Interactive Evaluation of Recommender Systems with SNIPER - An Episode Mining Approach

Sandy Moens
University of Antwerp
Belgium

Olivier Jeunen
University of Antwerp
Belgium

Bart Goethals
University of Antwerp, Belgium
Froomle, Belgium
Monash University, Australia

ABSTRACT

Recommender systems are typically evaluated using either offline methods, online methods, or through user studies. In this paper we take an episode mining approach to analysing recommender system data and we demonstrate how we can use SNIPER, a tool for interactive pattern mining, to analyse and understand the behaviour of recommender systems. We describe the required data format, and present a useful scenario of how a user can interact with the system to answer questions about the quality of recommendations.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; **Data mining**; • **Human-centered computing** → *Interactive systems and tools.*

KEYWORDS

recommender systems;evaluation;episode mining

ACM Reference Format:
Sandy Moens, Olivier Jeunen, and Bart Goethals. 2019. Interactive Evaluation of Recommender Systems with SNIPER - An Episode Mining Approach. In *Thirteenth ACM Conference on Recommender Systems (RecSys '19), September 16–20, 2019, Copenhagen, Denmark.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3298689.3346965

1 INTRODUCTION

Recommender systems are typically evaluated on a high level of abstraction. In an offline setting, competing algorithms are evaluated for some given datasets and some chosen metrics, in order to identify the best performers. When evaluating online, multiple model variants are deployed and assigned some portion of live traffic. The resulting interactions between users and the system are measured, and subsequently used to determine which algorithm should be favoured [4]. However, the fine-grained nature of said interactions is entirely lost by aggregating them in this way. In order to better *understand* how the system behaves, interesting sequential patterns that occur in the data can be examined. Although useful, this is a non-trivial task for non-expert users.

In this demo paper, we present our solution to this issue: we extend SNIPER, an existing tool for interactive pattern mining, with episode mining, which we exploit to analyse and understand recommender systems data. We distinguish between different types of interactions, such as *pageviews* (user-item interactions), *impressions* (shown recommendations), and *clicks* (on shown recommendations). Our system facilitates exploratory analysis of frequent patterns in the data, and aims to provide users with a more thorough understanding of the effects of live recommender systems.

2 INTERACTIVE EPISODE MINING

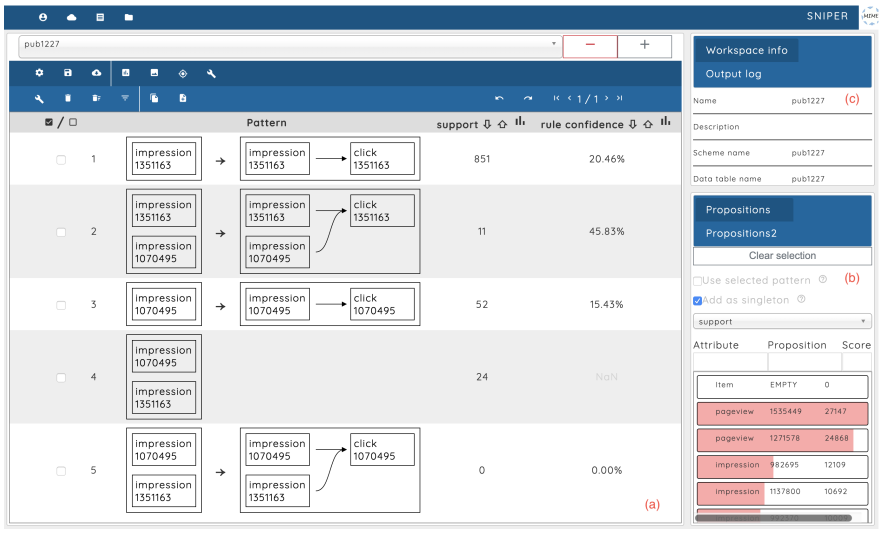
2.1 SNIPER

Description SNIPER (SNappy Interactive Pattern ExploreR) is an improved version of MIME [3], a tool for interactive pattern exploration. Fig. 1 shows the front-end in which users can interactively construct new patterns using basic building blocks and/or adapt existing patterns [3]. In addition, SNIPER offers a range of algorithms to produce patterns, and a variety of interestingness measures and visualisations to qualitatively evaluate and interpret these patterns.
Implementation The back-end of SNIPER is written in Java using Spring. We used RealKD[1] for its data model, implementation of measures, pattern mining algorithms, etc. We extended this framework with episode (rule) mining. The front-end is written in HTML using JavaScript and libraries for interactivity, querying, layout, and visualisation (JQuery, BootStrap, D3JS, etc.).
Deployment We provide a binary as well as the source code[2]. For correct installation, we require Java and MySQL to be installed. We provide additional configuration files for proper deployment.

2.2 Episode Mining

SNIPER has been developed for analysing transaction datasets using itemsets and association rules [2, 3]. We extended its functionality with support for *single sequence* datasets of events. An event consists of a label and a timestamp. This results in ordered sequences, such that 1) gaps are allowed and 2) multiple events can occur at the same time. An *episode* is a DAG (Directed Acyclic Graph), with events as nodes, and edges defining a precedence relation between nodes [1]. An occurrence of an episode in the sequence is defined by its *window* (i.e., the pair $(\text{time}_{\text{start}}; \text{time}_{\text{end}})$). Typically, one is interested in episodes that occur often and such that their windows are small, i.e., events happen close to one another. Given two episodes G and H, such that G is a sub episode of H, an *episode rule* $G \Rightarrow H$ represents the implication that "if G occurs then also H occurs within a user defined *reference window*". The *confidence* then represent the fraction of occurrences of G that can be extended to H. More information can be found in the work by Cule et al. [1].

[1]https://bitbucket.org/realKD/realkd
[2]https://bitbucket.org/sandymoens/sniper



	Pattern	support	rule confidence
1	impression 1351163 → impression 1351163 → click 1351163	851	20.46%
2	impression 1351163, impression 1070495 → impression 1351163, impression 1070495 → click 1351163	11	45.83%
3	impression 1070495 → impression 1070495 → click 1070495	52	15.43%
4	impression 1070495, impression 1351163	24	NaN
5	impression 1070495, impression 1351163 → impression 1070495, impression 1351163 → click 1070495	0	0.00%

Name	pub1227
Description	
Scheme name	pub1227
Data table name	pub1227

☐ Use selected pattern
☑ Add as singleton
support

Attribute	Proposition	Score
Item	EMPTY	0
pageview	1535449	27147
pageview	1271578	24868
impression	982695	12109
impression	1137800	10692
impression	992370	10009

Figure 1: Front-end: (a) work dock containing constructed and mined patterns together with their quality scores and visualisations, (b) source dock containing the basic building blocks for constructing patterns, and (c) information window.

id	timestamp	imp	click
u1	10/01/2019 10:20:30	1	
u1	10/01/2019 10:20:30	2	
u1	10/01/2019 10:25:00		1
u2	15/01/2019 20:45:00	2	
u2	15/01/2019 20:45:00	3	
u2	15/01/2019 20:45:10		2
u1	01/02/2019 15:00:00	2	
u1	01/02/2019 15:00:00	3	

(a)

timestamp	imp	click
1	1	
1	2	
2		1
3	2	
3	3	
7	2	
7	3	
8		2

(b)

Figure 2: Example data transformation: a) original input sequences encoded in a single table, and b) transformed single sequence grouping together sequences of the same user (indicated by colour).

2.3 Data Format

Recommender systems log for identifiable users her interactions such as *pageviews*, *impressions*, and *clicks*, resulting in a single sequence per user. In SNIPER we encode these different interactions by different columns in the data. Fig. 2a shows an example of such a data table. To ensure proper analysis, different types of interactions should not occur at the same timestamp. Therefore, each row can have at most one type, leaving the other column values empty. Within SNIPER this data is converted to a single sequence by grouping interactions of single users in consecutive blocks of events, and placing a large gap between sequences of different users. An example transformation is shown in Fig. 2.

3 SYSTEM DEMONSTRATION

We show an example of an interesting pattern (episode rule) extracted from the Outbrain[3] dataset. We filtered the data coming from page views that are based on a single publisher, for which each document has a decent number of impressions.

Fig. 1 shows in row 1 a basic episode rule for the impression and clicking of item 1351163. We see that the confidence of first showing and then clicking the item equals 20.46%. Note that this is essentially the click-through-rate (CTR). That is, in about one in five cases in which 1351163 has been recommended, it has also been clicked. Now adding in another impression 1070495 (row 2), we see a very large improvement in CTR, even more than doubling the original one (row 1). We can now investigate this behaviour a bit further, since it is well possible that 1070495 has a very low CTR on its own, and therefore increases the CTR of 1351163 when recommending both. Row 3 shows the global CTR of 1070495, while row 5 shows the CTR when also 1351163 has been recommended. Row 4 shows that the combination of the two recommendation occurs 24 times. However, in none of these cases has 1070495 been clicked (row 5). We might assume here that the addition of the second recommendation boosts the CTR for the first one. Scenario's such as these can help to better understand why recommenders might produce good or bad results. We extended SNIPER with additional post processing methods to interactively analyse such behaviour. A full use case is shown in the demo video[4].

4 CONCLUSION

We extended SNIPER with support for single sequence data and episode (rule) mining and showed how it can be used to analyse the behaviour of recommender systems. We have shown how we can easily prepare typical recommender system data into the required format, however, in doing so losing the gaps between different events. Moreover, we have shown how a typical analysis of such data can take form. As future work, we can first consider using true timestamps when transforming the data internally. Secondly, we can extend our tool with preprocessing algorithms that automatically prune uninteresting episode rules based on the information provided by its sub patterns. Finally, we can explore how to incorporate data from different recommender system algorithms (e.g., A/B tested) and how to compare these results.

REFERENCES

[1] Boris Cule, Nikolaj Tatti, and Bart Goethals. 2014. Marbles: Mining association rules buried in long event sequences. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 7, 2 (2014), 93–110.
[2] Bart Goethals. 2003. Survey on frequent pattern mining. *Univ. of Helsinki* 19 (2003), 840–852.
[3] Bart Goethals, Sandy Moens, and Jilles Vreeken. 2011. MIME: a framework for interactive visual pattern mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 757–760.
[4] Asela Gunawardana and Guy Shani. 2015. Evaluating recommender systems. In *Recommender systems handbook*. Springer, 265–308.

[3]https://www.kaggle.com/c/outbrain-click-prediction/data

[4]https://youtu.be/S23qbU1PbhY