strategy of *NSC generation*. In each iteration, the equivalent classes of the current seed set are chosen successively, and for equivalent class $[P_l]_e$, each (l+1)-length NSC S_{nsc} is selected and joined with another (l+1)-length sequence S_{joint} from $[P_l]_e$ to generate a new (l+2)-length NSC. Suppose the sequence of $[P_l]_e$ is denoted as $< P_l, (i_{l+1}) >$ where (i_{l+1}) is a separate element, or $< P_l, [i_{l+1}] >$ where i_{l+1} is the last item of the last element. For instance, given equivalent class $[<\neg a, b>]_e$, sequence $<\neg a, b, c>$ is denoted as $< P_2,\ (c) >$, and $<\neg a, (b, c)>$ is denoted as $< P_2,\ [c] >$ where $P_2 =<\neg a, c>$. Referring to the formats of

$S_{nsc} =< P_l, \alpha >$ and $S_{joint} =< P_l, \beta >$, the following situations of joining operations can be adopted:

(1) If both α and β are the items of the last element (i.e., $S_{nsc} =< P_l, [\alpha] >$ and $S_{joint} =< P_l, [\beta] >$), and element (α, β) satisfies the LNEC, then an (l+2)-length NSC $S_{res} =< P_l, [\alpha, \beta] >$ is generated, where (α, β) is a sub-element of the last element of NSC S_{res}.

(2) If either α or β is an item inside the last element and another item is a separate element (e.g., $S_{nsc} =< P_l, [\alpha] >$ and $S_{joint} =< P_l, (\beta) >$), then an (l+2)-length NSC $S_{res} =< P_l, [\alpha], (\beta) >$ is generated where (α) is a sub-element of the penultimate element, and (β) is the last element of NSC S_{res}, and vice versa.

(3) If both α and β are separate elements and at least one of them is a positive element, then three (l+2)-length NSCs are generated: $< P_l, (\alpha, \beta) >$, $< P_l, (\alpha), (\beta) >$ and $< P_l, (\beta), (\alpha) >$. In particular, if α equals β, only one (l+2)-length NSC is generated in the format of $< P_l, (\alpha), (\alpha) >$.

(4) If both α and β are separate negative elements, then a (l+2)-length NSC is generated in the format of $S_{res} =< P_l, (\alpha, \beta) >$, where (α, β) is the last element of NSC S_{res}.

The NSCs generated may include some invalid candidates, hence the *MPS pruning strategy* specified in Section 3.3 is applied to prune all the negative candidates whose MPS are not high-frequency PSPs (Zheng et al. 2009). After all the equivalent classes of current seed sets are processed, the set of generated (l+2)-length NSCs are partitioned into several equivalent classes, each of which constitutes a new seed set with the corresponding PSP. It can be proved that all the potential NSPs can be traversed and generated by the above NSC generation based on the above initial seed set. Because given an NSC $S_{res} =< P_l, \alpha, \beta >$ and a data sequence S_{data}, if $sup_{base}(S_{res}, S_{data}) = 1$, then $sup_{base}(< P_l, \alpha >, S_{data}) = 1$ and $sup_{base}(< P_l, \beta >, S_{data}) = 1$ always hold (i.e., $< P_l, \alpha >$ and $< P_l, \alpha >$ exist in current seed set). In this research, a depth-first strategy is adopted to improve space efficiency, which processes all the child equivalent classes along a path before handling prior to moving to another path among a seed set (Zaki 2001).

4.3.3 Vertical Representation Construction

bM-NSP incorporates a bitmap hash table (BHT) to instantiate the VR of each sequence, which associates a hash table of bitmaps for each sequence. Each hash entry adopts a sequence identifier Sid as a key and a list of bitmaps as its value. The size of each BHT equals the *size* of sequence dataset D. A bitmap in the BHT of sequence s corresponds to one of its appearances in a data sequence S_d of the dataset, denoted as $bm(s, S_d)$, and its length is equal to the size of sequence S_d, of which the value at l-th bit is set to *1* if s is supported by the l-prefix of S_d and its last element is contained by the l-th element of S_d and set to *0* otherwise. In this thesis, the BHT of sequence s is denoted as $BHT(s)$, and its hash entry corresponding to S_d is denoted as $entry(s, S_d) = [Sid(S_d) : \vec{bm}(s, S_d)]$.

The BHTs of 1-length PSP S_{psp} and the corresponding NSC S_{nsc} are constructed as follows: $BHT(S_{psp})$ is constructed by scanning dataset D, and if data sequence S_d in D supports S_{psp}, a hash entry with a key of $Sid(S_{psp})$ is inserted into $BHT(S_{psp})$. If the positive element of S_{psp} is supported by an element of S_d, the corresponding bit of only bitmap $bm \in \vec{bm}(S_{psp}, S_d)$ is set as *1*, otherwise this bit is set as *0*. The $BHT(S_{nsc})$ is constructed based on $BHT(PS(S_{nsc}))$, and hash entry $entry(S_{nsc}, S_d) = [Sid(S_{data}) : \vec{bm}(S_{nsc}, S_d)]$

Table 4.1: Toy Example

ID	**Data Sequence**
1	$< (a, b, c), (a, c), a >$
2	$< (a, c, d), (a, b, d) >$
3	$< (e, f), (a, b, c), d >$
4	$< (a, c), d, (c, f) >$
5	$< (a, c, e), (b, d), a >$

Table 4.2: The Constructed BHTs of the 1-length Sequences in Table 4.1

ID	$< a >$	$< b >$	$<\neg a >$	$<\neg b >$
s_1	{111}	{100}	{∅}	{011}
s_2	{11}	{01}	{∅}	{10}
s_3	{010}	{010}	{101}	{101}
s_4	{100}	{∅}	{011}	{111}
s_5	{101}	{010}	{010}	{101}

is inserted into $BHT(S_{nsc})$ if (1) $entry(S_{psp}, S_d)$ exists in $BHT(S_{psp})$ and it has at least one bit as *0*, where $\vec{bm}(S_{nsc}, S_d)$ is generated by a bit-wise reverse operation of $\vec{bm}(S_{psp}, S_d)$; or (2) $entry(S_{psp}, S_d)$ is not in $BHT(S_{psp})$, where all bits of only bitmap $bm \in \vec{bm}(S_{nsc}, S_d)$ are set as *1*. The size of a PSP's BHT is equal to its support count, while the size of an NSC's BHT is equal to its base-support count. Let us consider the toy example shown in Table 4.1, and BHTs of 1-length sequences $< a >$, $< b >$, $<\neg a >$ and $<\neg b >$ in Table 4.1 is illustrated in Table 4.2.

An (l+1)-length NSC S_{nsc} is generated by appending a new item i_a to the end of an l-length seed sequence $S_{seed} =< P, Q >$, and $BHT(S_{nsc})$ is generated based on $BHT(S_{seed})$ and $BHT(< i_a >)$, where Q is the last element of S_{seed}. A transformed bitmap list of S_{nsc} in data sequence S_d, $\vec{bm}^t(S_{nsc}, S_d)$, is incorporated to indicate which elements of S_d can contain the appended item of S_{nsc}. We denote the index of the first bit which is set as *1* in a bitmap bm as $b_1(bm)$, and the index of the first bit which is set as *0* behind $b_1(bm)$ in bm as $b_0(bm)$; for example, if $bm = 0111001$, then $b_1(bm) = 2$ and $b_0(bm) = 5$. We denote the set of bits in bm set as *1* as $\vec{b_1}(bm)$, and denote the value of bit b in bm as $b(bm)$.

Given a 1-length seed S_{is} from the initial seed set, the transformed bitmaps in $\vec{bm}^t(S_{is}, S_d)$ are generated from each bitmap $bm \in \vec{bm}(S_{is}, S_d)$ as follows: (1) if S_{is} is a PSP, a list of transformed bitmaps are generated by setting the bits behind b' as *1* and other bits as *0* for each bit $b' \in b_1(\vec{bm})$; for example, $\vec{bm}(a, S_1)$ in Table 4.2 is $\{111\}$ and thus $\vec{bm}^t(a, S_1) = \{011, 001\}$; and (2) if S_{is} is a negative seed, a transformed bitmap is generated by setting the bits between $b_1(bm)$ and $b_0(bm)$ as *1* and other bits as *0*; for example, $\vec{bm}(\neg a, S_3)$ in Table 4.2 is $\{101\}$ and thus $\vec{bm}^t(\neg a, S_1) = \{110\}$.

According to the situations of the NSC S_{nsc} generated, its $BHT(S_{nsc})$ is generated as follows:

(1) If item i_a is appended to the last element Q of S_{seed} (i.e., $S_{nsc} =< P, (Q, i_a) >$), then $\vec{bm}(S_{nsc}, S_d) = \vec{bm}(S_{seed}, S_d) \otimes \vec{bm}(< i_a >, S_d)$, where operation $\otimes$ means $\vec{bm}(S_{nsc}, S_d) = \{bm | bm = bm_{seed} \& bm_i, \forall bm_{seed} \in \vec{bm}(S_{seed}, S_d) \wedge bm_i \in \vec{bm}(< i_a >, S_d)\}$. For each $bm \in \vec{bm}(S_{nsc}, S_d)$, its transformed bitmap $bm^t \in bm^t(S_{nsc}, S_d)$ is generated as follows:

   (i) If Q is a *full negative element* and i_a is a negative item, bm^t is generated by setting the bits between $b_1(bm)$ and $b_0(bm)$ as *1* and other bits as *0*;

(ii) If Q is a positive or *partial negative element* and i_a is a negative item, $bm^t = bm \ominus \vec{bm}(< P, MPE(Q, i_a) >, S_d)$. The operation sets the bits between $b_1(bm) + 1$ and b' as *1* and other bits as *0*, where b' is the first bit for which $b'(bm) = 0$, and $b'(bm') = 1, bm' \in \vec{bm}(< P, MPE(Q, i_a) >, S_d)$.

(2) If item i_a is appended as a separate element of S_{seed} (i.e., $S_{nsc} =< P, Q, (i_a) >$), then $\vec{bm}(S_{nsc}, S_d) = \vec{bm}^t(S_{seed}, S_d) \otimes \vec{bm}(< i_a >, S_d)$. For each $bm \in \vec{bm}(S_{nsc}, S_d)$, its transformed bitmap $bm^t \in bm^t(S_{nsc}, S_d)$ is generated as follows:

(i) If item i_a is a negative one, then bm^t is generated by setting the bits between $b_1(bm)$ and $b_0(bm)$ as *1* and other bits as *0*;

(ii) If i_a is a positive item, then each bm^t is generated by setting the bits behind $b' + 1$ as *1* and other bits as *0* for each bit $b' \in b_1(\vec{bm})$.

4.3.4 NSC Support Testing

bM-NSP calculates the support count and base-support count of a generated NSC S_{nsc} based on its $BHT(S_{nsc})$: $\forall$ $entry(S_{nsc}, S_d) \in BHT(S_{nsc})$, if $\exists$ $bm^t(S_{nsc}, S_d) \in entry(S_{nsc}, S_d)$ ends with *1*, then S_{neg} is supported by S_d (i.e., $sup(S_{nsc}, S_d) = 1$), otherwise $sup(S_{nsc}, S_d) = 0$. In addition, if $\exists$ $entry(S_{nsc}, S_d) \in BHT(S_{nsc})$, then S_{neg} is covered by S_d (i.e., $sup_{base}(S_{nsc}, S_d) = 1$), otherwise $sup_{base}(S_{nsc}, S_d) = 0$. Therefore, for each pruned S_{nsc}, if $\sum_{S_d \in D} sup(S_{nsc}, S_d) \geq min\_sup \times |D|$, then it is an NSP; if $|BHT(S_{nsc})| \geq min\_sup \times |D|$, then it is a negative seed for further NSC generation in the subsequent iterations; otherwise, it is pruned since its super-sequences cannot be a high-frequency NSP, as per the *cover pruning strategy* specified in Section 3.3.

In summary, the pseudo code of the proposed bM-NSP method is presented in Algorithm 4.1.

Algorithm 4.1 The Pseudo-code of bM-NSP method

```
Input: Sequence dataset $D$, threshold $min\_sup$.
Output: $NSP$
$PSP \leftarrow minePSP()$
$SeedSet \leftarrow Initial\_Seed\_Set\_Construction(PSP)$
$SeedSet.VR\_Construction()$
$SeedSetStack.push(SeedSet)$
while $SeedSetStack.isNotEmpty()$ do
    $NSC\_Set \leftarrow NSC\_Generation(SeedSetStack.pop())$
    $NSC\_Set.VR\_Construction()$
    if $NSC\_Set.isNotEmpty()$ then
        for each $NSC$ in $NSC\_Set$ do
            if $NSC.BaseSup \geq min\_sup \times |D|$ then
                $SeedSet.add(NSC)$
            else if $NSC.Sup \geq min\_sup \times |D|$ then
                $NSP.add(NSC)$
            end if
        end for
    end if
    $SeedSetStack.push(SeedSet)$
end while
```

4.4 Theoretical Analysis

Here, we provide the theoretical analysis of the proposed bM-NSP method and compare it with one baseline, which is an LNEC-enabled extension of

NegGSP (Zheng et al. 2009). We select NegGSP as baseline because NegGSP and PNSP are the only two available methods comparable to bM-NSP, and NegGSP has proven to outperform PNSP. Accordingly, bM-NSP is only compared with NegGSP for simplicity.

The runtime of the NSP mining method is mainly consumed by the support calculation of the NSCs generated, $sup(NSC)$, and the comparison times during this support calculation determine the runtime of a method while the time consumed by arithmetic operations is negligible (Cao et al. 2016). With regard to bM-NSP, the number of comparison times for $sup(NSC)$ calculation is obtained by calculating the bit-wise operation upon BHT (i.e., by comparing the pair-wise bit stored in BHT). With regard to NegGSP, the number of comparison times is obtained by comparing each generated NSC with data sequences during dataset re-scanning. Here, six data factors are incorporated to describe and quantify the characteristics of a sequence dataset as follows: C is the average number of elements per sequence, T is the average number of items per element, S is the average length of maximal potentially PSP, I is the average number of items per element in maximal potentially PSP, DB is the number of data sequences in a sequence dataset, and N is the number of divergent items.

4.4.1 Runtime Analysis of bM-NSP

With regard to bM-NSP, during each iteration, an NSC is generated by a pair of seed sequences in an equivalent class. In the l-th iteration, let $|[P_{l-1}]_e|$ be the number of l-length equivalent classes and $|S_{pos}^{P_{l-1}^m}|$ and $|S_{neg}^{P_{l-1}^m}|$ be the number of positive and negative seeds in the m-th equivalent classes $[P_{l-1}^m]_e$. Hence, the number of l-length NSC generated by $[P_{l-1}^m]_e$, denoted by $|NSC^{P_{l-1}^m}|^{bM\text{-}NSP}$, is calculated in Eq. (4.1).

$$|NSC^{P^m_{l-1}}|^{bM\text{-}NSP} = (|S^{P^m_{l-1}}_{pos}| + |S^{P^m_{l-1}}_{neg}|) \times |S^{P^m_{l-1}}_{neg}| \tag{4.1}$$

Let L_{max} be the maximum length of all generated NSCs under a given threshold, then the total number of the NSC generated by bM-NSP, $|NSC|^{bM\text{-}NSP}$, is given in Eq. (4.2).

$$|NSC|^{bM\text{-}NSP} = \sum\nolimits_{l=1}^{L_{max}} \sum\nolimits_{m=1}^{|[P_{l-1}]_e|} |NSC^{P^m_{l-1}}|^{bM\text{-}NSP} \tag{4.2}$$

Finally, for the runtime performance of bM-NSP, the number of bits in each BHT is up to $C \times D\bar{B}$, where $D\bar{B}$ is the average support of an NSC in the sequence dataset. Accordingly, the total runtime for bM-NSP, T^{bM-NSP}, is presented in Eq. (4.3), where t^b is the unit runtime consumed conducting a bit-wise operation.

$$T^{bM\text{-}NSP} = |NSC|^{bM\text{-}NSP} \times (C \times D\bar{B} \times t^b) \tag{4.3}$$

4.4.2 Runtime Analysis of NegGSP

With regard to NegGSP, during each iteration, an NSC is generated by joining a negative seed with a positive or negative seed. Hence, in the l-th iteration, let $|S^{l-1}_{pos}|$ and $|S^{l-1}_{neg}|$ be the number of (l-1)-length positive and negative seeds respectively, then the number of l-length NSC, $|NSC^l|^{NegGSP}$, is demonstrated in Eq. (4.4).

$$|NSC^l|^{NegGSP} = (|S^{l-1}_{pos}| + |S^{l-1}_{neg}|) \times |S^{l-1}_{neg}| \tag{4.4}$$

Similarly, the total number of the NSCs generated by NegGSP, $|NSC|^{NegGSP}$, is given in Eq. (4.5).

$$|NSC|^{NegGSP} = \sum\nolimits_{l=1}^{L_{max}} |NSC^{l}|^{NegGSP} \tag{4.5}$$

Lastly, for the runtime of NegGSP, let t^c denote the unit runtime consumed by the comparison operation between two items, then the total runtime for NegGSP, T^{NegGSP}, is presented in Eq. (4.6).

$$T^{NegGSP} = |NSC|^{NegGSP}| \times (C \times T \times DB \times t^c) \tag{4.6}$$

4.4.3 Runtime Comparison w.r.t. Data Factors

The runtime ratio between bM-NSP and NegGSP is defined as the ratio of Eq. (4.3) to Eq. (4.6):

$$\frac{T^{bM\text{-}NSP}}{T^{NegGSP}} = \sum\nolimits_{l=1}^{L_{max}} \frac{\sum_{m=1}^{|[P_{l-1}]_e|} |NSC^{P_{l-1}^{m}}|^{bM\text{-}NSP}}{|NSC^{l}|^{NegGSP}} \times \frac{\bar{D}B \times t^b}{T \times DB \times t^c} \tag{4.7}$$

Although t^b and t^c depend on the specific experimental environments, the bit-wise operation is always more efficient than the comparison operation, and $\frac{t^b}{t^c}$ is a constant for the same environment. In the worst case, if in the l-th iteration, there is only one equivalent class in the current seed set (i.e., $|[P_{l-1}]_e| = 1$), then $\sum_{m=1}^{|[P_{l-1}]_e|} |NSC^{P_{l-1}^{m}}|^{bM\text{-}NSP} \leqslant (|S_{pos}^{l-1}| + |S_{neg}^{l-1}|) \times |S_{neg}^{l-1}| = |NSC^{l}|^{NegGSP}$. In addition, it is clear that $\bar{D}B \leqslant DB$; therefore, $\frac{T^{bM\text{-}NSP}}{T^{NegGSP}} \leqslant 1$ always holds, that is, bM-NSP is always more efficient than NegGSP.

Here, we analyze the impact of each data factor on the performance and superiority of the bM-NSP method: 1) the rise of C causes the runtime growth of bM-NSP per Eq. (4.3). In addition, as C increases, $\bar{D}B$ increases, and a larger number of equivalent classes are generated; thus bM-NSP becomes superior since the ratio $\frac{T^{bM\text{-}NSP}}{T^{NegGSP}}$ decreases; 2) Similarly, the increase of T results in the runtime growth of bM-NSP since $\bar{DB}$ rises, and $\frac{T^{bM\text{-}NSP}}{T^{NegGSP}}$ declines; 3) The change of S and I have less impact on the performance and superiority of bM-NSP, but with the increase of both factors, sequence dataset may get denser, and bM-NSP may consume more runtime; 4) the rise of DB causes the nonlinear growth of $\bar{DB}$; thus bM-NSP takes more time but its superiority becomes clearer since $\frac{T^{bM\text{-}NSP}}{T^{NegGSP}}$ decreases; and 5) the rise of N causes the sparsity of sequence dataset and less equivalent classes are generated, thus bM-NSP takes much less time but its superiority weakens. In summary, bM-NSP generally performs better than baselines on the datasets with higher C, higher T, and lower N, which is further verified in Section 4.5.

4.5 Experiments and Evaluation

Here, we evaluate the efficiency and scalability of the proposed bM-NSP method compared with two comparable baselines on multiple synthetic and real-life datasets.

4.5.1 Datasets and Baseline Methods

Eighteen synthetic datasets (DS1, DS2 and DS1.X), generated by the IBM data generator (Agrawal & Srikant 1995), and six real-life datasets (DS3, DS4, DS5, DS6, DS7 and DS8) are used for experiments.

- Dataset 1 (DS1): C8_T4_S6_I6_DB10k_N100, which on average has 8 elements per sequence (C), 4 items per element (T), 6 items in maximal potentially PSP (S), 6 items per element in maximal potentially PSP (I), 10K data sequences (DB), and 100 divergent items (N).

- Dataset 2 (DS2): C10_T8_S20_I10_DB10k_N200.

- Dataset 3 (DS3): A real-life UCI dataset which consists of 989,818 anonymous ordered webpage visits to MSNBC.com, where visits are recorded at the page category and in a temporal order (Cao et al. 2016).

- Dataset 4 (DS4): A real-life application dataset of health insurance claim sequences (Cao et al. 2016), which averages 21 elements per sequence, averages 2 items per element, 5,269 data sequences, and 340 divergent items.

- Dataset 5 (DS5): A real-life chain-store dataset containing 46,086 distinct items and 1,112,949 transactions (J. Pisharath n.d.), which is widely adopted for testing sequential pattern mining, especially for utility-based pattern mining.

- Dataset 6 (DS6): A real-life KDD-CUP 2000 dataset from SPMF [1], which contains 59,601 e-commerce click-stream sequences, 497 distinct items, and averages 2.42 items per sequence with a standard deviation of 3.22.

- Dataset 7 (DS7): Another real-life KDD-CUP 2000 dataset [2] with 77,512 click-stream sequences and 3,340 distinct items, averages 4.62 items per sequence with a standard deviation of 6.07.

- Dataset 8 (DS8): A real-life FIFA World Cup 98 dataset [3] with 20,450 click-stream sequences, 2,990 distinct items, and it averages 34.74 items per sequence with a standard deviation of 24.08.

- Dataset 1.X (DS1.X): A group of synthetic datasets generated to evaluate the sensitivity of bM-NSP and baselines with respect to different

[1] http://www.philippe-fournier-viger.com/spmf/datasets/BMS1_spmf

[2] http://www.philippe-fournier-viger.com/spmf/datasets/BMS2.txt

[3] http://www.philippe-fournier-viger.com/spmf/datasets/FIFA.txt

data factors (i.e., how the different characteristics of datasets influence the efficiency of each method) (Cao et al. 2016). The dataset DS1.X expands DS1 as the base dataset to generate 16 synthetic datasets, named DS1.X (X=1,...,16), by adjusting one factor once.

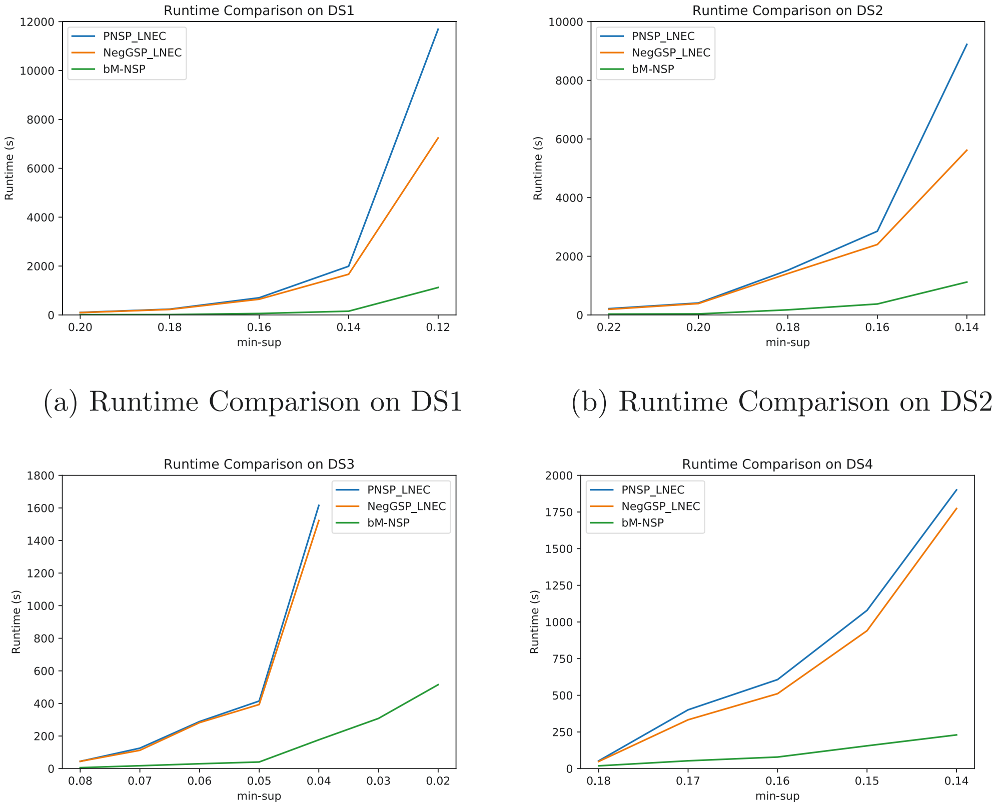
In addition, we revise PNSP and NegGSP to create two variants as baselines: PNSP_LNEC and NegGSP_LNEC, which are the two updated methods to enable the *EFC*, *LNEC* and *FCFC*. PNSP_LNEC is a PNSP-based variant, and its second phase is modified to derive all the possible negative elements satisfying the LNEC from each mined 1-size PSP, of which the negative elements with high support are applied to generate 1-size NSPs and the ones with high base-support assemble the negative element set to generate long-size NSCs in the subsequent phases. The NegGSP_LNEC is a NegGSP-based variant, and all the long-length NSCs with partial negative elements which satisfy the incorporated constraints are allowed to be generated in each joining operation of NSC generation. In this research, GSP (Srikant & Agrawal 1996) is applied as the PSP mining algorithm to discover the high-frequency PSPs, and then bM-NSP and two baselines are used to discover NSPs separately. To the best of our knowledge, PNSP and NegGSP are the only related methods which can be revised to be comparable to bM-NSP.

4.5.2 Performance Evaluation

Computational Cost Evaluation

Figures 4.2 and 4.3 illustrate the runtime of the proposed bM-NSP and two baselines, which is consumed by the NSP discovery on the eight datasets.

The runtime of bM-NSP is always significantly lower than that of PNSP_LNEC and NegGSP_LNEC on both synthetic and real-life datasets. Particularly, as



(a) Runtime Comparison on DS1

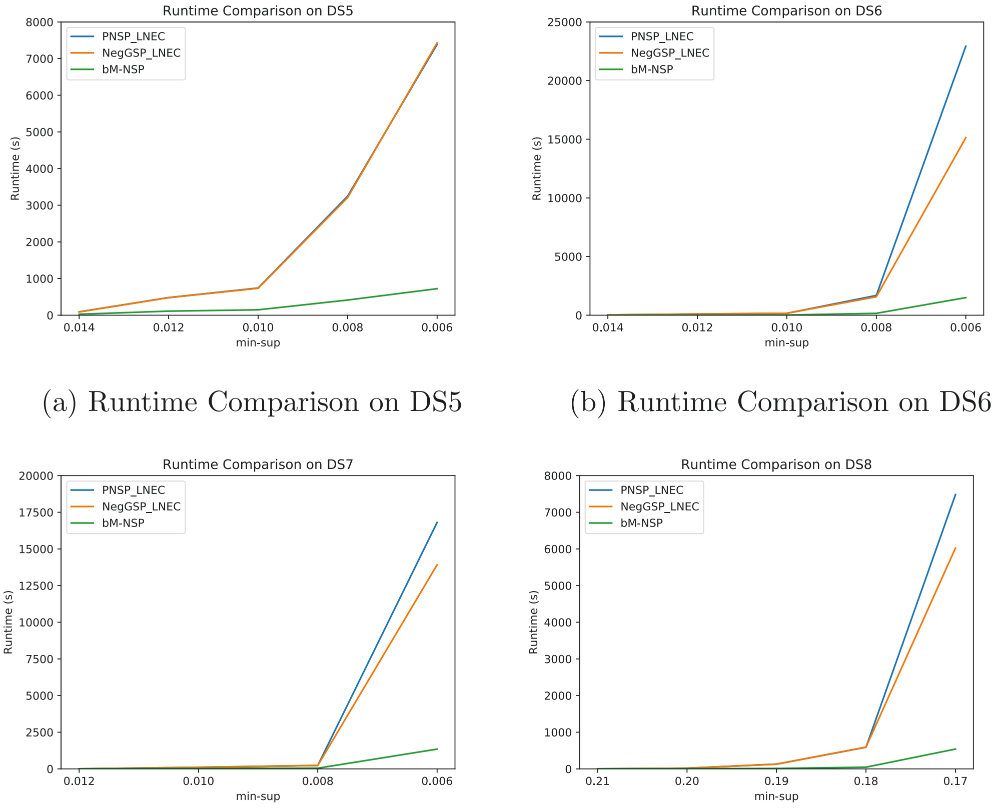
(b) Runtime Comparison on DS2

(c) Runtime Comparison on DS3

(d) Runtime Comparison on DS4

Figure 4.2: Runtime Comparison on Datasets DS1-DS4.

the threshold *min_sup* decreases, the execution time of both baselines rise rapidly while bM-NSP consumes a smaller proportion of the runtime used by PNSP_LNEC and NegGSP_LNEC. For instance, when mining NSPs on DS4, bM-NSP consumes only 13.26% to 12.09% of PNSP_LNEC runtime and 15.97% to 12.96% of NegGSP_LNEC runtime as *min_sup* declines from 0.17 to 0.14. In addition, when mining NSP on DS5 under $min\_sup = 0.006$, two baselines consume more than two hours while bM-NSP only takes around twelve minutes. Generally speaking, the proposed bM-NSP requires only about 10% of the execution time of both baselines on all datasets. The reason for these phenomena is that bM-NSP calculates the support of the NSCs generated only by bitmap operation upon the BHT of relevant seed sequences, which is significantly more efficient than the re-scan operation upon the whole dataset of two baselines.



(a) Runtime Comparison on DS5

(b) Runtime Comparison on DS6

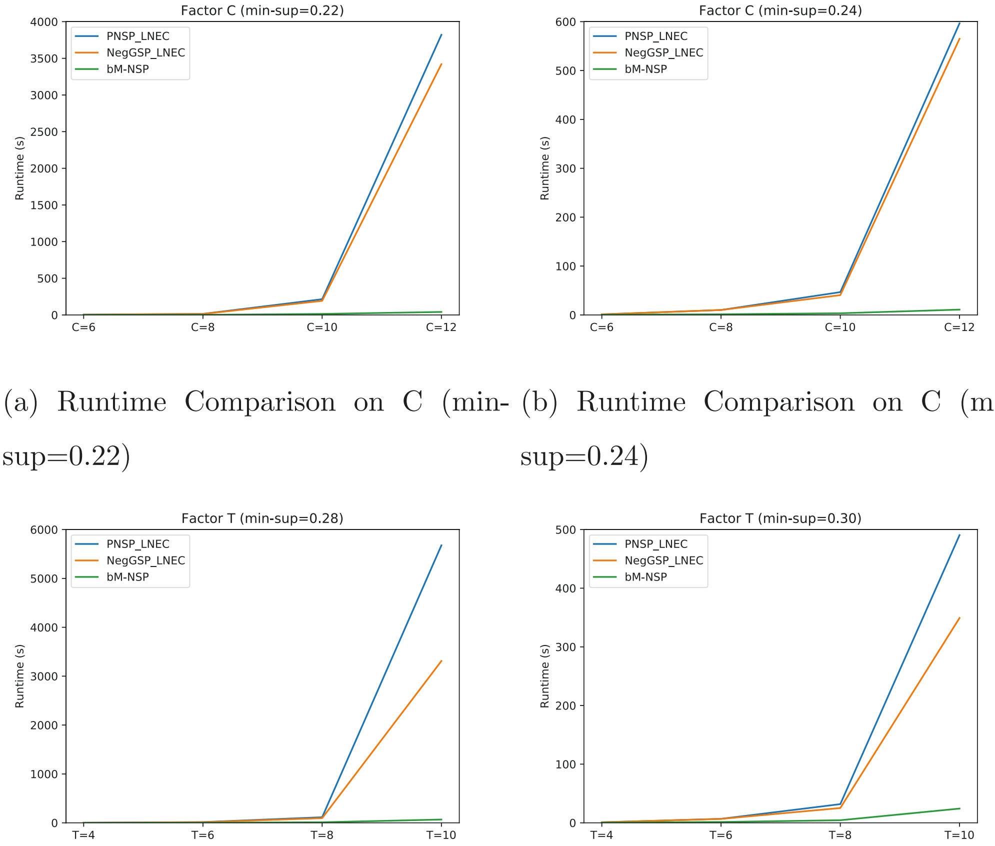
(c) Runtime Comparison on DS7

(d) Runtime Comparison on DS8

Figure 4.3: Runtime Comparison on Datasets DS5-DS8.

Sensitivity Evaluation

The performance of an NSP mining method may be sensitive to data factors of the sequence dataset (Cao et al. 2016), and an ideal method is supposed to maintain its runtime superiority on multiple datasets with divergent factors.



(a) Runtime Comparison on C (min-sup=0.22)

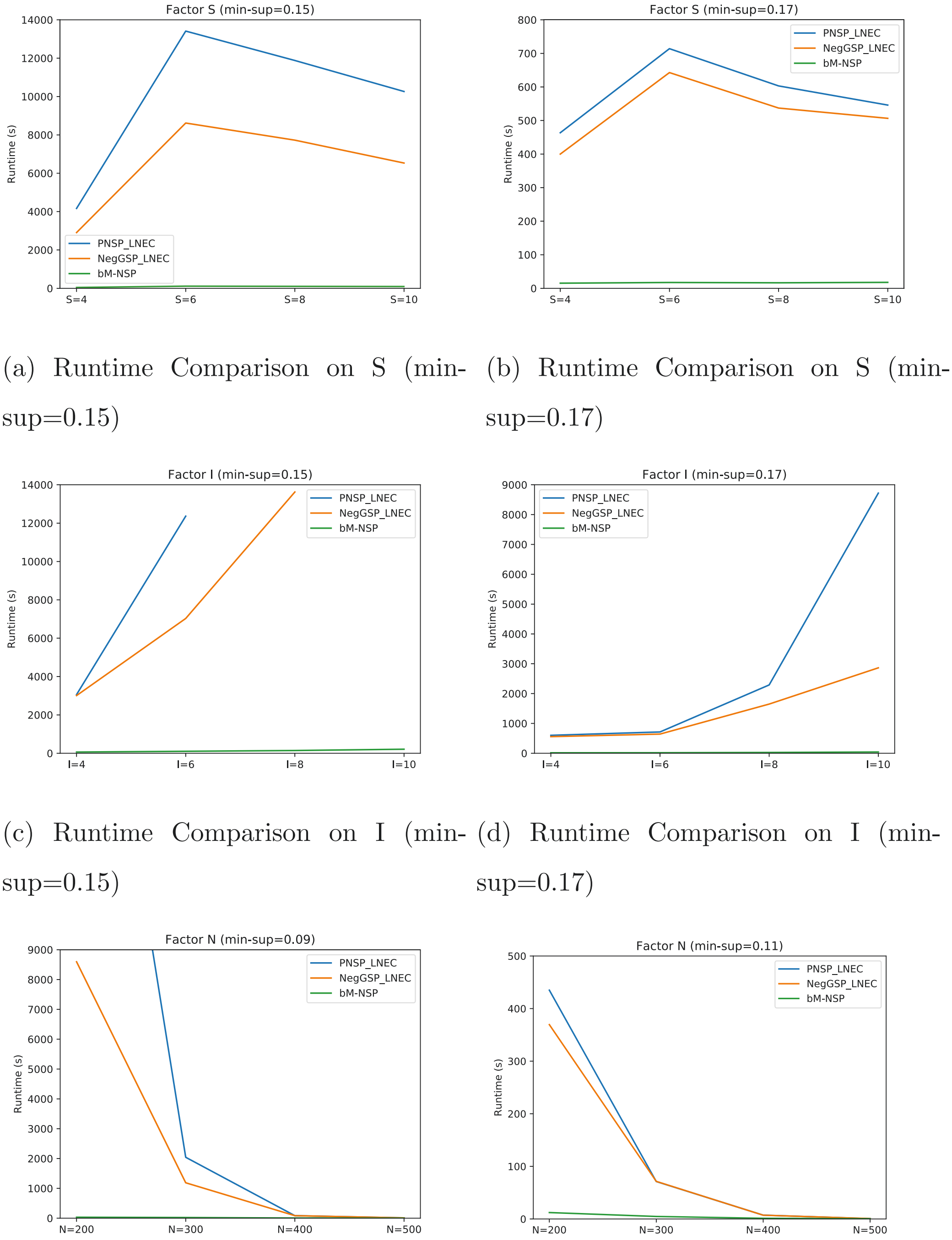
(b) Runtime Comparison on C (min-sup=0.24)

(c) Runtime Comparison on T (min-sup=0.28)

(d) Runtime Comparison on T (min-sup=0.30)

Figure 4.4: Runtime Comparison on Data Factors (C and T).

We analyze and compare the impact of the different data factors specified in Section 4.4 on the performance of bM-NSP and two baselines in terms of runtime, as illustrated in Figures 4.4 and 4.5 as well as Tables 4.3 and 4.4. Here *PN*, *NG* and *bN* stand for PNSP_LNEC, NegGSP_LNEC and bM-NSP, respectively, and t_1, t_2 and t_3 represent their runtime in seconds. t_2/t_1 is calculated to compare the runtime of bM-NSP with NegGSP_LNEC, and all sequence datasets have the same *DB* of 10k. Sixteen synthetic datasets with



(a) Runtime Comparison on S (min-sup=0.15)

(b) Runtime Comparison on S (min-sup=0.17)

(c) Runtime Comparison on I (min-sup=0.15)

(d) Runtime Comparison on I (min-sup=0.17)

(e) Runtime Comparison on N (min-sup=0.09)

(f) Runtime Comparison on N (min-sup=0.11)

Figure 4.5: Runtime Comparison on Data Factors (S, I and N).

Table 4.3: Runtime Sensitivity against Data Factors on DS1.X (Part I)

Factors	Dataset Name	min_sup	PN (t_1,s)	NG (t_2,s)	bN (t_3,s)	t_3/t_2
C=6	DS1.1=**C6**_T4_S6_I6_DB10k_N100	0.24	1.148	1.120	0.371	33.1%
		0.23	2.009	1.904	0.518	27.2%
		0.22	4.186	4.109	0.861	21.0%
C=8	DS1=**C8**_T4_S6_I6_DB10k_N100	0.24	10.122	10.087	1.393	13.8%
		0.23	13.349	11.571	1.603	13.9%
		0.22	14.462	13.377	1.610	12.0%
C=10	DS1.2=**C10**_T4_S6_I6_DB10k_N100	0.24	46.669	40.509	3.414	8.43%
		0.23	116.914	113.792	8.162	7.17%
		0.22	214.746	192.094	13.87	7.22%
C=12	DS1.3=**C12**_T4_S6_I6_DB10k_N100	0.24	596.631	564.984	10.89	1.93%
		0.23	1941.16	1722.56	25.31	1.47%
		0.22	3821.01	3420.46	40.64	1.19%
T=4	DS1=C8_**T4**_S6_I6_DB10k_N100	0.30	0.868	0.875	0.364	41.6%
		0.29	1.372	1.351	0.487	36.1%
		0.28	1.358	1.365	0.413	30.3%
T=6	DS1.4=C8_**T6**_S6_I6_DB10k_N100	0.30	6.776	6.692	1.414	21.1%
		0.29	7.630	7.700	1.379	17.9%
		0.28	14.805	12.348	1.701	13.8%
T=8	DS1.5=C8_**T8**_S6_I6_DB10k_N100	0.30	31.857	25.214	4.41	17.5%
		0.29	59.087	46.396	6.622	14.3%
		0.28	113.981	95.326	11.30	11.9%
T=10	DS1.6=C8_**T10**_S6_I6_DB10k_N100	0.30	490.525	349.279	24.27	6.95%
		0.29	2111.49	1327.20	46.11	3.47%
		0.28	5676.77	3310.78	67.04	2.02%
S=4	DS1.7=C8_T4_**S4**_I6_DB10k_N100	0.17	463.561	399.966	15.29	3.82%
		0.16	1094.79	778.106	21.18	2.72%
		0.15	4165.47	2906.68	44.12	1.52%
S=6	DS1=C8_T4_**S6**_I6_DB10k_N100	0.17	714.07	642.838	17.35	2.70%
		0.16	1967.62	1659.81	26.39	1.59%
		0.15	13413.2	8618.88	112.5	1.30%

divergent distributions, DS1.X, are generated to evaluate the performance sensitivity of methods with respect to data factors, which are extended from

Table 4.4: Runtime Sensitivity against Data Factors on DS1.X (Part II)

Factors	Dataset Name	min_sup	PN (t_1,s)	NG (t_2,s)	bN (t_3,s)	t_3/t_2
S=8	DS1.8=C8_T4_**S8**_I6_DB10k_N100	0.17	603.09	537.138	16.44	3.06%
		0.16	1843.08	1529.09	26.19	1.71%
		0.15	11880.3	7725.77	103.6	1.34%
S=10	DS1.9=C8_T4_**S10**_I6_DB10k_N100	0.17	545.986	506.485	17.72	3.50%
		0.16	1310.01	1207.00	21.88	1.81%
		0.15	10263.1	6532.37	93.18	1.43%
I=4	DS1.10=C8_T4_S6_**I4**_DB10k_N100	0.17	601.573	557.172	15.02	2.70%
		0.16	1287.64	1176.84	27.46	2.33%
		0.15	3059.81	3008.11	62.09	2.06%
I=6	DS1=C8_T4_S6_**I6**_DB10k_N100	0.17	714.070	642.838	17.35	2.70%
		0.16	1967.62	1659.81	26.39	1.59%
		0.15	12363.4	7028.55	103.0	1.47%
I=8	DS1.11=C8_T4_S6_**I8**_DB10k_N100	0.17	2290.23	1648.81	25.80	1.56%
		0.16	9741.58	4541.31	47.51	1.05%
		0.15	N/A	13623.9	143.5	1.05%
I=10	DS1.12=C8_T4_S6_**I10**_DB10k_N100	0.17	8720.64	2863.89	41.65	1.45%
		0.16	N/A	6248.80	78.79	1.26%
		0.15	N/A	N/A	210.6	0.0%
N=200	DS1.13=C8_T4_S6_I6_DB10k_**N200**	0.11	435.064	369.558	12.23	3.31%
		0.10	2601.74	1935.56	17.55	0.91%
		0.09	24894.8	8596.39	32.17	0.37%
N=300	DS1.14=C8_T4_S6_I6_DB10k_**N300**	0.11	71.225	71.540	4.781	6.68%
		0.10	301.623	261.478	10.98	4.20%
		0.09	2042.18	1187.74	22.03	1.85%
N=400	DS1.15=C8_T4_S6_I6_DB10k_**N400**	0.11	7.308	7.217	1.316	18.2%
		0.10	34.048	34.216	3.094	9.04%
		0.09	85.050	84.574	5.824	6.89%
N=500	DS1.16=C8_T4_S6_I6_DB10k_**N500**	0.11	0.679	0.798	0.399	50.0%
		0.10	3.269	3.409	0.847	24.8%
		0.09	10.437	10.710	1.624	15.2%

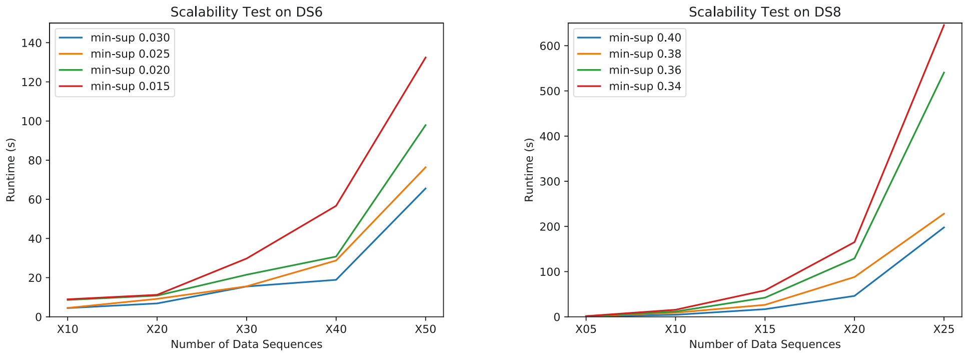
DS1 as the base dataset by tuning one factor, and the differentiators are marked by bolding the distinct factor.

The proposed bM-NSP method is significantly more efficient than the two baselines under different data factors and thresholds, and its performance is impacted by C, T and N. bM-NSP requires a longer runtime with the growth of C, T and the decrease of N. This is because when C rises, the average size of BHT increases and a larger number of bit-wise operations need to be conducted; when T rises, more potential long NSCs need to be generated and handled; and when N declines, the number of 1-size PSPs rises, the scale of seed sets and the average length of NSC increases, and the average support of NSC goes up. However, it is noted from t_3/t_2 that, generally speaking, bM-NSP has a higher and clearer superiority than NegGSP_LNEC on the datasets with higher C, higher T and lower N, which is consistent with the conclusion made in our theoretical analysis and demonstrates that bM-NSP is more scalable when handling more complex sequences. In general, bM-NSP works much more competitively under lower thresholds with respect to different data factors. In addition, we notice from the results of DS1 and DS1.6 that as T rises, bM-NSP saves a greater proportion of runtime than the other two baselines, as illustrated in our theoretical analysis. Lastly, the performance of our proposed bM-NSP method is not greatly affected by data factors S and I.

Scalability Evaluation

The proposed bM-NSP method calculates the support of the NSCs generated based on its BHT, of which the length is positively related to the size of the dataset, and thus bM-NSP requires more runtime when mining NSPs on larger data. Here, the scalability evaluation is conducted to test the performance of bM-NSP on large datasets, and we present the runtime used by bM-NSP on datasets DS6 and DS8 in Figure 4.6, from the perspective of different data sizes: from 10 (i.e., 8M) to 50 (40M and 2,980,050 sequences) times of DS6, and from 5 (13M) to 25 (65M and 511,250 sequences) times of DS8, with various low thresholds 0.015, 0.020, 0.025 and 0.030 on DS6, and 0.34, 0.36, 0.38, and 0.40 on DS8, respectively.

As illustrated in Figure 4.6a, on DS6, the runtime consumed by bM-NSP under four thresholds increases gently as the sampled data size rises to 50 times of its original size. Figure 4.6b further reveals that on DS8, the runtime of bM-NSP increases slowly when the sampled size is less than 20 times its original size, and it experiences around four-times the growth on the 25 times (×25) data size. These results demonstrate that our proposed bM-NSP can work well on large datasets even under relatively low thresholds.



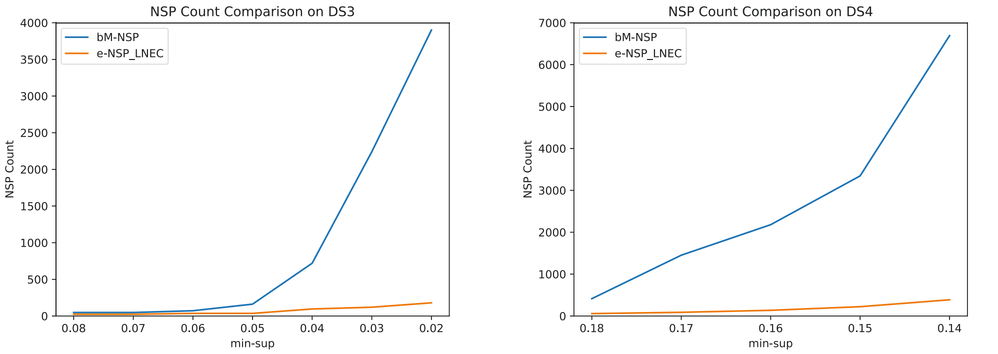
(a) Scalability Evaluation on DS6

(b) Scalability Evaluation on DS8

Figure 4.6: Scalability Evaluation on Datasets DS6 and DS8.

Coverage Evaluation

Set theory-based methods are another group of NSP mining methods which generate the NSC by negative conversion and calculate NSC support by referring to the PSPs discovered (Cao et al. 2016). They miss the NSPs whose *positive sequence partner* is not a high-frequency PSP and retain a smaller coverage compared with search-based ones. Here, we extend e-NSP to e-NSP_LNEC to enable NSP mining with LNEC and analyze the coverage of the search-based and set theory-based methods by comparing the NSP count of our proposed bM-NSP and e-NSP_LNEC, as illustrated in Figure 4.7. Figure 4.7a indicates that when mining NSP on DS3, as the threshold declines from 0.04 to 0.02, the e-NSP_LNEC only discovers 13.3% to 4.6% of the patterns mined by bM-NSP. Similarly, as illustrated in Figure 4.7b, when mining NSPs on DS4, e-NSP_LNEC only obtains around 6% of the NSPs mined by bM-NSP when $min\_sup \leqslant 0.17$. It is observed that set theory-based methods hold a tiny pattern coverage and require further research to enable complete NSP mining.



(a) NSP Count Comparison on DS3 (b) NSP Count Comparison on DS4

Figure 4.7: Coverage Evaluation on Datasets DS3 and DS4.

We can draw the following three-fold conclusions based on the above experimental analysis: 1) our proposed bM-NSP is significantly more efficient on multiple datasets in terms of runtime compared with the two baselines; 2) bM-NSP performs stably on the datasets with different data factors and demonstrates clear superiority on the datasets with a large number of elements per sequence, a large number of items per element, and a small number of divergent items; and 3) bM-NSP works well on large datasets even under relatively low thresholds.

4.5.3 Case Study

Negative sequential pattern has been utilized in the fraud claim detection of health insurance to discover high-frequency patterns of medical treatment (Zheng 2012) and to identify fraud claim sequences based on the analysis of the nonoccurring medical service code (Cao et al. 2016). Since there exists cooccurrences between the treatment codes in the service procedures of a patient, the nonoccurrences of some service codes in a certain context raises potential suspicion of a current claim. Here, the fraud-suspect patterns are discovered as follows: we firstly discover the set of NSPs, and for each discovered NSP S_{neg}, it is assumed to be fraud if $\frac{sup(S_{neg})}{sup(MPS(S_{neg}))} \leqslant min\_ratio$ holds because these negative medical codes are expected to occur together in this claim context. The fraud pattern discovered above can be converted into business rules, which are further used for fraud claim detection. For instance, a fraud pattern $< (114, \neg 121), 114, (121, \neg 12) >$ is discovered from DS4, where each item is a claimed medical code. It indicates that if a patient claims 121 followed by two courses of 114, then 121 is likely to be claimed together with the first course of 114, and 12 is also highly likely to cooccur with 121. Accordingly, if a claim sequence appears as $< 114, 114, 121 >$, it is likely to be fraudulent and should be flagged. Note that since $< (114, 121), 114, (121, 12) >$ is not necessarily a PSP, this fraud pattern may be missed by existing NSP mining methods based on NEC, which illustrates the importance of proposing an efficient framework to enable complete NSP mining with the LNEC.

4.6 Summary

To discover nonoccurring yet important behaviors, NSP mining is a powerful tool, but it receives little attention and is challenging owing to the hidden nature of nonoccurring behaviors and the many open theoretical issues in NSB analytics. Limited existing research has been built by involving strong constraints to make NSP mining doable, leading to serious information loss, which could be problematic as important NOBs may be few and fail to be detected. Our work in this chapter substantially releases the constraints on the negative elements in NSPs for the discovery of more flexible patterns. To tackle the corresponding significant challenges caused, a novel and efficient vertical mining framework, VM-NSP, and an instance method, bM-NSP, are proposed to discover the complete set of NSPs of complex sequences. The theoretical analysis confirms the performance superiority of our proposed bM-NSP on datasets with different data factors. Experimental results further reveal the significant efficiency improvement, sensitivity, scalability, and pattern coverage of bM-NSP, especially for NSP discovery on dense and large datasets. Nonoccurring sequential behavior analytics is new and challenging; we are working on a theoretical foundation for this important but open issue.

In this chapter, we only concentrate on the discovery of high-frequency NSB patterns in a massive search space, leading to the challenging issue that informative and insightful patterns are inundated in an extremely large-scale yet highly redundant NSP collection, which makes the further analysis and application of these NSPs complicated and time-consuming (Zheng 2012). Accordingly, we discuss further research in the next chapter to discover a representative NSP subset from the whole NSP collection, which consists of only high-quality and diverse patterns to enable the efficient capture of the vital knowledge and improve the actionability of the patterns discovered.

Chapter 5

Representative Negative Sequential Pattern Discovery

5.1 Introduction

5.1.1 Problem Statement

As discussed in Chapter 4, NSP mining is unique and critical for modeling sequences of occurring and nonoccurring behaviors, and its research outcome has been widely applied to many areas. However, as one of few approaches available for NSB analytics, NSP mining methods face significant theoretical and practical challenges, including lacking modeling hidden element and pattern-wise nonoccurrences and their complicated combinations and relations with occurring elements and patterns. These often result in high computational cost, large-scale high-frequency yet highly overlapping findings, and missing significant yet relatively low-frequency behaviors, making the collection of the NSPs discovered less to non actionable (Zheng et al. 2010, Cao et al. 2016, Cao 2012, Liu et al. 2015*a*). Discovering a representative subset consisting of high-quality and diverse NSPs with low complexity and high efficiency (i.e., *representative NSP discovery*), which is supposed to filter similar and redundant patterns but keep discriminating and informative ones that can represent the whole original NSP collection, is thus highly critical yet complicated (Zheng 2012).

The critical challenges of *representative NSP discovery* lie in efficiently (1) representing the explicit and implicit behavior relations between occurring and nonoccurring elements and their combinations, (2) quantifying the optimal selection criteria of each subset candidate in the whole NSP collection, and (3) handling various sequential characteristics. As discussed in Section 2.2.4, none of the existing NSB analytics research addresses these challenges.

5.1.2 Design and Contributions

This research makes the first attempt by inventing a DPP-based method - explicit and implicit element/pattern relations-based *representative NSP discovery* (EINSP) - to discover a representative subset of high-quality and diverse NSPs by involving both cooccurrence-based explicit relations and nonoccurrence-based implicit relations between NSP elements and patterns. The EINSP consists of four components: *NSP graph construction*, *explicit relation modeling*, *implicit relation modeling*, and *overall relation-based selection* to efficiently select a k-size informative and diverse subset from a collection of the discovered NSP $\mathbb{S}$, as illustrated in Figure 5.1.

First, *NSP graph construction* transforms a collection of the discovered NSP $\mathbb{S}$ into a directed DPP-based graph $\mathcal{G}$ for a powerful and compressed representation of the NSPs. Second, *explicit relation modeling* captures the explicit relations between NSPs in the DPP graph and computes the probability $P_e^k(\mathcal{S})$ of selecting an NSP subset $\mathcal{S} \in \mathbb{S}$ in terms of cooccurrences

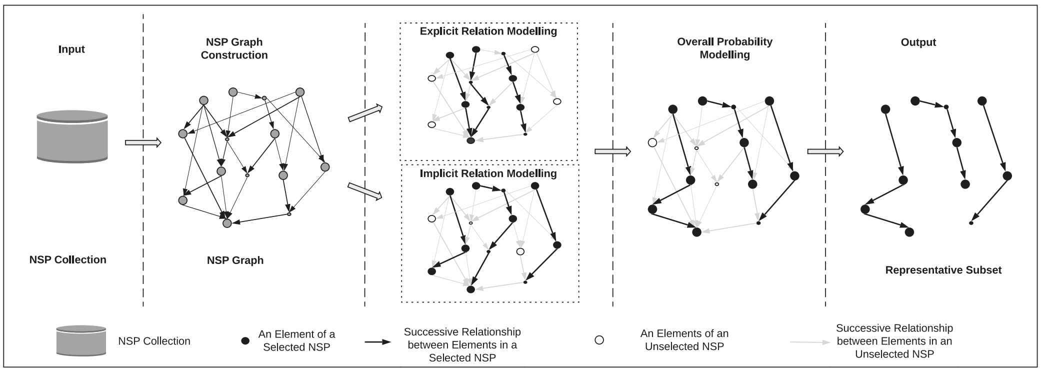

Input
NSP Collection
NSP Graph Construction
NSP Graph
Explicit Relation Modelling
Implicit Relation Modelling
Overall Probability Modelling
Output
Representative Subset
NSP Collection
An Element of a Selected NSP
Successive Relationship between Elements in a Selected NSP
An Elements of an Unselected NSP
Successive Relationship between Elements in an Unselected NSP

Figure 5.1: Representative NSP Discovery for High-quality and Diverse NSP by Modeling Both Explicit and Implicit Element/Pattern-wise Relations

between NSPs. Third, *implicit relation modeling* captures the implicit relations between NSPs in the DPP graph and computes the probability $P_i^k(\mathcal{S})$ of selecting a subset $\mathcal{S}$ in terms of the nonoccurrences with third-party NSPs. Lastly, *overall relation-based selection* computes the overall probability over the NSP subset $\mathcal{S}$ by integrating both $P_e^k(\mathcal{S})$ and $P_i^k(\mathcal{S})$ and selects the representative NSP subset in the DPP graph per the overall probability.

The above design for *representative NSP discovery* makes the following original contributions.

- The problem of *representative NSP discovery* is converted to a probabilistic subset selection problem in a DPP graph, which takes advantage of the probabilistic DPP theoretical foundation and strength in subset selection with diversity. A path in the graph is viewed as an NSP, and the subset selection task is to find a sub-graph that covers the most representative patterns as much as possible. The EINSP can thus select a subset of informative and diverse NSPs. This represents the first work of exploiting a graph (in particular the DPP) for representative NSP discovery and can generate a high-quality (high probability) subset.

- The cooccurring and nonoccurring element and pattern relations in NSPs are modeled in terms of the direct and indirect DPP-based node/edge dependencies, which captures the quality and diversity of each pattern in the NSP collection in terms of both explicit element/pattern cooccurrences and implicit nonoccurrences conditional on third parties. This captures rich element interactions and pattern relations in NSB analytics, rarely explored in existing related work.

- The EINSP integrates both the above explicit and implicit element/pattern relations in the DPP-based NSP graph and effectively samples those highly explicitly and implicitly related NSPs as a representative subset of the high-quality and diverse NSB patterns.

We derive the DPP-based theory for *representative NSP discovery* and verify our proposed EINSP on six real-life datasets and 17 synthetic datasets in terms of sequence and item coverage, average pattern size, average implicit relation strength, and sensitivity to various data factors including scalability. The analysis demonstrates significant NSP performance and great potential for actionable NSB analytics.

5.2 The EINSP Method

Here, we introduce the EINSP design for *representative NSP discovery* and then present its algorithm. Figure 5.1 illustrates the process of the proposed representative NSP discovery. It consists of four components: *NSP graph construction*, *explicit relation modeling*, *implicit relation modeling*, and *overall relation-based selection*. The EINSP takes a collection of the discovered NSPs, denoted as $\mathbb{S} = \{S_1, S_2, \dots, S_N\}$, as input, which is discovered by any existing NSP mining method. Here, $S_i =< S_i[1], S_i[2], \dots, S_i[n_i] >\in \mathbb{S}$ is an

NSP, and $S_i[k] \in S_i$ is an element in the NSP. We first represent the NSP as a DPP-based NSP graph. Then, both cooccurrence-based explicit relations and nonoccurrence-based implicit relations between NSP elements and patterns are learned, respectively. The EINSP finally outputs a representative k-size subset $\mathcal{S} \subseteq \mathbb{S}$ based on the overall probability $P^k(\mathcal{S})$ that integrates both explicit and implicit pattern relations for better NSP selection.

5.2.1 NSP Graph Construction

Since the NSP collection $\mathbb{S} = \{S_1, S_2, \ldots, S_N\}$ is always large scale (Cao et al. 2016) and many patterns share identical sub-sequences as prefixes, the *NSP graph construction* converts $\mathbb{S}$ into a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In this directed graph, each node $n \in \mathcal{V}$ corresponds to an element of a pattern and each edge $e \in \mathcal{E}$ stands for a directed link from one element to another in an NSP. In this way, each NSB pattern $S_i \in \mathbb{S}$ is transformed into a directed path of $\mathcal{G}$. We assume an *explicit element quality term* to quantify the significance of each element (a node) and each edge (an element pair) in terms of the explicit cooccurrence relations of elements. Taking pattern $S_i =< S_i[1], S_i[2], \ldots, S_i[n_i] >\in \mathbb{S}$ as an example, we assign an explicit element quality term $q_e(S_i[j])$ to element $S_i[j]$ and $q_e(S_i[j], S_i[j{+}1])$ to the element pair $< S_i[j], S_i[j{+}1] >$. In addition, we assign an *explicit element diversity feature vector* $\phi_e(S_i[j]) \in R^{|E|}$ for each node $S_i[j] \in S_i$ to measure the explicit cooccurrence similarity between element $S_i[j]$ and any other elements, where E is the set of potential elements, and $|E|$ is the size of E. Lastly, we assign an *implicit path quality term* $q_i(S_i)$ and an *implicit path diversity feature vector* $\phi_i(S_i)$ to path S_i in directed graph $\mathcal{G}$ to quantify its quality and diversity in terms of implicit relations between NSB patterns.

5.2.2 Explicit Relation Modeling

Since each pattern $S_i \in \mathbb{S}$ represents an ordered structure of elements, the explicit cooccurrence probability of a subset $\mathcal{S}$ can be computed by a fixed k-size SDPP (k-SDPP) model which captures the distribution on the condition that subset $\mathcal{S} \subseteq \mathbb{S}$ has cardinality k as follows:

$$P_e^k(\mathcal{S}) = \frac{det(L_{\mathcal{S}}^e)}{\sum_{|\mathcal{S}'|=k} det(L_{\mathcal{S}'}^e)} \tag{5.1}$$

Here, L^e is a positive semi-definite kernel, $L_{\mathcal{S}}^e$ represents the restriction on L^e to entries indexed by subset $\mathcal{S}$ (i.e., $L_{\mathcal{S}}^e \equiv [L_{ik}^e]_{S_i, S_k \in \mathbb{S}}$), and $det(L_{\mathcal{S}}^e)$ denotes the determinant of $L_{\mathcal{S}}^e$ (Kulesza & Taskar 2011). Following (Kulesza et al. 2012), the kernel L^e can be rewritten as a Gram matrix: $L^e = B^{eT} B^e$, where $B^e \in R^{|E| \times |\mathbb{S}|}$ is the matrix whose each column is a feature vector describing the corresponding pattern in NSP collection $\mathbb{S}$ (Kulesza et al. 2012), and column B_i^e can be factorized as the production of an *explicit path quality score* $q_e(S_i) \in R^+$ and a normalized *explicit path diversity feature vector* $\phi_e(S_i) \in R^{|E|}$, $\|\phi_e(S_i)\| = 1$. Here, $q_e(S_i) \in R^+$ can be regarded as a non-negative measure of the explicit significance of pattern S_i, and $\phi_e(S_i)^T \phi_e(S_k) \in [-1, 1]$ as a signed measure of the explicit similarity between patterns S_i and S_k. Accordingly, the entries of kernel L^e can be further decomposed as follows:

$$L_{ik}^e = q_e(S_i)\phi_e(S_i)^T \phi_e(S_k) q_e(S_k) \tag{5.2}$$

Eq. (5.2) gives rise to a distribution which places higher weight on the subsets that are composed of higher quality and more diverse patterns (Kulesza & Taskar 2012).

To efficiently define a DPP over the NSB patterns, the explicit path quality score $q_e(S_i)$ is decomposed multiplicatively using a log-linear model that depends on the explicit quality of each element $q_e(S_i[j])$ and its element pair $q_e(S_i[j], S_i[j{+}1])$ (two elements $S_i[j]$ and $S_i[j{+}1]$), which is as follows:

$$\begin{aligned} q_e(S_i) &= exp(\sum\nolimits_{j=1}^{n_i} q_e(S_i[j]) + \sum\nolimits_{j=1}^{n_i-1} q_e(S_i[j], S_i[j{+}1])) \\ &= \prod\nolimits_{j=1}^{n_i} exp(q_e(S_i[j])) \times \prod\nolimits_{j=1}^{n_i-1} exp(q_e(S_i[j], S_i[j{+}1])) \end{aligned} \tag{5.3}$$

As discussed in Section 4.1, the research on NSP mining is built on the frequentist statistics; we specify the *explicit quality of a pattern feature d* (element and element pair in the feature set) in terms of element and element pair frequencies (i.e., $q_e(\centerdot) \equiv \frac{|\{s_d | s_d \in D \wedge <\centerdot> \subseteq s_d\}|}{|D|}$), where D denotes the sequence dataset and s_d is a data sequence from dataset D. Similarly, the normalized *explicit diversity vector of a pattern $\phi_e(S_i)$* is decomposed additively over the elements:

$$\phi_e(S_i) = Normalization(\sum\nolimits_{j=1}^{n_i} \phi_e(S_i[j])) \tag{5.4}$$

Here, $Normalization(\centerdot)$ guarantees that for any pair of patterns S_i and S_k, $\phi_e(S_i)^T \phi_e(S_k) \in [-1, 1]$ is a signed measure of the similarity between these patterns. Here, the diversity feature $d_k^e(S_i[j]) \in \phi_e(S_i[j])$, which is the k-th component of the vector $\phi_e(S_i[j])$, is identified using the normalized pointwise mutual information (NPMI) of the element $S_i[j]$ and another element $E_k \in E$ for its strong ability to capture both linear and non-linear dependencies (Bouma 2009), which is defined below:

$$d_k^e(S_i[j]) \equiv NPMI(S_i[j], E_k) = \frac{h(S_i[j]) + h(E_k) - h(S_i[j], E_k)}{h(S_i[j], E_k)} \tag{5.5}$$

$h(\centerdot) = -logp(\centerdot)$, where $p(S_i[j])$ and $p(E_k)$ are the marginal probabilities of elements $S_i[j]$ and E_k, and $p(S_i[j], E_k)$ is the joint probability. Note that

$d_k^e(S_i[j]) \in [-1, 1]$, and some orientation values are as follows: when elements $S_i[j]$ and E_k occur separately but never together (negative dependent), $d_k^e(S_i[j]) = -1$; when they are distributed under independence, $d_k^e(S_i[j]) = 0$ as the numerator is 0; and when they completely cooccur (positive dependent), $d_k^e(S_i[j]) = +1$ (Bouma 2009). Hence, $\phi_e(S_i[j])$ works as an element dependence measure of the element $S_i[j]$ w.r.t. all other elements.

Because of the large search space and high computational cost involved in NSP mining, the size of NSP collection $\mathbb{S}$ is always great (Cao et al. 2016, Hsueh et al. 2008, Zheng et al. 2009), which makes kernel L^e quite large. In addition, DPP-based methods are typically memory intensive, the inference on kernel L^e may be highly intractable if $|\mathbb{S}|$ is large. Accordingly, a dual representation of L^e, denoted as $C^e = B^e B^{eT}$, is constructed to represent the properties carried by kernel L^e to enable the efficient inference (Kulesza et al. 2012). Note that $C^e \in R^{|E| \times |E|}$ is a symmetric and positive semi-definite matrix, where typically $|E| \ll |\mathbb{S}|$, and thus C^e is always much smaller in scale and less sensitive to the threshold than kernel L^e. Through transforming the computational focus into the representation C^e, the memory consumption of the proposed method can be greatly reduced (Kulesza et al. 2012). Accordingly, the dual representation C^e can be factorized as follows:

$$C^e = \sum\nolimits_{S_i \in \mathbb{S}} q_e^2(S_i) \phi_e(S_i) \phi_e(S_i)^T \tag{5.6}$$

The calculation of dual representation C^e can be done by the second-order message passing algorithm in the time complexity $O(|E|^2 |\mathbb{S}|)$ (Li & Eisner 2009, Borodin 2009).

Once representation C^e is computed, it can be eigen-decomposed in the form of $C^e = \sum_n \lambda_n^e v_n^e v_n^{eT}$ in the time complexity $O(|E|^3)$, and then the eigenvalue/eigenvector pairs $(\lambda_n, v_n)_{N_v}$ of representation C^e are available. As proved in (Kulesza et al. 2012), the non-zero eigenvalues of C^e and L^e

are identical, and if v_n is the n-th eigenvector of C^e, then $B^{eT}v_n$ is the n-th eigenvector of L^e, sharing the same eigenvalue λ_n^e. That is, $(\lambda_n^e, B^{eT}v_n^e)_{N_v}$ are the corresponding pairs of kernel L^e, making $L^e = \sum_n \lambda_n^e (B^{eT}v_n^e)(B^{eT}v_n^e)^T$.

To efficiently select a k-size subset of representative patterns, following the mechanism of k-DPP (Kulesza & Taskar 2011), we formalize the explicit probability $P_e^k(\mathcal{S})$ as follows:

$$P_e^k(\mathcal{S}) = \frac{1}{e_{k,N_v}^e} \sum_{|J|=k \wedge J \subseteq \{1,2,\ldots,N_v\}} P^{V_J^e}(\mathcal{S}) \prod_{n \in J} \lambda_n^e \tag{5.7}$$

Here, J is the index subset of $(\lambda_n^e, B^{eT}v_n^e)_{N_v}$, and V_J^e stands for the eigenvector subset indexed by J (i.e., $V_J^e \equiv \{B^{eT}v_n^e\}_{n\in J}$). In addition, $e_{k,N_v}^e = \sum_{|J|=k \wedge J \subseteq \{1,2,\ldots,N_v\}} \prod_{n\in J} \lambda_n$ is the k-th elementary symmetric polynomials on eigenvalues, which is equivalent to the normalization constant in Eq. (5.1) and can be computed by a recursive algorithm in the time complexity $O(|E|k)$ (Kulesza & Taskar 2011). Lastly, $P^{V_J^e}$ denotes an elementary DPP with marginal kernel $K^{V_J^e} = \sum_{n\in V_J^e} B^{eT} v_n^e v_n^{eT} B^e$.

5.2.3 Implicit Relation Modeling

Inspired by (Wang & Cao 2017), an NSB pattern S_i can be viewed as highly (implicitly) significant if the items in pattern S_i are highly dependent on third-party itemset Z, which means that these items may be relatively less likely cooccur but have a high probability of cooccurring with itemset Z. Here, itemset Z is called a *link itemset* since it serves as a bridge between the items in pattern S_i, where the items are called *implicitly related with* itemset Z. In addition, the implicit relation between a pair of patterns S_i and S_j can be modeled by their explicit dependencies with itemset Z in the form of $S_i \oplus S_j|Z$, which indicates that the items in both S_i and S_j likely cooccur with itemset Z. Intuitively, if NSB patterns S_i and S_j share a larger number of link itemsets and keep stronger dependencies on them, they are deemed to be more highly implicitly similar and shall be less likely to appear in the representative subset.

Jointly considering the design in (Wang & Cao 2017) and Section 5.2.2, the explicit relation between an item i and itemset Z can be quantified as the NPMI between i and Z, denoted as $NPMI(i, Z)$. If $NPMI(i, Z) > 0$, where Z is called a dependent itemset of item i, then all the dependent itemsets of i constitute its dependent itemset group, denoted as $A_i = \{Z|NPMI(i, Z) > 0\}$. Further, given an itemset I_s, the intersection set of all the dependent itemset groups of the items in I_s constitutes the link group of I_s, denoted as $G_{I_s} \equiv \cap_{i \in I_s} A_i$. Finally, for each shared dependent itemset $H \in G_{I_s}$, the *conditional implicit relation strength* (CIRS) of itemset I_s on H is defined as the minimum of the NPMI between the item i in I_s and itemset H, denoted as $CIRS(I_s|H) \equiv min\{NPMI(i, H)|i \in I_s\}$, and the *implicit relation strength* (IRS) of I_s is defined as the sum of its CIRS on all dependent itemsets, denoted as $IRS(I_s) \equiv \frac{\sum_{H \in G_{I_s}} CIRS(I_s|H)}{|G_{I_s}|}$.

Accordingly, for a pattern S_i, its *implicit path quality term* $q_i(S_i)$ is defined as the IRS of the corresponding itemset which is transformed from S_i, denoted as $flat(S_i)$. However, some long-size patterns may share no dependent itemsets (i.e., $G_{flat(S_i)} = \varnothing$). In such a case, we define $q_i(S_i)$ as the maximum IRS of its subsets with the largest size as follows:

$$q_i(S_i) = max\{IRS(S')|\{argmax_{size(S')}|S' \in \mathcal{I} \wedge S' \subseteq flat(S_i)\}\}. \tag{5.8}$$

Here, $\mathcal{I}$ is the collection of the *implicitly related itemsets* (IRI), which are the itemsets with high IRS that can be mined by an adapted IRRMiner method (Wang & Cao 2017). The itemset $\{argmax_{size(S')}|S' \in \mathcal{I} \wedge S' \subseteq flat(S_i)\}$ contains the subset of the IRI, of which each itemset is the subset of $flat(S_i)$

with the largest size. The *implicit path quality term* $q_i(S_i)$ is defined as the highest IRS of these itemsets which measures the implicit quality of the major items in pattern S_i with respect to their related link itemsets.

The *implicit path diversity feature vector* of pattern S_i is constructed as $\phi_i(S_i) = Normalization(d^i(S_i))$, where $d^i(S_i) \in R^{|H_S|}$ and H_S is the set of all dependent itemsets. Here, $d^i(S_i)$ quantifies the implicit dependency of pattern S_i regarding all potential *link itemsets*, and $\phi_i(S_i)^T\phi_i(S_j) \in [-1, 1]$ works as the implicit similarity between patterns S_i and $\S_j$. In this research, the k-th component $d^i_k(S_i) \in d^i(S_i)$ is built as the conditional IRS of $flat(S_i)$ on dependent itemset $H_k \in H_S$, that is, $d^i_k(S_i) \equiv CIRS(flat(S_i)|H_k)$ so that the production between the implicit path diversity feature vectors of two patterns is proportional to the fraction of dependent itemsets they share.

Accordingly, the implicit probability of a subset $\mathcal{S} \subseteq \mathbb{S}$ is computed as a k-DPP-based model, which is as follows:

$$P^k_i(\mathcal{S}) = \frac{det(L^i_{\mathcal{S}})}{\sum_{|\mathcal{S}'|=k} det(L^i_{\mathcal{S}'})}. \tag{5.9}$$

Similar to the discussion in Section 5.2.2, kernel L^i can also be rewritten as $L^i_{ij} = B^{i^T}B^i$, where the columns of B^i are given by $B^i_k = q_i(S_k)\phi_i(S_k)$, and the dual representation of L^i is constructed as $C^i = B^iB^{i^T}$. Assume that (λ^i_n, v^i_n) is an eigenvalue/eigenvector pair of representation C^i, then $(\lambda^i_n, B^{i^T}v^i_n)$ is the corresponding pair of L^i. Thus, Eq. (5.9) is rewritten as follows:

$$P^k_i(\mathcal{S}) = \frac{1}{e^i_{k,N_v}} \sum_{|J|=k \wedge J\subseteq\{1,2,\ldots,N_v\}} P^{V^i_J}(\mathcal{S}) \prod_{n\in J} \lambda^i_n. \tag{5.10}$$

Here, $P^{V^i_J}(\mathcal{S})$ stands for an elementary DPP with marginal kernel $K^{V^i_J} = \sum_{n\in V^i_J} B^{i^T} v^i_n {v^i_n}^T B^i$.

5.2.4 Overall Relation-based Selection

Inspired by (Yuan et al. 2016), the overall probability of a subset $\mathcal{S}$ can be modeled by jointly integrating the above explicit and implicit relation-oriented probabilities as follows:

$$P^k(\mathcal{S}) = w_e \times P_e^k(\mathcal{S}) + w_i \times P_i^k(\mathcal{S}). \tag{5.11}$$

Here, w_e and w_i are the parameters to govern the balance between $P_e^k(\mathcal{S})$ and $P_i^k(\mathcal{S})$, where $w_e + w_i = 1$. In this research, we adopt that $w_e = \frac{fr\bar{e}q(\mathcal{S})}{fr\bar{e}q(\mathcal{S})+I\bar{R}S(flat(\mathcal{S})}$ and $w_i = \frac{I\bar{R}S(flat(\mathcal{S}))}{fr\bar{e}q(\mathcal{S})+I\bar{R}S(flat(\mathcal{S})}$, and $fr\bar{e}q(\mathcal{S})$ is the average frequency of the patterns in subset $\mathcal{S}$ while $I\bar{R}S(flat(\mathcal{S}))$ is the average IRS of the flat itemsets transformed from the patterns in subset $\mathcal{S}$. In this way, the overall probability $P^k(\mathcal{S})$ is more affected by the major relations of patterns in subset $\mathcal{S}$. Substituting Eqs. (5.7) and (5.10) in Eq. (5.11), we obtain the overall probability of subset $\mathcal{S}$, which will be used to select the k-size subset in the following section.

$$P^k(\mathcal{S}) = \sum_{d\in\{e,i\}} \sum_{|J|=k\wedge J\subseteq\{1,2,\ldots,N_v\}} \frac{w_e P^{V_J^d}(\mathcal{S})}{e^d_{k,N_v}} \prod_{n\in J} \lambda_n^d \tag{5.12}$$

5.2.5 The EINSP Algorithm

The EINSP implements the process in Figure 5.1 to select a representative NSP subset based on the overall subset probability. Per Eq. 5.12, $P^k(\mathcal{S})$ is modeled as a mixture of elementary DPPs (proved in (Kulesza et al. 2012)); EINSP can sample the subset within two main loops. In the first loop, a subset of k eigenvectors is selected where the probability of selecting each eigenvector depends on its associated eigenvalue. Particularly, a k-size index subset J is sampled by $P(J) = \sum_{d\in\{e,i\}} \frac{w_d}{e^d_{k,N_v}} \prod_{n\in J} \lambda_n^d$, and the marginal probability of index $n \in J$ is as follows:

$$P(n \in J) = \textstyle\sum_{d \in \{e,i\}} w_d \lambda^d_n \frac{e^d_{k-1,n-1}}{e^d_{k,n}} \tag{5.13}$$

Due to the dual representation, the sets V^e and V^i of eigenvectors of kernels L^e and L^i are represented by their corresponding sets of eigenvectors of C^e and C^i, denoted as $\hat{V}^e$ and $\hat{V}^i$, with the mapping $V^e = \{B^{eT}\hat{v}^e | \hat{v}^e \in \hat{V}^e\}$ and $V^i = \{B^{iT}\hat{v}^i | \hat{v}^i \in \hat{V}^i\}$. Consequently, for any two eigenvectors $\hat{v}^e_i, \hat{v}^e_j \in \hat{V}^e$, we have $\hat{v}^{e\,T}_i \hat{v}^e_j = \hat{v}^{e\,T}_i C^e \hat{v}^e_j$. Accordingly, the normalization of the vectors in V^e and V^i can be computed by using only their preimages in $\hat{V}^e$ and $\hat{V}^i$ (Kulesza et al. 2012), by updating $\hat{v}^e_n \leftarrow \{\frac{\hat{v}^e_n}{\hat{v}^{e\,T}_n C^e \hat{v}^e_n}\}$ and $\hat{v}^i_n \leftarrow \{\frac{\hat{v}^i_n}{\hat{v}^{i\,T}_n C^i \hat{v}^i_n}\}$.

In the second phase, subset $\mathcal{S}$ is produced based on the selected eigenvectors. On each iteration of this second loop, the cardinality of $\mathcal{S}$ increases by one and the dimensionality of $\hat{V}^e$ and $\hat{V}^i$ is reduced by one. Here, e_j is the j-th standard basis vector, which is all zeros except for a one in the j-th position. During each iteration, EINSP selects pattern S_j according to the distribution below:

$$\begin{aligned} P(S_j) &= w_e \frac{1}{|V^e|} \sum_{\hat{v}^e \in \hat{V}^e} (v^{eT} e_j)^2 + w_i \frac{1}{|V^i|} \sum_{\hat{v}^i \in \hat{V}^i} (v^{iT} e_j)^2 \\ &= \sum_{d \in \{e,i\}} \sum_{\hat{v}^d \in \hat{V}^d} \frac{w_d}{|V^d|} q^2_d(S_j)(\hat{v}^{d\,T}_j \phi_d(S_j))^2. \end{aligned} \tag{5.14}$$

Algorithm 5.1 summarizes the working mechanism and process of the proposed EINSP method for *representative NSP discovery.*

5.3 Experiments and Evaluation

The empirical analysis of the proposed EINSP method in comparison with four baselines is undertaken on six real-life datasets and 17 synthetic datasets.

Algorithm 5.1 The Pseudo-code of EINSP for NSP Selection

```
**Input:** $\mathbb{S} = \{S_1, S_2, \ldots, S_N\}$, cardinality $k$.
**Output:** Representative NSP Subset $\mathcal{S}$
Map $\mathbb{S}$ to a directed graph $\mathcal{G}$ per the *NSP graph construction*
Construct dual representations $C^e$ and $C^i$, and compute their eigenvalue/eigenvector pairs $\{(\lambda^e_n, \hat{v}^e_n)\}_{n=1}^{N_v}$ and $\{(\lambda^i_n, \hat{v}^i_n)\}_{n=1}^{N_v}$ of $C^e$ and $C^i$, respectively
$J \leftarrow \varnothing$
**for** $n = 1, 2, \ldots, N_v$ **do**
    **if** $u \sim U[0,1] < \sum_{d\in\{e,i\}} w_d \lambda^d_n \frac{e^d_{k-1,n-1}}{e^d_{k,n}}$ **then**
        $J \leftarrow J \cup \{n\}$
        $k \leftarrow k - 1$
        **if** $k = 0$ **then**
            **break**
        **end if**
    **end if**
**end for**
$\hat{V}^e \leftarrow \{\frac{\hat{v}^e_n}{\hat{v^e_n}^T C^e \hat{v}^e_n}\}_{n\in J}$
$\hat{V}^i \leftarrow \{\frac{\hat{v}^i_n}{\hat{v^i_n}^T C^i \hat{v}^i_n}\}_{n\in J}$
$\mathcal{S} \leftarrow \varnothing$
**while** $V \neq \varnothing$ **do**
    Select $S_j$ from $\mathbb{S}$ with $P(S_j) = \sum_{d\in\{e,i\}} \sum_{\hat{v}^d \in \hat{V}^d} \frac{w_d}{|V^d|} q^2_d(S_j)(\hat{v^d_j}^T \phi_d(S_j))^2$
    $\mathcal{S} \leftarrow \mathcal{S} \cup S_j$
    $V_e \leftarrow V_{e,\perp}$, where $\{B^{eT} v_e | v_e \in V_{e,\perp}\}$ is an orthonormal basis for the subspace of $V_e$ orthogonal to $e_j$
    $V_i \leftarrow V_{i,\perp}$, where $\{B^{iT} v_i | v_i \in V_{i,\perp}\}$ is an orthonormal basis for the subspace of $V_i$ orthogonal to $e_j$
**end while**
```

Here, we first introduce the datasets and baseline methods and then evaluate the EINSP performance from multiple perspectives.

5.3.1 Experimental Setup

Datasets

We adopt the same six real-life datasets as those used in Chapter 4 to evaluate the efficiency of our proposed EINSP method against the baselines, which are summarized below for reference.

- Dataset 1 (DS1): A UCI dataset which consists of 989,818 anonymous ordered webpage visits to MSNBC.com (Cao et al. 2016).
- Dataset 2 (DS2): An application dataset of health insurance claim sequences (Cao et al. 2016), which averages 21 elements per sequence, averages 2 items per element, 5,269 data sequences, and 340 divergent items.
- Dataset 3 (DS3): A chain-store dataset containing 46,086 distinct items and 1,112,949 transactions (J. Pisharath n.d.).
- Dataset 4 (DS4): A KDD-CUP 2000 dataset from SPMF [1], which contains 59,601 e-commerce click-stream sequences, 497 distinct items, and averages 2.42 items per sequence with a standard deviation of 3.22.
- Dataset 5 (DS5): Another KDD-CUP 2000 dataset [2] with 77,512 click-stream sequences and 3,340 distinct items, averages 4.62 items per sequence with a standard deviation of 6.07.

[1] http://www.philippe-fournier-viger.com/spmf/datasets/BMS1_spmf

[2] http://www.philippe-fournier-viger.com/spmf/datasets/BMS2.txt

- Dataset 6 (DS6): A FIFA World Cup 98 dataset [3] with 20,450 click-stream sequences, 2,990 distinct items, and it averages 34.74 items per sequence with a standard deviation of 24.08.

Similar to the empirical analysis of Chapter 4, we also adopt 17 synthetic datasets which are generated by the IBM data generator (Agrawal & Srikant 1995) to evaluate the sensitivity of our proposed EINSP method and baselines with respect to different data factors (i.e., how the different characteristics of datasets influence the effectiveness of each method).

Baseline Methods

The following baselines are chosen to evaluate the effectiveness of our proposed EINSP method on the above real-life and synthetic datasets.

- Top-k selection (Top-k for short): A simple baseline to select the top-k patterns with the highest frequency, which evaluates each pattern only in terms of its support.

- SAPNSP (Liu et al. 2015*a*): The only method available for selecting a subset of the top-k patterns from an NSP collection regarding the highest contribution metric, which evaluates the importance of each pattern in terms of its frequency and intra-sequence correlation between its prefix and final element.

- k-means baseline (k-means): A diversity-oriented baseline to apply k-means clustering to the NSP collection by using the proposed explicit diversity to measure the distance between a pair of patterns and select the patterns with the highest frequency from each cluster to form

[3] `http://www.philippe-fournier-viger.com/spmf/datasets/FIFA.txt`

a k-size subset, which is somehow similar to the baseline adopted in (Gillenwater et al. 2012).

- k-SDPP baseline (k-SDPP): A variant of the EINSP method which selects a k-size representative subset by only modeling the cooccurrence-based explicit NSP relations.

- EINSP: The full model to select a k-size representative subset by jointly modeling both explicit and implicit NSP relations.

To the best of our knowledge, no existing work can discover the representative NSP subset by jointly considering the quality and diversity of the selected NSB patterns from the perspectives of both explicit and implicit behavior relations between NSP elements and patterns. Among the above baseline methods, Top-k selection and SAPNSP discover the subset by only considering the quality of a pattern, k-means discovers the subset only in terms of the diversity of the selected patterns, and k-SDPP discovers the subset only in terms of the explicit relations.

To illustrate the effectiveness of our proposed EINSP in discovering a high-quality and diverse subset, we conduct empirical comparisons between EINSP and the baselines from the perspectives of *pattern coverage*, *average pattern size*, and *average implicit relation strength* [4] of the selected subset in Sections 5.3.2, 5.3.3 and 5.3.4, respectively. In addition, a sensitivity comparison is conducted in Section 5.3.5 to evaluate the performance stability of EINSP and baselines on the datasets with different data factors.

In this research, NegGSP (Zheng et al. 2009) is adopted to discover NSP collection $\mathbb{S}$. In Sections 5.3.2, 5.3.3 and 5.3.4, threshold *min_sup* is set as

[4] The average of the experimental results, rather than the standard deviation, is adopted in this section because their difference is more obvious to evaluate the effectiveness.

10%, 20%, 1.5%, 1.5%, 1.5%, and 20% empirically for the six real-life datasets and 30% for all synthetic datasets, respectively.

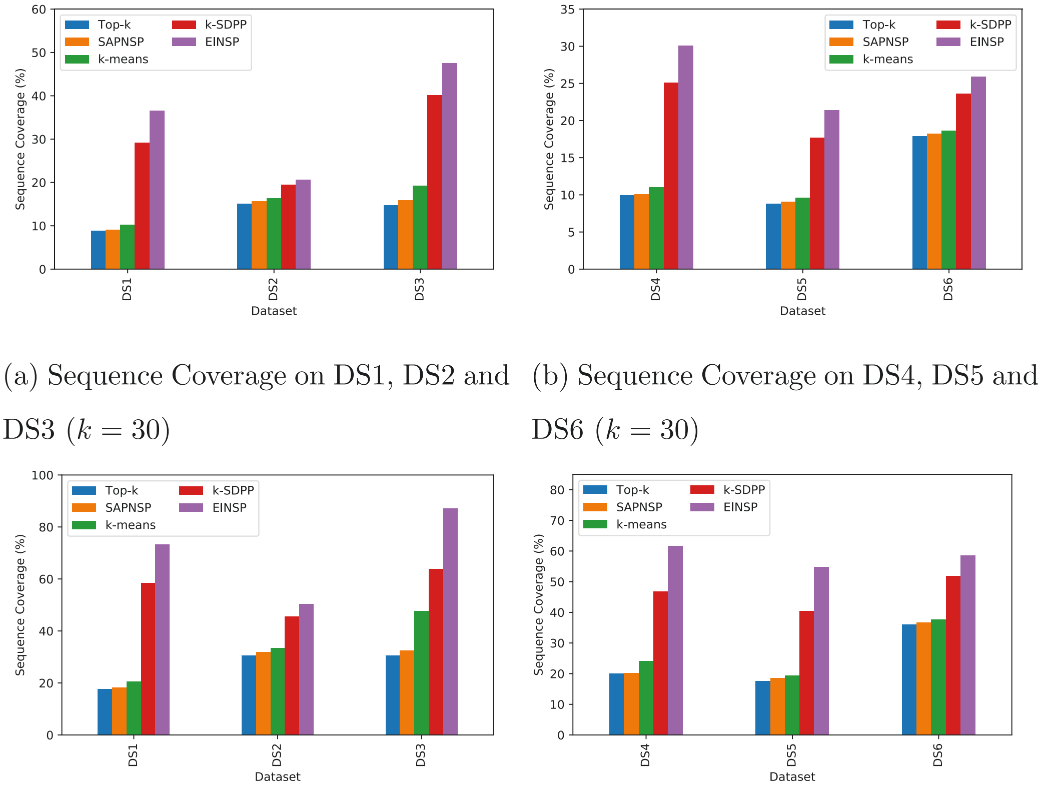
5.3.2 Pattern Coverage Comparison

The evaluation in terms of the representative degree of a selected subset (i.e., the *sequence coverage* and *item coverage* of the subsets), resulting from EINSP and baselines is given in Figures 5.2 and 5.3, and the *average frequency of the covered items* is presented in Figure 5.4. Here, the *sequence coverage* of subset $\mathcal{S}$ in dataset D, $C_s(\mathcal{S}|D)$, is defined as the ratio of the data sequences in dataset D which covers at least one NSB pattern in subset $\mathcal{S}$ with respect to the size of dataset D, that is, $C_s(\mathcal{S}|D) \equiv \frac{|\{S_d|S_d \in D \wedge (\exists S_p \in \mathcal{S} \ s.t. \ S_p \subseteq S_d)\}|}{|D|}$, where S_d is a data sequence from dataset D, and S_p is a pattern from the selected subset $\mathcal{S}$. The *sequence coverage* $C_s(\mathcal{S}|D)$ is always much lower than the sum of the frequency of the patterns in $\mathcal{S}$ (i.e., $C_s(\mathcal{S}|D) \ll \sum_{S_p \in \mathcal{S}} sup(S_p|D)$, where $sup(S_p|D)$ is the support (in percentage) of pattern S_p in dataset D), because the cover sets of different patterns may not be disjoint, that is, $cov(S_p|D) \bigcap cov(S_p'|D) \neq \varnothing$, where $S_p, S_p' \in \mathcal{S}$ and $cov(S_p|D) \equiv \{S_d|S_d \in D \wedge S_p \subseteq S_d\}$. An ideal representative subset is supposed to achieve a higher *sequence coverage* such that a larger proportion of the data sequences in the dataset can be covered by this small-scale subset.

In addition, the *item coverage* of a subset $\mathcal{S}$ in dataset D, $C_i(\mathcal{S}|D)$, is defined as the ratio of the items in dataset D covered by at least one pattern in subset $\mathcal{S}$ with respect to the whole item population, that is, $C_s(\mathcal{S}|D) \equiv \frac{|\{i|\exists S_d \in D, \ S_p \in \mathcal{S} \ s.t. \ i \subseteq S_d \wedge i \subseteq S_p\}|}{|I_D|}$, where i is an item and $I_D \equiv \{i|\exists S_d \in D \ s.t. \ i \subseteq S_d\}$ is the set of items covered by dataset D. Moreover, the frequency of item i in subset $\mathcal{S}$ refers to the ratio of its occurrence times in subset $\mathcal{S}$ with respect to the size of subset $\mathcal{S}$, and thus the *average frequency of the covered items* in subset $\mathcal{S}$ is defined as $AF(\mathcal{S}|D) \equiv \frac{1}{|I_{\mathcal{S}}|}\sum_{i \in I_{\mathcal{S}}} \frac{|\{S_p|S_p \in \mathcal{S} \wedge i \subseteq S_p\}|}{|\mathcal{S}|}$. Intuitively, a subset with a higher *item coverage* provides more comprehensive insight into the item information of the original sequence dataset, while a subset with a lower *average covered item frequency* tends to be more diverse and balanced.

Sequence Coverage Evaluation

Figure 5.2 illustrates the *sequence coverage* of our proposed EINSP and baselines on six real-life datasets. The results indicate that among these methods, Top-k selection performs worst as it assumes that the cover sets of its



(a) Sequence Coverage on DS1, DS2 and DS3 ($k = 30$)

(b) Sequence Coverage on DS4, DS5 and DS6 ($k = 30$)

(c) Sequence coverage on DS1, DS2 and DS3 ($k = 150$)

(d) Sequence coverage on DS4, DS5 and DS6 ($k = 150$)

Figure 5.2: Sequence Coverage Evaluation on Datasets DS1-DS8.

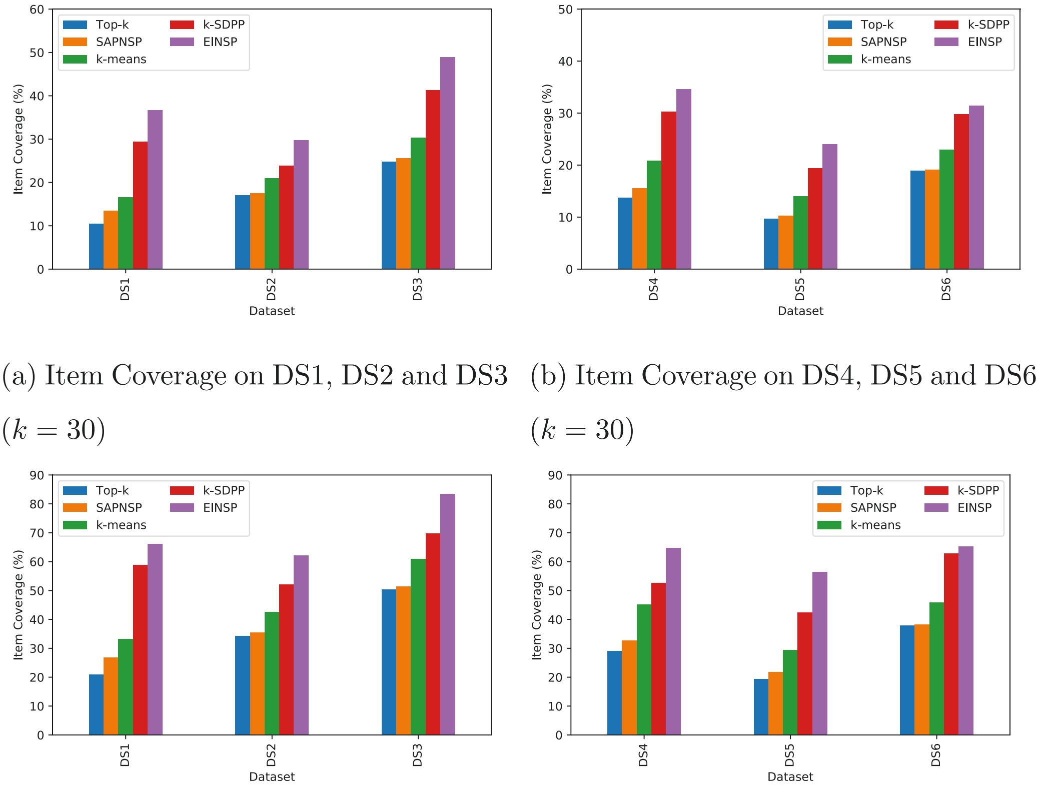
selected patterns are disjoint and completely neglects the diversity of the selected subset which is always inconsistent with the real-world cases. This results in the fact that Top-k selection achieves a relatively lower *sequence coverage* on the datasets with a smaller average length, such as DS1 and DS4, since a larger proportion of data sequences can cover multiple short-size and high-frequency patterns while the NSPs with relatively rare entities (items or elements) are ignored. Compared with Top-k selection, SAPNSP achieves slightly better *sequence coverage.* In addition to considering the frequency of each pattern, SAPNSP also considers the internal correlation among the elements as interestingness and thus allows some long-size patterns to achieve a higher contribution metric and enable the selected subset to have relatively better diversity. However, the superiority of SAPNSP over Top-k selection is limited because it assumes the contribution of the selected patterns are independent of each other and ignores the diversity of the selected subset in the design of contribution metric. Top-k selection and SAPNSP lag behind the k-means method because it selects the pattern from each cluster of the NSP collection and guarantees the cover set of selected patterns to share a much smaller overlap. Accordingly, the k-means method achieves higher *sequence coverage* on sparse data; by increasing the k value, its superiority becomes more obvious. For example, DS3 is a sparse dataset with 46,084 distinct items; the *sequence coverage* of the k-means method on DS3 makes improvement of 30% over Top-k selection and more than 20% over SAPNSP with $k = 30$, and around 55% over Top-k selection and about 46% over SAPNSP with $k = 150$, which proves the importance of diversity modeling to discover a representative NSP subset. However, the k-means method is weak in jointly modeling the quality and diversity of each pattern and cannot capture the underlying implicit relations between patterns.

The experimental results indicate that our proposed EINSP and its simplified k-SDPP version significantly beat all the other baselines by clear margins in terms of *sequence coverage*. Particularly, EINSP achieves much higher *sequence coverage* compared with the baselines by maximally 315%, 36.2%, 224%, 204%, 144%, and 44.9% as well as an average of 225%, 24.7%, 148%, 149%, 106%, and 34.1% with respect to these datasets on $k = 30$. The performance improvement of EINSP is less obvious on datasets DS2 and DS6. However, when the value of k increases to 150, the improvement on these datasets rises to maximally 64.4% and 63.1% as well as averagely 46.1% and 47.9%. Overall, both EINSP and its k-SDPP edition demonstrate strong superiority over all baselines on six datasets, which suggests the effectiveness of the proposed design for *representative NSP discovery*.

Furthermore, compared with the k-SDPP edition, which only models explicit behavior relations, EINSP jointly models the compound explicit and implicit relations and thus contributes to an additional average of 25.4%, 8.00%, 27.4%, 25.9%, 28.3%, and 11.4% performance improvement on these datasets, respectively over the k-SDPP. Moreover, compared with the k-means method, which only evaluates the importance of each pattern of each cluster in terms of its frequency, k-SDPP jointly considers each pattern's quality and diversity through its DPP-based design and thus contributes to additional average performance improvements of 186%, 27.9%, 71.7%, 111%, 96.0%, and 32.5% on the six datasets over the k-means baseline. The above analysis illustrates that different parts of the proposed EINSP can greatly improve the performance of representative NSP discovery in terms of the *sequence coverage*. Lastly, both EINSP and its k-SDPP edition achieve a better performance and a higher coverage superiority on a higher k, which indicates that EINSP and k-SDPP are scalable with the increase of parameter k.

Item Coverage Evaluation

Figure 5.3 illustrates the *item coverage* of the subsets selected by EINSP and baselines on real-life datasets. The results reveal that the Top-k selection and SAPNSP perform the worst with respect to *item coverage* on all datasets and k values, because they always tend to select the short-size patterns consisting of only high-frequency items but ignore those rarely observed items. In contrast, EINSP always achieves higher *item coverage* than all the baselines on six datasets, and its selected subset covers the patterns with more distinct



(a) Item Coverage on DS1, DS2 and DS3 ($k = 30$)

(b) Item Coverage on DS4, DS5 and DS6 ($k = 30$)

(c) Item Coverage on DS1, DS2 and DS3 ($k = 150$)

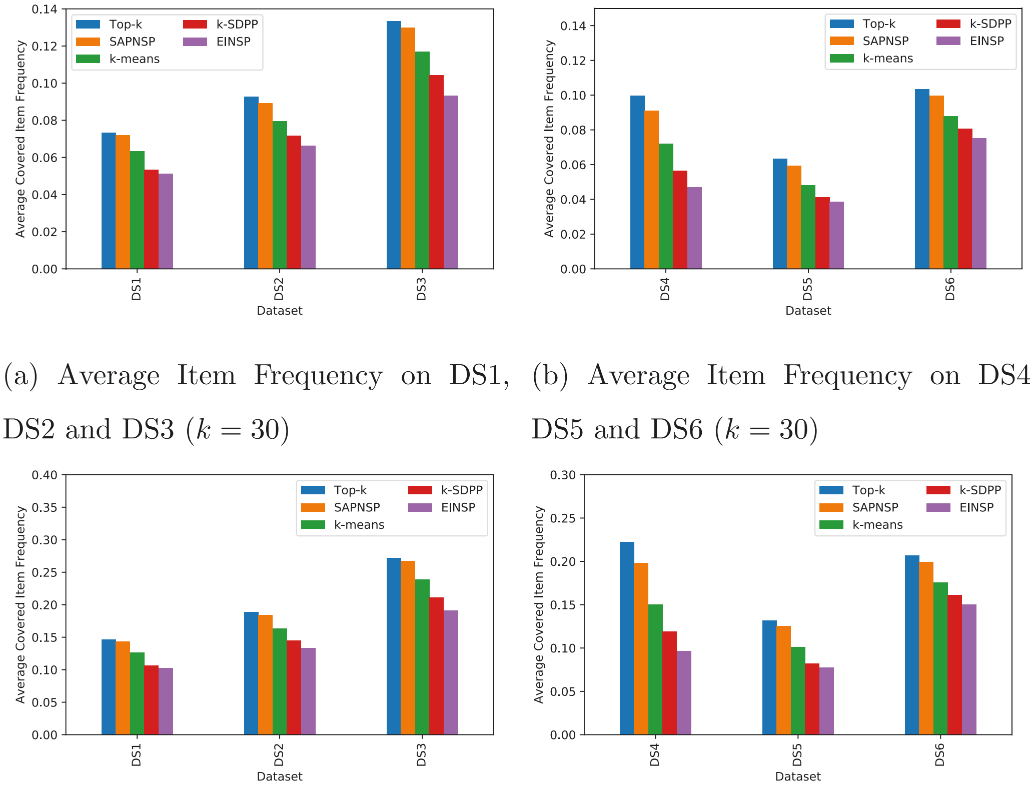
(d) Item Coverage on DS4, DS5 and DS6 ($k = 150$)

Figure 5.3: Item Coverage Evaluation on Datasets DS1-DS8.

items. In addition, EINSP always outperforms its simplified k-SDPP version by contributing an average of 18.6%, 22.1%, 19.0%, 18.6%, 28.4%, and 4.50% more with respect to the *item coverage* on these datasets. This is owing to the introduction of the implicit relations that drives EINSP towards the patterns with low-frequency item combinations. Moreover, with the help of the joint consideration of the quality and diversity of each pattern, k-SDPP beats the k-means method by an average of 77.6%, 17.9%, 25.3%, 31.0%, 41.8%, and 33.4% on the six datasets. The above observation demonstrates that the proposed method of capturing both explicit and implicit relations can greatly increase the coverage of the selected subset for more items, especially for the low-frequency items which are discarded by the baselines.

Averagely Covered Item Frequency Evaluation

The comparison of the *averagely covered item frequency* by EINSP and baselines on datasets is illustrated in Figure 5.4. The average frequency of the covered items resulting from the Top-k selection and SAPNSP is always much higher than the other three methods since they tend to select the patterns combining a smaller number of high-frequency items, and thus the selected subsets tend to achieve a relatively higher item frequency. In comparison, the methods considering the diversity of the selected subset maintain a lower *averagely covered item frequency* and thus preserve the information carried by the rarely observed items. In particular, the *averagely covered item frequency* of EINSP is only 69.8%, 71.1%, 70.1%, 45.3%, 59.8%, and 72.8% on datasets DS1-DS6 over Top-k selection and 81.1%, 82.6%, 79.8%, 64.8%, 78.5%, and 85.6% on these datasets over the k-means method. These results indicate that the patterns selected by our proposed EINSP are more balanced in terms of the covered items.



(a) Average Item Frequency on DS1, DS2 and DS3 ($k = 30$)

(b) Average Item Frequency on DS4, DS5 and DS6 ($k = 30$)

(c) Average Item Frequency on DS1, DS2 and DS3 ($k = 150$)

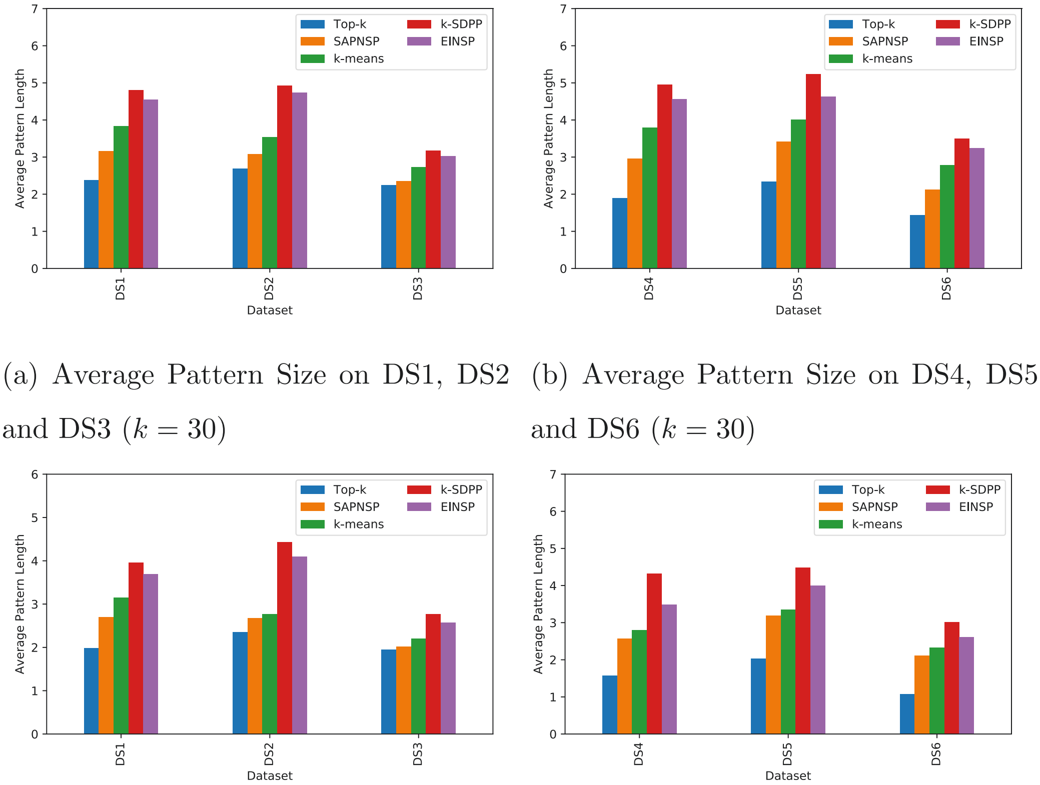
(d) Average Item Frequency on DS4, DS5 and DS6 ($k = 150$)

Figure 5.4: Averagely Covered Item Frequency Evaluation on Datasets DS1-DS8.

Combining the experimental findings in Figures 5.3 and 5.4, it is clear that the subset resulting from EINSP not only covers a larger proportion of distinct items but also contains a higher proportion of the patterns consisting of the relatively low-frequency items, that is, the selected patterns are more balanced to represent the item information of the original sequence dataset. Hence, compared with other baselines, the subset discovered by our proposed EINSP can not only cover a larger proportion of the data sequences in the dataset but also reserve more balanced information carried by distinct items because of the proposed *representative NSP discovery* design, which jointly models the explicit and implicit relations in terms of both pattern quality and diversity.

5.3.3 Average Pattern Size Comparison

The *average pattern size* of the subsets selected by different methods is a metric to evaluate the subset quality because longer-size patterns disclose higher long-range dependencies among elements. However, long-size pat-



(a) Average Pattern Size on DS1, DS2 and DS3 ($k = 30$)

(b) Average Pattern Size on DS4, DS5 and DS6 ($k = 30$)

(c) Average Pattern Size on DS1, DS2 and DS3 ($k = 150$)

(d) Average Pattern Size on DS4, DS5 and DS6 ($k = 150$)

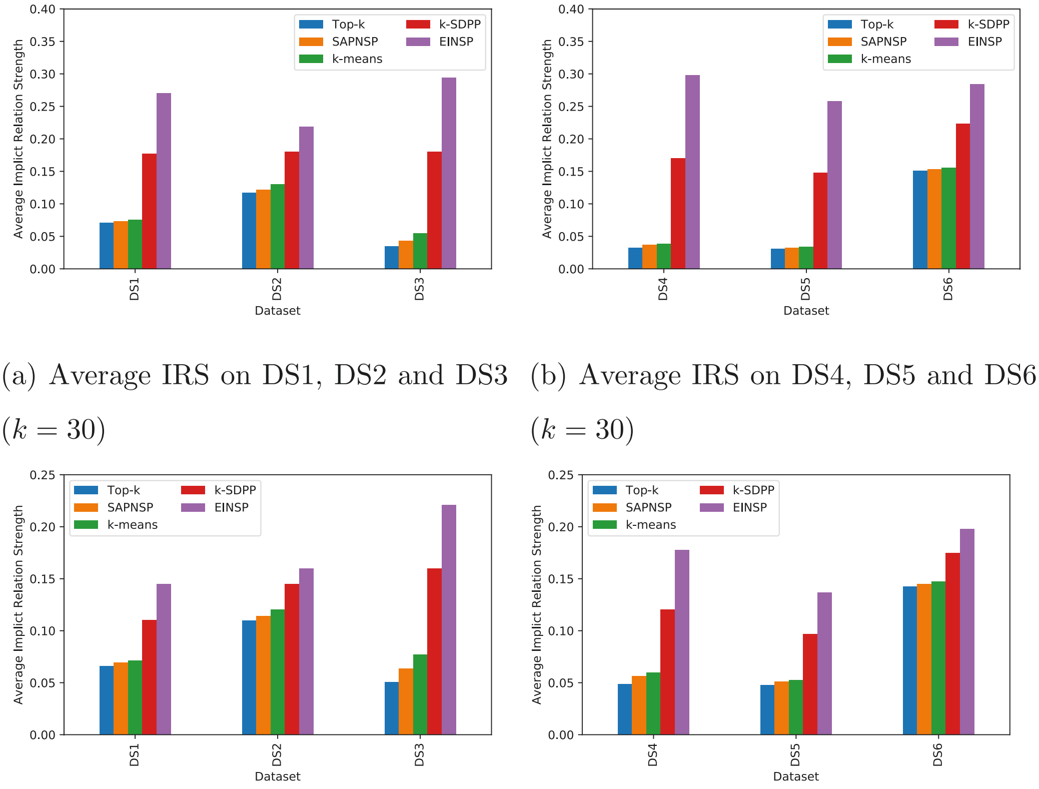
Figure 5.5: Average Pattern Size Comparison on Datasets DS1-DS8.

terns are more likely discarded because of their low frequency. Figure 5.5 illustrates the *average pattern size* of EINSP and baselines on six real-life datasets. First, it is noted that the *average pattern size* of Top-k selection is always highly smaller than the other methods, and only a small number of its selected patterns contain more than three elements. This is because short-size patterns usually have higher frequencies, which is consistent with the results of the item coverage evaluation in Section 5.3.2. In addition, SAPNSP achieves a slightly higher *average pattern size* than Top-k selection because the interestingness term of SAPNSP's contribution metric partly favors the high-frequency but short-size patterns. However, the downward property of this contribution metric indicates that an NSB always achieves a higher contribution value than any of its super-sequences, leading to short-size patterns being more likely to be selected. Accordingly, the average size of the patterns selected by SAPNSP is always clearly shorter than that of k-SDPP and EINSP, suggesting that it is unrealistic to discover representative NSPs by only a single metric while retaining long-size NSB patterns, and it is necessary to consider subset diversity. Note that the k-means method achieves a higher *average pattern size* than the above two methods because it groups patterns into k clusters with respect to their diversity and thus allows the occurrences of the clusters composed of long-size patterns. However, the k-means method only selects the highest-frequency pattern in each cluster, which tends to be the shortest in the cluster it belongs to.

Contrary to the k-means method, the k-SDPP method adopts the proposed explicit quality measure to evaluate the importance of a pattern, which tends to assign a high quality to a relatively longer-size pattern as shown in Eq. (5.3). Consequently, more long-size patterns are likely selected, contributing to the largest *average pattern size*. Accordingly, the *average pattern* *size* of k-SDPP selections reveals an improvement of averagely 49.2%, 62.8%, 36.2%, 67.6%, 47.0%, and 53.9% over SAPNSP and 25.5%, 49.6%, 21.1%, 42.2%, 32.1%, and 27.8% over the k-means on these datasets. Compared with k-SDPP, EINSP achieves a relatively smaller *average pattern size* because its implicit quality measure tends to assign a lower IRS to a longer-size pattern, as discussed in (Wang & Cao 2017). Compared with its baselines, EINSP contributes to an average of 40.2%, 53.5%, 27.9%, 44.9%, 30.3%, and 38.0% over SAPNSP and 17.8%, 40.9%, 13.7%, 22.3%, 17.2%, and 14.3% over the k-means with respect to *average pattern size*. Lastly, as seen in Section 5.3.4, the introduction of implicit relations further improves the *average implicit relation strength* of the selected subset at the expense of a smaller *average pattern size*.

5.3.4 Average Implicit Relation Strength Comparison

As discussed in Section 5.1, some relatively low-frequency but highly implicitly-related patterns may reveal complex yet insightful information and can be useful for business applications. Accordingly, the *average implicit relation strength* is a metric to measure the quality of the subsets selected by different methods, which is illustrated in Figure 5.6. It demonstrates that three baselines, Top-k selection, SAPNSP and k-means always achieve similar but low *average implicit relation strength* because they do not consider the implicit relations among patterns. Compared with the above three baselines, k-SDPP selection achieves a higher *average implicit relation strength* by maximally 148.8%, 418.7%, 419.2%, and 375.5% for $k = 30$, and an average of 141.2%, 320.0%, 374.4%, and 354.2% for $k = 150$ on datasets DS3, DS5, DS6, and DS7, respectively. Accordingly, k-SDPP selection shows an improvement on *average implicit relation strength* by maximally 66.4%, 214.6%, 148.0% and



(a) Average IRS on DS1, DS2 and DS3 ($k = 30$)

(b) Average IRS on DS4, DS5 and DS6 ($k = 30$)

(c) Average IRS on DS1, DS2 and DS3 ($k = 150$)

(d) Average IRS on DS4, DS5 and DS6 ($k = 150$)

Figure 5.6: Comparison of Average Implicit Relation Strength (IRS) on Datasets DS1-DS8.

101.9% and averagely 60.1%, 158.0%, 120.7% and 92.0% on these datasets.

Benefiting from jointly modeling the explicit and implicit relations, EINSP achieves a much higher *average implicit relation strength* than k-SDPP and contributes to additional performance improvements of 52.8%, 21.1%, 63.0%, 75.3%, 74.4%, and 27.3% on these six datasets with $k = 30$ and 31.4%, 10.1%, 38.1%, 47.4%, 41.2%, and 13.1% with $k = 150$. This demonstrates that the modeling of implicit quality and diversity in EINSP contributes to a highly implicitly-related subset. Generally speaking, both k-SDPP and

EINSP achieve better performance for a larger subset size k in terms of the *average implicit relation strength* on all datasets but perform better over other baselines for a smaller subset size.

In summary, the significantly bigger *pattern coverage*, larger *average pattern size*, and higher *average implicit relation strength* of our proposed EINSP compared with all baselines demonstrate the effectiveness and contribution of jointly modeling explicit and implicit element/pattern relations to the discovery of a representative subset composed of high-quality and diverse patterns. Further, in Section 5.3.5, we validate the scalability of the proposed EINSP versus that of baselines on the multiple synthetic datasets with different data factors.

5.3.5 Sensitivity Evaluation

As revealed in Sections 5.3.2, 5.3.3, and 5.3.4, a *representative NSP discovery* method may be sensitive to data characteristics, and a reliable method is supposed to maintain its performance superiority on the datasets with respect to different data factors (Cao et al. 2016). Accordingly, we demonstrate the influence of data factors C, T, DB and N on *sequence coverage* of EINSP and baselines on size $k = 150$, where the sensitivity evaluation is presented in Table 5.1.

The evaluation indicates that the *sequence coverage* of our proposed EINSP is always significantly higher than that of other baselines under different data factors, which is consistent with the results of the other comparisons. In addition, the *sequence coverage* of EINSP is highly impacted by the data factors C, T, and N, and EINSP works better on the datasets with higher C, higher T, and lower N. With the increase of factor C, more long-size sequences are generated which more likely cover the long-size pat-

Table 5.1: Sequence Coverage Sensitivity of EINSP and Baselines w.r.t. Data Factors for $k = 150$

Factors	Dataset Name	Top-k	SAPNSP	k-means	k-SDPP	EINSP
C	**C6**_T6_S8_I8_DB10k_N0.1k	22.80%	24.25%	25.74%	35.41%	43.77%
	C8_T6_S8_I8_DB10k_N0.1k	26.95%	28.10%	29.81%	40.98%	49.25%
	C10_T6_S8_I8_DB10k_N0.1k	28.67%	29.74%	31.05%	42.04%	52.09%
	C12_T6_S8_I8_DB10k_N0.1k	33.66%	34.74%	36.95%	49.97%	62.33%
	C14_T6_S8_I8_DB10k_N0.1k	44.71%	46.34%	48.16%	63.97%	77.61%
T	C10_**T4**_S8_I8_DB10k_N0.1k	24.53%	25.39%	26.67%	35.96%	44.04%
	C10_**T6**_S8_I8_DB10k_N0.1k	28.67%	29.74%	31.05%	42.04%	52.09%
	C10_**T8**_S8_I8_DB10k_N0.1k	37.50%	38.66%	39.85%	53.68%	64.68%
	C10_**T10**_S8_I8_DB10k_N0.1k	39.18%	40.34%	42.29%	57.30%	70.94%
	C10_**T12**_S8_I8_DB10k_N0.1k	51.39%	52.54%	54.49%	73.01%	85.40%
DB	C10_T6_S8_I8_**DB10k**_N0.1k	28.67%	29.74%	31.05%	42.04%	52.09%
	C10_T6_S8_I8_**DB20k**_N0.1k	30.19%	29.07%	30.95%	41.74%	51.93%
	C10_T6_S8_I8_**DB30k**_N0.1k	30.21%	29.94%	31.08%	41.93%	52.36%
	C10_T6_S8_I8_**DB40k**_N0.1k	30.25%	29.40%	31.10%	41.47%	52.12%
	C10_T6_S8_I8_**DB50k**_N0.1k	30.26%	29.06%	31.12%	41.46%	51.73%
N	C10_T6_S8_I8_DB10k_**N0.1k**	28.67%	29.74%	31.05%	42.04%	52.09%
	C10_T6_S8_I8_DB10k_**N0.2k**	23.57%	24.94%	25.85%	33.71%	45.74%
	C10_T6_S8_I8_DB10k_**N0.3k**	22.54%	23.24%	24.10%	32.79%	41.87%
	C10_T6_S8_I8_DB10k_**N0.4k**	21.46%	22.77%	24.59%	32.84%	39.50%
	C10_T6_S8_I8_DB10k_**N0.5k**	17.46%	18.39%	19.28%	26.60%	34.06%

terns selected by EINSP, as discussed in Section 5.3.3. Moreover, a higher C leads to a larger number of high-frequency elements being generated, and the corresponding diversity vector in EINSP contributes to a more diverse subset and makes EINSP more effective. On $k = 150$, when C rises from 6 to 14, the *sequence coverage* of EINSP grows by 77.3% and outperforms other baselines by an average of 55.8% and maximally 73.6%. Furthermore, when factor T grows, the dataset becomes denser and thus each pattern selected by EINSP can cover more data sequences. For example, when T rises from 4 to 12 on $k = 150$, the *sequence coverage* of EINSP increases by 85.4% and is 17.0% higher than that of k-SDPP, which is the second-best method among the baselines. In addition, it reveals that the growth of factor DB makes limited impact on all methods because it does not change the distribution of the sequence dataset. Lastly, EINSP achieves a lower *sequence coverage* on the datasets with a higher N. This is because, with the increase of factor N, the dataset becomes sparser and thus a smaller proportion of its sequences is covered by a small-scale subset. Additionally, less frequent elements are available in a sparse dataset, which makes the diversity vector less effective.

5.4 Summary

Pattern mining-based NSB analytics, such as NSP mining, has been rarely studied, while it is proven to play a strong role in discovering significant occurring and nonoccurring behaviors. There are many significant theoretical and practical challenges in the research on NSP mining, such as lack of theoretical foundation, complicated combinatorial scenarios, nonoccurring relations, and extreme computational cost. This chapter addresses one of them, discovering the representative negative sequential patterns that are of high quality and diversity to make the identified patterns more actionable. We propose a novel DPP-based approach EINSP, which jointly models both explicit and implicit relations in positive and negative sequential elements and patterns in terms of the cooccurring and nonoccurring probabilistic distributions over all possible subsets in the pattern cohort, and further selects a representative subset composed of high-quality and diverse patterns. Exhaustive empirical analysis demonstrates the strong potential to discover more representative and informative patterns with wider coverage, greater diversity, and better quality in terms of different data factors.

EINSP opens a new direction for the pattern analysis and the selection of the nonoccurring sequential behaviors, that is, pattern selection by analyzing pattern relations and element relations. However, more theoretical work is required to explore the comprehensive and structural behavior relations into the NSB analytics. As discussed in Chapter 1, the NSPs discovered by the pattern mining-based NSB analytics methods only focus on the global information and knowledge carried by the entire NSB but fail to capture the complicated relations between the behaviors within each NSB, especially for the intra-/inter-element behavior relations inside an NSB, which is of great significance for the prediction of future sequential behaviors. The above drawback makes the discovered NSPs less practical in some prediction-oriented applications. Thus, in the next chapter, we address the significance and modeling of both intra-/inter-element relations (also called intra-/inter-basket relations in the field of recommender systems) and positive/negative feedback for interactive sequential basket recommendation.

Chapter 6

Interactive Sequential Basket Recommendation

6.1 Introduction

As discussed in Chapters 4 and 5, pattern mining-based NSB analytics targets the discovery of NSB patterns with discriminating global significance (e.g., the patterns with high frequency or high quality and diversity), but fails to capture the complicated behavior relations inside each NSB, which is significant in distinguishing the dominant behaviors or behavior combinations and making predictions for future behaviors. Accordingly, the NSB patterns discovered are less practical in prediction-oriented applications, and in this chapter we explore the compound behavior relations to enable the sequential behavior prediction.

In recent years, the research on recommendation systems (RSs) is most relevant to our focus on behavior prediction and thus forms the foundation of our work, which has paid increasing attention to analyzing and computing user's sequential behaviors (Cao 2010, Cao et al. 2015) such as sequences of purchasing baskets in retail and online businesses and sequential clicking behavior in interactive mobile apps. Such recommendation applications involve continuous interactions between users and service platforms (e.g., a mobile app or an e-commerce website) and generate sequences of corresponding user preferences, behaviors, feedback, and behavior predictions (also called recommendations) which are all coupled with each other.

6.1.1 Problem Statement

As illustrated in Figure 6.1, an interactive sequential RS for the above applications on behavior prediction may work as follows. At each time point (or period), a basket of items (e.g., a group of purchasing behaviors or services) is consumed by a user. The RS analyzes the consumed items, user preferences, and other relevant information to suggest future items for the next basket that the user may likely consume. During the next consumption, the user may give feedback on the predicted or recommended items, such as selecting some of them for the next basket but non-selecting others (which we call positive and negative feedback, respectively, referring to occurring and nonoccurring future behaviors). Here, negative feedback is reflected by those negative items (Cao et al. 2016) which are recommended by a RS but skipped or ignored by a user, which reflect underlying user preference and selection bias (Zhao, Zhang, Ding, Xia, Tang & Yin 2018). In this research, we incorporate such nonoccurring but important (Cao et al. 2015) items as negative feedback into recommendation consideration to consider the feedback from both missing and selected items on user preference and behaviors. The RS further analyzes the consumption and interaction (e.g., feedback) data and makes recommendations for the next future basket. This interactive consumption and recommendation iterates a trial-and-error recommen-

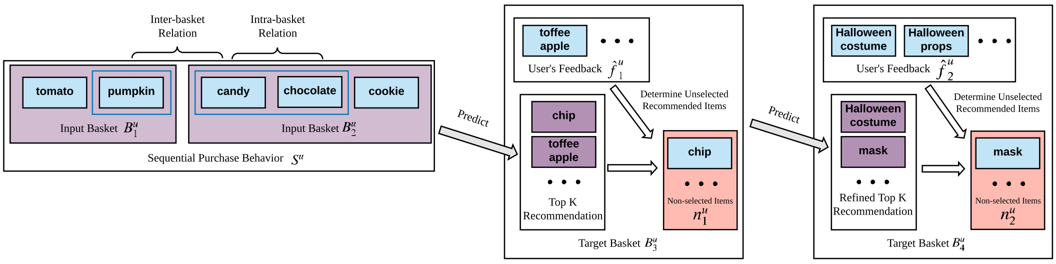


Figure 6.1: An Example of Interactive Sequential Basket Recommendation.

dation process, forming sequential baskets, user behaviors, user preferences, and positive/negative feedback, and continuously captures user feedback and updates recommendations (Cao 2015, Hariri et al. 2015). Such interactive sequential recommender systems suggest a basket of items one at a time; we thus call them *interactive sequential basket recommender systems (iSBRSs)* or broadly *sequential basket recommender systems (SBRSs)*. The SBRSs aim to continuously recommend baskets of items one after another in an interactive way by closely modeling the sequential baskets, user preferences, feedback, and dynamics during the sequential interactions between an RS and users. The SBRSs support sequential recommendations in a continuous and interactive manner, which continues monitoring user sequential behaviors and preferences and reflecting on their impact on the next basket. Such SBRSs can consecutively analyze user preference and behavior and make continuous predictions about the interactions between users and RSs for the aforementioned applications. The SBRSs are also widely and increasingly demanded in many other domains such as health and medical treatments, sales and marketing services, and customer relationship management.

Figure 6.1 illustrates the above SBRS problem and the process of interactive sequential basket recommendation: given a user's purchase sequence $<(tomato, pumpkin), (candy, chocolate, cookie)>$ which consists of two consecutive baskets (i.e., two consecutive elements) $B_1^u :< (tomato, pumpkin) >$ and $B_2^u :< (candy, chocolate, cookie) >$, an RS suggests $\{chip, toffee\ apple\}$ as the next basket B_3^u to the user. Once the user receives the recommendations, if he/she selects item $toffee\ apple$ (i.e., selected items) but non-selects $chip$ (i.e., a non-selected item), this selection/non-selection feedback may indicate that the user likely intends to celebrate Halloween instead of having an ordinary party. Accordingly, the RS may recommend $Halloween\ costume$ and $mask$ as the next basket, B_4^u, based on basket B_3^u and user feedback. Figure 6.1 also indicates some intrinsic characteristics and complexities in modeling interactive sequential basket recommendation. (1) SBRSs involve sequences of baskets where each basket is composed of items selected for a shopping basket. There are hierarchical couplings (Cao et al. 2012, Cao 2015) between a user's selected items: the items in a basket are coupled (i.e., intra-basket relations) and the items across baskets may also be more or less coupled (i.e., inter-basket relations). Such intra-/inter-basket relations reflect that items both within and between baskets may be tightly or loosely coupled for various reasons, determining the item relevance to a basket and which items may appear in the next basket and the one after. (2) There are sequential interactions between an RS and users when sequences of positive (recommended items occurred in the next basket) and negative (recommended ones nonoccurred (Cao et al. 2015) in the next basket) feedback are given by users. Both positive and negative feedback and their sequences are important to understand user-item and user-basket relations (Cao 2016, Zhao, Zhang, Ding, Xia, Tang & Yin 2018), as well as user like/dislike preferences, selection/non-selection behaviors, and their evolution along the interactions. These aspects and characteristics of SBRSs form the nature of the SBRS problem, and thus should be considered in designing SBRSs.

Since NBRSs (Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016, Gatzioura & Sànchez-Marrè 2015, Wang, Guo & Lan 2014, Christidis, Apostolou & Mentzas 2010, Lee et al. 2005, Adomavicius & Tuzhilin 2005) are mostly relevant to our target problem on SBRSs in this research, we further illustrate their limitations and differences from SBRSs below in terms of the example in Figure 6.1. NBRS makes predictions for a basket of items that a user may likely consume during the next shopping opportunity by modeling items selected in previous baskets and user preferences. Next-basket is recommended based on its item relations to the previous adjacent baskets (Yu et al. 2016). NBRSs can be treated as a special case of SBRSs, where only the next-basket is predicted based on what appeared in previous baskets. SBRSs instead sequentially predict the next basket one after another by successively modeling the sequential interactions between an RS and users.

For example, given two consecutive baskets $B_1^u :< (tomato, pumpkin) >$ and $B_2^u :< (candy,\ chocolate, cookie) >$ in Figure 6.1, since the existing NBRS methods focus on the behavior relations between all the items in two adjacent baskets but pay more to all attention to the most recent one, an NBRS may suggest some snacks like *chip* (probably for a party) as all the items are food-relevant especially given that the latter basket consists of only snacks. However, if we consider the intra-/inter-basket behavior relations among these items and identify some important cross-basket item combinations (e.g., sub-sequence $< pumpkin, (candy, chocolate) >$ as the user may be preparing for Halloween), then an SBRS would more likely recommend items like $toffee\ apple$ and form a $\{chip, toffee\ apple\}$ recommendation as basket B_3^u. Once recommended, if the user selects items $toffee\ apple$ (i.e., positive feedback) but non-selects *chip* (i.e., negative feedback), then this could further confirm the guess that the user may intend to celebrate Halloween instead of organizing an ordinary party. Accordingly, $Halloween\ costume$ and $mask$ may be further recommended in the next basket, B_4^u, based on the positive/negative feedback on basket B_3^u. In this example, the items *candy* and *chocolate* are coupled in the basket B_2^u as an intra-basket item relation, and the two sub-sequences *pumpkin* and $(candy, chocolate)$ are coupled as an inter-basket relation between baskets B_1^u and B_2^u.

The above analysis also illustrates how an SBRS is more realistic and practical than an NBRS since SBRSs continuously recommend a sequence of next baskets by considering the hierarchical behavior relations among items and the positive/negative feedback in a recommendation-feedback interactive and iterative way. An SBRS can continuously capture and update user preferences and selection behaviors in continuous sessions. In contrast, NBRSs (Rendle et al. 2010, Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016) and sequential recommendation (Liu, Wu, Wang, Li & Wang 2016, Chen, Xu, Zhang, Tang, Cao, Qin & Zha 2018, Ying, Zhuang, Zhang, Liu, Xu, Xie, Xiong & Wu 2018) only suggest the items for the next timestamp. While one could repeat NBRSs for continuous next-basket predictions by involving multiple-step sequential basket behaviors, each next-basket prediction is independent of the following basket (hence the intra-/inter-basket relations are overlooked), and the repeated NBRS does not involve the behavior relations between next baskets, the RS-user interactions, and both positive and negative feedback. As demonstrated in the relevant research (Cao 2016, Cao 2015, Wang, Cao & Chi 2015, Zhao, Zhang, Ding, Xia, Tang & Yin 2018), continuously incorporating hierarchical behavior relations among items, interactions, and interactive feedback not only characterizes the intricate SBRS problem but also improves learning performance. In addition, if SBRS only makes predictions for basket B_3^u but ignores sequential refinement for B_4^u, it degrades to a standard NBRS problem. In summary, this SBRS problem is more consistent with the prediction-oriented NSB analytics compared with existing RS problem, which is to be addressed in this chapter.

6.1.2 Design and Contributions

To address the intricate SBRS problem, this research proposes a hierarchical attentive encoder-decoder model (HAEM) for interactive sequential basket recommendation. Our proposed HAEM consists of three components: a basket encoder, a sequence encoder, and a prediction decoder. The basket encoder builds a compound basket representation over all the items in a basket by exploring multiple intra-basket relations between items. The sequence encoder maps the sequence of the built basket representations into a sequence of annotation representations by exploring the inter-basket relations between baskets. The prediction decoder makes predictions for target baskets by modeling both basket context representations and positive/negative (i.e., selection/non-selection) feedback and makes non-selection refinement on future basket predictions. Specifically, we incorporate the factorization machine (FM) mechanism (Rendle 2010, Rendle, Gantner, Freudenthaler & Schmidt-Thieme 2011) into the basket encoder to model both linear and pairwise intra-basket item couplings within a basket and further introduce the attention mechanism to the basket and sequence encoders to recognize and pay more attention to those highly relevant items and significant baskets. To the best of our knowledge, this is the first effort made to comprehensively explore the characteristics of interactive sequential basket recommendation, jointly model the intra-/inter-basket relations in sequential user basket behaviors, and incorporate the positive/negative feedback to enable negative feedback-based refinement for interactive sequential basket recommendations.

Empirical analysis indicates HAEM outperforms state-of-the-art NBRSs and session-based baselines on two real-life datasets on both prediction accuracy and novelty. The negative feedback-based refinement mechanism further improves the prediction quality in interactive sequential recommendations.

6.2 The HAEM Model

Here, we first introduce the main working process of our proposed HAEM, and then introduce each component. Lastly, we show how to train the HAEM and learn the parameters, as well as how to make sequential basket recommendation with iterative refinement by the learned model. The notations involved in this research are listed in Tables 6.1 and 6.2.

Table 6.1: Notations in Sequential Basket Recommendation (Part I)

Notation	**Description**
U	A set of users, i.e., $U = \{u_1, u_2, \ldots, u_{\|U\|}\}$
I	A set of items, i.e., $I = \{i_1, i_2, \ldots, i_{\|I\|}\}$
B_t^u	The basket of items consumed by user u at time t
S^u	The sequential behaviors of user u, i.e., $S^u := < B_1^u, B_2^u, \ldots, B_{t_u}^u >$
$<_{u,t}$	The personalized ranking of basket B_t^u for user u at time t
$\boldsymbol{o}_t^u$	The preference score of user u at time t for the prediction of basket B_t^u
$y_a^{u,t}$	The intra-basket relation between items in basket B_t^u
$\boldsymbol{h}_j^t \in \mathbb{R}^H$	The embedding representation vector of item $i_j^t \in B_t^u$
w_j	The weight of item i_j^t w.r.t. the intra-basket relation $y_a^{u,t}$
$\boldsymbol{v}_j^t \in \mathbb{R}^F$	The latent feature vector of item i_j^t
$\langle \boldsymbol{v}_j^t, \boldsymbol{v}_k^t \rangle$	The inner-product of latent vectors $\boldsymbol{v}_j^t$ and $\boldsymbol{v}_k^t$, i.e., the factorized interactions between items i_j^t and i_k^t
$\boldsymbol{h}_j^t \odot \boldsymbol{h}_k^t$	The element-wise product of two embedding vectors $\boldsymbol{h}_j^t$ and $\boldsymbol{h}_k^t$
$\hat{w}_{jk}^t$	The weight of pairwise relation between items i_j^t and i_k^t w.r.t. $y_a^{u,t}$
$\boldsymbol{l}_t^u \in \mathbb{R}^H$	The overall linear attentive intra-basket relation between items in B_t^u
$\boldsymbol{p}_{jk}^t$	The pairwise intra-basket relation between a pair of items i_j^t and i_k^t
$\boldsymbol{q}_t^u \in \mathbb{R}^H$	The overall pairwise attentive intra-basket relation between items in B_t^u
$\boldsymbol{b}_t^u$	The basket representation of basket B_t^u
$\boldsymbol{s}_t^u$	The annotation representation of user u at time t

Table 6.2: Notations in Sequential Basket Recommendation (Part II)

Notation	Description
$\boldsymbol{p}^u_{t'}$	The RNN hidden state of user u for target basket $B^u_{t'}$
$\hat{\boldsymbol{f}}^u_{t'}$	The user feedback vector of user u for target basket $B^u_{t'}$
$\boldsymbol{c}^u_{t'}$	The context vector for target basket $B^u_{t'}$
$w^{t'}_j$	The integration weight capturing the impact on basket B^u_j for the prediction of target basket B^u_t
$\boldsymbol{n}^u_{t'}$	The negative feedback vector of user u at time t'

6.2.1 HAEM Architecture

Figure 6.2 presents the architecture and working process of the proposed HAEM. The HAEM takes sequential behaviors S^u of user u as input, collects user feedback to basket $B^u_{t'}, t' \in [t_u, t_u + p - 1]$, and predicts the personalized ranking $<_{u,t'+1}$ for $B^u_{t'+1}$ by generating preference score $\boldsymbol{o}^u_{t'+1}$. Specifically, since the HAEM makes predictions for sequential basket recommendation by jointly modeling both intra-/inter-basket relations in user sequential behaviors and then refines recommendations by incorporating the positive/negative feedback, given behaviors S^u, HEAM first builds a basket representation for each basket $B^u_t \in S^u$ which captures the compound intra-basket behavior relations over the items within B^u_t. After building a sequence of basket representations where each corresponds to a specific basket, HAEM then explores the inter-basket relations between baskets in behaviors S^u by learning a sequence of annotation representations. The annotation representation $\boldsymbol{s}^u_t$ of user u at time t contains inter-basket information about all the sequential behaviors of user u with a strong focus on the particular part surrounding basket B^u_t as the more recent behaviors may have a greater impact on the target behavior. Finally, based on the sequence of annotation representations learned and the feedback collected from each user, the HAEM predicts each

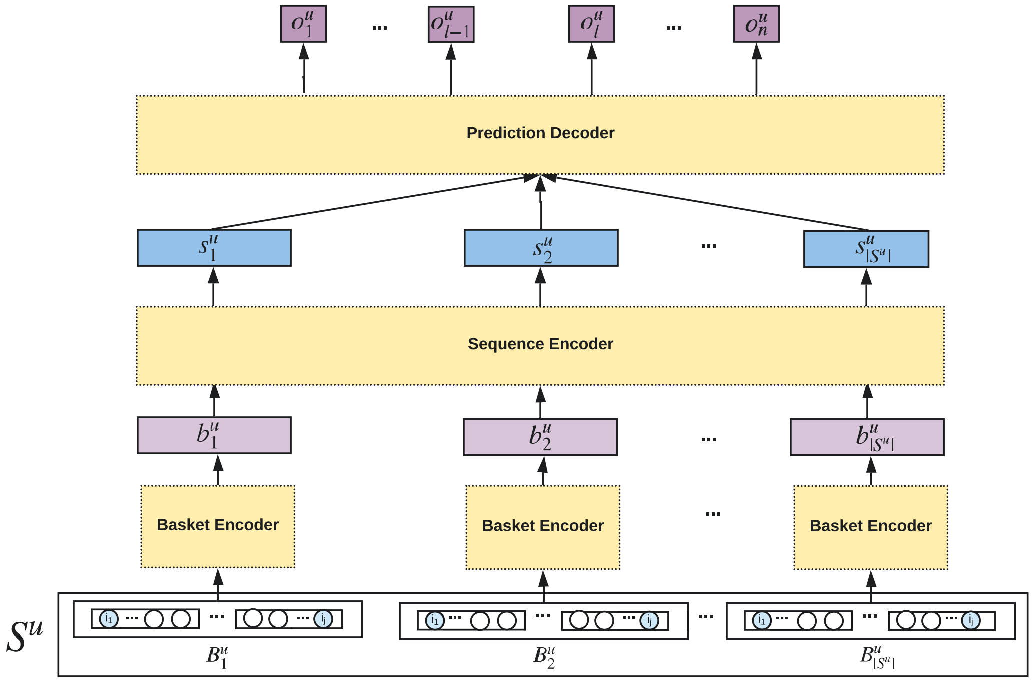


Figure 6.2: The HAEM Model for Sequential Basket Recommendation

target basket by modeling the basket's context representation.

Accordingly, the HAEM has three modules to achieve the above: a *basket encoder* to learn the representation of each basket and the intra-basket relations between items in the basket; a *sequence encoder* to learn a sequence of annotation representations from the learned basket representations by modeling the inter-basket relations in sequential behaviors; and a *prediction decoder* to model the basket context representations from the annotation sequence learned and predict next baskets based on the basket context representations and make refinements based on collecting positive/negative user feedback. Below, we introduce each component in Figure 6.2 from bottom to top.

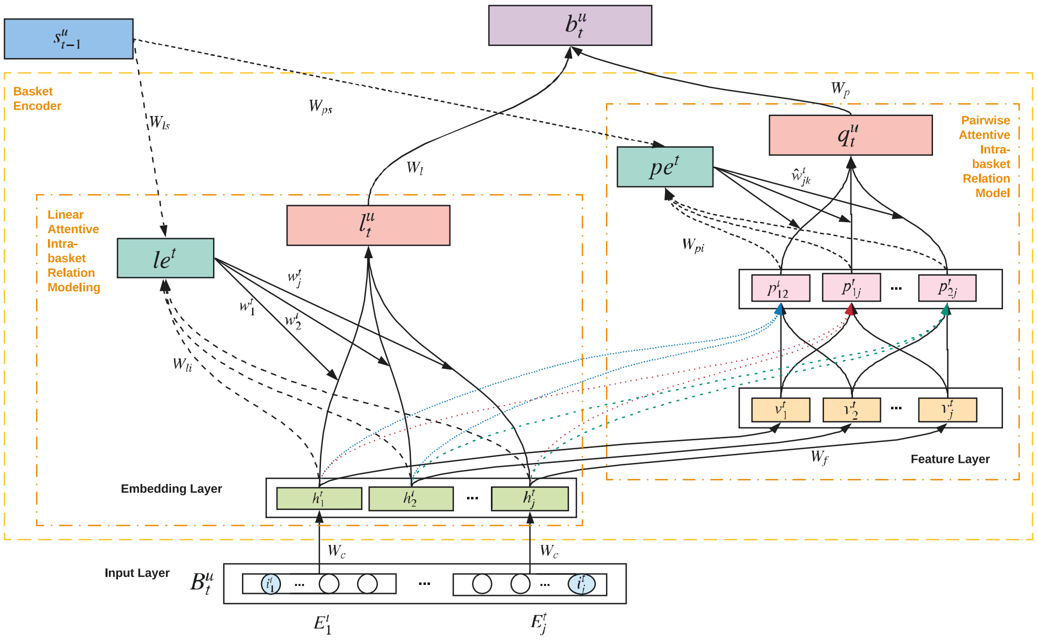


Figure 6.3: The Basket Encoder. It maps an input basket to a basket representation and models linear and pairwise attentive intra-basket relations.

6.2.2 Basket Encoder

Taking the items in basket $B^u_t \in S^u$ of user u as an input example, the basket encoder learns a corresponding basket representation $\boldsymbol{b}^u_t$ by modeling the compound intra-basket behavior relations $y^{u,t}_a$ within basket B^u_t, which consist of both linear and pairwise intra-basket relations between items and are captured by incorporating the FM mechanism, as shown in Figure 6.3. Inspired by the FM-based work in (Rendle 2010, He & Chua 2017), the basket encoder models the intra-basket relations between items $y^{u,t}_a$ in B^u_t as follows:

$$y^{u,t}_a = \boldsymbol{W}_l \sum_{j=1}^{|B^u_t|} w^t_j \boldsymbol{h}^t_j + \boldsymbol{W}_p \sum_{j=1}^{|B^u_t|} \sum_{k=j+1}^{|B^u_t|} \hat{w}^t_{jk} \langle \boldsymbol{v}^t_j, \boldsymbol{v}^t_k \rangle \boldsymbol{h}^t_j \odot \boldsymbol{h}^t_k \tag{6.1}$$

where the first term learns the linear intra-basket relation of a single item to the basket, while the second term learns the pairwise intra-basket relation between every two items. $\boldsymbol{h}_j^t \in \mathbb{R}^H$ is the embedding representation vector of item $i_j^t \in B_t^u$, and w_j is the weight of item i_j^t with respect to the intra-basket relation $y_a^{u,t}$, indicating the contribution rate of item i_j^t to $y_a^{u,t}$. Further, $\boldsymbol{v}_j^t \in \mathbb{R}^F$ denotes the latent feature vector of item i_j^t, and $\langle \boldsymbol{v}_j^t, \boldsymbol{v}_k^t \rangle$ is the inner-product of latent vectors $\boldsymbol{v}_j^t$ and $\boldsymbol{v}_k^t$, indicating the factorized interactions between items i_j^t and i_k^t (Rendle 2010, Rendle et al. 2011). The operation $\boldsymbol{h}_j^t \odot \boldsymbol{h}_k^t$ denotes the element-wise product of two embedding vectors, and $\langle \boldsymbol{v}_j^t, \boldsymbol{v}_k^t \rangle \boldsymbol{h}_j^t \odot \boldsymbol{h}_k^t$ encodes the second-order relation between items i_j^t and i_k^t in the embedding space (He & Chua 2017, Guo et al. 2017). Lastly, $\hat{w}_{jk}^t$ is the weight of the pairwise relation between items i_j^t and i_k^t with respect to $y_a^{u,t}$, indicating the contribution rate of this pair of items to $y_a^{u,t}$, and $\boldsymbol{W}_l \in \mathbb{R}^{B \times H}$ and $\boldsymbol{W}_p \in \mathbb{R}^{B \times H}$ fully connect the linear and pairwise vectors to $y_a^{u,t}$.

Here, two different vectors are designed for item representations in this basket encoder, that is, the embedding representation vector and the latent feature vector. The embedding vector is an informative lower-dimensional representation of an item to represent its individual behavior information, while its latent feature vector represents the latent interaction information by an FM model to capture the pairwise intra-basket relation between one item and another. As discussed in Section 6.1, compared with single items, a pair of relevant items may have a higher impact on revealing user preferences or behavior trends, and thus the pairwise intra-basket relation contributes to identifying significant item pairs for better recommendations.

Below, we further detail how to encode a basket representation. Given basket B_t^u, each item $i_j^t \in B_t^u$ is encoded as a *one-hot* vector, where only the unit at position j is set 1 while others are set 0, and the one-hot vectors of all the items in B_t^u constitute the input units in the bottom layer of Figure 6.3. Since the one-hot vector of item i_j^t, denoted as $\boldsymbol{E}_j^t$, only represents meaningless ID information, an embedding layer is created to map the sparse one-hot vector of an item to an informative continuous low-dimensional representation, and a H-dimensional vector $\boldsymbol{h_j^t} \in \mathbb{R}^H$ is used as the embedding representation vector of i_j^t. The weight matrix $\boldsymbol{W}_c \in \mathbb{R}^{H \times |I|}$ fully connects the input layer and embedding layer, namely:

$$\boldsymbol{h}_j^t = \sigma(\boldsymbol{W}_c \boldsymbol{E}_j^t). \tag{6.2}$$

Further, we obtain the overall linear attentive intra-basket relation between the items in basket B_t^u, $\boldsymbol{l}_t^u \in \mathbb{R}^H$, by a weighted integration of the embedding vectors $\{\boldsymbol{h}_j^t | i_j^t \in B_t^u\}$, namely:

$$\boldsymbol{l}_t^u = \sum\nolimits_{j=1}^{|B_t^u|} w_j^t \boldsymbol{h}_j^t. \tag{6.3}$$

The integration weight w_j^t captures the contribution rate of item i_j^t with respect to the intra-basket relation in basket B_t^u, which is learned automatically by the linear attention layer, as depicted in the left part of Figure 6.3. Considering that the contribution and significance of an item depend not only on the item's contextual items in the basket (Wang et al. 2018) but also the previous baskets (Cui et al. 2017), our linear attentive model calculates the integration weights by a softmax layer as demonstrated in Eqs. (6.4) and (6.5). Here $\boldsymbol{\alpha}$ is a context vector shared by all items which acts as a high-level representation of the informative factors over the items (Kumar, Irsoy, Ondruska, Iyyer, Bradbury, Gulrajani, Zhong, Paulus & Socher 2016) and is jointly trained with all the other components (Bahdanau, Cho & Bengio 2014). In addition, vector $\boldsymbol{s}_{t\text{-}1}^u \in \mathbb{R}^B$ is the annotation representation of previous basket $B_{t\text{-}1}^u$; matrices $\boldsymbol{W}_{ls}$ and $\boldsymbol{W}_{li}$ fully connect vectors $\boldsymbol{s}_{t\text{-}1}^u$ and $\boldsymbol{h}_j^t$ to vector $\boldsymbol{le}_j^t$, and $\boldsymbol{b}_l$ is the bias vector.

$$w_j^t = softmax(\boldsymbol{\alpha}^T \boldsymbol{le}_j^t) = \frac{exp(\boldsymbol{\alpha}^T \boldsymbol{le}_j^t)}{\sum_{k=1}^{|B_t^u|} exp(\boldsymbol{\alpha}^T \boldsymbol{le}_k^t)} \tag{6.4}$$

$$\boldsymbol{le}_j^t = tanh(\boldsymbol{W}_{ls}\boldsymbol{s}_{t\text{-}1}^u + \boldsymbol{W}_{li}\boldsymbol{h}_j^t + \boldsymbol{b}_l) \tag{6.5}$$

In addition, inspired by the FM mechanism, we create a feature layer to map the embedding vector of each item to a latent feature vector, denoted as $\boldsymbol{v_j^t} \in \mathbb{R}^F$, which represents the latent feature interaction information of item i_j^t and is used to capture the pairwise intra-basket relations with another item. Since the items hold the homogeneous feature information, a shared weight matrix $\boldsymbol{W_f} \in \mathbb{R}^{F\times H}$ is used to fully connect the embedding layer and the feature layer as depicted in Eq. (6.6), where $\boldsymbol{b_f}$ is the bias vector:

$$\boldsymbol{v}_j^t = \sigma(\boldsymbol{W}_f\boldsymbol{h}_j^t + \boldsymbol{b}_f). \tag{6.6}$$

By implementing the FM mechanism, the pairwise intra-basket relation $\boldsymbol{p}_{jk}^t$ between a pair of items i_j^t and i_k^t can be captured by the embedding vectors and feature vectors, namely:

$$\boldsymbol{p}_{jk}^t = \langle \boldsymbol{v}_j^t, \boldsymbol{v}_k^t \rangle \boldsymbol{h}_j^t \odot \boldsymbol{h}_k^t. \tag{6.7}$$

Similar to the linear attentive intra-basket relation modeling, the overall pairwise attentive intra-basket relation between the items in basket B_t^u, $\boldsymbol{q}_t^u \in \mathbb{R}^H$, is captured by the weighted integration of the $\boldsymbol{p}_{jk}^t$ between each pair of items, namely:

$$\boldsymbol{q}_t^u = \sum\nolimits_{j=1}^{|B_t^u|} \sum\nolimits_{k=j+1}^{|B_t^u|} \hat{w}_{jk}^t \boldsymbol{p}_{jk}^t. \tag{6.8}$$

Similar to the design of a linear attention model, the integration weight $\hat{w}_{jk}^t$ is learned by the pairwise attention model as shown in the right part of Figure 6.3.

$$\hat{w}_{jk}^t = \frac{exp(\boldsymbol{\beta}^T \boldsymbol{pe}_{jk}^t)}{\sum_{m=1}^{|B_t^u|}\sum_{n=m+1}^{|B_t^u|} exp(\boldsymbol{\beta}^T \boldsymbol{pe}_{mn}^t)} \tag{6.9}$$

$$\boldsymbol{pe}_{jk}^t = tanh(\boldsymbol{W}_{ps}\boldsymbol{s}_{t\text{-}1}^u + \boldsymbol{W}_{pi}\boldsymbol{p}_{jk}^t + \boldsymbol{b}_p) \tag{6.10}$$

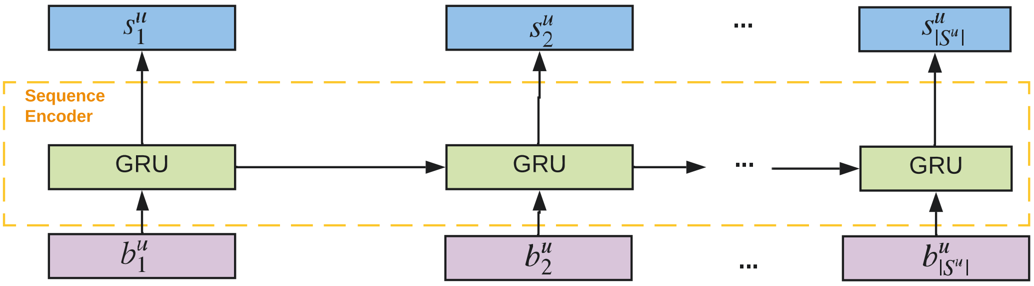


Figure 6.4: The Sequence Encoder. It maps the basket representation sequence to an annotation representation sequence.

Here, $\boldsymbol{\beta}$ is a context vector shared by all pairs of items, matrices $\boldsymbol{W}_{ps}$ and $\boldsymbol{W}_{pi}$ fully connect vectors $\boldsymbol{s}_{t\text{-}1}^u$ and $\boldsymbol{p}_{jk}^t$ to vector $\boldsymbol{pe}_{jk}^t$ respectively, and $\boldsymbol{b}_p$ is the bias vector. Accordingly, basket representation $\boldsymbol{b}_t^u$ is encoded by the learned intra-basket relation, which is the integration of a linear and pairwise relation as shown in Eq. (6.11).

$$\boldsymbol{b}_t^u = \sigma(y_a^{u,t}) = \sigma(\boldsymbol{W}_l \boldsymbol{l}_t^u + \boldsymbol{W}_p \boldsymbol{q}_t^u) \tag{6.11}$$

6.2.3 Sequence Encoder

Given a sequence of baskets, their intra-basket relation-based representations $< \boldsymbol{b}_1^u, \boldsymbol{b}_2^u, \ldots, \boldsymbol{b}_{t_u\text{-}1}^u >$ can be built by the above basket encoder where each basket representation learned only captures the intra-basket relations among items. To learn the inter-basket relations between baskets with the sequential behaviors as the context of each basket, the sequence encoder applies a GRU-based recurrent architecture on the top of the basket encoder to propagate the sequential information between each two adjacent baskets to capture the global inter-basket relation with respect to the sequential features of all baskets (Yu et al. 2016), as depicted in Figure 6.4. Given the annotation representation $\boldsymbol{s}_{t\text{-}1}^u$ of user u at time t-1 and basket representation $\boldsymbol{b}_t^u$, the annotation representation $\boldsymbol{s}_t^u$ at time t is calculated by Eq. (6.12), which indicates the dynamic representation of the preference of user u at time t.

$$\boldsymbol{s}_t^u = GRU(\boldsymbol{s}_{t\text{-}1}^u, \boldsymbol{b}_t^u) = (1 - \boldsymbol{z}_t^{se})\boldsymbol{s}_{t\text{-}1}^u + \boldsymbol{z}_t^{se}\hat{\boldsymbol{s}}_t^{se} \tag{6.12}$$

Here, $\hat{\boldsymbol{s}}_t^{se}$ and $\boldsymbol{z}_t^{se}$ are the update state and update gate of the sequence encoder respectively, which are given as follows:

$$\hat{\boldsymbol{s}}_t^{se} = tanh(\boldsymbol{W}_h^{se}\boldsymbol{b}_t^u + \boldsymbol{U}_h^{se}(\boldsymbol{r}_t^{se} \odot \boldsymbol{s}_{t\text{-}1}^u)) \tag{6.13}$$

$$\boldsymbol{z}_t^{se} = \sigma(\boldsymbol{W}_z^{se}\boldsymbol{b}_t^u + \boldsymbol{U}_z^{se}\boldsymbol{s}_{t\text{-}1}^u) \tag{6.14}$$

where $\boldsymbol{r}_t^{se}$ is the reset gate, which is given as follows:

$$\boldsymbol{r}_t^{se} = \sigma(\boldsymbol{W}_r^{se}\boldsymbol{b}_t^u + \boldsymbol{U}_r^{se}\boldsymbol{s}_{t\text{-}1}^u). \tag{6.15}$$

Here, update gate $\boldsymbol{z}_t^{se}$ allows each annotation representation to maintain the activation of its previous basket, while the reset gate $\boldsymbol{r}_t^{se}$ controls how much and what information from the previous basket should be reset (Bahdanau et al. 2014).

Subsequently, with the sequence of basket representations $< \boldsymbol{b}_1^u, \boldsymbol{b}_2^u, \dots, \boldsymbol{b}_{t_u\text{-}1}^u >$ as input, the sequence encoder generates the corresponding sequence of annotation representations, that is, $< \boldsymbol{s}_1^u, \boldsymbol{s}_2^u, \dots, \boldsymbol{s}_{|S^u|}^u >$. This sequence of annotation representations carries the contextual behavioral information of a user's behavior sequence for the basket behaviors by learning both intra-basket and inter-basket behavior relations among items. It will be used to make predictions for the next baskets.

6.2.4 Prediction Decoder

With the sequence of annotation representations learned by the sequence encoder and the user feedback on the recommended previous baskets, the

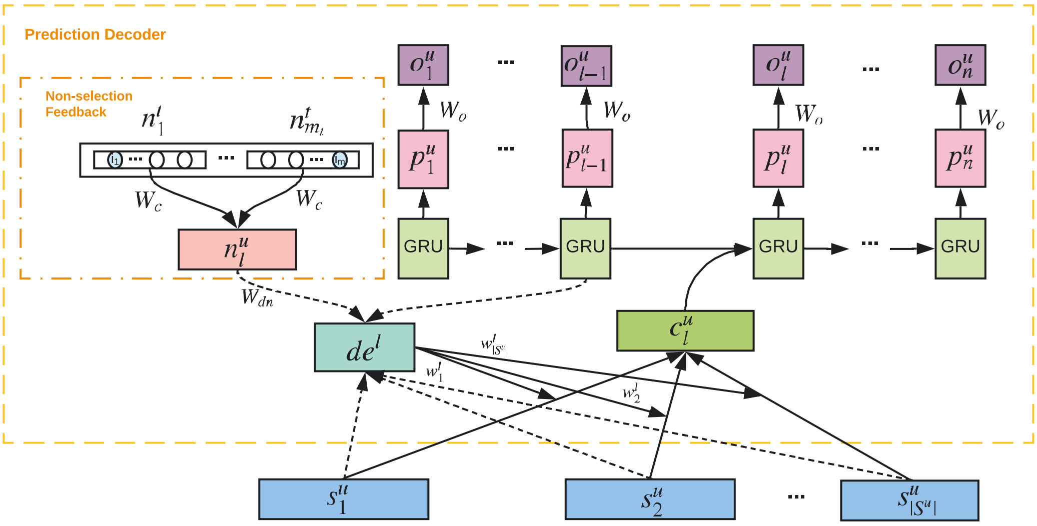


Figure 6.5: The Prediction Decoder. It makes predictions for sequential basket recommendations based on the annotation sequence learned.

prediction decoder makes predictions for the next several target baskets sequentially as depicted in Figure 6.5. This is motivated by the intuition that the behavior preference of a user u for the next target basket $B_{t'}^u$ is likely influenced by or associated with recent past behaviors S^u as presented in Figure 6.1 (Liu et al. 2017, Cui et al. 2017) and positive/negative feedback on those already recommended baskets (Zhao, Zhang, Ding, Xia, Tang & Yin 2018) for refining future recommendations. Accordingly, the prediction decoder predicts the next basket $B_{t'}^u$ at time $t' \in [t_u, t_u + p - 1]$ based on the user preference score $\boldsymbol{o}_{t'}^u$ with respect to all items. The preference score is generated based on RNN hidden state $\boldsymbol{p}_{t'}^u$ for time t', which is calculated by involving its previous hidden state $\boldsymbol{p}_{t'\text{-}1}^u$, user feedback vector $\hat{\boldsymbol{f}}_{t'\text{-}1}^u$ consisting of feedback on whether a recommended item is selected or non-selected by the user, and context vector $\boldsymbol{c}_{t'}^u$ consisting of the sequence of previous baskets consumed by this user and being used as the context of the current basket.

$$\begin{aligned}\boldsymbol{o}^u_{t'} &= \boldsymbol{W}_o\boldsymbol{p}^u_{t'} &(6.16)\\ \boldsymbol{p}^u_{t'} &= RNN(\boldsymbol{p}^u_{t'\text{-}1}, \hat{\boldsymbol{f}}^u_{t'\text{-}1}, \boldsymbol{c}^u_{t'})\\ &= (1-\boldsymbol{z}^{pd}_{t'})\boldsymbol{p}^u_{t'\text{-}1} + \boldsymbol{z}^{pd}_{t'}\hat{\boldsymbol{p}}^{pd}_{t'} &(6.17)\end{aligned}$$

Here, the weight matrix $\boldsymbol{W_o}$ fully connects the RNN hidden states to the output layer, and $\hat{\boldsymbol{p}}^{pd}_{t'}$ and $\boldsymbol{z}^{pd}_{t'}$ are the update state and update gate of the prediction decoder respectively, which is given as follows:

$$\begin{aligned}\hat{\boldsymbol{p}}^{pd}_{t'} &= tanh(\boldsymbol{W}^{pd}_h\hat{\boldsymbol{f}}^u_{t'\text{-}1} + \boldsymbol{U}^{pd}_h(\boldsymbol{r}^{pd}_{t'} \odot \boldsymbol{p}^u_{t'\text{-}1}) + \boldsymbol{C_h}\boldsymbol{c}^u_{t'}) &(6.18)\\ \boldsymbol{z}^{pd}_{t'} &= \sigma(\boldsymbol{W}^{pd}_z\hat{\boldsymbol{f}}^u_{t'\text{-}1} + \boldsymbol{U}^{pd}_z\boldsymbol{p}^u_{t'\text{-}1} + \boldsymbol{C_z}\boldsymbol{c}^u_{t'}) &(6.19)\end{aligned}$$

where $\boldsymbol{r}^{pd}_{t'}$ is the reset gate, which is given as follows:

$$\boldsymbol{r}^{pd}_{t'} = \sigma(W^{pd}_r\hat{\boldsymbol{f}}^u_{t'\text{-}1} + U^{pd}_r\boldsymbol{p}^u_{t'\text{-}1} + C_r\boldsymbol{c}^u_{t'}). \tag{6.20}$$

Here, the user feedback vector $\hat{\boldsymbol{f}}^u_{t'\text{-}1}$ is constructed as a multi-hot vector for the recommended items in basket $B^u_{t'\text{-}1}$, where the units corresponding to those items recommended and also selected by the user are set 1 while those recommended yet non-selected are set at 0. When we predict target basket $B^u_{t'}$, the user feedback at time t'-1 is available and is thus used to construct feedback vector $\hat{\boldsymbol{f}}^u_{t'\text{-}1}$, which is then fed to Eq. (6.17). In addition, context vector $\boldsymbol{c}^u_{t'}$ of user u at time t' is modeled as a weighted integration of the annotation representations (i.e., previous baskets), namely:

$$\boldsymbol{c}^u_{t'} = \sum\nolimits_{j=1}^{|S^u|} w^{t'}_j\boldsymbol{s}^u_j \tag{6.21}$$

where the integration weight $w^{t'}_j$ captures the influence of the user preference of each previous basket B^u_j on target basket B^u_t. Accordingly, we set the weights in terms of two perspectives. On one hand, the weight w^t_j is associated with both the annotation vector $\boldsymbol{s}^u_j$ and the hidden state of previous prediction $\boldsymbol{p}^u_{t'\text{-}1}$. On the other hand, as discussed in Section 6.1, those recommended but non-selected items in the user feedback reflect the divergence of user preferred items from our recommendations and are thus valuable to refine future predictions. Consequently, the weight $w^{t'}_j$ is computed as follows:

$$w^{t'}_j = softmax(\boldsymbol{\gamma}^T \boldsymbol{de}^{t'}_j) = \frac{exp(\boldsymbol{\gamma}^T \boldsymbol{de}^{t'}_j)}{\sum_{k=1}^{|S^u|} exp(\boldsymbol{\gamma}^T \boldsymbol{de}^{t'}_k)} \tag{6.22}$$

$$\boldsymbol{de}^{t'}_j = tanh(\boldsymbol{W}_{ds}[\boldsymbol{s}^u_j : \boldsymbol{p}^u_{t'\text{-}1}] + \boldsymbol{W}_{dn}\boldsymbol{n}^u_{t'\text{-}1} + \boldsymbol{b}_d). \tag{6.23}$$

$\boldsymbol{W}_{ds}$ and $\boldsymbol{W}_{dn}$ are the weight matrices, and $\boldsymbol{b}_d$ is the bias vector. In addition, the non-selection feedback vector $\boldsymbol{n}^u_{t'\text{-}1} \in \mathbb{R}^H$ is the integration of the embedding representation of $m_{t'\text{-}1}$ recommended but non-selected items $\{i^{t'\text{-}1}_j\}$ $(i^{t'\text{-}1}_j \in I)$. Here, $m_{t'\text{-}1}$ is the number of non-selected items in the basket recommended for time t'-1. Assume $\mathbb{B}^u_{t'\text{-}1}$ to be the set of items recommended for basket $B^u_{t'\text{-}1}$, then the set of negative items is generated via $\{i^{t'\text{-}1}_j\} \equiv \{i | i \in \mathbb{B}^u_{t'\text{-}1} \wedge i \notin B^u_{t'\text{-}1}\}$. For the example in Figure 6.1, $\mathbb{B}^u_3 = \{chip,\ toffee\ apple\}$ while $B^u_3 = \{toffee\ apple\}$ and the recommended item *chip* is not selected; thus $m_3 = 1$ and i^3_1 is *chip*. $\boldsymbol{n}^{t'\text{-}1}_j$ is the one-hot embedding vector of non-selected item $i^{t'\text{-}1}_j$, and the non-selection feedback vector $\boldsymbol{n}^u_{t'\text{-}1} \in \mathbb{R}^H$ is calculated by Eq. (6.24), namely:

$$\boldsymbol{n}^u_{t'\text{-}1} = \frac{1}{m_{t'\text{-}1}} \sum_{j=1}^{m_{t'\text{-}1}} \sigma(\boldsymbol{W}_c \boldsymbol{n}^{t'\text{-}1}_j). \tag{6.24}$$

As a result, context vector $\boldsymbol{c}^u_{t'}$ can identify and consider the important cross-basket item combinations in the sequential behavior S^u for the prediction of target basket $B^u_{t'}$, such as the identification of the sub-sequence $< pumpkin, (candy, chocolate) >$ for target basket B^u_3 in the example in Figure 6.1. This is owing to the integration weight $w^{t'}_j$ in Eq. (6.22), which captures those significant baskets, while the linear integration weight w^t_j in

Eq. (6.4) and the pairwise integration weight $\hat{w}^t_{jk}$ in Eq. (6.9) of the basket encoder respectively capture the significant items and item combinations within a basket. Consequently, the prediction decoder generates a user preference score $\boldsymbol{o}^u_{t'}$ for the prediction of target basket $B^u_{t'}$, where the v-th element $o^u_{t',v}$ indicates the preference score of user u to item i_v at time t', and a higher score reveals user u is more likely to consume item i_v.

6.2.5 The HAEM Learning and Prediction

As the HAEM involves user feedback and generates a top-k ranking of predicted items for each target basket, we apply Bayesian personalized ranking (BPR) to the HAEM learning process. BPR is a state-of-the-art pairwise ranking method on feedback data (Rendle et al. 2009) that demonstrates strong suitability as an objective function for behavior prediction tasks (Cui et al. 2017, Liu et al. 2017, Yu et al. 2016, Liu, Wu, Wang & Tan 2016). The HAEM implements BPR in terms of a logistic function $\sigma(\cdot)$ by assuming that selected item v in the basket at a specific time is more preferred by a user than non-selected item v'. In this way, the following probability needs to be maximized:

$$p(u,t,v \succ v') = \sigma(o^u_{t,v} - o^u_{t,v'}). \tag{6.25}$$

By adding up all the log-likelihood and regularization terms, we can minimize the objective function as follows:

$$J = \sum ln(1 + e^{-(o^u_{t,v} - o^u_{t,v'})}) + \frac{\lambda}{2}\|\Theta\|^2. \tag{6.26}$$

Here, Θ denotes the set of all parameters to be estimated, and λ is the hyper-parameter to control the power of the regularization term. The above objective function is optimized by backpropagation through time, and the parameters are updated by stochastic gradient descent until convergence.

After the training, the HAEM makes the sequential basket recommendation as follows. Given a sequential behavior S^u, the HAEM first produces the preference score vector $\boldsymbol{o}^u_{t_u}$ for basket $B^u_{t_u}$ according to Eq. (6.16), which indicates the preference ranking over items at time t_u. The top-ranked items at time t_u are recommended to user u as the next basket for feedback. Second, by collecting user feedback on these recommendations, the user feedback vector $\hat{\boldsymbol{f}}^u_{t_u}$ and non-selection feedback vector $\boldsymbol{n}^u_{t_u}$ are updated. Third, the HAEM then generates the preference score vector $\boldsymbol{o}^u_{t_u+1}$ for following basket $B^u_{t_u+1}$ at time $t_u + 1$ by the prediction decoder. Lastly, the HAEM further produces an item ranking for target basket $B^u_{t_u+1}$. The HAEM repeats this process for sequential recommendations of the next baskets.

6.3 Experiments and Evaluation

In this part, we conduct an empirical analysis of our proposed HAEM on two real-life datasets. We first prepare the datasets and introduce the baseline methods to set up the experiments and then evaluate the performance of the proposed HAEM from the perspectives of both accuracy and novelty.

6.3.1 Datasets and Preparation

We adopt the following two real-life datasets for our experiments to evaluate our proposed HAEM model:

- Ta-Feng[1] is a public grocery shopping dataset released by ACM RecSys, covering 817,741 baskets of 32,266 users and 23,812 purchased products collected from November 2000 to February 2001.

[1] http://stackoverflow.com/questions/25014904/download-link-for-ta-feng-grocery-dataset

- IJCAI-15[2] is a real-life dataset collected from Tmall.com and contains user shopping logs for the six months before and on the "Double 11" day (November 11th).

To make the data applicable for sequential basket recommendations with user feedback, both datasets are preprocessed. First, for both datasets, all the users who bought fewer than ten baskets and all the items purchased by fewer than ten users are removed from the raw data. The K-*core* subsets of these two datasets are obtained (Rendle et al. 2010, Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016). In this research, we set the above K as 10 because we hope to extract the final few baskets of each user's sequential behaviors as the *target baskets* and guarantee the remaining previous subsequence contains a sufficient number of baskets to be fed to the proposed model and baselines. Second, on the processed data, we extract the user sequential behaviors from each dataset based on the transaction timestamp. Because both datasets only record the purchase date without any specific time, we treat the items purchased by a user in one day as a basket, and the baskets within a sequence are sorted temporally. In this way, the sequential behavior of user u consists of multiple baskets, and each basket contains multiple items, making the extracted user sequential behaviors consistent with the formalization in Chapter 3. Then, we split the processed datasets, which are the collection of extracted sequential behaviors, into training and testing sets by randomly holding out 20% of the sequential behaviors for testing while the remaining is used for training, as in related work (Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017, Wang et al. 2018). It is noted that the dataset that is 80/20 training-test split is the collection of the extracted sequential behaviors, of which each instance is a user sequential

[2] https://tianchi.aliyun.com/datalab/dataSet.htm?id=1

Table 6.3: Statistics of Experimental Datasets

Statistics	**Ta-Feng**	**IJCAI-15**
#Users $\|U\|$	9,238	24,889
#Items $\|I\|$	7,982	35,272
#Baskets	67,964	183,472
Average basket size	7.4	9.1
Average number of items per User	43.7	67.1

behavior with ordered baskets, and thus the order of the baskets within each user sequential behavior is retained. Lastly, to construct a training or testing instance for the sequential behavior of user u, the final p baskets ($p = 3$) are picked up as the *target baskets* and the previous baskets constitute the input sequential behavior S^u. Here, we set the length of *target baskets* as 3 because we hope to verify the effectiveness of the non-selection feedback-based refinement for continuous sequential basket recommendations on the basis of guaranteeing that the corresponding input sequential behaviors consist of a sufficient number of baskets for model training. The statistics of the consequent datasets for experiments are presented in Table 6.3.

6.3.2 Baseline Methods

There are no recommenders reported for sequential basket recommendation by jointly learning the intra-/inter-basket behavior relations and positive/negative feedback in sequential user behaviors. We thus compare the HAEM with the following representative state-of-the-art NBRS and session-based methods for the experiments to evaluate the quality of next-basket/next-item recommendation. We also test the effect of non-selection feedback-based recommendation refinement embedded in the HAEM for each next-basket recommendation and sequential basket recommendation.

- *FPMC* (Rendle et al. 2010): A hybrid model to combine the first-order Markov chains with matrix factorization for next-basket recommendation, which factorizes the personal transition matrix between items with a pairwise interaction model.

- *HRM* (Wang, Guo, Lan, Xu, Wan & Cheng 2015): A hierarchical representation-based model to explore both general user preference and the last basket for next-basket recommendation.

- *DREAM* (Yu et al. 2016): An RNN-based model to learn a dynamic representation of a user to capture user preference on all baskets for next-basket recommendation.

- *ATEM* (Wang et al. 2018): An attention-based model to learn an attentive context embedding over all the observed items within a transaction for session-based next-item recommendation.

- *DEERS* (Zhao, Zhang, Ding, Xia, Tang & Yin 2018): A deep reinforcement learning-based model to automatically learn the optimal recommendation strategies by incorporating both positive and negative feedbacks, which can continuously improve its strategies by modeling interactions with users.

- *HAEM_L*: A sub-model of HAEM that only models the linear intra-basket relations between items in the basket encoder.

- *HAEM_S*: A sub-model of HAEM that does not contain the negative feedback-based refinement mechanism in the prediction decoder.

- *HAEM*: The full HAEM model which consists of all three components.

HAEM_L, HAEM_S and HAEM are created for the ablation test. HAEM_L only models the linear intra-basket relations but ignores the pairwise intra-basket relations in the basket encoder. The comparison between HAEM_L and HAEM reveals the effect and contribution of the pairwise intra-basket relations on modeling sequential basket recommendation. In addition, by replacing the attention layer of the basket encoder in HAEM_L by max/average pooling and replacing the context vector of the prediction decoder by the last annotation representations of the sequence encoder, HAEM_L degrades to DREAM for NBRS tasks. The effect and contribution of the attention layers in both the basket encoder and the prediction decoder is demonstrated in the comparison between HAEM_L and DREAM for NBRS. Finally, the significance of the negative feedback-based refinement in the prediction decoder is evaluated by the comparison between HAEM_S and HAEM.

6.3.3 Experimental Settings

In the training process, the input sequential behaviors of each user in the training set are fed into the models in batches to learn the sequence of annotation representations. During the testing stage, the context vector is built based on the sequence of annotation representations and used to predict the first target basket. The actual first target basket from the raw data is used as the benchmark for comparison; those recommended but nonoccurring items in the benchmark are collected as non-selected items (negative feedback) for the refinement of further recommendations. With the obtained benchmark of first target basket and the identified non-selected items, the trained HAEM is then used to sequentially predict the second target basket. This interactive recommendation-feedback process repeats until the desired p-th target basket is predicted.

To evaluate the recommendation performance, we select the top-k items (we choose $k = 5$) as the recommended basket for each user, denoted as $R^u_{t_{u+p'}}$, where $R^u_{t_{u+p'}}[j]$ represents the prediction for the j-th item. Here, k is set as 5 because of the limitation of the average basket size in the experimental datasets and $k = 5$ is also a widely adopted top-k setting (Rendle et al. 2010, Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016). In reality, most customers are only interested in the first few items recommended on the first page (Hu, Cao, Wang, Xu, Cao & Gu 2017), and it could be a challenge to find out the exact actual items from many possible items in each basket (Wang, Hu & Cao 2017, Wang et al. 2018).

6.3.4 Evaluation Measures

We evaluate the *accuracy* and *novelty* of the predictions made by HAEM against those made by baselines. Accuracy is a typical measure for evaluating the NBRS approaches (Yu et al. 2016), and two measures *F1-score*@k and *NDCG*@k compare the predicted baskets with those preferred by users.

- *F1-score*: is the harmonic mean of precision and recall, which are calculated as follows:

$$Precision(B^u_{t_{u+p'}}, R^u_{t_{u+p'}}) = \frac{|B^u_{t_{u+p'}} \cap R^u_{t_{u+p'}}|}{|R^u_{t_{u+p'}}|} \tag{6.27}$$

$$Recall(B^u_{t_{u+p'}}, R^u_{t_{u+p'}}) = \frac{|B^u_{t_{u+p'}} \cap R^u_{t_{u+p'}}|}{|B^u_{t_{u+p'}}|} \tag{6.28}$$

$$F1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{6.29}$$

- *Normalized Discounted Cumulative Gain (NDCG@k)*: is a cumulative measure of ranking quality that takes into account the order of recommended items in the list, which is calculated as follows:

$$NDCG@k = \frac{1}{N_k}\sum_{j=1}^{K}\frac{2^{I(R^u_{t_{u+p'}}[j]\in B^u_{t_{u+p'}})}-1}{log_2(j+1)} \tag{6.30}$$

where $I(\cdot)$ is an indicator function, and N_k is a constant that denotes the maximum value of NDCG@k given $R^u_{t_{u+p'}}$.

We report the accuracy per $F1$-*score* and NDCG@k for using the HAEM to predict both a single next basket and the sequential following baskets by making refinements based on the feedback collected from the previous recommended baskets. This evaluates the effect of involving the selection feedback on refining recommendations.

In addition, we also assess the novelty of the recommended baskets to address the usual concern about suggesting duplicated items and the common interest in novel items. Accordingly, we use *MCAN@k* (Wang, Hu & Cao 2017, Wang et al. 2018) as the novelty measure. For NBRS, *MCAN@k* quantifies the difference between the recommended target basket and its previous one. Intuitively, the more different the recommended items are from those consumed in the previous basket, the more novel these items are. Therefore, given recommendation list $R^u_{t_{u+p'}}$, the fewer items in $R^u_{t_{u+p'}}$ that are observed in its previous basket $B^u_{t_{u+p'\text{-}1}}$, the higher the novelty of this recommendation. Accordingly, the novelty of recommendation $R^u_{t_{u+p'}}$, denoted as *CAN*, is defined as follows:

$$CAN = 1 - \frac{|R^u_{t_{u+p'}} \cap B^u_{t_{u+p'\text{-}1}}|}{|R^u_{t_{u+p'}}|}. \tag{6.31}$$

Therefore, the overall novelty over all N top-k recommendations (i.e., *MCAN@k*), is defined as the mean of the CAN of all recommendation as follows:

$$MCAN = \frac{1}{N}\sum_{j=1}^{N} CAN. \tag{6.32}$$

By substituting Eq. (6.31) back into Eq. (6.32), we obtain Eq. (6.33) to evaluate novelty.

$$MCAN = \frac{1}{N}\sum_{j=1}^{N}(1 - \frac{|R^{u}_{t_{u+p'}} \cap B^{u}_{t_{u+p'\text{-}1}}|}{|R^{u}_{t_{u+p'}}|}) \tag{6.33}$$

$MCAN@k$ quantifies the difference between the recommended basket and the previous basket actually consumed by a user. A higher $MCAN@k$ indicates that the items in the predicted basket are more different from those already consumed in the previous transactions, that is, more novel items are recommended.

6.3.5 Accuracy Evaluation

Performance of Next-basket Recommendation

We first apply our proposed HAEM just for next-basket recommendations for a fair comparison with existing NBRS methods as they cannot make sequential basket recommendations. We set three different values of the dimensionality d for the embedding representations: $d \in \{50, 100, 150\}$ on Ta-Feng and IJCAI-15; Tables 6.4 and 6.5 reveal the five-fold cross-validation results of *F1-score*@5 and *NDCG*@5 over the datasets Ta-Feng and IJCAI-15, respectively. The number of factors is set as 5 for training FPMC when the best performance is achieved; its accuracy worsens when more factors are adopted. A similar phenomenon is also observed in (Hu, Cao, Wang, Xu, Cao & Gu 2017, Wang, Hu & Cao 2017, Wang et al. 2018). In addition, for

Table 6.4: Accuracy of HAEM against Baselines for Next-basket Recommendation on the Ta-Feng Dataset

Methods	**F1-score@5**			**NDCG@5**		
	$d = 50$	$d = 100$	$d = 150$	$d = 50$	$d = 100$	$d = 150$
FPMC	0.0543	0.0578	0.0620	0.0763	0.0781	0.0813
HRM	0.0573	0.0633	0.0653	0.0805	0.0823	0.0829
DREAM	0.0630	0.0660	0.0685	0.0830	0.0837	0.0845
ATEM	0.0609	0.0654	0.0677	0.0828	0.0841	0.0848
DEERS	0.0645	0.0667	0.0691	0.0835	0.0849	0.0855
HAEM_L	0.0660	0.0686	0.0701	0.0868	0.0892	0.0897
HAEM	**0.0692**	**0.0719**	**0.0721**	**0.0887**	**0.0904**	**0.0907**

HRM, the number of negative samples is empirically set as 25, and the regularization constant and drop ratio are set as 0.001 and 60%. For DREAM, the regularization constant is set as 0.001, and the number of RNN hidden layers is set as 3. The aggregation operations for HRM and DREAM are max pooling because it shows advantage over average poolings (Wang, Guo, Lan, Xu, Wan & Cheng 2015, Yu et al. 2016). For DEERS, the length of the modeled feedback is set as 10, and the discounted factor and pairwise regularization constant are set as 0.9 and 0.1, empirically. Finally, for the HAEM-based methods, the hyper-parameter λ is empirically set as 0.001.

The experimental results indicate that among all methods, FPMC performs the worst. This may be because the intra-basket behavior relations between items are linearly independent, which is inconsistent with real-world

Table 6.5: Accuracy of HAEM against Baselines for Next-basket Recommendation on the IJCAI-15 Dataset

Methods	F1-score@5			NDCG@5		
	$d = 50$	$d = 100$	$d = 150$	$d = 50$	$d = 100$	$d = 150$
FPMC	0.0556	0.0584	0.0593	0.1248	0.1438	0.1469
HRM	0.0606	0.0618	0.0653	0.1318	0.1470	0.1550
DREAM	0.0640	0.0670	0.0695	0.1515	0.1610	0.1665
ATEM	0.0628	0.0652	0.0682	0.1434	0.1560	0.1628
DEERS	0.0659	0.0692	0.0715	0.1582	0.1654	0.1701
HAEM_L	0.0736	0.0767	0.0794	0.1679	0.1759	0.1803
HAEM	**0.0793**	**0.0825**	**0.0828**	**0.1791**	**0.1897**	**0.1913**

cases, and FPMC fails to depict the sophisticated interactions between items and cannot capture the influence of item interactions. In addition, both datasets are sparse, and therefore the matrices constructed by these datasets are quite large and sparse for training this MF-based model. Moreover, HRM and ATEM lag behind DREAM because they both only learn the embedding representation based on the successive transaction while neglecting all the baskets purchased before in the entire behavior history, which omits significant information. Furthermore, HRM utilizes the max pooling operation on the item representation within a basket to capture the most significant features of input items for intra-basket behavior relation modeling, but max pooling can neither recognize the relevant items and capture their impact for next baskets nor model any pairwise intra-basket behavior relations. Compared with HRM, ATEM achieves higher accuracy because it intensifies the relevant items and downplays the irrelevant ones for predictions by the attention model. Compared with HRM and ATEM, which only make predictions from the last basket, DREAM captures the global sequential information and models users' dynamic representations benefiting from its recurrent structure, and thus performs slightly better. In both datasets, the *F1-score*@5 of DREAM reveals an improvement of more than 6% over HRM and around 2% over ATEM. However, DREAM is weak in capturing the compound intra-/inter-basket behavior relations between items. Thanks to the mechanism of incorporating both selection and non-selection feedback to learn the optimal recommendation strategies, DEERS achieves higher recommendation accuracy than DREAM and contributes to an improvement of, on average, more than 2% over DREAM. However, similar to the drawback of DREAM, DEERS adopts an RNN with a GRU to capture user sequential preference, which tends to pay more attention to recent behaviors and is less effective at capturing the inter-basket relations between divergent baskets.

In contrast, the HAEM achieves much better performance than all baselines by maximally 27.4% and an average of 15.9% with respect to *F1-score*@5 and maximally 16.3% and an average of 9.5% with respect to *NDCG*@5 on Ta-Feng with $d = 50$, and maximally 42.6% and on average 28.9% with respect to *F1-score*@5 and maximally 43.5% and on average 27.4% with respect to *NDCG*@5 on IJCAI-15 with $d = 50$. The HAEM demonstrates its superiority consistently over all baseline methods on both datasets, which illustrates the effectiveness of capturing compound intra-/inter-basket relations for NBRS. In particular, compared with DEERS, the HAEM contributes to an improvement of on average 6.51% *F1-score*@5 and 6.28% *NDCG*@5 ($d = 50$) on Ta-Feng and 18.49% *F1-score*@5 and 13.45% *NDCG*@5 ($d = 50$)

on IJCAI-15, revealing that the modeling of the intra-basket relations helps to enhance the accuracy performance for the next-basket recommendation.

Ablation study

We evaluate the effect of only modeling the intra-basket relations by HAEM_L, excluding the negative feedback-based refinement by HAEM_S, and involving both intra-/inter-basket relations and positive/negative feedback by HAEM.

As illustrated in Tables 6.4 and 6.5, compared with HAEM_L, which only models the linear intra-basket behavior relations in the basket encoder, HAEM models more complicated intra-basket behavior relations and inter-basket behavior relations, and thus contributes to an additional 4.85% *F1-score*@5 and 2.19% *NDCG*@5 ($d = 50$) on Ta-Feng and 7.74% *F1-score*@5 and 6.67% *NDCG*@5 ($d = 50$) on IJCAI-15 over HAEM_L for the next-basket recommendation. In addition, compared with DREAM, which models intra-basket behavior relation by max pooling, HAEM_L captures the linear intra-basket behavior relations to intensify the relative items by the linear attention layer, and thus contributes to an additional 4.76% *F1-score*@5 and 4.58% *NDCG*@5 ($d = 50$) on Ta-Feng and 15.0% *F1-score*@5 and 10.8% *NDCG*@5 ($d = 50$) on IJCAI-15 over DREAM for the next-basket recommendation. The above ablation analysis suggests that the attention layer as well as the pairwise intra-basket behavior relation modeling in the basket encoder can significantly improve the accuracy of the HAEM for NBRS.

In addition, the experiments also indicate that both HEAM and HAEM_L achieve better accuracy on a higher dimensionality d on two datasets but greater accuracy advantage over baselines on a lower dimensionality. In addition, the highest *NDCG*@5 value also demonstrates that our models can more accurately predict the more highly-ranked items in the recommendation list to a user for multifaceted reasons. On one hand, the basket encoder in HAEM makes the basket representations learned more informative: the compound intra-basket behavior relations between items within a basket are learned and encoded into the basket representation, and significant items and item combinations are intensified by the attentive mechanism. On the other hand, the basket context representations learned by the sequence encoder and prediction decoder help to capture the inter-basket behavior relations within the whole behavior sequence and recognize the significant baskets for the target recommendation, which is more consistent with real-world cases compared with those baseline models that only utilize the previous basket or focus more on the recent ones.

In conclusion, the experimental results indicate that both HAEM_L and HAEM consistently outperform all baseline methods on both datasets, which illustrates their effectiveness in capturing compound intra-/inter-basket behavior relations for NBRS. In addition, in Section 6.3.5, we further demonstrate that our proposed HAEM also significantly outperforms its variants HAEM_L and HAEM_S for sequential basket recommendation (i.e., SBRS).

Effect of Negative Feedback-based Refinement for Sequential Basket Recommendation

We further apply HAEM_L, HAEM_S and HAEM to predict sequential baskets by involving user feedback. Tables 6.6 and 6.7 present the recommendation results of *F1-score*@5 and *NDCG*@5 for three continuous target baskets on both datasets. We set the dimensionality $d \in \{50, 150\}$ on both Ta-Feng and IJCAI-15; the suffix of each method refers to the adopted dimensionality of the embedding representation on the corresponding datasets. For example, *HAEM_L_50* refers to the method HAEM_L which sets its dimensionality as

Table 6.6: Accuracy Effect of Negative Feedback-based Refinement on the Ta-Feng Dataset

Methods	**F1-score@5**			**NDCG@5**		
	$p = 1$	$p = 2$	$p = 3$	$p = 1$	$p = 2$	$p = 3$
HAEM_L_50	0.0660	0.0679	0.0720	0.0868	0.0878	0.0918
HAEM_S_50	0.0692	0.0689	0.0686	0.0887	0.0886	0.0884
HAEM_50	0.0692	0.0702	0.0734	0.0887	0.0897	0.0932
HAEM_L_150	0.0701	0.0724	0.0757	0.0897	0.0913	0.0941
HAEM_S_150	0.0721	0.0718	0.0716	0.0907	0.0906	0.0903
HAEM_150	0.0721	0.0754	0.0769	0.0907	0.0950	0.0971

50. Given the user feedback and non-selection feedback on previous recommended baskets, both HAEM_L and HAEM with low and high representation dimensionalities achieve better performance with the increase of continuous target baskets, illustrating the significant impact made by negative feedback-based refinement on continuous next-basket recommendations. Contrary to HAEM_L and HAEM, the accuracy of HAEM_S decreases with the increase of target baskets, and it demonstrates the accumulation of the prediction bias on worse recommendations for SBRS. Accordingly, negative feedback-based refinement is significant for more accurate prediction of SBRSs.

Further, HAEM always outperforms HAEM_L contributing to maximally 4.85% and an average of 3.39% with respect to *F1-score*@5 as well as maximally 2.19% and an average of 1.96% with respect to *NDCG*@5 ($d = 50$) on Ta-Feng, and maximally 7.74% and an average of 5.27% with respect to

Table 6.7: Accuracy Effect of Negative Feedback-based Refinement on the IJCAI-15 Dataset

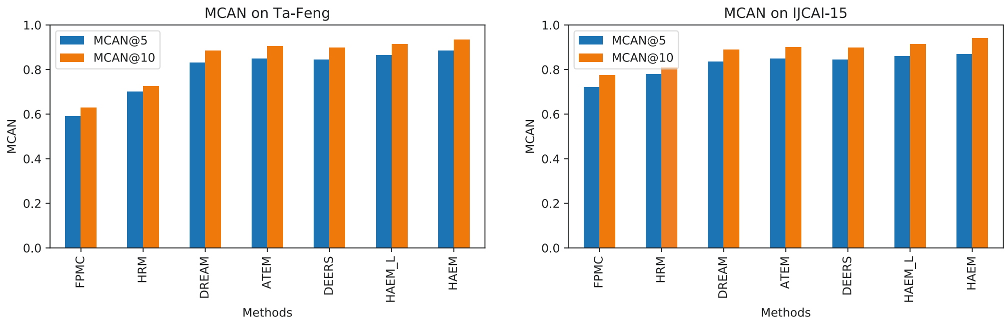
Methods	**F1-score@5**			**NDCG@5**		
	$p = 1$	$p = 2$	$p = 3$	$p = 1$	$p = 2$	$p = 3$
HAEM_L_50	0.0736	0.0769	0.0803	0.1679	0.1780	0.1854
HAEM_S_50	0.0793	0.0787	0.0771	0.1791	0.1789	0.1786
HAEM_50	0.0793	0.0810	0.0825	0.1791	0.1859	0.1901
HAEM_L_150	0.0794	0.0837	0.0867	0.1803	0.1899	0.1971
HAEM_S_150	0.0828	0.0827	0.0820	0.1913	0.1872	0.1850
HAEM_150	0.0828	0.0867	0.0886	0.1913	0.1978	0.2033

F1-score@5 as well as maximally 6.67% and an average of 4.55% with respect to *NDCG*@5 ($d = 50$) on IJCAI-15 over HAEM_L for the sequential basket recommendation because the significant pairwise intra-basket behavior relations captured in the HAEM contribute to the next-basket prediction. In addition, with the help of the negative feedback-based refinement mechanism, the HAEM outperforms HAEM_S by maximally 7.40% and an average of 4.14% with respect to *F1-score*@5 and maximally 7.53% and an average of 4.13% with respect to *NDCG*@5 on Ta-Feng with $d = 150$, and maximally 8.05% and an average of 4.30% with respect to *F1-score*@5 and maximally 9.89% and an average of 5.18% with respect to *NDCG*@5 on IJCAI-15 with $d = 150$. The above observation suggests that the proposed HAEM mechanisms of modeling both compound intra-/inter-basket behavior relations and enabling negative feedback-based refinement are effective for SBRS.

6.3.6 Novelty Evaluation

In RS research, more recent interest has been on avoiding duplicated and similar recommendations (Wang et al. 2018, Zhao, Willemsen, Adomavicius, Harper & Konstan 2018), improving recommendation diversity (Hu, Cao, Wang, Xu, Cao & Gu 2017), and actionability (Cao 2012), thus enabling greater user satisfaction and business effects. These have proven to be particular important for next-item, next-basket, and session-based recommendations (Wang, Hu & Cao 2017). In next-basket recommendations, users are often concerned with similar or duplicated items they consumed in the last basket that are recommended again in the next basket. Hence, NBRS and sequential recommendations should not only recommend more accurate items preferred by users but also recommend more novel and diverse items to satisfy the different, changing, and new consumer demand that may also complement those items already recommended. Novel NBRSs will thus enable better user experience and satisfaction, the actionability of NBRSs, and business benefits.

Accordingly, in this section, we evaluate the novelty of the recommended next basket in terms of *MCAN@k* by different methods on both datasets. Figure 6.6 illustrates the results of the novelty comparison of the top-5 and top-10 recommendations over Ta-Feng and IJCAI-15 by all competitors for NBRS. Overall, the HAEM achieves the greatest novelty compared with other baselines. FPMC provides the lowest novelty because the sparse characteristics of both datasets make it difficult for FPMC to learn its parameters well, and thus FPMC only recommends relatively similar items. The HRM generates better novelty than FPMC, but it makes recommendations only by modeling the previous basket in sequential behaviors. This is often not the case in the real world and may result in information loss by neglecting the



(a) MCAN@5 on Ta-Feng

(b) MCAN@5 on IJCAI-15

Figure 6.6: Novelty Comparison of Methods on Two Datasets.

other baskets in the sequential behaviors, which makes HRM likely to output items similar to the previous basket and incurs low novelty. In addition, benefiting from its RNN-based model, DREAM can make use of all the baskets in the sequential behaviors and capture the global influence of these baskets to make more novel recommendations. Compared with DREAM, DEERS explores user sequential preference from the perspectives of both positive and negative feedback, which helps DEERS gain more comprehensive knowledge to provide slightly higher novelty recommendations.

However, the RNN model, on which DREAM and DEERS are based, assumes that the temporal dependency changes monotonously along with the sequence and the recent baskets have a more dominant effect on the prediction than the previous ones. This assumption does not conform to complex real situations for behavior predictions, where there is no guarantee that one basket has more or less a significant effect than the previous one (Liu et al. 2017). Accordingly, DREAM and DEERS cannot distinguish the dominant basket and thus its prediction relies more on the recent baskets, which makes its recommendations similar to the previous basket. Moreover, the max pooling of the basket representation in DREAM can only capture major factors among the items but cannot intensify significant items and item combinations and model their compound behavior relations for predictions. ATEM can generate slightly more novel recommendation benefiting from its attentive structure, but its prediction is only based on its contextual basket and neglects its previous purchase history.

Compared with the aforementioned baselines, the proposed HAEM_L and HAEM can not only consider the inter-basket behavior relations to intensify the difference between the last baskets and the next ones along the whole sequence of behavior, but also make full use of the compound intra-basket behavior relations within each basket to enhance item diversity, which makes the recommendations more diverse and novel. Compared with HAEM_L, HAEM can achieve slightly higher novelty, indicating that the modeling of pairwise intra-basket behavior relations in the basket encoder contributes to more novel recommendations.

6.4 Summary

In this chapter, we discuss the problem of *interactive sequential basket recommendation* by modeling both intra-basket behavior relations and inter-basket behavior relations among items, as well as continuously incorporating the interactions of user selection and non-selection of recommended items in the next-baskets for refining the further next basket recommendations. This work represents one-step forward in the recent popular interest on sequential recommendation, next-basket recommendation, and session-based recommendation. To achieve the above goal, the hierarchical attentive encoder-decoder model HAEM is created based on deep models. The HAEM jointly models the intra-/inter-basket behavior relations and continuously quantifies the impact of user-selected and non-selected items during the iterative interactions between the recommender and users on next recommendations. The positive/negative interactive feedback further improves the continuous next-basket recommendations. The HAEM not only significantly outperforms existing next-basket recommenders in terms of accuracy and novelty, but also provides continuously improved sequential basket recommendations when the positive/negative interactive feedback is iteratively incorporated. Lastly, interactive sequential basket recommendation can address many real-life applications and can form a solid foundation for future research to address the diversified characteristics and complexities such as addressing changing user preferences and behaviors in interactive sequential recommendations.

Different from other chapters which focus on the discovery of set/subset of entire NSB patterns with highly-global significance, this chapter explores the complicated behavior relations inside the sequential behaviors and the interaction of NOBs for the prediction of future behaviors, which alleviates the challenges faced by pattern mining-based NSB analytics. However, it only explores the behavior relations from the distribution information of individual items but fails to consider the feature information of items. Actually, the behavior relations among items are greatly driven by their intrinsic nature, that is, they are complicatedly coupled with item features (Cao 2015), and such coupling relevance is particularly significant for the prediction of the novel or rarely-observed behaviors (Wang, Hu & Cao 2017). Accordingly, it would be highly interesting to incorporate the item features into the exploration of comprehensive behavior relations to enable more effective and accurate behavior prediction/recommendation for complex behavior data.

Chapter 7

Conclusions and Future Directions

7.1 Conclusions

In this thesis, we propose various methods to incorporate the NOB characteristics in terms of sequence analysis to perform nonoccurring sequential behavior analytics. Here, we categorize the research on NSB analytics into three kinds of tasks: 1) Efficient Negative Sequential Pattern Mining in Chapter 4; 2) Representative Negative Sequential Pattern Discovery in Chapter 5; and 3) Interactive Sequential Basket Recommendation in Chapter 6.

7.1.1 Efficient Negative Sequential Pattern Mining

We have incorporated a *loose negative element constraint (LNEC)* into *NSP mining* to enable the discovery of NSPs containing *partial negative elements*, which specifies that elements in an NSB are allowed to contain both positive and negative items but do not contain both an item and its negation. To solve the challenging problem of complete NSP mining with an LNEC, we present an efficient vertical mining framework, VM-NSP, to efficiently discover the complete set of the high-frequency NSPs by incorporating a vertical representation (VR) for each sequence. On the basis of the VM-NSP framework, we further propose a bitmap-based NSP mining method, bM-NSP, to optimize the performance of NSP mining by enabling efficient NSC support calculation without any dataset re-scanning or reducing the number of candidates generated. As a result, bM-NSP reveals significantly better efficiency and scalability and more complete coverage for effective pattern discovery.

7.1.2 Representative Negative Sequential Pattern Discovery

We have made the first attempt by proposing a novel Determinantal Point Process (DPP)-based *representative NSP discovery* approach EINSP, which jointly models both explicit and implicit NSB pattern relations by modeling the distribution over all possible subsets in view of both relations; it then samples a representative subset composed of the high-quality and diverse NSB patterns. Both theoretical design and empirical analysis are provided, where experimental results on both real-life and synthetic datasets demonstrate the strong potential to deliver more representative and informative NSB patterns with wider coverage and diversity and higher quality in terms of different data factors.

7.1.3 Interactive Sequential Basket Recommendation

We have proposed a hierarchical attentive encoder-decoder model (HAEM) for the problem of *interactive sequential basket recommendation.* The proposed HAEM (1) jointly considers both intra-/inter-basket behavior relations among items in the sequential behaviors, (2) continuously incorporates the interactions of a user's selection and non-selection of recommended items to refine the next-basket prediction, and (3) enables the continuous recommendation of a sequence of next-baskets by iteratively feeding positive/negative feedback. As a result, the HAEM not only significantly outperforms the state-of-the-art NBRSs and session-based recommenders in terms of both accuracy and novelty, but also presents continuously improved sequential basket recommendations when the positive/negative interactive feedback is iteratively incorporated.

7.2 Future Directions

Our comprehensive work in this thesis discloses the significant challenges facing NSB analytics and the enormous opportunities it presents. Nonoccurring sequential behavior analytics-related research is still in an early stage, and there is significant potential for NSB for various nonoccurring behaviors and applications (Cao et al. 2015, Liu et al. 2015*a*, Dong et al. 2014, Kamepalli & Kurra 2014, Li et al. 2010). Here, we highlight several future directions in the research on NSB analytics as per our current work, which include but are not limited to:

7.2.1 Incremental NSP Mining over Streamed Behaviors

Our current work on *NSP mining* is confined over the static sequential behaviors, and it is much more challenging yet practical to discover NSPs over data streams in which new sequential behaviors are continuously inserted (Cheng, Yan & Han 2004). New data sequences may be generated at high speed, and NSP methods can only process incoming new sequential behaviors once and do not re-scan the dataset repeatedly, as is usually done by existing methods (Soliman, Ebrahim & Mohammed 2011). By incorporating the new streamed sequential behaviors, the amount of space and memory resources required to process sequences may be quickly exhausted. This involves many issues including how to store, represent, and process evolving NSP and how to define the related constraint settings and negative containment.

7.2.2 Top-K NSP Mining

We have proposed an efficient NSP mining method that sets a predefined threshold and mines the complete set of all NSB patterns, and it is an interesting further work to allow users to determine the number of NSB patterns to be discovered and target the discovery of only the top patterns with high significance within a limited search space and lower computational complexity. Compared with top-K PSP mining, *top-K NSP mining* is more challenging and sophisticated because the downward closure property does not hold in NSP mining. Accordingly, classic top-K PSP mining methods, such as TSP (Tzvetkov, Yan & Han 2005), TUS (Yin, Zheng, Cao, Song & Wei 2013), and TKU (Wu, Shie, Tseng & Yu 2012) cannot be used or adjusted directly. Therefore, new theories and methods are required to discover top-K NSPs.

7.2.3 Quantitative NSP Discovery

There are opportunities to extend our current work to consider the different significance of each negative containment and evaluate an NSB in terms of its utility instead of only its frequency, which can uncover more informative NSB patterns and achieve greater actionability of the resultant patterns (Yin, Zheng & Cao 2012, Cao 2012, Cao, Yu, Zhao & Zhang 2010). In some real-life applications, items with high utility (Yin et al. 2012) may appear less frequently. For example, in a PC shop, failing to sell a laptop would lead to a greater reduction in profits than failing to sell a mouse. Proper strategies may be informed by analyzing how to promote laptop sales to achieve higher profitability. For this, we need to quantify the utility of a negative item, element, and sequence and build a new theoretical framework for quantitative NSP discovery, for which existing NSP methods do not work. This involves many issues such as new evaluation measures, data structures, negative containment, NSC selection, and pruning strategies.

7.2.4 NSP Discovery with Complex Hierarchical Structure

Our proposed NSB analytics methods are confined by the assumption that items in the same element have no order and are on the same level, and thus one interesting further work is to consider the hierarchical structures involved in sequential behaviors. For example, in marketing, let $< (coke, tissue, sprite), (bread, milk) >$ be a user's purchase sequence; both *coke* and *sprite* belong to a subclass of food and thus they can be considered separately from the item *tissue*. The existing problem formalization is not applicable to this scenario and the introduction of a complex hierarchical structure can simplify the solution to such issues. When hierarchical and coupling relationships (Cao 2015, Cao 2012) are considered, many interesting research issues arise that are aligned with real-life behavior interactions and applications. It requires theoretical breakthroughs in representing, modeling, reasoning about, storing, and managing complex structures, hierarchies, and relationships in NSB analytics.

7.2.5 Feature-aware Sequential Behavior Prediction

In this thesis, we only explore the complicated behavior relations from the distribution information of individual items but overlook the feature information of these items. Accordingly, another challenging further research is to incorporate multiple feature information of each item to explore and model the comprehensive coupling relevance between the behavior relations among items and the item features, inspired by the fact that the item behavior relations may be significantly driven by their intrinsic nature (Cao 2015), and such coupling relevance is highly critical for the prediction of novel or rarely-observed behaviors (Wang, Hu & Cao 2017). This indicates that not only the contextual distribution information of individual items but also their features can be modeled to enable a more accurate prediction for sequential behaviors, that is, *feature-aware sequential behavior prediction*, especially for the case of cold-start behavior prediction/recommendation.

Bibliography

Adomavicius, G. & Tuzhilin, A. (2005), 'Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions', *IEEE Transactions on Knowledge & Data Engineering* (6), 734–749.

Affandi, R. H., Fox, E., Adams, R. & Taskar, B. (2014), Learning the parameters of determinantal point process kernels, *in* 'International Conference on Machine Learning', pp. 1224–1232.

Agrawal, R. & Srikant, R. (1995), Mining sequential patterns, *in* 'Data Engineering, 1995. Proceedings of the Eleventh International Conference on', IEEE, pp. 3–14.

Anwar, F., Petrounias, I., Morris, T. & Kodogiannis, V. (2010), Discovery of events with negative behavior against given sequential patterns, *in* 'Intelligent Systems, 2010 5th IEEE International Conference', IEEE, pp. 373–378.

Aseervatham, S., Osmani, A. & Viennet, E. (2006), bitspade: A lattice-based sequential pattern mining algorithm using bitmap representation, *in* 'Sixth International Conference on Data Mining', IEEE, pp. 792–797.

Ayres, J., Flannick, J., Gehrke, J. & Yiu, T. (2002), Sequential pattern mining using a bitmap representation, *in* 'Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 429–435.

Bahdanau, D., Cho, K. & Bengio, Y. (2014), 'Neural machine translation by jointly learning to align and translate', *arXiv preprint arXiv:1409.0473* .

Beg, I. & Butt, A. R. (2009), 'Fixed point for set-valued mappings satisfying an implicit relation in partially ordered metric spaces', *Nonlinear Analysis: Theory, Methods & Applications* **71**(9), 3699–3704.

Błaszczyszyn, B. & Keeler, P. (2018), 'Determinantal thinning of point processes with network learning applications', *arXiv preprint arXiv:1810.08672* .

Borcea, J., Brändén, P. & Liggett, T. (2009), 'Negative dependence and the geometry of polynomials', *Journal of the American Mathematical Society* **22**(2), 521–567.

Borodin, A. (2009), 'Determinantal point processes', *arXiv preprint arXiv:0911.1153* .

Borodin, A. & Rains, E. M. (2005), 'Eynard–mehta theorem, schur process, and their pfaffian analogs', *Journal of statistical physics* **121**(3-4), 291–317.

Bouma, G. (2009), 'Normalized (pointwise) mutual information in collocation extraction', *Proceedings of GSCL* pp. 31–40.

Cao, L. (2010), 'In-depth behavior understanding and use: The behavior informatics approach', *Inf. Sci.* **180**(17), 3067–3085.

Cao, L. (2012), 'Combined mining: Analyzing object and pattern relations for discovering actionable complex patterns', *sponsored by Australian Research Council Discovery Grants (DP1096218 and DP130102691) and an ARC Linkage Grant (LP100200774)* .

Cao, L. (2013), 'Combined mining: Analyzing object and pattern relations for discovering and constructing complex yet actionable patterns', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **3**(2), 140–155.

Cao, L. (2014), 'Non-iidness learning in behavioral and social data', *Comput. J.* **57**(9), 1358–1370.

Cao, L. (2015), 'Coupling learning of complex interactions', *Journal of Information Processing and Management* **51**(2), 167–186.

Cao, L. (2016), 'Non-iid recommender systems: A review and framework of recommendation paradigm shifting', *Engineering* **2**(2), 212–224.

Cao, L., Dong, X. & Zheng, Z. (2016), 'e-nsp: Efficient negative sequential pattern mining', *Artificial Intelligence* **235**, 156–182.

Cao, L., Ou, Y. & Yu, P. S. (2012), 'Coupled behavior analysis with applications', *Knowledge and Data Engineering, IEEE Transactions on* **24**(8), 1378–1392.

Cao, L., Ou, Y., Yu, P. S. & Wei, G. (2010), Detecting abnormal coupled sequences and sequence changes in group-based manipulative trading behaviors, *in* 'KDD'2010', pp. 85–94.

Cao, L. & Philip, S. Y. (2012), *Behavior computing: modeling, analysis, mining and decision*, Springer.

Cao, L. & Yu, P. S. (2012), *Behavior Computing - Modeling, Analysis, Mining and Decision*, Springer.

Cao, L., Yu, P. S. & Kumar, V. (2015), 'Nonoccurring behavior analytics: A new area', *Intelligent Systems, IEEE* **30**(6), 4–11.

Cao, L., Yu, P. S., Zhang, C. & Zhang, H. (2008), *Data Mining for Business Applications*, 1 edn, Springer.

Cao, L., Yu, P., Zhao, Y. & Zhang, C. (2010), *Domain Driven Data Mining*, Springer.

Cao, L., Zhao, Y. & Zhang, C. (2008*a*), 'Mining impact-targeted activity patterns in imbalanced data', *IEEE Trans. Knowl. Data Eng.* **20**(8), 1053–1066.

Cao, L., Zhao, Y. & Zhang, C. (2008*b*), 'Mining impact-targeted activity patterns in imbalanced data', *IEEE Transactions on knowledge and data engineering* **20**(8), 1053–1066.

Cao, L., Zhao, Y., Zhang, H., Luo, D., Zhang, C. & Park, E. K. (2010), 'Flexible frameworks for actionable knowledge discovery', *IEEE Trans. Knowl. Data Eng.* **22**(9), 1299–1312.

Chand, C., Thakkar, A. & Ganatra, A. (2012), 'Sequential pattern mining: Survey and current research challenges', *International Journal of Soft Computing and Engineering* **2**(1), 185–193.

Chen, L., Zhang, G. & Zhou, E. (2018), Fast greedy map inference for determinantal point process to improve recommendation diversity, *in* 'Advances in Neural Information Processing Systems', pp. 5622–5633.

Chen, S., Moore, J. L., Turnbull, D. & Joachims, T. (2012), Playlist prediction via metric embedding, *in* 'Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 714–722.

Chen, X., Xu, H., Zhang, Y., Tang, J., Cao, Y., Qin, Z. & Zha, H. (2018), Sequential recommendation with user memory networks, *in* 'Proceedings of the eleventh ACM international conference on web search and data mining', ACM, pp. 108–116.

Cheng, H., Yan, X. & Han, J. (2004), Incspan: incremental mining of sequential patterns in large database, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 527–532.

Chiu, D.-Y., Wu, Y.-H. & Chen, A. L. (2004), An efficient algorithm for mining frequent sequences by a new strategy without support counting, *in* 'Data Engineering, 2004. Proceedings. 20th International Conference on', IEEE, pp. 375–386.

Chou, S.-Y., Yang, Y.-H., Jang, J.-S. R. & Lin, Y.-C. (2016), Addressing cold start for next-song recommendation, *in* 'Proceedings of the 10th ACM Conference on Recommender Systems', ACM, pp. 115–118.

Christidis, K., Apostolou, D. & Mentzas, G. (2010), Exploring customer preferences with probabilistic topic models, *in* 'European Conference on

Machine Learning and Principles and Practice of Knowledge Discovery in Databases'.

Cui, Q., Wu, S., Huang, Y. & Wang, L. (2017), 'A hierarchical contextual attention-based gru network for sequential recommendation', *arXiv preprint arXiv:1711.05114* .

Dong, X., Gong, Y. & Cao, L. (2018*a*), 'e-rnsp: An efficient method for mining repetition negative sequential patterns', *IEEE transactions on cybernetics* .

Dong, X., Gong, Y. & Cao, L. (2018*b*), 'F-nsp+: A fast negative sequential patterns mining method with self-adaptive data storage', *Pattern Recognition* **84**, 13–27.

Dong, X., Gong, Y. & Zhao, L. (2014), Comparisons of typical algorithms in negative sequential pattern mining, *in* 'Electronics, Computer and Applications, 2014 IEEE Workshop on', IEEE, pp. 387–390.

Dong, X., Sun, F., Han, X. & Hou, R. (2006), Study of positive and negative association rules based on multi-confidence and chi-squared test, *in* 'International Conference on Advanced Data Mining and Applications', Springer, pp. 100–109.

Feng, S., Li, X., Zeng, Y., Cong, G., Chee, Y. M. & Yuan, Q. (2015), Personalized ranking metric embedding for next new poi recommendation, *in* 'IJCAI', Vol. 15, pp. 2069–2075.

Fournier-Viger, P., Gomariz, A., Campos, M. & Thomas, R. (2014), Fast vertical mining of sequential patterns using co-occurrence information, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer, pp. 40–52.

Gatzioura, A. & Sànchez-Marrè, M. (2015), 'A case-based recommendation approach for market basket data', *IEEE Intelligent Systems* **30**(1), 20–27.

Gillenwater, J. A., Kulesza, A., Fox, E. & Taskar, B. (2014), Expectation-maximization for learning determinantal point processes, *in* 'Advances in Neural Information Processing Systems', pp. 3149–3157.

Gillenwater, J., Kulesza, A. & Taskar, B. (2012), Discovering diverse and salient threads in document collections, *in* 'Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning', Association for Computational Linguistics, pp. 710–720.

Gong, B., Chao, W.-L., Grauman, K. & Sha, F. (2014), Diverse sequential subset selection for supervised video summarization, *in* 'Advances in Neural Information Processing Systems', pp. 2069–2077.

Gong, Y., Liu, C. & Dong, X. (2015), 'Research on typical algorithms in negative sequential pattern mining', *Open Automation and Control Systems Journal* **7**, 934–941.

Gong, Y., Xu, T., Dong, X. & Lv, G. (2017), 'e-nspfi: Efficient mining negative sequential pattern from both frequent and infrequent positive sequential patterns', *International Journal of Pattern Recognition and Artificial Intelligence* **31**(02), 1750002.

Gu, W., Dong, S. & Zeng, Z. (2014), 'Increasing recommended effectiveness with markov chains and purchase intervals', *Neural Computing and Applications* **25**(5), 1153–1162.

Guo, H., Tang, R., Ye, Y., Li, Z. & He, X. (2017), 'Deepfm: a factorization-machine based neural network for ctr prediction', *arXiv preprint arXiv:1703.04247* .

Hariri, N., Mobasher, B. & Burke, R. (2015), Adapting to user preference changes in interactive recommendation, *in* 'IJCAI', Vol. 15, pp. 4268–4274.

He, X. & Chua, T.-S. (2017), Neural factorization machines for sparse predictive analytics, *in* 'Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval', ACM, pp. 355–364.

Hidasi, B., Karatzoglou, A., Baltrunas, L. & Tikk, D. (2015), 'Session-based recommendations with recurrent neural networks', *arXiv preprint arXiv:1511.06939* .

Hong, K. & Nenkova, A. (2014), Improving the estimation of word importance for news multi-document summarization, *in* 'Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics', pp. 712–721.

Hsueh, S.-C., Lin, M.-Y. & Chen, C.-L. (2008), Mining negative sequential patterns for e-commerce recommendations, *in* 'Asia-Pacific Services Computing Conference, 2008. APSCC'08. IEEE', IEEE, pp. 1213–1218.

Hu, L., Cao, L., Cao, J., Gu, Z., Xu, G. & Wang, J. (2017), 'Improving the quality of recommendations for users and items in the tail of distribution', *ACM Transactions on Information Systems (TOIS)* **35**(3), 25.

Hu, L., Cao, L., Cao, J., Gu, Z., Xu, G. & Yang, D. (2016), 'Learning informative priors from heterogeneous domains to improve recommendation in cold-start user domains', *ACM Transactions on Information Systems (TOIS)* **35**(2), 13.

Hu, L., Cao, L., Wang, S., Xu, G., Cao, J. & Gu, Z. (2017), Diversifying personalized recommendation with user-session context, *in* 'Proceedings of the 26th International Joint Conference on Artificial Intelligence', AAAI Press, pp. 1858–1864.

Hu, Y., Koren, Y. & Volinsky, C. (2008), Collaborative filtering for implicit feedback datasets., *in* 'ICDM', Vol. 8, Citeseer, pp. 263–272.

J. Pisharath, Y. Liu, B. O. R. N. W. L. A. C. G. M. (n.d.), 'Nu-minebench version 2.0 data set and technical report', `http://cucis.ece.northwestern.edu/projects/DMS/MineBenchDownload.html`.

Jiang, H., Luan, X. & Dong, X. (2012), Mining weighted negative association rules from infrequent itemsets based on multiple supports, *in* 'Industrial Control and Electronics Engineering, 2012 International Conference on', IEEE, pp. 89–92.

Jiang, X., Gao, Q., Xu, T. & Dong, X. (2018), 'Campus data analysis based on positive and negative sequential patterns', *International Journal of Pattern Recognition and Artificial Intelligence* .

Kamepalli, S. & Kurra, R. (2014), 'Frequent negative sequential patterns: a survey', *Int. J. Comput. Eng. Technol* **5**(3), 15–121.

Kazienko, P. (2008), Mining sequential patterns with negative conclusions, *in* 'International Conference on Data Warehousing and Knowledge Discovery', Springer, pp. 423–432.

Khare, V. K. & Rastogi, V. (2013), 'Mining positive and negative sequential pattern in incremental transaction databases', *International Journal of Computer Applications* **71**(1).

Kulesza, A. & Taskar, B. (2010), Structured determinantal point processes, *in* 'Advances in neural information processing systems', pp. 1171–1179.

Kulesza, A. & Taskar, B. (2011), k-dpps: Fixed-size determinantal point processes, *in* 'Proceedings of the 28th International Conference on Machine Learning (ICML-11)', pp. 1193–1200.

Kulesza, A. & Taskar, B. (2012), 'Learning determinantal point processes', *arXiv preprint arXiv:1202.3738* .

Kulesza, A., Taskar, B. et al. (2012), 'Determinantal point processes for machine learning', *Foundations and Trends® in Machine Learning* **5**(2–3), 123–286.

Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R. & Socher, R. (2016), Ask me anything: Dynamic memory networks for natural language processing, *in* 'International Conference on Machine Learning', pp. 1378–1387.

Lee, J.-S., Jun, C.-H., Lee, J. & Kim, S. (2005), 'Classification-based collaborative filtering using market basket data', *Expert Systems with Applications* **29**(3), 700–704.

Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T. & Ma, J. (2017), Neural attentive session-based recommendation, *in* 'Proceedings of the 2017 ACM on Conference on Information and Knowledge Management', ACM, pp. 1419–1428.

Li, Y., Algarni, A. & Zhong, N. (2010), Mining positive and negative patterns for relevance feature discovery, *in* 'Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 753–762.

Li, Y., Hong, J. & Chen, H. (2019), 'Short sequence classification through discriminable linear dynamical system', *IEEE transactions on neural networks and learning systems* .

Li, Z. & Eisner, J. (2009), First-and second-order expectation semirings with applications to minimum-risk training on translation forests, *in* 'Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1', Association for Computational Linguistics, pp. 40–51.

Lian, D., Zheng, K., Ge, Y., Cao, L., Chen, E. & Xie, X. (2018), 'Geomf++: Scalable location recommendation via joint geographical modeling and matrix factorization', *ACM Transactions on Information Systems (TOIS)* **36**(3), 33.

Lin, N. P., Chen, H.-J. & Hao, W.-H. (2007), Mining negative sequential patterns, *in* 'Proc. of the 6th WSEAS International Conference on Applied Computer Science, Hangzhou, China', pp. 654–658.

Lin, N. P., Chen, H.-J., Hao, W.-H., Chueh, H.-E. & Chang, C.-I. (2008), 'Mining strong positive and negative sequential patterns', *WSEAS Transactions on Computers* **7**(3), 119–124.

Lin, N. P., Hao, W.-H., Chen, H.-J., Chang, C.-I. & Chueh, H.-E. (2007), 'An algorithm for mining strong negative fuzzy sequential patterns', *International Journal of Computers* **1**(3).

Linden, G., Smith, B. & York, J. (2003), 'Amazon. com recommendations: Item-to-item collaborative filtering', *IEEE Internet computing* (1), 76–80.

Liu, C., Dong, X., Li, C. & Li, Y. (2015*a*), Sapnsp: Select actionable positive and negative sequential patterns based on a contribution metric, *in* 'Fuzzy Systems and Knowledge Discovery, 2015 12th International Conference on', IEEE, pp. 811–815.

Liu, C., Dong, X., Li, C. & Li, Y. (2015*b*), Sapnsp: Select actionable positive and negative sequential patterns based on a contribution metric, *in* 'Fuzzy Systems and Knowledge Discovery, 2015 12th International Conference on', IEEE, pp. 811–815.

Liu, Q., Wu, S., Wang, D., Li, Z. & Wang, L. (2016), Context-aware sequential recommendation, *in* '2016 IEEE 16th International Conference on Data Mining (ICDM)', IEEE, pp. 1053–1058.

Liu, Q., Wu, S. & Wang, L. (2017), 'Multi-behavioral sequential prediction with recurrent log-bilinear model', *IEEE Transactions on Knowledge and Data Engineering* **29**(6), 1254–1267.

Liu, Q., Wu, S., Wang, L. & Tan, T. (2016), Predicting the next location: A recurrent model with spatial and temporal contexts., *in* 'AAAI', pp. 194–200.

Mahasseni, B., Lam, M. & Todorovic, S. (2017), Unsupervised video summarization with adversarial lstm networks, *in* 'Proceedings of the IEEE conference on Computer Vision and Pattern Recognition', pp. 202–211.

Mariet, Z. & Sra, S. (2015), Fixed-point algorithms for learning determinantal point processes, *in* 'International Conference on Machine Learning', pp. 2389–2397.

Mobasher, B., Dai, H., Luo, T. & Nakagawa, M. (2002), Using sequential and non-sequential patterns in predictive web usage mining tasks, *in* '2002 IEEE International Conference on Data Mining, 2002. Proceedings.', IEEE, pp. 669–672.

Ouyang, W. & Huang, Q. (2009), Mining positive and negative fuzzy multiple level sequential patterns in large transaction databases, *in* 'Intelligent Systems, 2009. GCIS'09. WRI Global Congress on', Vol. 1, IEEE, pp. 500–504.

Ouyang, W. & Huang, Q. (2010), Mining positive and negative sequential patterns with multiple minimum supports in large transaction databases, *in* 'Intelligent Systems, 2010 Second WRI Global Congress on', Vol. 2, IEEE, pp. 190–193.

Ouyang, W., Huang, Q. & Luo, S. (2008), Mining positive and negative fuzzy sequential patterns in large transaction databases, *in* 'Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on', Vol. 5, IEEE, pp. 18–23.

Ouyang, W.-m. & Huang, Q.-h. (2007), Mining negative sequential patterns in transaction databases, *in* 'Machine Learning and Cybernetics, 2007 International Conference on', Vol. 2, IEEE, pp. 830–834.

Pan, R. & Scholz, M. (2009), Mind the gaps: weighting the unknown in large-scale one-class collaborative filtering, *in* 'Proceedings of the 15th

ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 667–676.

Pei, J., Han, J. & Wang, W. (2002), Mining sequential patterns with constraints in large databases, *in* 'Proceedings of the eleventh international conference on Information and knowledge management', ACM, pp. 18–25.

Pei, J., Han, J. & Wang, W. (2007), 'Constraint-based sequential pattern mining: the pattern-growth methods', *Journal of Intelligent Information Systems* **28**(2), 133–160.

Pemantle, R. (2000), 'Towards a theory of negative dependence', *Journal of Mathematical Physics* **41**(3), 1371–1390.

Quadrana, M., Cremonesi, P. & Jannach, D. (2018), 'Sequence-aware recommender systems', *ACM Computing Surveys (CSUR)* **51**(4), 66.

Rendle, S. (2010), Factorization machines, *in* 'Data Mining (ICDM), 2010 IEEE 10th International Conference on', IEEE, pp. 995–1000.

Rendle, S. (2012), 'Factorization machines with libfm', *ACM Transactions on Intelligent Systems and Technology (TIST)* **3**(3), 57.

Rendle, S., Freudenthaler, C., Gantner, Z. & Schmidt-Thieme, L. (2009), Bpr: Bayesian personalized ranking from implicit feedback, *in* 'Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence', AUAI Press, pp. 452–461.

Rendle, S., Freudenthaler, C. & Schmidt-Thieme, L. (2010), Factorizing personalized markov chains for next-basket recommendation, *in* 'Proceedings of the 19th international conference on World wide web', ACM, pp. 811–820.

Rendle, S., Gantner, Z., Freudenthaler, C. & Schmidt-Thieme, L. (2011), Fast context-aware recommendations with factorization machines, *in* 'Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval', ACM, pp. 635–644.

Shani, G., Heckerman, D. & Brafman, R. I. (2005), 'An mdp-based recommender system', *Journal of Machine Learning Research* **6**(Sep), 1265–1295.

Soliman, A. F., Ebrahim, G. A. & Mohammed, H. K. (2011), Speds: a framework for mining sequential patterns in evolving data streams, *in* 'Communications, Computers and Signal Processing, 2011 IEEE Pacific Rim Conference on', IEEE, pp. 464–469.

Song, Y., Cao, L., Wu, X., Wei, G., Ye, W. & Ding, W. (2012), Coupled behavior analysis for capturing coupling relationships in group-based market manipulations, *in* 'KDD'12', pp. 976–984.

Srikant, R. & Agrawal, R. (1996), Mining sequential patterns: Generalizations and performance improvements, *in* 'International Conference on Extending Database Technology', Springer, pp. 1–17.

Su, X. & Khoshgoftaar, T. M. (2009), 'A survey of collaborative filtering techniques', *Advances in artificial intelligence* **2009**.

Tzvetkov, P., Yan, X. & Han, J. (2005), 'Tsp: Mining top-k closed sequential patterns', *Knowledge and Information Systems* **7**(4), 438–457.

Wang, C. & Cao, L. (2012), Modeling and analysis of social activity process, *in* 'Behavior computing', Springer, pp. 21–35.

Wang, C., Cao, L. & Chi, C.-H. (2015), 'Formalization and verification of group behavior interactions', *Systems, Man, and Cybernetics: Systems, IEEE Transactions on* **45**(8), 1109–1124.

Wang, C., She, Z. & Cao, L. (2013), Coupled clustering ensemble: Incorporating coupling relationships both between base clusterings and objects, *in* 'Data Engineering, 2013 IEEE 29th International Conference on', IEEE, pp. 374–385.

Wang, P., Guo, J. & Lan, Y. (2014), Modeling retail transaction data for personalized shopping recommendation, *in* 'Proceedings of the 23rd ACM international conference on conference on information and knowledge management', ACM, pp. 1979–1982.

Wang, P., Guo, J., Lan, Y., Xu, J., Wan, S. & Cheng, X. (2015), Learning hierarchical representation model for nextbasket recommendation, *in* 'Proceedings of the 38th International ACM SIGIR conference on Research and Development in Information Retrieval', ACM, pp. 403–412.

Wang, Q., Sheng, V. S. & Wu, X. (2017), Keyphrase extraction with sequential pattern mining., *in* 'AAAI', pp. 5003–5004.

Wang, Q., Zeng, C., Zhou, W., Li, T., Shwartz, L. & Grabarnik, G. Y. (2017), 'Online interactive collaborative filtering using multi-armed bandit with dependent arms', *arXiv preprint arXiv:1708.03058* .

Wang, S. & Cao, L. (2017), 'Inferring implicit rules by learning explicit and hidden item dependency', *IEEE Transactions on Systems, Man, and Cybernetics: Systems* .

Wang, S., Hu, L. & Cao, L. (2017), Perceiving the next choice with comprehensive transaction embeddings for online recommendation, *in* 'Joint European Conference on Machine Learning and Knowledge Discovery in Databases', Springer, pp. 285–302.

Wang, S., Hu, L., Cao, L., Huang, X., Lian, D. & Liu, W. (2018), Attention-based transactional context embedding for next-item recommendation, AAAI.

Wang, S., Kam, K., Xiao, C., Bowen, S. R. & Chaovalitwongse, W. A. (2016), An efficient time series subsequence pattern mining and prediction framework with an application to respiratory motion prediction., *in* 'AAAI', pp. 2159–2165.

Wu, C. W., Shie, B.-E., Tseng, V. S. & Yu, P. S. (2012), Mining top-k high utility itemsets, *in* 'Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 78–86.

Wu, X., Liu, Q., Chen, E., He, L., Lv, J., Cao, C. & Hu, G. (2013), Personalized next-song recommendation in online karaokes, *in* 'Proceedings of the 7th ACM conference on Recommender systems', ACM, pp. 137–140.

Wu, X., Zhang, C. & Zhang, S. (2004), 'Efficient mining of both positive and negative association rules', *ACM Transactions on Information Systems* **22**(3), 381–405.

Xiao, J., Ye, H., He, X., Zhang, H., Wu, F. & Chua, T.-S. (2017), 'Attentional factorization machines: Learning the weight of feature interactions via attention networks', *arXiv preprint arXiv:1708.04617* .

Xu, T., Dong, X., Xu, J. & Dong, X. (2017), 'Mining high utility sequential patterns with negative item values', *International Journal of Pattern Recognition and Artificial Intelligence* **31**(10), 1750035.

Xu, T., Dong, X., Xu, J. & Gong, Y. (2017), 'E-msnsp: Efficient negative sequential patterns mining based on multiple minimum supports', *International Journal of Pattern Recognition and Artificial Intelligence* **31**(02), 1750003.

Yap, G.-E., Li, X.-L. & Philip, S. Y. (2012), Effective next-items recommendation via personalized sequential pattern mining, *in* 'International Conference on Database Systems for Advanced Applications', Springer, pp. 48–64.

Yin, J., Zheng, Z. & Cao, L. (2012), Uspan: an efficient algorithm for mining high utility sequential patterns, *in* 'Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 660–668.

Yin, J., Zheng, Z., Cao, L., Song, Y. & Wei, W. (2013), Efficiently mining top-k high utility sequential patterns, *in* '2013 IEEE 13th International Conference on Data Mining', IEEE, pp. 1259–1264.

Ying, H., Zhuang, F., Zhang, F., Liu, Y., Xu, G., Xie, X., Xiong, H. & Wu, J. (2018), Sequential recommender system based on hierarchical attention networks, *in* 'the 27th International Joint Conference on Artificial Intelligence'.

Yu, F., Liu, Q., Wu, S., Wang, L. & Tan, T. (2016), A dynamic recurrent model for next basket recommendation, *in* 'Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval', ACM, pp. 729–732.

Yuan, Y., Zheng, X. & Lu, X. (2016), 'Discovering diverse subset for unsupervised hyperspectral band selection', *IEEE Transactions on Image Processing* **26**(1), 51–64.

Zaki, M. J. (2001), 'Spade: An efficient algorithm for mining frequent sequences', *Machine learning* **42**(1-2), 31–60.

Zhao, Q., Willemsen, M. C., Adomavicius, G., Harper, F. M. & Konstan, J. A. (2018), Interpreting user inaction in recommender systems, *in* 'Proceedings of the 12th ACM Conference on Recommender Systems', ACM, pp. 40–48.

Zhao, X., Zhang, L., Ding, Z., Xia, L., Tang, J. & Yin, D. (2018), Recommendations with negative feedback via pairwise deep reinforcement learning, *in* 'Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining', ACM, pp. 1040–1048.

Zhao, X., Zhang, W. & Wang, J. (2013), Interactive collaborative filtering, *in* 'Proceedings of the 22nd ACM international conference on Information & Knowledge Management', ACM, pp. 1411–1420.

Zhao, Y., Zhang, H., Cao, L., Zhang, C. & Bohlscheid, H. (2008), Efficient mining of event-oriented negative sequential rules, *in* 'Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on', Vol. 1, IEEE, pp. 336–342.

Zhao, Y., Zhang, H., Cao, L., Zhang, C. & Bohlscheid, H. (2009), Mining both positive and negative impact-oriented sequential rules from transactional data, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer, pp. 656–663.

Zhao, Y., Zhang, H., Figueiredo, F., Cao, L. & Zhang, C. (2007), Mining for combined association rules on multiple datasets, *in* 'DDDM'07', pp. 18–23.

Zhao, Y., Zhang, H., Wu, S., Pei, J., Cao, L., Zhang, C. & Bohlscheid, H. (2009), Debt detection in social security by sequence classification using both positive and negative patterns, *in* 'Joint European Conference on Machine Learning and Knowledge Discovery in Databases', Springer, pp. 648–663.

Zheng, Z. (2012), Negative Sequential Pattern Mining, PhD thesis, University of Technology, Sydney.

Zheng, Z., Wei, W., Liu, C., Cao, W., Cao, L. & Bhatia, M. (2016), 'An effective contrast sequential pattern mining approach to taxpayer behavior analysis', *World Wide Web* **19**(4), 633–651.

Zheng, Z., Zhao, Y., Zuo, Z. & Cao, L. (2009), Negative-gsp: An efficient method for mining negative sequential patterns, *in* 'Proceedings of the Eighth Australasian Data Mining Conference-Volume 101', Australian Computer Society, Inc., pp. 63–67.

Zheng, Z., Zhao, Y., Zuo, Z. & Cao, L. (2010), An efficient ga-based algorithm for mining negative sequential patterns, *in* 'Advances in Knowledge Discovery and Data Mining', Springer, pp. 262–273.

Zimdars, A., Chickering, D. M. & Meek, C. (2001), Using temporal data for making recommendations, *in* 'Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence', Morgan Kaufmann Publishers Inc., pp. 580–588.