Recommender Ensembles for News Articles based on Most-Popular Strategies

Andreas Lommatzsch, Niels Johannes, Jens Meiners, Lea Helmers, and Jaschar Domann

Technische Universität Berlin
Agent Technologies in Business Applications and Telecommunication Group (AOT)
Ernst-Reuter-Platz 7, D-10587 Berlin, Germany
{fistname.lastname}@campus.tu-berlin.de,
http://www.aot.tu-berlin.de

Abstract. With the change from classical paper-based to online representations of newspapers publishers are able to provide adaptive recommender services supporting users in finding the relevant articles in the huge amount of published news. Since most traditional recommender approaches are tailored to scenarios characterized by static sets of items and exactly identifiable users, these approaches cannot be utilized for anonymously consumed news portals offering a continuously changing set of items. In order to develop an efficient recommender system optimized to the specific requirements of news portals, we analyze the user behavior and define a model for computing recommendations. We have developed tools for continuously monitoring the user preferences and analyzing the features of the most popular news articles. The derived recommender models are implemented in Java using a recommender ensemble architecture that is able to adapt to the specific characteristics of different news portals. The evaluation of the implemented recommender in the CLEF NewsREEL challenge shows that our system provides reliable results and reaches a high Click-Through-Rate. The implemented architecture is open to further optimization with respect to different load levels and context parameters.

Keywords: stream recommender, recommender ensemble, online stream monitoring online and offline evaluation

1 Introduction

Current news portals offer huge, continuously changing collections of news articles. In order to support users in finding relevant articles, publishers integrate recommender components suggesting potentially interesting articles to users.

The development of recommender components for news portals leads to several challenges. First, the item sets offered by news portals change continuously. Second, the identification of users does not work precisely since users access the portals anonymously without logging in. Third, the algorithms must be capable of handling high load scenarios and ensure that recommendation requests

are successfully answered within short time limits. Moreover, each news portal has its specific characteristics requiring a customization of the recommender algorithms for the respective domain.

We focus on the news recommendation scenario defined in the CLEF NewsREEL challenge [5,9]. Here we participate in both tasks defined in the challenge: Task 1 (Living Lab Evaluation) is based on live user feedback; Task 2 (Offline Evaluation Lab) is based on replaying large datasets recorded in the Living Lab scenario. In Task 1 [6] the participating teams get live information about impressions (a user requests an article), clicks (a user clicks on a recommendation) and published articles (a publisher releases a new article). In addition, teams receive recommendation requests that must be answered within 100ms. The performance of the recommender algorithms is computed based on the Click-Through-Rate (CTR). The CTR defines the fraction of recommendation requests for that the users click on the provided recommendations. The error rate describes the fraction of incorrectly answered requests (e.g. violating the time constraints).

In Task 2 [10] the performance is measure based on the offline CTR (computed based on the overlap of recommendations and future impression). Technical aspects are measure based on the throughput and the response time for different load levels.

In this work we develop recommender algorithms for news portals addressing the discussed challenges. We analyze the log files from different news portals in order to get insights into the user behavior. Based on the identified user preferences we defined a recommender model. The implemented recommender is evaluated both online and offline in the CLEF NewsREEL challenge. We show that the implemented recommender provides high-quality recommendation results with respect to the CTR and ensures a short response time. Due to the flexible architecture of our system, the recommender can be further optimized by dynamically adapting the parameters to the context requirements.

The remaining paper is structured as follows: Section 2 gives a brief overview on existing news recommender algorithms and approaches. Our analysis of the offline log data and the developed online log analysis tool are explained in Section 3. This section also discusses the derived recommender model and the architecture of the implemented recommender system. The evaluation results are presented in Section 4. Finally, a conclusion and an outlook to future work are given in Section 5.

2 Related Work

In this Section we review existing approaches to recommender systems and analyze how these algorithms fit with the requirements of our scenario.

User-based Collaborative Filtering Most recommender systems implement Collaborative Filtering (CF). CF-based algorithms analyze the user feedback (e.g. ratings user assign to items) [4]. User-based CF algorithms compute the similarity between users and suggest items similar users liked. The algorithm requires a

sufficient number of ratings for all users and items. This is one of the weaknesses of the approach. User-based CF algorithms do not reliably provide recommendations for new users due to the fact that without a sufficient number of ratings, users having a similar "taste" cannot be determined ("cold start problem") [14].

In the analyzed news recommendation scenario, the sets of users and items change continuously. This leads to a permanent cold start problem. Thus, in our case, user-based CF does not seem to be a suitable approach.

Item- and Content-based Recommender Approaches An alternative to user-based CF are item-to-item recommender algorithms. These approaches are based on the idea that users prefer items similar to items they liked in the past. The similarity between the items can either be computed based on ratings (item-based CF) or based on the content features of the items (content-based filtering) [12]. Content-based recommender algorithms do not suffer from the cold start problem. However, content-based features provide less information on the relevance of an item (compared with collaborative features). That is the reason why rating-based approaches outperform content-based approaches in many scenarios [3]. Moreover, item-to-item recommenders neither consider news trends nor the information about the user's context. This means that highly important aspects for assessing the relevance of news articles are not taken into account.

Popularity-based algorithms News articles are characterized by a short life cycle. The top news articles are relevant for a huge number of readers during a short time period [13]. This observation is used by most-popular algorithms. These algorithms analyze the behavior of the user crowd and suggest the most read articles. The idea behind this strategy is that articles that are interesting for most users are also relevant for new users.

In the news recommendation scenario this approach seems to be promising due to the fact that it can handle fast changing sets of users and items [2]. It does neither require an exact identification of users nor additional meta-data for items. In order to apply a most-popular algorithm in the news recommendation scenario, an appropriate strategy for handling the changes in the user interests and the changes in the item set must be implemented.

Discussion In the analyzed news recommendation scenario the use of most-popular algorithms seems to be an adequate approach. In contrast to user-based CF, most-popular algorithms do not suffer from the cold start problem and do not require exact user profiles. Differently from content-based approaches, they analyze the user behavior instead of content features leading to a better recommendation precision. This analysis motivates us to further investigate most-popular algorithms.

3 Approach

In this section we first analyze the offline log data from the news portals (part of the NewsREEL challenge) to study the trends in the user behavior in detail.

Subsequently, we describe the approaches and tools used for defining a suitable recommender model and discuss the implemented strategy in detail.

3.1 Analysis of Offline Data

In order to define an efficient recommender model we analyze the offline log data. We assess user behavior with respect to impressions and clicks searching for patterns that indicate the users' preferences.

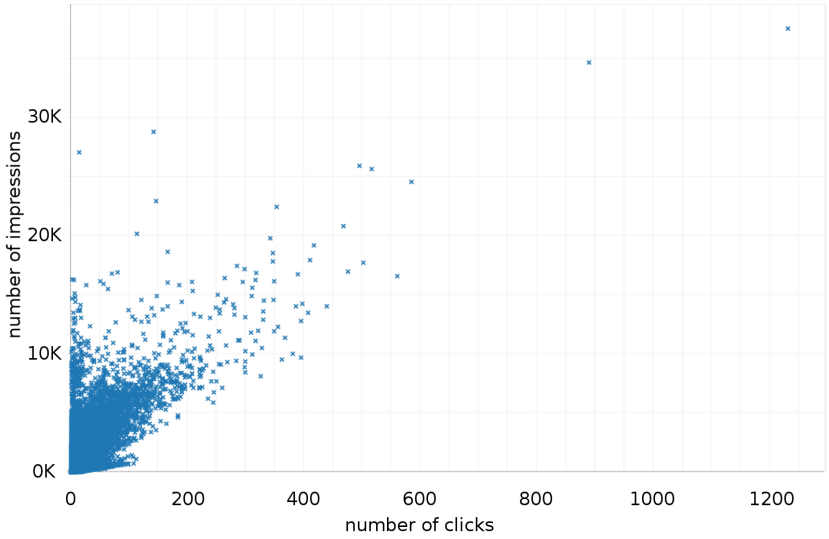

Fig. 1. The figure visualizes the correlation between the number of clicks and the number of impressions. The plot is based on data of 10,000 randomly chosen articles from the offline log dataset.

In a first step, we analyze the correlation of impressions and clicks. Figure 1 shows that news articles receiving a large number of clicks typically also have a large number of impressions; but a large number of impressions does not necessarily imply that an article receives a high number of clicks. With respect to the development of a recommender algorithm, Figure 1 shows that a large number of impressions is a valuable indicator that an article is interesting and a promising candidate for answering a recommendation request.

In a next step, we analyze the distribution of the number of impressions over the set of articles. Figure 2 shows the number of impression (y-axis) with respect to the rank of impressions (x-axis) for different publishers. The impression rank is computed by sorting the article set in descending order of the impression count. The article with the highest number of impressions has an impression rank of zero. The graphs show that there are very few articles with very high

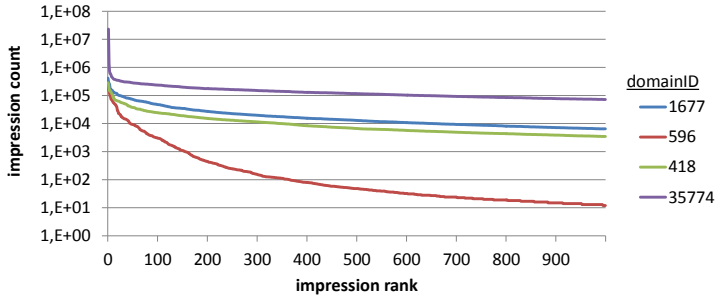

Fig. 2. The figure shows the number of impressions depending on the rank of impression for different publishers ("domainIDs")

impression count while the curve decays very fast with the impression rank. The observation indicates that there are articles of very high interest for most users. Thus recommending the most popular articles seems to be a promising strategy since most users are likely to read these articles.

3.2 Live Analysis of Online Data

The analysis of the offline log data motivates us to analyze the most popular articles in detail. We develop an online log viewer allowing us the visualization of characteristic features of the most popular articles for each news portal.

The online analysis tool consists of a database and two components, one for aggregating the relevant information from the online message stream and one for visualizing the data. The data aggregation component extracts data relevant for the analysis from the message stream and stores it in a database. The data comprise the article title, the article text, the publisher, and additional meta-data. Furthermore, it aggregates the number of clicks and the number of impressions on a 15 minutes basis.

The visualization component is implemented as a web application based on a Grizzly web server[1]. The web application uses SQL statements for retrieving the data from the database required in the visualization. The data are converted to the JSON format and sent to the HTML client that places the data on the HTML formatted web page.

The created user interface visualizes the most popular articles for the different domains and for a user-defined timeframe. For each publisher, articles are sorted by the number of impressions within the set time segment. Figure 3 shows a visualization example. For the analysis the time frame and the publisher must be defined. The tool visualizes the number of impressions for the selected publisher in a histogram. The news articles (most popular in the selected time frame) are shown in a list. For each article the most important information is displayed.

[1] `https://grizzly.java.net/`

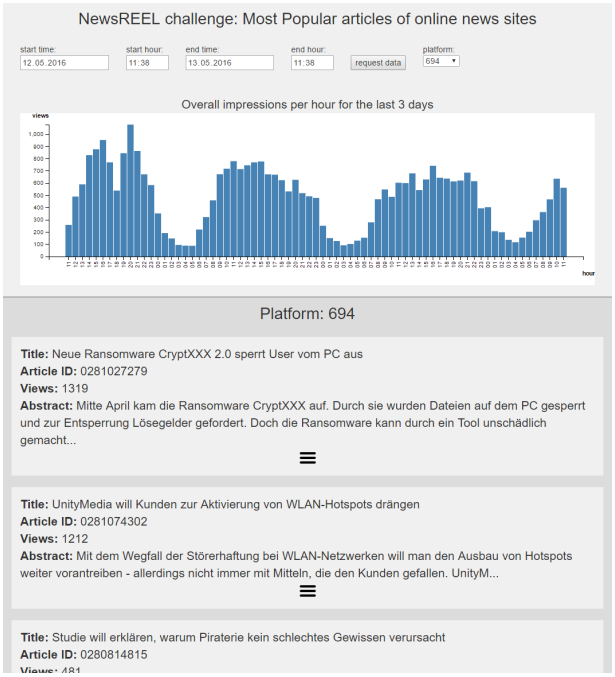

Fig. 3. The figure shows the front page of the online monitoring web application.

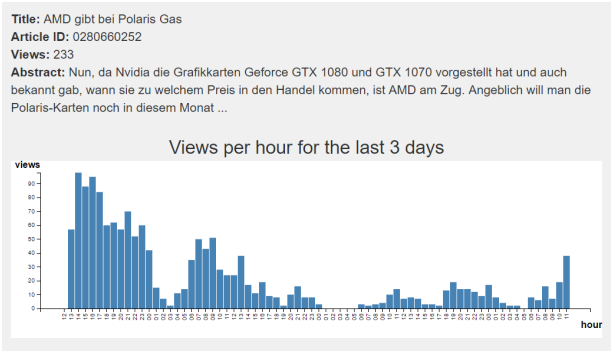

Fig. 4. Detail view: The number of impressions for a popular article of the platform gulli.com. The chart shows the amount of interest over the day.

This includes the title of the article, the articlesID, the number of views within the set time segment and the abstract summarizing the content.

Furthermore, the click on an article opens a new panel showing the impression statistic for each article. The diagram visualizes the number of impressions per hour during the last three days. Figure 4 shows an example chart for a selected article published on May 13th, 2016, at 12 o'clock. The graph shows the high variance in the number of impressions per hour over the day. Another important aspect the chart reveals is that the number of impressions per hour decreases over time. This is due to the reduced interest of users in outdated articles.

3.3 Implementation of the Recommender

Based on our analysis, we develop a recommender component implementing a most-popular algorithm. Due to the differences in the number of articles and average lifetime of the items from different portals, we use a meta-recommender architecture having a special component for each domain as shown in Figure 5.

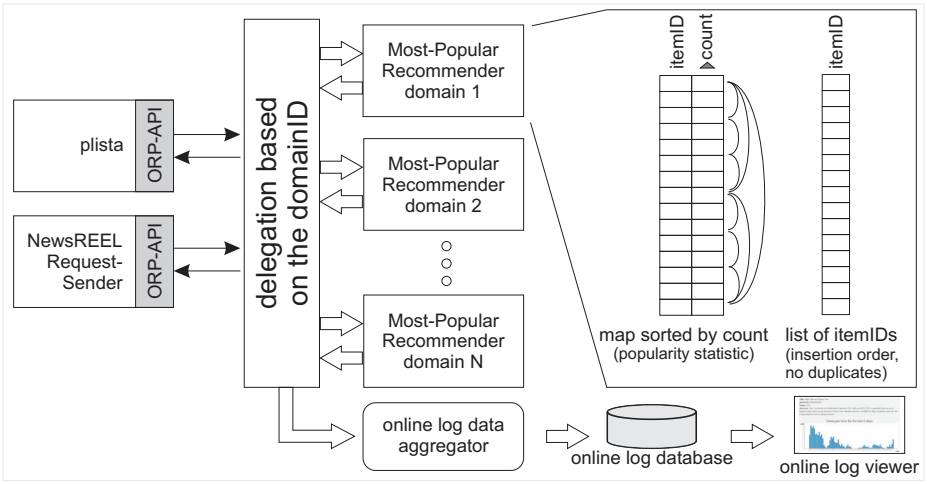

Fig. 5. Visualization of the system architecture. A most-popular recommender is implemented for each domain. Requests are delegated to the recommender component of the specific news portal. Each most-popular component stores the number of impressions for each news article in a map sorted by popularity. In order to remove old articles an insertion order list is used.

The implemented strategy We implement a most-popular algorithm by analyzing the most recent impressions with regard to their popularity for each domain. For this purpose we use a map `(impressionID, number of impressions)` storing the number of impressions for each itemID. In order to limit the size of the

map, we use a queue ("fifo"), storing the articles' IDs in insertion order. If the maximal number of keys in the map is reached, the oldest key from the map is removed. We implement the map based on a concurrent skip list containing the impressions in descending order of the number of impressions. This data structure has several advantages:
(1) The map supports the concurrent access enabling the use of threading. In addition to that, no explicit synchronization is needed. This ensures that an update of the impression statistic does not block recommendation requests. This is important to ensure that we can correspond to the tight time limits for the requests.
(2) Since the map contains the itemIDs sorted by popularity, recommendation requests are efficiently answered by returning the head of the list of itemIDs. This minimizes the complexity of providing recommendations.

Discussion In this section we motivated the use of a most-popular model. We presented our online log viewer visualizing the characteristic features as well as the impression statistics for the most popular articles. Moreover, we presented our meta-recommender architecture combining recommender components customized for the different news portals ("domains"). Each recommender component uses a most-popular algorithm tailored for the characteristics of the specific portal.

The implementation of the most-popular recommender components uses highly optimized data structures allowing concurrent access due to the use of concurrent collections. Since the potentially relevant articles are already sorted by popularity, the efficient handling of requests is assured. The meta-recommender architecture ensures that the optimal window size for each portal is used handling the removal of outdated articles. The suitable window size is determined based on the analysis of the log data.

4 Evaluation

We benchmark the implemented news recommender system both in the online and the offline task.

Online Evaluation Our recommender participated actively in the online evaluation from April 14th until May 20th, 2016. We restrict our evaluation to the domain `35774` due to the observation that 96% of all requests belong to this domain (cf. Fig. 6). The number of requests for the other publishers is small so that the significance of the results for the other domains is limited.

Table 1 summarizes the results obtained in the online task. The results show that the implemented strategy works very reliably. Our recommender handled a large number of requests and, having a median response rate of 99.4%, significantly outperformed the baseline recommender.

In order to analyze the online task in detail, we plot the CTR on a daily basis. Figure 7 shows the number of requests and the reached CTR. The results

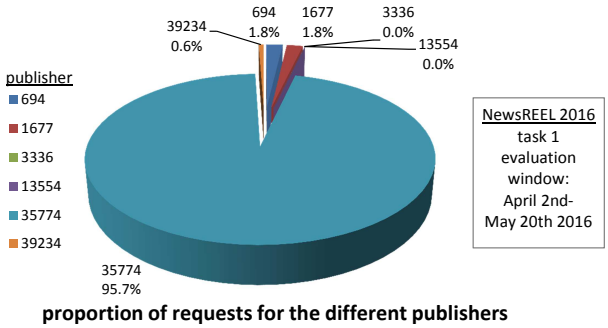

Fig. 6. The figure shows that requests for the publisher `35774` dominate the NEWSREEL challenge.

Table 1. Evaluation results for the time period of the April 14th to May 20th, 2016 for the publisher `35774`.

	our meta-recommender (`Berlin`)	baseline recommender (`baseline`)
number of requests	320,423	119,794
number of clicks	2,432	777
one week	0.40%	0.51%

show that the CTR is stable. On May 3rd, 2016, the CTR dropped significantly. Since the recommender has not been changed during the entire evaluation, the decrease must be caused by external factors that cannot be influenced by the recommender algorithms. In order to handle external changes, the recommender should be able to adapt itself to new environments.

Offline Task In addition to the online evaluation, we also benchmark the performance of the recommender system offline based on a simulated stream. We replay the data stream recorded in one week (May 9th - May 15th, 2016) and analyze the offline CTR as well as the response time.

Figure 8 visualizes the offline CTR of our algorithm. Compared to the online CTR the offline CTR is slightly lower. This indicates that our algorithm is optimized for the online reward model. The user impressions (used as reward model in the offline evaluation) seem to follow different patterns than the user clicks (used as reward model in the online evaluation). The differences between the online and the offline evaluation results could be explained by the observation that user impressions cover a wide spectrum of articles but users only click on recommendations if special (very popular) articles are recommended.

We also evaluated the response time of the recommender algorithm. The evaluation has been executed on a laptop running Windows 7 having an Intel® Core™ `i7-3520M 2.9GHz` CPU and 16 GB of RAM. Figure 9 shows the response time for different levels of concurrent requests. The results show that the implementation efficiently handles requests.

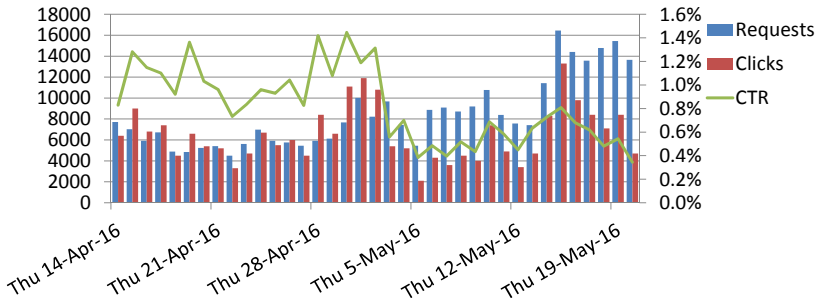

Fig. 7. The figure shows the online CTR of our team (`Berlin`) on a daily basis (publisher `35774`).

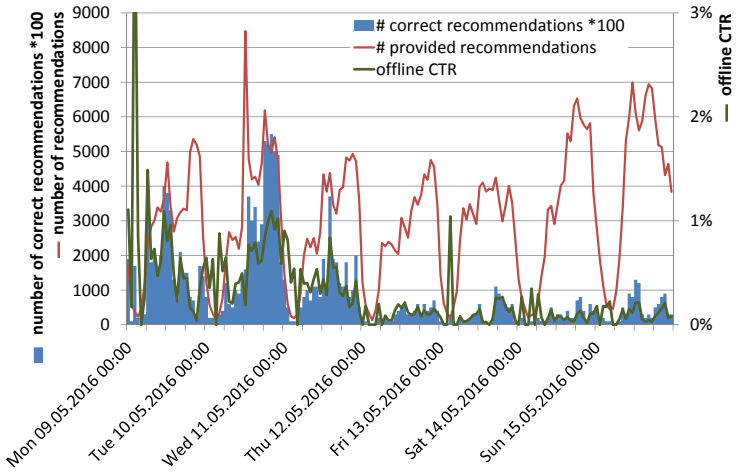

Fig. 8. The figure shows the offline CTR for the publisher `35774` for the second week of May 2016.

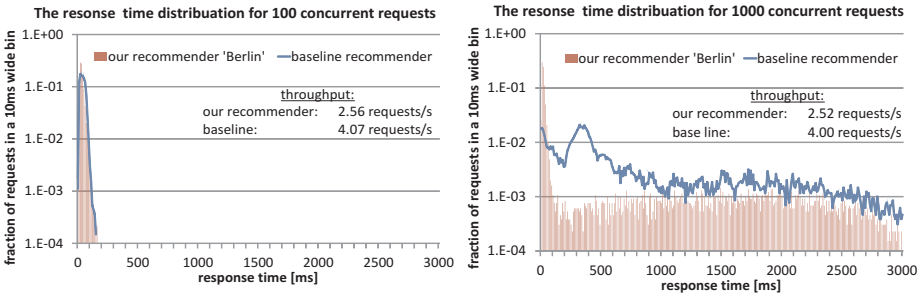

Fig. 9. The figure shows the offline response time distribution for different load levels. The results show that our recommender efficiently handles concurrent recommendation requests.

5 Conclusion

In this paper we presented our news recommender system tailored to the specific requirements of the NewsREEL challenge. Our system uses a most-popular recommender approach that adapts to the characteristics of the different news portals ("publishers"). In order to motivate that a most-popular algorithm is an efficient solution for the news recommendation task, we developed an online tool visualizing the most popular articles and their characteristics. Our analysis showed that the most popular articles cover a high fraction of the user-item interactions. The user behavior differs between the different portals. This has been the motivation for developing a recommender implementing a meta-recommender architecture combining components that are optimized for the relevant news portals.

In order to handle the problem of continuous changes in the user interests, we implemented a sliding window approach ensuring that only the most recent popular articles are recommended. Outdated articles are automatically removed from the set of candidates considered when computing recommendations. The evaluation showed that the implemented recommender outperforms the baseline for all considered domains. The error rate is very low. The response time requirements are reliably fulfilled.

The analysis of the user behavior showed that the optimal window size considered while computing the most popular news articles depends on the news portal. As future work we plan to apply a context-aware optimization of the window size in order to take into account the day of the week (working day, weekend, and holiday) and the time of the day. Furthermore, we plan to implement algorithms predicting what articles will be popular in the next hour. For this purpose we identify current trends and extrapolate these trends into the near future.

Acknowledgments

The research leading to these results was performed in the CrowdRec project, which has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement No. 610594.

References

1. H. Ahn. A new similarity measure for collaborative filtering to alleviate the new user cold-start problem. *Information Science*, 178:37–51, 2008.
2. R. Bandari, S. Asur, and B. A. Huberman. The Pulse of News in Social Media: Forecasting Popularity. 2012.
3. F. Cacheda, V. Carneiro, D. Fernández, and V. Formoso. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Trans. Web*, 5(1):2:1–2:33, Feb. 2011.
4. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. (TOIS)*, 22(1):5–53, 2004.
5. F. Hopfgartner, T. Brodt, J. Seiler, B. Kille, A. Lommatzsch, M. Larson, R. Turrin, and A. Serény. Benchmarking news recommendations: The clef newsreel use case. *SIGIR Forum*, 49(2):129–136, Jan. 2016.
6. F. Hopfgartner, B. Kille, A. Lommatzsch, T. Plumbaum, T. Brodt, and T. Heintz. *Benchmarking News Recommendations in a Living Lab*, pages 250–267. Springer International Publishing, 2014.
7. B. Kille, T. Brodt, T. Heintz, F. Hopfgartner, A. Lommatzsch, and J. Seiler. NEWSREEL 2014: Summary of the news recommendation evaluation lab. In *Working Notes for CLEF 2014 Conference*, pages 790–801, 2014.
8. B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz. The plista dataset. In *Proceedings of the 2013 International News Recommender Systems Workshop and Challenge*, NRS '13, pages 16–23, New York, NY, USA, 2013. ACM.
9. B. Kille, A. Lommatzsch, G. Gebremeskel, F. Hopfgartner, M. Larson, J. Seiler, D. Malagoli, A. Sereny, T. Brodt, and A. de Vries. Overview of NewsREEL'16: Multi-dimensional Evaluation of Real-Time Stream- Recommendation Algorithms. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction 7th Intl. Conf. of the CLEF Association, CLEF 2016, Evora, Portugal, September 5-8, 2016.*, LNCS 9822. Springer, 2016.
10. B. Kille, A. Lommatzsch, R. Turrin, A. Serény, M. Larson, T. Brodt, J. Seiler, and F. Hopfgartner. *Stream-Based Recommendations: Online and Offline Evaluation as a Service*, pages 497–517. Springer International Publishing, Cham, 2015.
11. A. Lommatzsch and S. Albayrak. Real-time recommendations for user-item streams. In *Proc. of the 30th Symposium On Applied Computing, SAC 2015*, SAC '15, pages 1039–1046, New York, NY, USA, 2015. ACM.
12. P. Lops, M. Gemmis, and G. Semeraro. *Content-Based Recommender Systems: State of the Art and Trends*, chapter 3, pages 73–105. Springer, 2011.
13. G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.
14. Z.-K. Zhang, C. Liu, Y.-C. Zhang, and T. Zhou. Solving the cold-start problem in recommender systems with social tags. *EPL (Europhysics Letters)*, 92(2):28002, 2010.