Stream-Based Recommendations: Online and Offline Evaluation as a Service

Benjamin Kille[1], Andreas Lommatzsch[1], Roberto Turrin[2], András Serény[3], Martha Larson[4], Torben Brodt[5], Jonas Seiler[5], and Frank Hopfgartner[6]

[1] TU Berlin, Berlin, Germany
{benjamin.kille,andreas.lommatzsch}@dai-labor.de
[2] ContentWise R&D - Moviri, Milan, Italy
roberto.turrin@moviri.com
[3] Gravity R&D, Budapest, Hungary
sereny.andras@gravityrd.com
[4] TU Delft, Delft, The Netherlands
m.a.larson@tudelft.nl
[5] Plista GmbH, Berlin, Germany
{torben.brodt,jonas.seiler}@plista.com
[6] University of Glasgow, Glasgow, UK
frank.hopfgartner@glasgow.ac.uk

Abstract. Providing high-quality news recommendations is a challenging task because the set of potentially relevant news items changes continuously, the relevance of news highly depends on the context, and there are tight time constraints for computing recommendations. The CLEF NewsREEL challenge is a campaign-style evaluation lab allowing participants to evaluate and optimize news recommender algorithms online and offline. In this paper, we discuss the objectives and challenges of the NewsREEL lab. We motivate the metrics used for benchmarking the recommender algorithms and explain the challenge dataset. In addition, we introduce the evaluation framework that we have developed. The framework makes possible the reproducible evaluation of recommender algorithms for stream data, taking into account recommender precision as well as the technical complexity of the recommender algorithms.

Keywords: recommender systems, news, evaluation, living lab, stream-based recommender

1 Introduction

When surveying research advances in the field of recommender systems, it becomes evident that most research hypotheses are studied under the premise that the existence and the relevance of recommendable items are constant factors that remain the same throughout the whole recommendation task. The reasons underlying these assumptions can be traced to the use by the research community of shared datasets with static content for the purposes of system development and evaluation. An example is the well-known MovieLens dataset [12], which is used extensively to benchmark movie recommendations. A multitude of experiments have pointed out that recommendation algorithms, such as collaborative filtering, developed under such a premise can provide good recommendations. However, these techniques are inherently limited by the fact that they cannot easily be applied in more dynamic domains, in which new items continuously emerge and are added to the data corpus, while, at the same time, existing items become less and less relevant [4]. An example where recommendation of dynamic data is required can be found in the news domain where new content is constantly added to the data corpus. CLEF NewsREEL[7] addresses this news recommendation scenario by asking participants to recommend news articles to visitors of various news publisher web portals. These recommendations are then embedded on the same news web page. The news content publishers constantly update their existing news articles, or add new content. Recommendations are required in real-time whenever a visitor accesses a news article on one of these portals. We refer to this constant change of the data corpus as streamed data, and the task of providing recommendations as stream-based recommendations. This news recommendation scenario provides ground to study several research challenges:

1. In contrast to traditional recommender systems working with a static set of users and items, the set of valid users and items is highly dynamic in the news recommendation scenario. New articles must be added to the recommender model; outdated news articles must be demoted in order to ensure that the recommended articles are timely. Thus, one big challenge of the news recommender system is the continuous cold-start problem: New articles potentially more relevant than old articles are only sparsely described by meta-data or collaborative knowledge. The system has not observed sufficiently many interactions to determine these articles' relevance.
2. Noisy user references pose an additional challenge in the analyzed web-based news recommendation scenario. Since users do not need to register explicitly on the news portals, these systems lack consistent referencing. They seek to overcome this issue by tracking users with cookies and JavaScript. Some of the users may apply obfuscating tools (such as Ad-Blocker) leading to noisy user references. The implemented recommender algorithms be aware of the challenge and should apply algorithms providing highly relevant recommendations even if the user tracking is noisy.
3. The user preferences in news highly depend on the domain and on the hour of the day. In the morning, users usually do not have much time. For this reason, at this time, users are interested in the top news from the domains of politics and sports. In the evening users usually spend more time reading and engaging in longer, more detailed news articles from diverse domains. Therefore, news recommender algorithms must consider different aspects of context such as the news domain, the time of the day and the users' devices.
4. In the online news recommendation scenario, the requests must be answered within a short period of time. The response time constraint is defined as publishers require suggestions to be seamlessly integrated in their web page.

Regarding these challenges, CLEF NewsREEL 2015 aims to promote benchmarking of recommendation techniques for streamed data in the news domain. As depicted

[7] `http://clef-newsreel.org/`

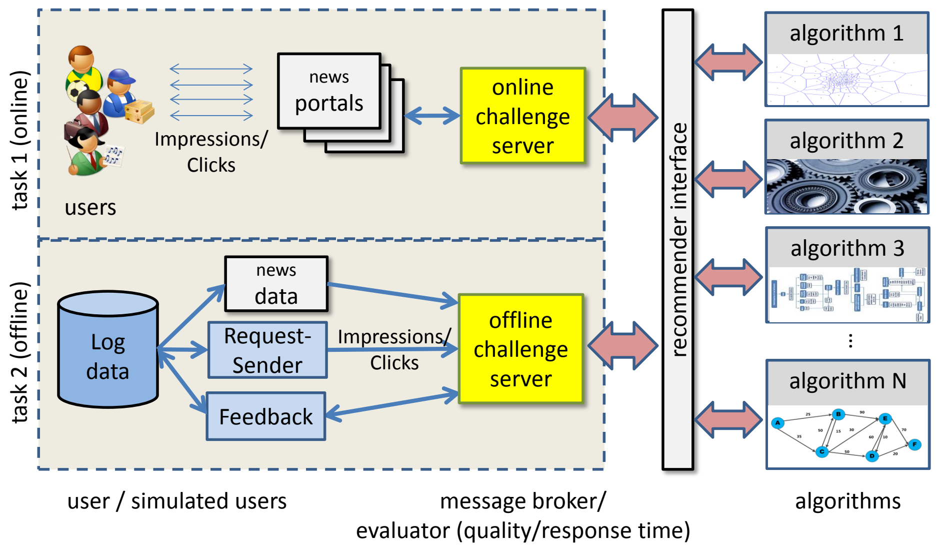

Fig. 1. The figure visualizes the similarities and differences between the online and the offline task. In task 1 (online) the impressions and recommendation requests are initiated by real users. The quality of the recommendations is evaluated based on the fraction of recommendations clicked by the users ("click-through-rate"). Task 2 (offline) simulates users based on the user behavior recorded in the online scenario. The recommender algorithms are similar in the online and the offline evaluation tasks. The recommender API ensures that all recommender algorithms use a similar interface and ensures that new strategies can be integrated in the system.

in Figure 1 the lab consists of two separate tasks targeting the benchmarking challenges from two different directions:
Task 1 focuses on the online evaluation. The participating teams register with the online system (ORP). Whenever a user visits a news web page assigned to the NewsREEL challenge, a recommendation request is sent to a randomly selected registered team. The team has to provide a list of up to 6 recommendations. The time constraint for completing the recommendation request is 100ms. In addition to the recommendation requests, there are messages describing the creation, removal, or update of news articles. The performance of the recommender algorithms is measure based on the click-through-rate (CTR) recorded in four pre-defined time frames. The scenario can be seen as an example of evaluation-as-a-service [19, 13] where a service API is provided rather than a dataset.

Task 2 focuses on the offline evaluation of stream-based recommender algorithms. The offline evaluation enables the reproducible evaluation of different recommender algorithms on exactly the same data. In addition, different parameter configurations for one algorithm can be analyzed in detail. In addition to the analysis of recommendation precision, Task 2 also enables the analysis of the technical complexity of different algorithms. Using virtual machines simulating different hardware settings the offline setting allows us to investigate the effect of the hardware resources and the load level on the response time and the recommendation precision.

Since Task 1 and Task 2 use very similar data formats recommender algorithms that are implemented can be tested in both online and offline evaluation. This allows the comprehensive evaluation of the strengths and weaknesses of the strategies of different algorithms. While last year's lab overview paper provided a detailed description of the online evaluation in a so-called living lab environment [14], this paper focuses on the simulation based evaluation that was applied in Task 2.

The remainder of this paper is structured as follows. Section 2 surveys related work in the field of online and offline evaluation. Section 3 provides a task description of the two tasks of NewsREEL and outlines the metrics used for benchmarking the different recommendation algorithms. Focusing on Task 2, Section 4 introduces the Idomaar benchmarking framework. Section 5 provides an overview of NewsREEL 2015. A discussion and conclusion is provided in Section 6.

2 Related Work

In this section we discuss related evaluation initiatives. In addition we focus on recommender algorithms able to take into account dynamic contexts and evaluations using a living lab approach.

2.1 Benchmarking using static datasets

CLEF NewsREEL is a campaign-style evaluation lab that focuses on benchmarking news recommendation algorithms. Benchmarking has been one of the main driving forces behind the development of innovative advances in the field. In the context of recommender systems evaluation, the release of the first MovieLens dataset[8] in 1998 can be seen as an important milestone. Since then, four different MovieLens datasets have been released. As of June 2015, 7500+ references to "movielens" can be found on Google Scholar, indicating its significance in education, research, and industry. The datasets consist of movie titles, ratings for these movies provided by users of the MovieLens system, and anonymized user identifiers. The ratings are stored as tuples in the form $\langle$user, item, rating, timestamp$\rangle$. While MovieLens focuses on movie recommendation, various datasets for other domains (e.g., [10]) have been released by now following a similar data structure.

Using these *static* datasets, a typical benchmarking task is to predict withheld ratings. The most important event that triggered research in the field is the Netflix Challenge where participants could win a prize for beating the baseline recommender system of a on-demand video streaming service by providing better predictions. Other benchmarking campaigns are organized as challenges in conjunction with Academic conferences such as the Annual ACM Conference Series on Recommender Systems (e.g., [2, 25]), and the European Semantic Web Conference (e.g., [22]), or as Kaggle competition (e.g., [21]).

[8] `http://movielens.org/`

Apart from providing static datasets and organizing challenges to benchmark recommendation algorithms using these datasets, the research community has been very active in developing software and open source toolkits for the evaluation of static datasets. Examples include Lenskit[9], Mahout[10], and RiVal[11].

2.2 Recommendations in dynamic settings

The research efforts that have been presented above have triggered innovation in the field of recommender systems, but the use of static datasets comes with various drawbacks.

Various research exists focusing on the use of non-static datasets, referred to as streamed data that showcase some of these drawbacks. Chen et al. [5] performed experiments on recommending microblog posts. Similar work is presented by Diaz-Aviles et al. [7]. Chen et al. [6] studied various algorithms for real-time bidding of online ads. Garcin et al. [9] and Lommatzsch [20] focus on news recommendation, the latter in the context of the scenario presented by NewsREEL.

All studies deal with additional challenges widely overlooked in a static context. In particular, research based on static databases does not take external factors into account that might influence users' rating behavior. In the context of news, such external factors could be emerging trends and news stories. In the same context, the freshness of items (i.e., news articles) plays an important role that needs to be considered. At the same time, computational complexity is out of focus in most academic research scenarios. Quick computation is of uttermost importance for commercial recommender systems. Differing from search results provided by an information retrieval system, recommendations are provided proactively without any explicit request by the user. Another challenge is the large number of requests and updates that online systems have to deal with.

Offline evaluation using a static dataset conducts an exact comparison between different algorithms and participating teams. However, offline evaluation requires assumptions, such as that past rating or consumption behavior is able to reflect future preferences. The benchmarking community is just starting to make progress in overcoming these limitations. Notable efforts from the Information Retrieval community include the CLEF Living Labs task [1], which uses real-world queries and user clicks for evaluation. Also, the TREC Live Question Answering task[12] involves online evaluation, and requires participants to focus on both response time and answer quality.

NewsREEL addresses the limitations of conventional offline evaluation in the area of recommender systems running an online evaluation. It also offers an evaluation setting that attempts to add the advantages of online evaluation, while retaining the benefits of offline evaluation. An overview of the NewsREEL recommendation scenario is provided in the next section.

[9] `http://lenskit.org/`
[10] `http://mahout.apache.org/`
[11] `http://rival.recommenders.net/`
[12] `https://sites.google.com/site/trecliveqa2015/`

3 Task Descriptions

As mentioned earlier, NewsREEL 2015 consists of two tasks in which news recommendation algorithms of streamed data can be evaluated in an online, and an offline mode, respectively. The online evaluation platform used in Task 1 enables participants to provide recommendations and observe users' responses. While this scenario has been described in detail by Hopfgartner et al. [14], Section 3.1 provides a brief overview of the underlying system and the evaluation metrics used. Task 2 is based on a recorded dataset providing the ground truth for the simulation-based evaluation. The dataset is presented in Section 3.2.

3.1 Task 1: Benchmark News Recommendations in a Living Lab

Researchers face different challenges depending on whether they work in industry or academia. Industrial researchers can access vast data collections. These collections better reflect actual user behavior due to their dimensionality. Industry requires researchers to quickly provide satisfactory solutions. Conversely, academia allows researchers to spend time on fundamental challenges. Academic research often lacks datasets of sufficiently large size to reflect populations such as internet users. The Open Recommendation Platform (ORP) [3] seeks to bridge this gap by enabling academic researchers to interactively evaluate their algorithms with actual users' feedback.

Participants connect their recommendation service to an open interface. Users visiting a selection of news portals initiate events. ORP randomly selects among all connected recommendation services and issues a request for recommendations. The selected recommendation service returns an ordered list of recommended items. This list must arrive within, at most, 100ms. In case of delayed responses, ORP forwards a precomputed default list as fall back.

In addition, participants receive notifications. These notifications either signal interactions between visitors and articles or articles being created or updated. ORP provides two types of interactions. Impressions refer to visitors accessing articles. 'Clicks' occur whenever visitors click on recommendations. Participants may use these data to implement their recommendation algorithms. Further, participants may exploit additional information sources to boost their performances.

The evaluation focuses on maximizing the visitors click on recommended items. Since the number of requested recommendations limits the number of clicks, ORP uses the ratio between the clicks and the number of requests for measuring the recommendation quality. This quantity is also known as Click-Through-Rate (CTR). A higher CTR indicates a superior ability to suggest relevant items. In real-life settings the CTR is often low ($\approx 1\%$) sufficient number of requests must be taken into account for ensuring the significance of the computed CTR scores.

We observe how users interact with news articles offered by various publishers. Publishers provide news articles with a headline, optionally an image, and a snippet of text. We interpret users clicking on such snippets as positive feedback. This assumption may not hold in all instances. For instance, users may fail to click on articles that match their interest. Similarly, users may misinterpret the title and ultimately find the article irrelevant. Dwell times could offer a more accurate picture of users' preferences.

Unfortunately, we cannot measure dwell times reliably. Most web sessions tend to be short and include only few articles. We cannot assure that users actually read the articles. Nonetheless, we expect users not to click on articles whose snippets they deem irrelevant.

The ORP provides four types of data for each participant:

- *Clicks:* Clicks refer to users clicking on an article recommended by the participant. Generally, we assume clicks to reflect positive feedback. The underlying assumption, as stated above, is that users avoid clicking on irrelevant articles.
- *Requests:* Requests refer to how often the participant received a recommendation request. The ORP delegates requests randomly to active, connected recommendation engines. Recommendation engines occasionally struggle to respond under heavy load. For this reason, the ORP temporarily reduces the volume of request under such circumstances. Participants with similar technical conditions should obtain approximately equal numbers of requests.
- *Click-through rate:* The CTR relates clicks and requests. It represents the ratio of requests which led to a click to the total number of requests. Hypothetically, a recommender could achieve a CTR of 100.0%. Each recommendation would have to be clicked to achieve such a perfect score. Humans have developed a blindness for contents such as advertisements. Frequently, publishers embed recommendations alongside advertisements. For this reason, there is a chance that users fail to notice the recommendations leading to fewer clicks than might have otherwise occurred. Historically, we observe CTR in the range of $0.5 - 5.0\%$.
- *Error Rate:* ORP reports the error rate for each participant. Errors emerge as recommendation engines fail to provide recommendations. The error rate denotes the proportion of such events within all requests. Ideally, a systems would have an error rate of 0.0%.

As a result, we can measure performance with respect to four criteria. First, we can determine the algorithm that received the most clicks. This might favor algorithms receiving a high volume of requests. Participants who lack access to powerful servers may fall short. Second, we can determine the algorithm that handles the largest volume of requests. Operating news recommenders have to handle enormous volumes of requests. This objective can be addressed by further optimizing the algorithms or by adding additional hardware. In the NewsREEL challenge we ought to avoid penalizing participants lacking hardware resources. Third, we can determine the algorithm obtaining highest CTR. The CTR reflects the system's ability to accurately determine users' preferences. As a drawback, we might not grasp how algorithms scale by analyzing CTR. A system might get a high CTR by chance on a small number of requests. Finally, we can determine how stably an algorithm performs in terms of the error rate. Although, a system may respond in time with inadequate suggestions and still obtain a perfect error rate. We chose CTR as decisive criteria. Additionally, we award the participants handling the largest volume of requests.

3.2 Task 2: Benchmark News Recommendations in a Simulated Environment

The NewsREEL challenge provides access to streams of interactions. Still, ORP routes requests to individual recommendation engines. Consequently, recommendation engines serve different groups of users in different contexts. We recorded interaction streams on a set of publishers. The stream-based evaluation issues these streams to different recommendation engines. Each engine faces the identical task. As a result, the stream-based evaluation improves comparability as well as reproducibility.

The dataset used in the offline evaluation has been recorded between July 1st, 2014 and August 31st, 2014. A detailed overview of the general content and structure of the dataset is provided by Kille et al. [15]. The dataset describes three different news portals: One portal providing general as well as local news, the second portal provides sport news; the third portal is a discussion board providing user generated content. In total, the dataset contains approximately 100 million messages. Messages are chronologically ordered. Thereby, participants could reduce the data volume by selecting subsets to explore larger parameter spaces.

Table 1. Data set statistics for Task 2.

	item create/update	user-item interactions	sum
July 2014	618,487	53,323,934	53,942,421
August 2014	354,699	48,126,400	48,481,099
sum	973,186	101,450,334	102,423,520

We evaluate the quality of news recommendation algorithms by chronologically reiterating interactions on news portals. Thereby, we simulate the situation which the system had faced while data recording. Unfortunately, we only obtain positive feedback and lack negative feedback. Unless the actual recommender had included the recommended items, we cannot tell how the user would have reacted. Nevertheless, we can obtain meaningful results as Li et al. [18] pointed out.

The evaluation of recommender algorithms online in a living lab leads to results that are difficult to reproduce since the set of users and items as well as the user preferences change continuously. This hampers the evaluation and optimization of algorithms due to the fact that different algorithms or different parameter settings cannot be tested in an exactly repeatable procedure. We seek to ensure reproducible results and to make sure that algorithms implemented by different teams are evaluated based on the same ground truth; the NewsREEL challenge also provides a framework for evaluating recommender algorithms offline using a well-defined, static dataset. The basic idea behind the offline evaluation is recording a stream in the online scenario that can be replayed in exactly the same way ensuring that all evaluation runs are based on the same dataset. Since the offline evaluation framework creates a stream that is based on the offline dataset, the adaptation of the recommender algorithms is not required. For the benchmarking of the recommender algorithms offline, we rely on similar metrics to those used in the online evaluation. Since there is no direct user feedback in the offline evaluation, the metrics must be slightly modified.

CTR: Instead of the *Click-Through-Rate* computed based on clicks in the live news portal, a simulated CTR is used that is computed based on a stream of recorded user interactions. In the offline evaluation, we assume that a recommendation is correct if the recommended item is requested by the user up to 5 minutes after the recommendation has been presented. This measure allows us to compute the CTR based on recorded data without requiring additional information. We do not have to adapt the definition of CTR since the offline CTR is still computed as the ratio between the recommended news items explicitly accessed by the user and the total number of computed recommendations. A disadvantage of the offline CTR is that the recorded user behavior is slightly influenced by the originally presented recommendation as well as by the presentation of news in the portal.

Computational Resources: We analyze the amount of computational resources required for providing recommendations. In order to have a controlled computation environment we use virtual machines. This ensures that the number of CPUs and the amount of RAM that can be used by the benchmarked algorithms is similar in all the evaluation runs. The measurement of the required resources is done using the management tools of the virtual machine.
In the NewsREEL offline evaluation we focus the benchmarking of the "computational complexity" in terms of the throughput. We analyze how effectively recommendations for the dataset can be computed based on the resources that are provided. The throughput can be measured by determining the number of recommendation that can be served by the system. In order to reach a maximal throughput, we have to ensure that the recommender algorithms are able to use multiple CPUs and an efficient management and synchronization strategy for concurrent threads is applied.

Response Time: One requirement in the news recommendation scenario is the provision of recommendation within the time limit of 100ms. For this reason, we analyze of response time distribution of the recommender algorithms that are implemented. Based on the idea of a service level agreement we calculate the relative frequency of cases in which the recommender cannot meet the time constraints.

Benchmarking recommender algorithms offline allows NewsREEL participants detailed insights in the characteristics of the implemented algorithms. Using exactly the same stream for comparing different parameter settings or recommender implementations ensures that the algorithms are benchmarked in the same setting. In addition, the offline evaluation supports the debugging of algorithms since the number of messages in the stream can be adapted. Furthermore, load peaks as well as special situation that can only rarely observed in the live evaluation. Even though the results obtained in the offline evaluation may not completely correlate with the online evaluation, the offline evaluation is very useful for understanding and optimizing recommender algorithms with respect to different aspects.

4 The Offline evaluation framework

Offline evaluation has been performed using *Idomaar*[13], a recommender system reference framework developed in the settings of the European Project CrowdRec[14] that addresses the evaluation of stream recommender systems. The key properties of Idomaar are:

- **Architecture independent**. The participants can use their preferred environments. Idomaar provides an evaluation solution that is independent of the programming language and platform. The evaluation framework can be controlled by connecting to two given communication interfaces by which data and control messages are sent by the framework.
- **Effortless integration**. The interfaces required to integrate the custom recommendation algorithms make use of open-source, widely-adopted technologies: Apache Spark and Apache Flume. Consequently, the integration can take advantage of popular, ready-to-use clients existing in almost any languages.
- **Consistency and reproducibility**. The evaluation is fair and consistent among all participants as the full process is controlled by the reference framework, which operates independently from the algorithm implementation.
- **Stream management**. Idomaar is designed to manage, in an effective and scalable way, a stream of data (e.g., users, news, events) and recommendation requests.

4.1 Idomaar architecture

The high-level architecture of Idomaar is sketched in Figure 2 and it is composed of four main components: Data container, Computing environment, Orchestrator, and Evaluator.

Data container The Data container contains the datasets available for experiments. The data format is composed by entities (e.g., users, news) and relations (e.g., events) represented by 5 tab-separated fields: object type (e.g., user, news, event, etc.), object unique identifier, creation timestamp (e.g., when the user registers with the system, when a news is added to the catalog, when the user reads a news, etc.), a set of JSON-formatted properties (e.g., the user name, the news category, the rating value, etc.), and a set JSON-formatted linked entities (e.g., the user and the news, respectively, subject and object of an event). Further details are described in [23].

Computing environment The Computing environment is the environment in which the recommendation algorithms are executed. Typically, for the sake of reproducibility and fair comparison, it is a virtual machine automatically provisioned by the Orchestrator by means of tools such as Vagrant[15] and Puppet[16]. The Computing environment communicates with the Orchestrator to (i) receive stream of data and (ii) serve recommendation requests. Future releases will also provide system statistics (e.g., CPU times, i/o activity).

[13] `http://rf.crowdrec.eu/`
[14] `http://www.crowdrec.eu/`
[15] `https://www.vagrantup.com/`
[16] `http://www.puppetlabs.com/`

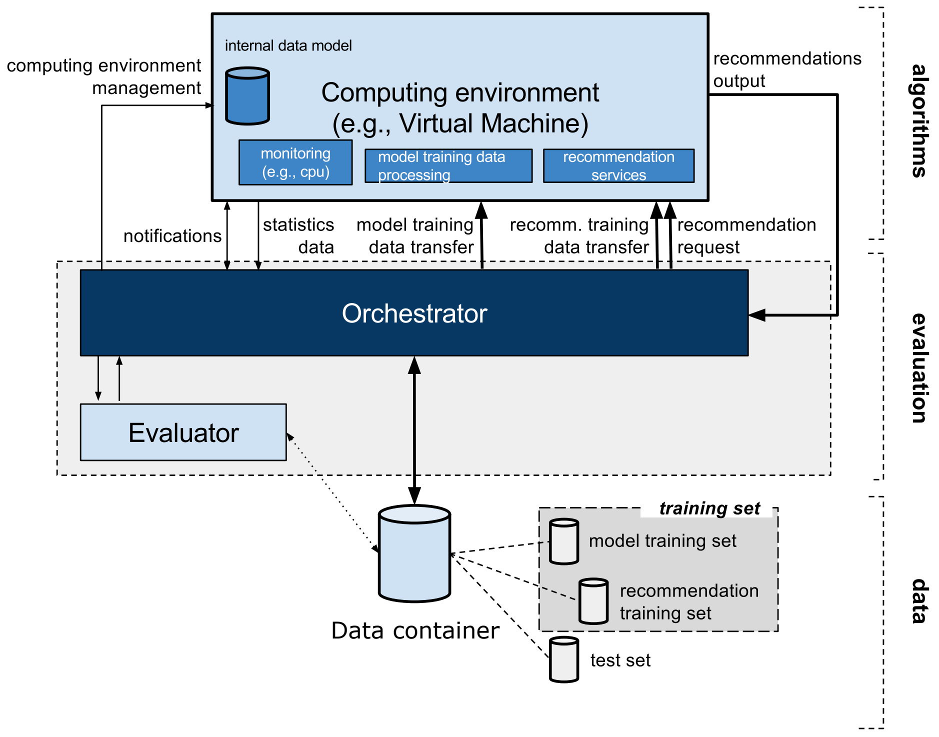

Fig. 2. The figure visualizes the architecture of the Idomaar framework used in the offline evaluation (Task 2).

Orchestrator The Orchestrator is in charge of initializing the Computing environment, providing training and test data at the right time, requesting recommendations, and eventually collecting the results to compute evaluation metrics. The Orchestrator may send a training dataset to the recommender algorithm in order to allow the algorithm to optimize on the dataset. Actually, for the NewsREEL challenge, there is no separate training data in order to keep the offline evaluation very similar to the online evaluation. However, additional training sets are supported by the Orchestrator enabling also traditional static training-, test-set based evaluations.

The Orchestrator uses the Kafka[17] messaging system to transmit data to the computing environment. Kafka is specifically designed to handle linear event sequences, and training and test data for recommender systems consist of such event sequences. Kafka has a relatively simple API and offers superior performance (for which strict delivery guarantees are sacrificed).

The Orchestrator has support for Flume[18], a plugin-based tool to collect and move large amounts of event data from different sources to data stores. In Idomaar, it provides flexibility: Flume has a couple of built-in sources and sinks for common situations

[17] `http://kafka.apache.org/`
[18] `https://flume.apache.org/`

(e.g., file-based, HTTP-based, HDFS) and it is straightforward to implement and use new ones if the need arises. Notably, there is a Flume source (and a Flume sink) that reads data from Kafka (and writes data to Kafka), meaning that Flume can serve as an integration layer between Kafka and a range of data sources.

Kafka and Flume are automatically installed on the Orchestrator virtual machine by Vagrant provisioning (using packages from Cloudera). At runtime, the Orchestrator is able to configure and bring up Flume by generating Flume property files and starting Flume agents. For instance, the Orchestrator can instruct Flume to write recommendation results to plain files or HDFS.

Computing environments have the option to receive control messages and recommendation requests from the Orchestrator via ZeroMQ[19] or HTTP, and data via Kafka. In the NewsREEL competition, recommendation engines implement an HTTP server, so Idomaar is used in its pure HTTP-mode. The HTTP interface in Idomaar is implemented as a Flume plugin.

Evaluator The Evaluator contains the logic to (i) split the dataset according to the evaluation strategy and (ii) compute the quality metrics on the results returned by the recommendation algorithm. As for NewsREEL, the data is a stream of timestamped user events; the Computing environment is flooded with such events that can be used to constantly train the recommendation models. Randomly, some events are selected and, in addition to the new information, the Orchestrator sends a recommendation request for the target user. All news consumed by such user in the upcoming 5 minutes form the groundtruth for such recommendation request. The quality of results is measures in terms of CTR, as described in Section 3.2.

Splitting and evaluations are implemented as Apache Spark scripts, so that they can be easily customized and run in a scalable and distributed environment.

4.2 Idomaar data workflow

The data workflow implemented in Idomaar complies with the following three sequential phases: (i) data preparation, (ii) data streaming, and (iii) result evaluation.

Phase 1: data preparation The first phase consists in reading the input data (entities and relations) and preparing them for experimenting with the recommendation algorithms. The Evaluator is used to split the data, creating a training set and ground truth data ("test set"). In the case that the data preparation is already done by explicit markers in the dataset (as it is done in NewsREEL Task 2), this phase can be skipped.

Phase 2: data streaming Initially, once the Computing environment has booted, the recommendation models can be optionally bootstrapped with an initial set of training data. Afterwards, the Orchestrator floods the computing environment with both information messages (e.g., new users, news, or events) and recommendation requests. The second phase terminates when the Computing environment has processed all messages.

[19] `http://zeromq.org/`

The output of the Computing environment is stored in an extended version of the Idomaar format, composed by an additional column where the recommendation response for a given recommendation request is saved.

Phase 3: result evaluation The last phase is performed by the Evaluator that compares the results returned by the computing environment with the created ground truth in order to estimate some metrics related to the recommendation quality (i.e., CTR).

In addition, the Orchestrator is seated in a position that makes it possible to measure metrics related to the communication between the Orchestrator (which simulates the final users) and the computing environment (which represents the recommender system), such as the response time.

4.3 Discussion

In this section, we have presented the evaluation framework supporting the efficient, reproducible evaluation of recommender algorithms. Idomaar is a powerful tool allowing users to abstract from concrete hardware or programming languages by setting up virtual machine having exactly defined resources. The evaluation platform allows a high degree of automatization for setting up the runtime environment and for initializing the evaluation components. This ensures the easy reproducibility of evaluation runs and the comparability of results obtained with different recommender algorithms. Idomaar supports the set-based as well as the stream-based evaluation of recommender algorithms.

In NewsREEL Task 2, the steam-based evaluation mode is used. In contrast to most existing evaluation frameworks Idomaar can be used out of the box and, for evaluation, considers not only the recommendation precision but also the resource demand of the algorithms.

5 Evaluation

The NewsREEL challenge 2015 attracted teams from 24 countries to develop and evaluate recommender algorithms. In this section, we provide details about the registered teams and the implemented algorithms. In addition, we explain the provided baseline recommender algorithm. Finally, we report the performance scores for the different algorithms and discuss the evaluation results. A more detailed overview can be found in [16].

5.1 Participation

A total of 42 teams registered for NewsREEL 2015. Of these, 38 teams signed up for both tasks. Figure 3 illustrates the spread of teams around the Globe. Central Europe, Iran, India, and the United States of America engaged most. Network latency may negatively affect the performance in Task 1 of team located far from Europe. Five teams received virtual machines to run their algorithms and alleviate latency issues. In the final evaluation phase of Task 1, we observed 8 actively competing teams. Each team could run several algorithms. Some teams explored a larger segment of algorithms. This led to a total of 19 algorithms competing during the final evaluation round of Task 1.

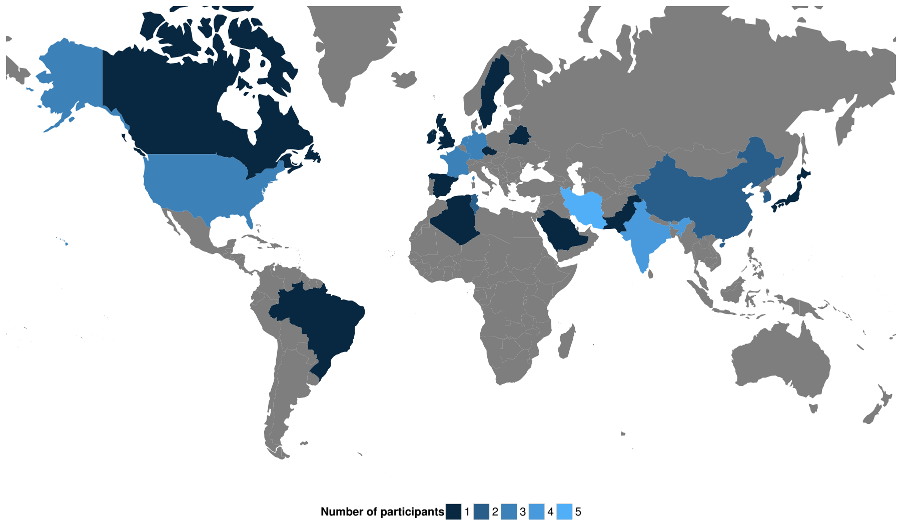

Fig. 3. The figure shows the participation around the world. Countries colored gray had no participation. Lighter blue colors indicate more participants than darker shades.

5.2 The Baseline algorithm

The NewsREEL challenge provides a baseline algorithm implementing a simple, but powerful recommendation strategy. The strategy recommends users the items most recently requested by other users. The idea behind this strategy is that items currently interesting to users might also be interesting for others. Thereby, the strategy assumes that users are able to determine relevant articles for others.

Implementation of the baseline recommender The most recently requested recommender is implemented based on a ring buffer. Whenever a user requests a new item, the system adds the item to the ring buffer. In order to keep insertion as simple as possible, duplicate entries in the buffer are allowed. If the ring buffer is completely filled, a new newly added item overwrites the oldest entry in the buffer. Upon receiving a recommendation request, we search for n distinct items starting at the most recently added. The process iterates in reverse order through the buffer until we collected n distinct items. Since the buffer may contain duplicate entries, the size of the ring buffer must be large enough that for all request at least n distinct items can be found. In addition, items may be blacklisted (e.g., because they are already known to the user) and excluded from the result set.

Properties of the baseline recommender The provided baseline recommender has several advantages. Since the recommender only considers the item requested by other users during the last few minutes, the recommendation usually fits well with respect to the time-based context. In addition, the recommendations are biased towards popular items requested by many different users. Since users typically request news items from different fields of interest, the suggestions provided by the least-recently requested recommender are often characterized by a certain level of diversity, which supports recommendation of serendipitous news items.

Recommendation Precision The baseline recommender has been tested in the online and the offline evaluation. Due to the limited memory used by the algorithm, the recommender quickly adapts to new users and items. The cold-start phase of the algorithm is short; as soon as there are sufficient distinct entities in the ring buffer, the recommender works correctly. Comparing the least-recently requested algorithms with alternative recommender strategies, the baseline recommender behaves similarly to a most-popular recommender with a short "window" used for computing the most popular items.

Figure 4 shows the CTR of the baseline recommender observed during the final evaluation period of NewsREEL 2015. The figure shows that the CTR typically varies between 0.5% and 1.5% reaching an average CTR of 0.87%.

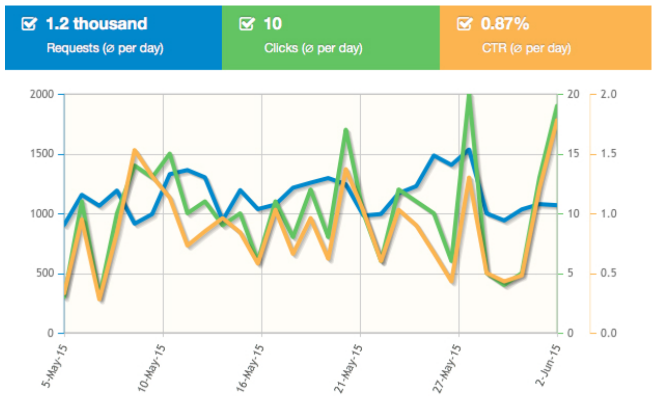

Fig. 4. The plot shows the CTR of the baseline recommender algorithm for the NewsREEL's evaluation period (May–June 2015).

Required computation resources The implementation of baseline recommender uses a ring buffer allocating a fixed amount of memory. This prevents problems with allocating and releasing memory while running the recommender. Concurrent threads accessing the ring buffer can be handled in a simple way allowing dirty read and write operations, since we do not require strong consistency of items contained in the buffer. The avoidance of locks and synchronized blocks simplifies the implementation and ensures that active threads are not blocked due to synchronization purposes. Due to the limited amount of memory required for the ring buffer, the baseline recommender keeps all necessary data in the main memory and does not require hard drive access. The small number of steps for computing recommendations and the simple (but dirty) synchronization strategy leads to a very short response time ensure that the time constraints are reliably fulfilled.

The baseline recommender is a simple, but powerful recommender reaching a CTR of $\approx 0.9\%$ in the online evaluation.

5.3 Evaluated Algorithms

Last year's NewsREEL edition produced a variety of ideas to create recommendation algorithms. We highlight three contributions. Castellanos et al. [11] created a content-based recommender. Their approach relies on a Formal Concept Analysis Framework. They represent articles in a concept space. As users interact with articles, their method derives preferences. The system projects these preferences onto a lattice and determines the closest matches. They report that content-based methods tend to struggle under heavy load. Doychev et al. [8] analyzed strategies with different contextual features. These features include time, keywords, and categories. They show that combining different methods yields performance increases. Finally, Kuchar and Kliegr [17] applied association rule mining techniques to news recommendation. Association rule mining seeks to discover regularities in co-occurring events. For instance, we may observe users frequently reading two particular articles in rapid sequence. Consequently, as we recognize a user reading one of them, we may consider recommending the remaining one. In this year's installment of NewsREEL, participants explored various ideas. The Team "cwi" investigated the potential improvement through considering geographic locations of news readers. Team "artificial intelligence" used time context and device information to build a meta recommender. Based on contextual factors, the system picked the most promising algorithm from a set of existing recommenders. Team "abc" extends the approach of team "artificial intelligence" by considering trends with respect to success of individual recommenders [20]. The remaining participants have not yet revealed their approaches. More details will be added to the working notes overview paper. Apart from Task 1 related approaches, we received some insights concerning Task 2. The team "irs" applied the Akka[20] framework to the task of news recommendation. They paid particular attention toward ensuring response time constraints and handling of request peaks. Akka allows concurrently running processes on multiple machines and CPUs for the purpose of load balancing. Team "7morning" tried to identify characteristic patterns in the data stream. Subsequently, they extrapolated these patterns to accurately predict future interactions between users and news articles.

5.4 Evaluation results

Task 1 challenged participants to suggest news articles to visitors of publishers. The more visitors engaged with their suggested, the better we deemed their performances. The Open Recommendation Platform (ORP) seeks to balance the volume of requests. Generally, each participating recommendation service ought to receive a similar proportion of requests. Still, this requires all recommendation services to be available at

[20] `http://akka.io/`

any time. We observed some teams exploring various algorithms. As a result, some algorithms were partly active throughout the evaluation time frame. Consequently, they received fewer requests compared to algorithms running the full time span. Figure 5 related the volume of requests and the number of clicks for each recommendation service. We congratulate the teams “artificial intelligence” (CTR = 1.27%), “abc” (CTR = 1.14%), and “riadi-gdl” (CTR = 0.91%) on their outstanding performance. We ran two baselines varying in available resources. The baselines are “riemannzeta” and “gaussiannoise”. We observe that both baselines achieve competitive CTR results.

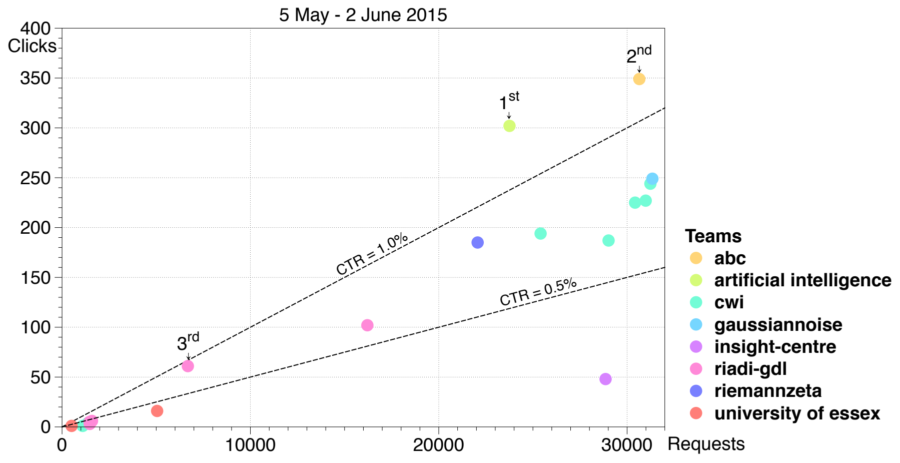

Fig. 5. Results of the final evaluation conducted from May 5 to June 2, 2015. The figure shows the volume of requests on the x-axis, and the number of clicks on the y-axis. Each point refers to the click-through-rate of an individual recommendation service. Colors reflect which team was operating the service. The closer to the top left corner a point is located, the higher the resulting CTR. Dashed lines depict CTR levels of 1.0% and 0.5%. The best performances have labels indicating their place assigned.

5.5 Discussion

The NewsREEL challenge gives participating teams the opportunity for evaluating individual algorithms for recommending news articles. Analyzing the implemented strategies and discussing with the researchers, we find a wide variety of approaches, ideas, and programming languages. The performance as well as the response time of the algorithms varies with the algorithms and contexts. Thus, the performance ranking may change during the course of a single day. In order to compute a reliable ranking, the challenge uses a comprehensive evaluation period (4 weeks in Task 1) and a huge dataset (consisting of $\approx$ 100 million messages in Task 2) respectively. The baseline recommender performs quite successfully, being always among the best 8 recommender algorithms.

6 Conclusion and Outlook

In this paper, we have presented the CLEF NewsREEL 2015 challenge that requires participants to develop algorithm capable of processing a stream of data, including news items, users, and interaction events, and generating news item recommendations. Participants can choose between two tasks, Task 1, in which their algorithms are tested online, and Task 2, in which their algorithms are tested offline using a framework that 'replays' data streams. The paper has devoted particular attention to the framework, called Idomaar, which makes use of open source technologies designed for straightforward usage. Idomaar enables a fair and consistent evaluation of algorithms, measuring the quality of the recommendations, while limiting or tracking the technical aspects, such as throughput, required CPU resources, and response time.

The NewsREEL 2015 challenge supports recommender system benchmarking in making a critical step towards wide-spread adoption of online benchmarking (i.e., "living lab evaluation"). Further, the Idomaar framework for offline evaluation of stream recommendation is a powerful tool that allowing multi-dimensional evaluation of recommender systems "as a service". Testing of stream-based algorithms is important for companies who offer recommender systems services, or provide recommendations directly to their customers. However, until now, such testing has occurred in house. Consistent, open evaluation of algorithms across the board was frequently impossible. Because NewsREEL provides a huge dataset and enables reproducible evaluation of recommender system algorithms, it has the power to reveal underlying strengths and weaknesses of algorithms across the board. Such evaluation provide valuable insights that help to drive forward the state of the art.

We explicitly point out that the larger goal of both Task 1 and Task 2 of the NewsREEL 2015 challenge is to evaluate stream-based recommender algorithms not only with respect to their performance as measured by conventional user-oriented metrics (i.e., CTR), but also with respect to their technical aspects (i.e., response time). As such, the NewREEL challenge takes a step towards realizing the paradigm of 3D benchmarking [24].

We face several major challenges as we move forward. These challenges must be addressed by a possible follow-up NewsREEL challenge, but also by any benchmark that aspires to evaluate stream recommendations with respect to both user and technical aspects. First, stream-based recommendation is a classic big data challenge. In order to ensure that a benchmark addresses a state-of-the-art version of the problem, it is necessary to continuously monitor new tools that are developed. Here, we are particularly interested in keeping up with the developments of key open source tools for handling data streams. Allowing the reference framework to track these developments requires a significant amount of engineering effort. Second, it is necessary to keep the threshold for participating in the benchmark low. In other words, new teams should be able to test their algorithms with a minimal of prior background knowledge or set up time. In 2015, we notice that it requires an investment for teams to be able to understand the complexities of stream-based recommendation, and how they are implemented within Idomaar. Again, a considerable amount of engineering effort is needed to ensure that Idomaar is straightforward to understand and easy to use. Finally, additional work is needed to fully understand the connection between online evaluation and the "replayed" stream used in offline evaluation. The advantage of offline testing is clear: on-demand exact repeatability of experiments. However, it also suffers from particular limitations. In the future, we will continue to work to understand the potential of using offline testing in place of online testing.

Acknowledgments

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

References

1. K. Balog, L. Kelly, and A. Schuth. Head first: Living labs for ad-hoc search evaluation. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 1815–1818, New York, NY, USA, 2014. ACM.
2. J. Blomo, M. Ester, and M. Field. Recsys challenge 2013. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 489–490, 2013.
3. T. Brodt and F. Hopfgartner. Shedding Light on a Living Lab: The CLEF NEWSREEL Open Recommendation Platform. In *Proceedings of the Information Interaction in Context conference*, IIiX'14, pages 223–226. Springer-Verlag, 2014.
4. P. G. Campos, F. Díez, and I. Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Model. User-Adapt. Interact.*, 24(1-2):67–119, 2014.
5. J. Chen, R. Nairn, L. Nelson, M. S. Bernstein, and E. H. Chi. Short and tweet: experiments on recommending content from information streams. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Atlanta, Georgia, USA, April 10-15, 2010*, pages 1185–1194, 2010.
6. Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur. Real-time bidding algorithms for performance-based display ad allocation. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1307–1315, 2011.
7. E. Diaz-Aviles, L. Drumond, L. Schmidt-Thieme, and W. Nejdl. Real-time top-n recommendation in social streams. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, pages 59–66, 2012.
8. D. Doychev, A. Lawlor, and R. Rafter. An analysis of recommender algorithms for online news. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 825–836, 2014.
9. F. Garcin, B. Faltings, O. Donatsch, A. Alazzawi, C. Bruttin, and A. Huber. Offline and online evaluation of news recommender systems at swissinfo.ch. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, pages 169–176, 2014.
10. K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2), July 2001.
11. Á. C. Gonzáles, A. M. García-Serrano, and J. Cigarrán. UNED @ clef-newsreel 2014. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 802–812, 2014.

12. GroupLens Research. MovieLens data sets. `http://www.grouplens.org/node/73`, Oct. 2006.
13. F. Hopfgartner, A. Hanbury, H. Mueller, N. Kando, S. Mercer, J. Kalpathy-Cramer, M. Potthast, T. Gollup, A. Krithara, J. Lin, K. Balog, and I. Eggel. Report of the evaluation-as-a-service (eaas) expert workshop. *SIGIR Forum*, 49(1):57–65, 2015.
14. F. Hopfgartner, B. Kille, A. Lommatzsch, T. Plumbaum, T. Brodt, and T. Heintz. Benchmarking news recommendations in a living lab. In *5th International Conference of the CLEF Initiative*, pages 250–267, 2014.
15. B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz. The plista dataset. In *NRS'13: Proceedings of the International Workshop and Challenge on News Recommender Systems*, pages 14–21. ACM, 10 2013.
16. B. Kille, A. Lommatzsch, R. Turrin, A. Serny, M. Larson, T. Brodt, J. Seiler, and F. Hopfgartner. Overview of clef newsreel 2015: News recommendation evaluation labs. In *6th International Conference of the CLEF Initiative*, 2015.
17. J. Kuchar and T. Kliegr. Inbeat: Recommender system as a service. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 837–844, 2014.
18. L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, pages 297–306, 2011.
19. J. Lin and M. Efron. Evaluation as a service for information retrieval. *SIGIR Forum*, 47(2):8–14, 2013.
20. 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.
21. B. McFee, T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. The million song dataset challenge. In *Proceedings of the 21st International Conference Companion on World Wide Web*, WWW '12 Companion, pages 909–916, 2012.
22. T. D. Noia, I. Cantador, and V. C. Ostuni. Linked open data-enabled recommender systems: ESWC 2014 challenge on book recommendation. In *Semantic Web Evaluation Challenge - SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 129–143, 2014.
23. A. Said, B. Loni, R. Turrin, and A. Lommatzsch. An extended data model format for composite recommendation. In *Poster Proceedings of the 8th ACM Conference on Recommender Systems, RecSys 2014, Foster City, Silicon Valley, CA, USA, October 6-10, 2014*, 2014.
24. A. Said, D. Tikk, K. Stumpf, Y. Shi, M. Larson, and P. Cremonesi. Recommender systems evaluation: A 3d benchmark. pages 21–23, 2012.
25. M. Tavakolifard, J. A. Gulla, K. C. Almeroth, F. Hopfgartner, B. Kille, T. Plumbaum, A. Lommatzsch, T. Brodt, A. Bucko, and T. Heintz. Workshop and challenge on news recommender systems. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 481–482, 2013.