Follow Africa: Building an African News Recommender Systems

Moustapha Diouf Fall and Modou Gueye(✉)

Université Cheikh Anta Diop, Dakar, Senegal
moustaphad.fall@gmail.com, modou2.gueye@ucad.edu.sn

Abstract. Recommender systems have become an important component of Web media. From VoD providers (Netflix, Amazon Video) (https://goo.gl/g3G1ys) to news websites (Yahoo! News, CNN) (https://goo.gl/yA2NB6), users have become accustomed to personalized content. However, news recommendation differs from traditional recommendation due to the short lifetime of news. Indeed, News is particularly characterized by a short time span during which they are relevant. Therefore, in addition, to suggest suited news to users, news recommender systems (NRS) have to deal with news recency in order to avoid recommending already read content somewhere else.

In most of the cases, NRS implicitly collect users' click history and readings to build topic-based user profiles. News websites generally integrate some keywords into the news articles which sum up their content. But this is not always the case for African news websites.

In this paper, we present *Follow Africa*, an African news recommender system. We introduce a recency-based recommendation model which also takes account of users' previous readings. We show the effectiveness of our proposal through the results we obtain in a month-lasted online experiment with more than one hundred users.

Keywords: News recommendation · Recency-based algorithm · Keyword assignment

1 Introduction

The news is an important aspect in modern society, they are the best resources to know what happens around us. However, the abundance of new information that is daily published online through different channels and portals can make it challenging for users to find the content they are interested to read, consequently, they can be overwhelmed and may miss some important news.

News recommender systems (NRS) aim to recommend suited news articles to readers instead of leaving them spending a long time searching content which may be interesting.

NRS is different from the other kinds of recommender systems (RS) as they have to work with a continuous stream of short-lived items. Indeed, the news is

G. Bassioni et al. (Eds.): InterSol 2019, LNICST 296, pp. 70–81, 2019.
https://doi.org/10.1007/978-3-030-34863-2_7

particularly characterized by a short time span during which they are relevant, after this time their relevance can decrease very quickly. Therefore, in addition, to suggest suited news to users based on the history of what they read, NRS have to deal with their recency in order to avoid recommending already read content somewhere else.

In the context of African news, making some recommendations is more difficult. News articles from many African news websites are not well structured and do not integrate keywords which describe the topics of their articles. This makes the crawling of their content more difficult.

We have also to take account of user privacy when computing his recommendations. Thus in this work, we propose a privacy-heeded NRS which handles the challenge of helping readers to be aware of fresh Africa-related news articles which might suit them.

In our knowledge there is no news recommender system build specially for African news. The ones like Flipboard, Google News, Yahoo! News and so on show Africa in overhaul but not in deep. Furthermore, they only rely on international news websites and do not use African regional ones which can give some local useful news. International news websites are well structured. It is easy to get a summary of news articles as they generally give some keywords inside their articles which sum up them. But for regional African news websites that is not the case. There is a real challenge in crawling and summarizing their articles.

The aim of *Follow Africa* is to propose news recommendations related to Africa. Its architecture is organized into two components. A back-end component which makes all the tasks of news crawling and summarizing into topics, and a Front-end component which heeds privacy and locally computes news recommendations to show to readers. We present in this work these two components. We detail how we make recommendations and point out the effectiveness of our proposal through some results that we get from a month-lasted online experiment over many readers[1].

The sequel of this paper is organized as follows. In Sect. 2 we present some related works in news recommendation. In Sect. 3 we provide a detailed presentation of *Follow Africa*. We expose its architecture and explain how we recommend suited news articles. Then in Sect. 4.2 we show our experimental results. Finally, in Sect. 5 we conclude this paper and present some future works.

2 Related Works

Two different approaches are commonly used in news recommendation: content-based and collaborative filtering. Each of them presents some drawbacks and advantages [12]. The collaborative filtering approach considers the opinions of peer users to generate news recommendations. In [6] , the authors use collaborative methods to make new recommendations. However, this approach presents

[1] One can install *Follow Africa*'s Android mobile application from https://goo.gl/t8nahh.

two majors drawbacks as cited by [11]. First, collaborative methods do not recommend an article that many users have not yet read. Second, the trend to recommend "buzz" articles while some people have no interest in this kind of articles. For these reasons and some security and privacy aspects, content-based approaches are better suited for news recommendation [3]. In content-based approaches, keywords are used to describe a news article and a user profile is built to indicate the type of news this user likes. We take this approach and use the keywords of news articles to understand what topics are of interest to the user.

In the literature, many authors have proposed topics-based news recommendations like in [14]. However, in this work, we combine the interest of users on news topics by the recency of news articles in order to recommend fresh and suited news content. Moreover, we take into account users' reading privacy, therefore we propose a news recommender system which even runs on the reader's mobile. Recommender systems retrieve users' feedback in various forms. They can explicitly ask the user to give feedback by rating the articles [2] or collect implicit feedback based on user clicks and readings. Even if the second method is less accurate than the first, it allows to gets more feedback as users do not give any ratings. Furthermore, it is not bothering. Our proposal only uses implicit, positive feedback.

3 Follow Africa

As said above, the purpose of *Follow Africa* is to find news articles related to Africa that may be of interest to a particular user. It continuously aggregates articles from several African news websites and other ones that publish some topics related to Africa in order to build personalized content for its readers.

In this section, we show the architectural system of *Follow Africa*, then we discuss its functioning and some issues we have faced with unstructured or mistaken data from some new websites. Finally, we present the recommendation model behind our mobile application and the algorithm to compute recommendations.

3.1 Architecture and Functioning

The architecture of *Follow Africa* has two parts: a back-end side and a front-end one. The first is responsible for harvesting articles from news websites and providing them to the front-end side. The latter computes locally on the mobile the recommendations to make to a user and displays the selected articles. Figure 1 shows the architecture of the overall system of *Follow Africa*.

The back-end side represents all the background works we do to gather news articles to give to users. Its different elements are listed below.

- Web: this element corresponds to the web sources from which we extract news, ranging from several popular and well-known websites to nameless ones.

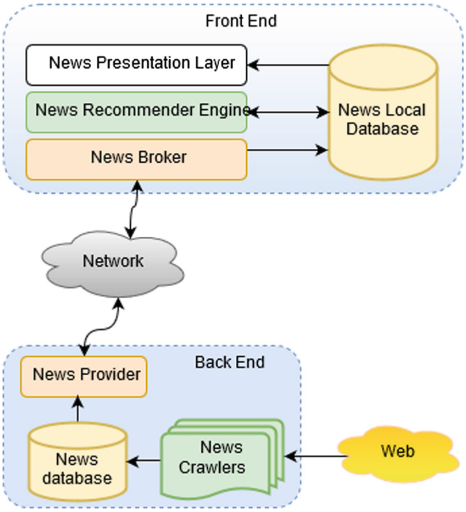


Fig. 1. Architecture of *Follow Africa*

Hence let us notice that our content is not confined to news, we also add some fashion or cooking magazines to bring more diversity in our content. Currently, we have more than forty websites as news sources. They provide news in French language and cover almost all the categories of news like politic, sport, health and so on. Obviously, they have not the same page structure, some of them are well organized while the others are messy.

- News Crawlers: we use different scripts to harvest articles from the news websites. For each website and depending on its page structure, we assign a custom news crawler.
  Periodically, each news crawler browses its target website and search for newly published articles. It retrieves the main content of each new article like the title, the lead paragraph, the publication date and time, the illustrative image and, when they are available, the keywords which give the related topics of the article. In the case where any keyword is not available, the crawler uses a topic assignment method to fill the missing data. In Sect. 3.2, we explain the topic assignment method we used. Furthermore, we compress the illustrative image in a smaller size in order to optimize the communication cost with the front-end side which is implemented in readers' mobiles.
- News Database: The news database stores all the crawled articles. As news is ephemeral, every day we delete from the database all articles which are older than seven days. By doing this, we make sure to only consider news over a period of one week.
- News Provider: The news provider is one of the most important components of our architecture as it handles all the requests from the front-end side (*i.e.*, the mobile app) and prevents the user to have an article more than once.

To summarize the back-end side, the news crawlers harvest periodically fresh news articles from websites and save their main content on the database. When the news provider receives requests from a reader's mobile application, it selects all the articles the reader has not yet received and sent them to him.

The front-end side refers to our mobile application. It presents to the readers the collected data from different news websites. Figure 2(a) gives an overview of its interface. This site is organized in four components to ensure that the users see first the news which may be of interest to them from the multitude of articles he has in his disposition.

- News Broker: it carries out all the communications with the back-end side. Especially, it communicates with the news provider and collects the latest news that it records in the new local database. It also verifies that all the collected items are in a good format before inserting them into the database.
- News Local Database: We decided to use a local database because in Africa, Internet access is not so easy for everyone and we want to be able to work offline. As a result, all the news article from the news broker are saved and used when the news presentation layer requests them. News articles do not always stay in the database. Of course, they are deleted after a specified period.
- News Recommender Engine: the task of the recommender engine is to compute and save scores of interest a reader might have for yet unread news articles. Each time the reader reads or clicks on an article or some new articles are added on the news local database by the news broker, it recomputes the scores of articles.
  Section 3.3 details the recommendation model behind the news we suggest to users for reading.
- News Presentation Layer: it implements all the IHM of the mobile app. Figure 2 shows an overview of it's Home's page and the recommendation's one. When the reader clicks on a news article, the news presentation layer updates his profile by adding the topics of the article. Then the recommender engine recomputes the scores of interest of articles and ranks them in decreasing order. Finally, the presentation layer refreshes its display with these ranked articles.

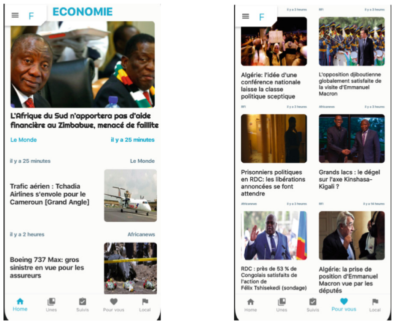
3.2 Topics Assignment

Most news websites sum up in their articles the topics they cover through some keywords. More formally, let be $\mathcal{A}$ and $\mathcal{T}$ respectively the set of news articles and one of possible topics that describe news. We consider for a news article $a \in \mathcal{A}$ the set $\mathcal{T}_a$ defined by

$$\mathcal{T}_a = \{t \mid t \in \mathcal{T}\} \tag{1}$$

as the set of topics t that sum up the article a. From this definition, we set the description D_a of a news article a by a triplet of information as follows

$$D_a = (c_a, s_a, \mathcal{T}_a)\,, s_a \in \mathbb{R}^+ \tag{2}$$



(a) Home (b) Recommendations

Fig. 2. Overviews of the news presentation layer

where c_a merges the title, image and lead paragraph of the news article and s_a represents the staleness of the article. We consider the staleness of an article as the number of elapsed days since the article was published online. Through the staleness of articles we consider their recency.

Despite the fact that many news websites summarize their articles with keywords representing their topics, some websites do not. In this case, we use a custom taxonomy database for assigning some topics to news articles. We explain our approach.

First, our taxonomy database is organized as a key-value table where the values are sets of words: $\left[k \Rightarrow \{t \mid t \in \mathcal{T}\}\right]$. For instance, we can find in the following key-value instances:

$$\left[\text{“}sadio\ mane\text{”} \Rightarrow \{\text{“}football\text{”}, \text{“}sport\text{”}, \text{“}senegal\text{”}\}, \text{“}election\text{”} \Rightarrow \{\text{“}politic\text{”}\}\right]$$

Therefore when a news crawler finds a news article without topics that describe it, the crawler breaks its content into several parts through string tokenization [5]. Then the crawler browses the taxonomy database with the tokens and retrieves related topics. The retrieved topics are associated with the article and the all is saved in the database.

Topics assignment is also called keywords assignment. Many assignment methods are suggested in the literature [4] with some that use advanced natural processing language libraries but they are especially dedicated to English language and are not so useful for other languages like French [9].

Once news articles are harvested and topics assignment is done for those we have not yet established topics, the articles are saved on the news database. They are sent to readers by the news provider when they ask new content. Our recommender system takes the relay on the readers' mobiles and makes personalized recommendations to show to them. In the next section, we present the recommendation model we use.

3.3 Recommendation Model and Algorithm

We use a content-based recommendation model under our news presentation pages. Indeed content-based techniques are more robust to face cold start problem than collaborative filtering [13]. In order to simplify the rest of this section, let us consider int the following the sets $\mathcal{U}$ and $\mathcal{D}$ as respectively the set of all users (*i.e.*, our readers) and one of the valid calendar dates.

The recommendation task can be summarized as to find the top-K highest interesting news articles for a user $u \in \mathcal{U}$ and recommend them to him. Let be a utility function which computes the score of interest a reader might have for some news articles., i.e., $score : \mathcal{U} \times \mathcal{A} \rightarrow \mathbb{R}$. We can formalize the task of recommendation as follows:

$$Top_u^K = \underset{a \in \mathcal{A}}{\overset{K}{argmax}}\ score(u, a) \tag{3}$$

The basic assumption of personalizing content is that users reasonably have consistent interests that may change over time [11]. Thus each user can be followed up by a profile which is a representation of all his interests and properties like age, gender, occupation and so on.

In the case of news recommendation, a user profile usually corresponds to a data structure which sums up his preferences on news topics. With *Follow Africa* we set the profile P_u of a given user $u \in \mathcal{U}$ by

$$P_u = \{(t, w_t, d_t) \ \mid \ t \in \mathcal{T}, w_t \in \mathbb{R}^+, d_t \in \mathcal{D}\} \tag{4}$$

where each triplet (t, w_t, d_t) represents a news topic t which interests the user with an interest-valued weight w_t and the last date d_t when the user read an article related to that topic.

The user profile helps us to predict what he may like or not. To build the profile we assume a click on an article means he is interested in the subject of the article, so we update his profile by adding the topics of the article to it. In addition to adding a topic t to the user profile, we set its interest-valued weight w_t to one. In case that is not the first time that the topic is added, we increase its interest-valued weight by a unit.

To prevent the cold start problem which refers to new users (their profiles are empty), we ask users to choose a least three main topics that interest them at the first time they start to use the app.

From the Eq. 4, we define all the topics of interest to a user u as

$$\mathcal{T}_u = \{t \mid \exists\ w_t \in \mathbb{R}^+ \wedge d_t \in \mathcal{D} : (t, w_t, d_t) \in P_u\} \tag{5}$$

and we compute the interest-valued weight of a news article a for a user u by the following score function

$$score(u, a) = \mathrm{e}^{-s_a} \times \Big(\sum_{t \in \{\mathcal{T}_u \cap \mathcal{T}_a\}} w_t \Big) \tag{6}$$

By this interest-valued function we take account of the user's favorite topics and the staleness of news articles. The first part e^{-s_a} takes account of the staleness of articles (*i.e.*, their recency). We decided to represent the evolution of a news article's staleness by the exponential function as the probability a user has already read an article on somewhat website greatly increases day after day. The second part of our function represents the intrinsic interest of the article for the user.

Since the user's favourite topics may change over time depending on several contextual elements, his profile must be updated each time he expresses a particular interest in some topics by reading a new article or just click on it. In this work, we build the profiles of users based only on their clicks on articles. In other words, we just consider the implicit positive feedback of users. In addition, for some privacy concerns, we store each user's profile on his mobile as the data belongs to him. Moreover, as the recommendations are carried out on the user mobile, we alleviate our server which runs the back-end tasks.

Once we computed the interest-valued weights that a user may have on news articles, we sort them in decreasing order. Let us notice that all articles that do not have topics are ignored. The articles that a user has already read are ignored too. We do not recommend an article that a user has already read. However, an article that has been recommended can be recommended again because it is difficult to interpret why a user has not yet clicked on. Algorithm 1 details more our recommendation process.

Algorithm 1. *Follow Africa*'s recommendation algorithm

Data: $\{D_a \mid a \in \mathcal{A}\}$, P_u
Result: $S = [(a, score(u, a)) \mid a \in \mathcal{A}]$
begin
    $S \longleftarrow [\,]$;
    foreach $a \in \mathcal{A}$ **do**
        $S.add(a, score(u, a))$;
    end
    Sort S by decreasing $score(u, a)$;
    return S
end

4 Experimentation

We present in this section the effectiveness of our recommendation model. We led an online experiment on a population of 113 real readers. In the next two subsections, we shortly describe how we did the experiment by the evaluation measures and methodology we used. Then we present the results we obtained.

4.1 Evaluation Measures and Methodology

One can evaluate RS by using three approaches: offline analysis, user studies or online experiments. Furthermore, a combination of these approaches is also possible [1].

Offline analysis is typically easy to conduct, as they require no interaction with real users. However, it is hard to find publicly available datasets for news recommendation. The one we know is the Plista dataset [7], but all its contained news are in German. User studies is an alternative. In this case, a small group of persons will use the mobile app in a controlled environment and then their experience will be reported. However, we may have to consider various biases in the experimental design of such studies. Therefore we decide to carry out an online evaluation where a pool of real users who are unaware of the experiment are used. As some authors said, this is perhaps the most trustworthy approach [8]. However, it can only collect certain types of data.

So we conducted an online experiment for a month with the recourse of a week-based size-increasing pool of real users. Table 1 details the increasing of the pool size. By weekly increasing the pool of users, we want to take into account the evolution of the users' profiles.

Table 1. Week-based size-increasing pool of users

Week	Dec. 16–22	Dec. 23–29	Dec. 30–Janv. 5	Janv. 6–12
# New users	38	23	19	33
# Total	38	61	80	113

We did the experiment from December 16, 2018, to January 12, 2019, with a total of 113 users at the end whose 58.8% of male and 41.2% of female ranging from 18 to 54 years old.

To measure the effectiveness of our news recommender system, we mainly measure changes in the retention rate of new users to determine the extent to which our recommendation is keeping users engagement. We did two things.

1. we first track the increase in overall users retention by measuring the evolution of the number of established active users,
2. then we divide users into four week-based cohorts depending on the first time they used the mobile app. For each cohort, we measure the retention rate of its users during the next weeks.

4.2 Experimental Results

We present in this part the results of our online experiment.

Active Users Evolution. Active users refer to those who frequently use the app. Note that by "frequently" we do not necessarily say every day. An active user can skip a day and do not use the app, but not two days.

We log monthly (28-day), weekly (7-day), and daily (1-day) active users percentage evolution in the range from December 16, 2018, to January 12, 2019. Figure 3 displays the evolution of active users.

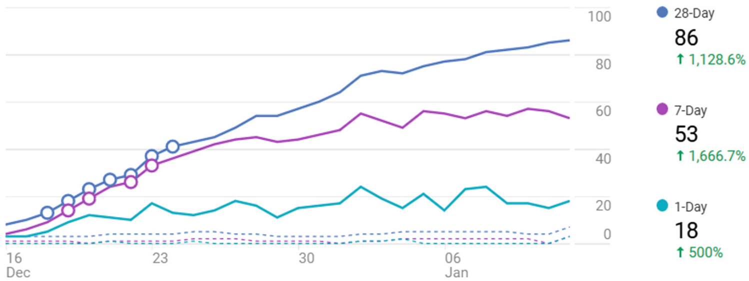


Fig. 3. Active users evolution

We can see that the number of active users is increasing. At the date of January 12, 2019, we count 86 active users among all our 113 users. Either 76% of our users actively use the app and access to recommended news articles. However, active users do not use the app in the same manner. Some of them discovered the app during the last week of the experiment while some other users began to use it in previous weeks. Therefore we led the evaluation below where we take into consideration this fact.

New Users Retention. To measure the ability of our recommendations to retain new users over time, we measure the percentage of active users who use the mobile app every day since the first time they opened it. Let us note that we do not consider here users which remained at least one day without using the app. We divided users into four week-based cohorts. Each cohort is a set of users who started using our application during the same week. Table 1 gives the number of users into each cohort.

Figure 4 shows the extent to which our recommendations retain new users. It indicates whether users acquired a week ago continue to use the app. Each row represents a cohort. The bottom row represents the most recent cohort while the top one corresponds to the earliest cohort.

As we can see, the retention rate of each group decreases after the first week of use of the application, but increases slightly from the Week 2. This is an

	Week 0	Week 1	Week 2	Week 3
	100%	38.8%	42.6%	39.5%
Dec 16 - Dec 22 38 Users	100%	36.8%	42.1%	39.5%
Dec 23 - Dec 29 23 Users	100%	39.1%	43.5%	
Dec 30 - Jan 5 19 Users	100%	42.1%		
Jan 6 - Jan 12 33 Users	100%			

Fig. 4. New users retention

expected effect of the well known issue named the cold start problem. Indeed recommender systems meet difficulties to recommend items to a user who has not yet read enough news articles and then who is not well profiled.

RS need some time to learn the users click behaviour and understand the topics which are in the interest of them. That is why the retention of Week 2 is greater than the one of Week 1. Our recommendation model seems to work better after one week of usage. This demonstrates in a certain extent the effectiveness of our recommendations.

We remark also that the user retention rate of last cohorts is higher than the one of first cohorts. When we investigate the kind of users of each cohort, we discovered that the first cohort is almost composed by friends whom we gave the app while the rest are some unknown people who have installed *Follow Africa* from the Google's play store (maybe they know the app from the Facebook page we dedicated to it). The latter seem to have more engagement than our friends who would just accept to install the app due to our friendship.

5 Conclusion

In this paper, we presented a news recommender system we called *Follow Africa*. We detailed its architecture and discussed how it carries out recommendations. The experiment we led online points out the effectiveness of its recommendation.

As future works, we target to collect and create a news dataset from our current readers. Thus we will be able to compare in the same baseline our news recommendation model to some others in the literature [10].

Currently, some redundant news articles often occur in our recommendations despite they are provided by different websites. By redundancy, we refer to the similarity of their information and not of their content in terms of expressions or sentences. Thus we want to investigate article de-duplication methods which allow eliminating redundant news articles.

References

1. Abel, F., Deldjoo, Y., Elahi, M., Kohlsdorf, D.: Recsys challenge 2017: offline and online evaluation. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, pp. 372–373. ACM, New York (2017)
2. Billsus, D., Pazzani, M.J.: A hybrid user model for news story classification. In: Kay, J. (ed.) UM99 User Modeling. CICMS, vol. 407, pp. 99–108. Springer, Vienna (1999). https://doi.org/10.1007/978-3-7091-2490-1_10
3. Calandrino, J.A., Kilzer, A., Narayanan, A., Felten, E.W., Shmatikov, V.: "you might also like:" privacy risks of collaborative filtering. In: 2011 IEEE Symposium on Security and Privacy, pp. 231–246, May 2011
4. Dakka, W., Gravano, L.: Efficient summarization-aware search for online news articles. In: JCDL, pp. 63–72. ACM (2007)
5. Dridan, R., Oepen, S.: Tokenization: returning to a long solved problem a survey, contrastive experiment, recommendations, and toolkit. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2, ACL 2012, pp. 378–382. Association for Computational Linguistics, Stroudsburg (2012)
6. Garcin, F., Zhou, K., Faltings, B., Schickel, V.: Personalized news recommendation based on collaborative filtering. In: Proceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01, WI-IAT 2012, pp. 437–441. IEEE Computer Society, Washington (2012)
7. Kille, B., Hopfgartner, F., Brodt, T., Heintz, T.: The plista dataset. In: Proceedings of the 2013 International News Recommender Systems Workshop and Challenge, NRS 2013, pp. 16–23. ACM, New York (2013)
8. Kohavi, R., Longbotham, R., Sommerfield, D., Henne, R.: Controlled experiments on the web: survey and practical guide. Data Min. Knowl. Disc. **18**(1), 140–181 (2009)
9. Kompan, M., Bieliková, M.: Content-based news recommendation. In: Buccafurri, F., Semeraro, G. (eds.) EC-Web 2010. LNBIP, vol. 61, pp. 61–72. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15208-5_6
10. Li, L., Wang, D.-D., Zhu, S.-Z., Li, T.: Personalized news recommendation: a review and an experimental investigation. J. Comput. Sci. Technol. **26**(5), 754–766 (2011)
11. Liu, J., Dolan, P., Pedersen, E.R.: Personalized news recommendation based on click behavior. In: Proceedings of the 15th International Conference on Intelligent User Interfaces, IUI 2010, pp. 31–40. ACM, New York (2010)
12. Shah, L., Gaudani, H., Balani, P.: Article: survey on recommendation system. Int. J. Comput. Appl. **137**(7), 43–49 (2016). Published by Foundation of Computer Science (FCS), NY, USA
13. Volkovs, M., Yu, G.W., Poutanen, T.: Content-based neighbor models for cold start in recommender systems. In: Proceedings of the Recommender Systems Challenge 2017, RecSys Challenge 2017, pp. 7:1–7:6. ACM, New York (2017)
14. Wang, Z., Hahn, K., Kim, Y., Song, S., Seo, J.-M.: A news-topic recommender system based on keywords extraction. Multimedia Tools Appl. **77**(4), 4339–4353 (2018)