StreamingRec: A Framework for Benchmarking Stream-based News Recommenders

Michael Jugovac
TU Dortmund
Dortmund, Germany
michael.jugovac@tu-dortmund.de

Dietmar Jannach
Alpen-Adria-Universität
Klagenfurt, Austria
dietmar.jannach@aau.at

Mozhgan Karimi
University of Antwerp
Antwerp, Belgium
mozhgan.karimi@uantwerpen.be

ABSTRACT

News is one of the earliest application domains of recommender systems, and recommending items from a virtually endless stream of news is still a relevant problem today. News recommendation is different from other application domains in a variety of ways, e.g., because new items constantly become available for recommendation. To be effective, news recommenders therefore have to continuously consider the latest items in the incoming stream of news in their recommendation models. However, today's public software libraries for algorithm benchmarking mostly do not consider these particularities of the domain. As a result, authors often rely on proprietary protocols, which hampers the comparability of the obtained results. In this paper, we present `StreamingRec` as a framework for evaluating streaming-based news recommenders in a replicable way. The open-source framework implements a replay-based evaluation protocol that allows algorithms to update the underlying models in real-time when new events are recorded and new articles are available for recommendation. Furthermore, a variety of baseline algorithms for session-based recommendation are part of `StreamingRec`. For these, we also report a number of performance results for two datasets, which confirm the importance of immediate model updates.

CCS CONCEPTS

• **Information systems → Recommender systems**; *Collaborative filtering*; • **General and reference** → Evaluation;

KEYWORDS

News Recommendation; Evaluation; Benchmarking

ACM Reference Format:
Michael Jugovac, Dietmar Jannach, and Mozhgan Karimi. 2018. `StreamingRec`: A Framework for Benchmarking Stream-based News Recommenders. In *Twelfth ACM Conference on Recommender Systems (RecSys '18), October 2–7, 2018, Vancouver, BC, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3240323.3240384

1 INTRODUCTION

The recommendation of news—in particular in the form of *Usenet* messages [41]—was one of the driving application scenarios for the development of collaborative filtering (CF) methods. Today, almost 25 years later, ranking and filtering objects appearing in a virtually endless stream of information is still a relevant problem, e.g., in the context of social media feeds or news aggregation sites [1, 17, 38, 44]. Since the early days of nearest-neighbor-based methods, substantial progress has been made in terms of the development of more sophisticated machine learning algorithms. Such advanced methods were also successfully explored in the news domain. An early version of Google's news recommender is, for example, based on collaborative filtering techniques [7]. More recently, a variety of other techniques were proposed, which also take some of the particularities of the domain into account, e.g., by (i) considering content information to deal with the permanent item cold-start situation, (ii) by considering the freshness of news items, and (iii) by taking a reader's short-term interests into account [2, 4, 34, 46].

The fast pace in which news articles can be outdated or superseded by newer information is in fact key characteristics of the news domain. This, in turn, means that there can be limits regarding the applicability of complex CF techniques that rely on pre-trained models, as many of these approaches do not support immediate model updates. Updating models frequently is, however, required to consider newly available items and to react to short-term popularity trends in the reader community, as was demonstrated in the CLEF/NewsREEL challenge series [37]. Thus, until CF models can be re-trained, less effective fallback techniques have to be used, e.g., based on general item popularity or content information.

In academia, the common matrix completion problem abstraction and corresponding evaluation procedures are not well suited to assess the capability of an algorithm to deal (a) with the constant stream of new items and emerging community trends and (b) the changing short-term interests of news readers [10, 25, 33, 34]. Instead, protocols for sessions- or stream-based recommendation scenarios have to be applied, see [39]. Unfortunately, since no evaluation standards are yet defined in the field, researchers often design and implement their own procedures. This, in turn can make replicability and comparability of results an issue.

With this work, our goal is to contribute to more realistic and replicable research practices in the art of stream-based news recommendation. We present the extensible `StreamingRec` framework, which implements a stream-based evaluation protocol that supports real-time model updates. It also comprises a set of conceptually simple, yet effective baseline algorithms upon which other researchers can build. We benchmark the implemented strategies on different datasets, and the results highlight the importance of being able to immediately consider new items and click trends. We share the source code of the framework online.[1]

[1]https://github.com/mjugo/StreamingRec

RecSys '18, October 2–7, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5901-6/18/10…$15.00
https://doi.org/10.1145/3240323.3240384

2 BACKGROUND AND RELATED WORK

2.1 Evaluating News Recommenders

Offline experiments using precision and recall in particular in terms of article *click-throughs* [6, 31] are the predominant way of comparing recommendation algorithms in the news domain [26]. Regarding the used evaluation protocols, researchers typically apply time-based data splitting procedures [3], where the data of a few days or weeks is used for training and the learned models are then used to predict article views on the next day for a given user [14, 32, 35]. However, as discussed, e.g., in [5], such traditional evaluation setups have their limitations in the news domain. In particular, the outcomes of several instances of the CLEF/NewsREEL challenge [20, 28] showed that it is crucial to consider (popular) articles that appeared in the last few minutes in the recommendation process to achieve high click-through rates in real-world settings [37]. Evaluation protocols that do not consider these temporal dynamics and assume the set of recommendable items to be static for the entire test period might not capture the reality well, leading to significant differences between online and offline performance results [15].

Another distinctive feature of the news domain is that model-based algorithms that are designed for matrix completion setups, which perform very well in other domains, have their limitations in the news domain because of the potentially fast-changing short-term reading interests of the users. To be effective, an algorithm has to adapt its recommendations to these interests immediately during the user's current session. A possible solution for this problem is the use of session-based recommendation algorithms, as a special form of context-aware recommenders [39]. Such algorithms are designed to take the very last actions of an anonymous or known user into account in real-time. Depending on the domain, being able to react to such short-term interest changes can be much more important than models that capture the user's longer-term interests [23].

2.2 Research Datasets

A significant amount of earlier academic works is based on proprietary, non-public datasets, e.g., from Yahoo! or Google, sometimes leading to limited replicability of existing research. In recent years, however, different public datasets have become available for researchers, e.g., the Outbrain, Plista, and SmartMedia News Adressa datasets.[2] These datasets contain detailed user interaction logs of news platforms and allow researchers to "replay" the click stream offline, e.g., with the help of the Idomaar framework [21, 43]. Like Idomaar, our framework is designed to process such click-stream logs, leading to a more realistic evaluation setting that bridges the gap between academia and industry.

2.3 Frameworks for Algorithm Benchmarking

Over the last few years, a number of software libraries for benchmarking recommendation algorithms were published, e.g., [9, 13, 16, 42]. However, as mentioned, these frameworks mostly implement algorithms that make non-contextualized recommendations based on a given user-item ratings matrix, which does not fully match the requirements of the news domain.

Only a small number of evaluation frameworks have been proposed in the past that are targeted specifically toward real-time recommendation. While, in some cases, replicability is limited because implementation details are not disclosed [8], in other cases, the source code is shared publicly. The above-mentioned Idomaar framework [43], for example, was designed as a reference architecture for realistic offline evaluation of news recommenders in industrial settings. `StreamingRec` is inspired in different ways by Idomaar, e.g., with respect to its extensible architecture and the implemented evaluation protocol. Differently from Idomaar, however, our aim is to create a light-weight environment for academic researchers with a richer set of pre-implemented algorithms. One specific goal was also to avoid dependencies to other state-of-the-art technologies like Apache Kafka that are often not required for academic evaluations.

Outside the news domain, other general-purpose evaluation frameworks were proposed that cover some of the critical aspects of the news domain. The *Alpenglow* and *FluRS* [11, 30] frameworks, for example, implement versions of nearest-neighbor and matrix factorization methods that support incremental model updates. However, these methods are based on the matrix completion problem abstraction and are—in contrast to the session-based recommendation algorithms implemented in our framework—not designed to take the short-term reader interests into account.

3 THE STREAMINGREC FRAMEWORK

With the proposed `StreamingRec` framework, our goal is to address the various challenges of benchmarking news recommendation algorithms in a realistic way in one single and extensible environment. Specifically, the framework should support session-based recommendation strategies and the immediate consideration of new items and user interactions. Furthermore, our goal is to provide a set of competitive baselines for other researchers to test their models.

3.1 Design and Evaluation Protocol

The offline evaluation strategy of `StreamingRec` is based on *replaying* the recorded user interactions as was done in one task of the CLEF/NewsREEL challenge. The provided log data is split into a training and a test set as usual, where the training set is revealed to the algorithms in the beginning to learn their models. The interactions of the test set are then incrementally provided to the algorithm. Upon each new action, the task of the algorithm is to predict which other items the user will read (click) next. The recommendations can then be compared with the true next actions of the particular user session to determine different accuracy measures like precision, recall, F1, or the mean reciprocal rank (MRR).

Besides common accuracy measures, the framework records a number of additional quality factors for recommenders, like the article catalog coverage or the number of provided recommendations per request. Running times for training and recommending are recorded as well, and further custom evaluation measures can be implemented easily via straight-forward interfaces. The framework is written in Java, allowing an easy integration with enterprise-scale architectures and the use of parallel execution threads.

3.2 Pre-Implemented Algorithms

Since immediate model updates are crucial in the news domain, `StreamingRec` implements a number of algorithms that can immediately incorporate the incoming events into their predictions. The algorithms include trivial baselines that, e.g., simply recommend the latest published articles (Recently Published), the overall most clicked articles (Most Popular), or the articles that were clicked most often during the last N minutes (Recently Popular). Another basic strategy simply recommends items in the order of when they received their last click in the event stream (Recently Clicked). While trivial, such methods have shown to be highly effective in practice in the context of the 2017 CLEF/NewsReel challenge [37].

The more advanced, but still conceptually simple pre-implemented algorithms include a variant of an item-to-item based collaborative filtering approach (Item-Item CF), a session-based nearest neighbor technique that proved to be highly effective for session-based recommendation tasks (SkNN) [24], and a recent sequence-aware extension of it (V-SkNN), which outperformed even a recent deep learning-based approach in [36]. In the latter case, the SkNN scheme is extended by a weighting function that puts more emphasis on the most recent clicks of the target user when calculating session similarities. The scalability of both neighborhood-based approaches is ensured through neighborhood sampling techniques.

Two association rule techniques are also part of the framework: one based on pairwise item co-occurrences in a session (Co-Occurrence) and one based on sequential pattern mining (Sequential Pattern), implemented in a similar way as in [18] for improved efficiency. Finally, the framework also includes two basic content-based approaches, which could be used in the future to build effective hybrid approaches, as in [29]. One is based on article keyword overlaps (Keywords), and one computes article similarities with the help of the Lucene[3] framework (Lucene). Further algorithms, such as more complex incremental CF approaches [5, 7, 22, 27], can be implemented via pre-defined Java interfaces. According to our experiments, session-based algorithms are particularly promising.

4 EXPERIMENTS

We conducted a number of experiments with the pre-implemented algorithms of our framework and one recent deep learning approach to assess their relative performances and to gauge the importance of immediate model updates in the news domain.

4.1 Datasets and Experimental Procedure

We report the results for two datasets. From the click logs provided by Outbrain and Plista [28], we selected one medium-sized publisher each (with publisher IDs 43 and 418, respectively). We determined user sessions based on idle times between successive clicks, using a threshold of 20 minutes. Table 1 shows statistics for these datasets after removing single-click sessions.

We split the datasets using a time-based criterion, with 70% of the data used for training and the remaining 30% for testing. In the case of the Outbrain dataset, this means that the data of about the last six days remained for testing. During the test phase, algorithms can update their model with the newly available data. We optimized the parameters for the tested algorithms on a separate validation set consisting of 10% of the training data.[4]

[2] https://www.kaggle.com/c/outbrain-click-prediction/data, http://www.clef-newsreel.org/, http://reclab.idi.ntnu.no/dataset/

[3] https://lucene.apache.org/

[4] The detailed final parameter settings can be found online as part of the shared code.

Table 1: Dataset Characteristics

	Outbrain		*Plista*	
Clicks		1,067,675		1,129,408
Users		281,910		220,117
Items		1,475		835
Sessions		421,620		355,300
	Avg.	*Med.*	*Avg.*	*Med.*
Clicks per item	723.8	49	1,352.6	268
Clicks per user	3.8	2	5.1	3
Sessions per user	1.5	1	1.6	1
Clicks per Session	2.5	2	3.2	2

4.2 Comparison With Complex Models

One main hypothesis of our research is that periodically updating more complex models is not sufficient in the highly dynamic news domain. We therefore included two representatives of more complex models in the comparison. First, we used *Bayesian Personalized Ranking* (BPR) [40] as a representative of a broadly used learning-to-rank method for implicit feedback. The method is designed for matrix completion problem settings and therefore session-agnostic, i.e., it will not take the most recent actions of a user into account when recommending. The second alternative algorithm is GRU4Rec [19], which is a deep learning-based method for session-based recommendation using recurrent neural networks.

Both methods, BPR and GRU4Rec, do not support incremental updates. Therefore, we re-trained the models periodically during the evaluation of the test set. Since BPR needed considerable time for training for our datasets, we updated its model after processing the events of one day in the test set before continuing the evaluation. For GRU4Rec, we employed a heuristic to sample from the more recent sessions to take recent temporal shifts into account, as proposed in [45]. Besides performance improvements, this sampling significantly reduces the training time. In our experiments, we re-trained GRU4Rec after processing *one hour* of the data in the test set.[5]

4.3 Results

Figure 1 shows the results of our evaluations for the Outbrain and Plista datasets. In the following section, we will describe the results for the Outbrain dataset in detail and only discuss the Plista data results when there are interesting differences, because, in general, the results are quite similar. We report the typical information retrieval measures F1 and the mean reciprocal rank (MRR), using a list length of 10. In addition, we measured how many different items appeared in the top-10 lists of each algorithm in order to detect potential concentration biases. Finally, we report the time needed by each algorithm to generate a recommendation list.

Accuracy. Looking at the accuracy results, we can observe three groups of algorithms that exhibit comparable performance.

(1) The lowest accuracy results are, as expected, achieved by methods that do not consider article recency, recent community trends, or the context of the current user session. Recommending the most popular articles in the training set is not much better than a random recommendation strategy, even though the entire dataset only covers two weeks. This shows how quickly news articles

[5] The Python-based GRU4Rec implementation from [19] is not part of `StreamingRec`, but was integrated through a web service interface.

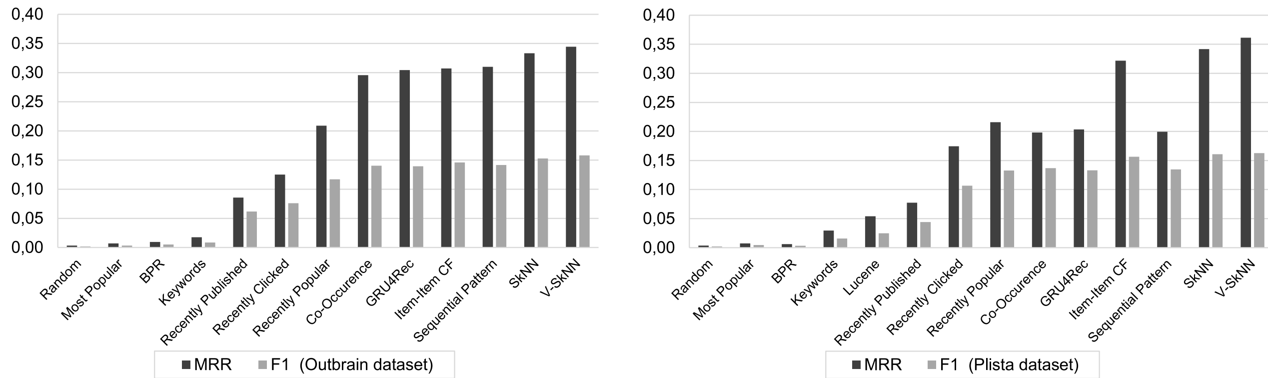

Figure 1: Evaluation Results in Terms of MRR and F1 @10 for the Outbrain Dataset (left) and the Plista Dataset (right)

become outdated. Considering the context of the ongoing session via the keywords appearing in the articles in a pure content-based approach is not much better, because recency and popularity aspects are not considered. The performance of BPR is also low, because it was not designed for the problem setting and cannot take new articles immediately into account. This is in line with the findings from a similar, yet session-agnostic, stream-based evaluation [12], where batch-trained matrix factorization methods also performed worse than even simple item-item transition methods.

(2) The next group of techniques, which perform considerably better, do not take the user's current click history into account and are based on article recency or click trends. The Recently Popular strategy is the best in this group and is also very similar to the best performing method in the *online part* of the 2017 CLEF/NewsREEL challenge. Interestingly, even the very simple and efficient Recently Clicked baseline performs quite well for both datasets and metrics. The Recently Published method, in contrast, is the least effective in this group, but could be a reasonable fallback method in real-world systems when no click data is available.

(3) The best performing group of algorithms all consider the context of the current session of the users and are able to update their recommendation models immediately with every click, or, as in the case of GRU4Rec, at least every hour. Co-occurrence patterns of the style "Customers who read ... also read ..." are computationally very simple, but already quite effective; also the Item-Item CF method is computationally efficient and leads to good results.

The deep learning technique GRU4Rec achieves competitive results when its model is re-trained every hour with the latest 40,000 sessions. Additional experiments (not shown in the figure) revealed that re-training GRU4Rec with the entire training dataset, but only every 24 hours, leads to far worse results.[6]

The best overall results were achieved with the recent V-SkNN method, which considers the ordering of the events when searching for similar sessions in the training data. The differences to the second-best method are statistically significant with $p < 0.01$ according to a Kolmogorov-Smirnov test. The findings are in general in line with those reported in [36], where in many cases the consideration of the sequence of events leads to a performance improvement for session-based kNN methods.

Plista Dataset Results. The V-SkNN method is also the winning strategy for the Plista dataset. Interestingly, the relative performance of the Sequential Pattern, Co-Occurrence, and GRU4Rec approaches is lower for this dataset and seems to be dependent on the characteristics of the data, as was observed also in [36]. The Lucene strategy, which we could only evaluate on the Plista dataset, was slightly better than the keyword-based strategy. It did, however, not reach the performance level of the simple baselines.

Aggregate Diversity. In terms of the number of different items that are recommended in the top-10 (not shown in the figure), the session-based nearest neighbor methods and the content-based strategies exhibit the highest diversity level. The differences compared to many other techniques are, however, often small. Only some algorithms, e.g., those based on general popularity, by design recommend a very small set of items.

Computation Times. Complex models like GRU4Rec and BPR can need significant computational resources for training. At prediction time, most techniques need less than 1 ms in our evaluation environment, a standard desktop computer. The V-SkNN method was the slowest one in this comparison, but still needed less than 5 ms to generate one recommendation list.

Overall, like in [24], nearest neighbor methods proved to be strong baselines for session-based recommendation scenarios and can be implemented in a computationally efficient manner based on efficient data structures and neighborhood sampling.

5 SUMMARY

Open source evaluation frameworks are an important means to foster replicable research in recommender systems. With our work, we contribute a new framework that is specifically designed to deal with stream-based recommendation scenarios, and the empirical evaluations reported in the paper highlight the importance of immediate model updates. The framework uses a realistic replay evaluation protocol and includes several conceptually simple, yet effective algorithms. These algorithms can in the future serve as baselines for benchmarking more sophisticated models, hybrids that combine multiple strategies, or techniques that personalize the recommendations across sessions.

[6] We tested even lower update frequencies, e.g., every ten minutes, using the most recent 10,000 sessions for training. This, however, did not improve the performance.

REFERENCES

[1] Shlomo Berkovsky, Jill Freyne, and Gregory Smith. 2012. Personalized Network Updates: Increasing Social Interactions and Contributions in Social Networks. In *Proceedings of the 20th International Conference on User Modeling, Adaptation, and Personalization (UMAP '12)*. 1–13.

[2] Daniel Billsus, Michael J. Pazzani, and James Chen. 2000. A Learning Agent for Wireless News Access. In *Proceedings of the 5th International Conference on Intelligent User Interfaces (IUI '00)*. 33–36.

[3] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1 (2014), 67–119.

[4] Iván Cantador and Pablo Castells. 2009. Semantic contextualisation in a news recommender system. In *Proceedings of the 2009 Workshop on Context-Aware Recommender Systems (CARS '09)*.

[5] Badrish Chandramouli, Justin J. Levandoski, Ahmed Eldawy, and Mohamed F. Mokbel. 2011. StreamRec: A Real-time Recommender System. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data (SIGMOD '11)*. 1243–1246.

[6] Wei Chu and Seung-Taek Park. 2009. Personalized Recommendation on Dynamic Content Using Predictive Bilinear Models. In *Proceedings of the 18th International Conference on World Wide Web (WWW '09)*. 691–700.

[7] Abhinandan S. Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. 2007. Google News Personalization: Scalable Online Collaborative Filtering. In *Proceedings of the 16th International Conference on World Wide Web (WWW '07)*. 271–280.

[8] Doychin Doychev, Aonghus Lawlor, Rachael Rafter, and Barry Smyth. 2014. An Analysis of Recommender Algorithms for Online News. In *Working Notes for the 2014 Conference and Labs of the Evaluation Forum (CLEF '14)*. 825–836.

[9] Michael D. Ekstrand, Michael Ludwig, Joseph A. Konstan, and John T. Riedl. 2011. Rethinking the Recommender Research Ecosystem: Reproducibility, Openness, and LensKit. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. 133–140.

[10] Blaz Fortuna, Carolina Fortuna, and Dunja Mladenic. 2010. Real-Time News Recommender System. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD '10)*. 583–586.

[11] Erzsébet Frigó, Róbert Pálovics, Domokos Kelen, Levente Kocsis, and András A. Benczúr. 2017. Alpenglow: Open Source Recommender Framework with Time-aware Learning and Evaluation. In *Proceedings of the Poster Track of the 11th ACM Conference on Recommender Systems (RecSys '17)*.

[12] Erzsébet Frigó, Róbert Pálovics, Domokos Kelen, Levente Kocsis, and András A. Benczúr. 2017. Online Ranking Prediction in Non-stationary Environments. In *Proceedings of the 1st Workshop on Temporal Reasoning in Recommender Systems co-located with RecSys '17 (RecTemp '17)*. 28–34.

[13] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. MyMediaLite: A Free Recommender System Library. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. 305–308.

[14] Qi Gao, Fabian Abel, Geert-Jan Houben, and Ke Tao. 2011. Interweaving Trend and User Modeling for Personalized News Recommendation. In *Proceedings of the 2011 International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT '11)*. 100–103.

[15] Florent Garcin, Boi Faltings, Olivier Donatsch, Ayar Alazzawi, Christophe Bruttin, and Amr Huber. 2014. Offline and Online Evaluation of News Recommender Systems at Swissinfo.Ch. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. 169–176.

[16] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. 2015. LibRec: A Java Library for Recommender Systems. In *Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd Conference on User Modelling, Adaptation and Personalization (UMAP '15)*.

[17] Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. 2010. Social Media Recommendation Based on People and Tags. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '10)*. 194–201.

[18] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. 2004. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery* 8, 1 (2004), 53–87.

[19] Balázs Hidasi and Alexandros Karatzoglou. 2017. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. *CoRR* (2017). arXiv:1706.03847 [cs.LG] http://arxiv.org/abs/1706.03847

[20] Frank Hopfgartner, Torben Brodt, Jonas Seiler, Benjamin Kille, Andreas Lommatzsch, Martha Larson, Roberto Turrin, and András Serény. 2016. Benchmarking News Recommendations: The CLEF NewsREEL Use Case. *SIGIR Forum* 49, 2 (2016), 129–136.

[21] Frank Hopfgartner, Benjamin Kille, Tobias Heintz, and Roberto Turrin. 2015. Real-time Recommendation of Streamed Data. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. 361–362.

[22] Yanxiang Huang, Bin Cui, Wenyu Zhang, Jie Jiang, and Ying Xu. 2015. TencentRec: Real-time Stream Recommendation in Practice. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD '15)*. 227–238.

[23] Dietmar Jannach, Lukas Lerche, and Michael Jugovac. 2015. Adaptation and Evaluation of Recommendations for Short-term Shopping Goals. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. 211–218.

[24] Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks Meet the Neighborhood for Session-Based Recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys '17)*. 306–310.

[25] Dietmar Jannach, Paul Resnick, Alexander Tuzhilin, and Markus Zanker. 2016. Recommender Systems - Beyond Matrix Completion. *Commun. ACM* 59, 11 (2016), 94–102.

[26] Mozhgan Karimi, Dietmar Jannach, and Michael Jugovac. 2018. News recommender systems—Survey and roads ahead. *Information Processing & Management* (2018). In press.

[27] Mohammad Khoshneshin and W. Nick Street. 2010. Incremental Collaborative Filtering via Evolutionary Co-clustering. In *Proceedings of the Fourth ACM Conference on Recommender Systems (RecSys '10)*. 325–328.

[28] Benjamin Kille, Frank Hopfgartner, Torben Brodt, and Tobias Heintz. 2013. The Plista Dataset. In *Proceedings of the 2013 International News Recommender Systems Workshop and Challenge (NRS '13)*. 16–23.

[29] Evan Kirshenbaum, George Forman, and Michael Dugan. 2012. A Live Comparison of Methods for Personalized Article Recommendation at Forbes.com. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD '12)*. 51–66.

[30] Takuya Kitazawa. 2017. FluRS: A Python Library for Online Item Recommendation. (2017). Retrieved April 30, 2018 from https://takuti.me/note/flurs/

[31] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. 2010. A Contextual-bandit Approach to Personalized News Article Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. 661–670.

[32] Lei Li and Tao Li. 2013. News recommendation via hypergraph learning: encapsulation of user behavior and news content. In *Proceedings of the 6th International Conference on Web Search and Data Mining (WSDM '13)*. 305–314.

[33] Lei Li, Li Zheng, and Tao Li. 2011. LOGO: A Long-Short User Interest Integration in Personalized News Recommendation. In *Proceedings of the 5th Conference on Recommender Systems (RecSys '11)*. 317–320.

[34] Lei Li, Li Zheng, Fan Yang, and Tao Li. 2014. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications* 41, 7 (2014), 3168 – 3177.

[35] Chen Lin, Runquan Xie, Xinjun Guan, Lei Li, and Tao Li. 2014. Personalized news recommendation via implicit social experts. *Information Sciences* 254 (2014), 1–18.

[36] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of Session-Based Recommenation Algorithms. (2018). arXiv:1803.09587 [cs.IR] https://arxiv.org/abs/1803.09587

[37] Cornelius A. Ludmann. 2017. Recommending News Articles in the CLEF News Recommendation Evaluation Lab with the Data Stream Management System Odysseus. In *Working Notes for the 2017 Conference and Labs of the Evaluation Forum (CLEF '17)*.

[38] Owen Phelan, Kevin McCarthy, and Barry Smyth. 2009. Using Twitter to Recommend Real-time Topical News. In *Proceedings of the Third ACM Conference on Recommender Systems (RecSys '09)*. 385–388.

[39] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *Comput. Surveys* (2018). arXiv:1802.08452 [cs.IR] https://arxiv.org/abs/1802.08452 forthcoming.

[40] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. 452–461.

[41] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW '94)*. 175–186.

[42] Alan Said and Alejandro Bellogín. 2014. Rival: A Toolkit to Foster Reproducibility in Recommender System Evaluation. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. 371–372.

[43] Mario Scriminaci, Andreas Lommatzsch, Benjamin Kille, Frank Hopfgartner, Martha Larson, Davide Malagoli, András Serény, and Till Plumbaum. 2016. Idomaar: A Framework for Multi-dimensional Benchmarking of Recommender Algorithms. In *Proceedings of the Poster Track of the 10th ACM Conference on Recommender Systems (RecSys '16)*.

[44] Bichen Shi, Georgiana Ifrim, and Neil Hurley. 2016. Learning-to-Rank for Real-Time High-Precision Hashtag Recommendation for Streaming News. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. 1191–1202.

[45] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved Recurrent Neural Networks for Session-based Recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems (DLRS '16)*. 17–22.

[46] Hao Wen, Liping Fang, and Ling Guan. 2012. A hybrid approach for personalized recommendation of news on the Web. *Expert Systems with Applications* 39, 5 (2012), 5806–5814.