OpenSearch: Lessons Learned from an Online Evaluation Campaign

ROLF JAGERMAN, University of Amsterdam
KRISZTIAN BALOG, University of Stavanger
MAARTEN DE RIJKE, University of Amsterdam

We report on our experience with TREC OpenSearch, an online evaluation campaign that enabled researchers to evaluate their experimental retrieval methods using real users of a live website. Specifically, we focus on the task of *ad hoc* document retrieval within the academic search domain, and work with two search engines, CiteSeerX and SSOAR, that provide us with traffic. We describe our experimental platform, which is based on the living labs methodology, and report on the experimental results obtained. We also share our experiences, challenges, and the lessons learned from running this track in 2016 and 2017.

CCS Concepts: • **Information systems → Evaluation of retrieval results**; *Test collections*; *Retrieval effectiveness*;

Additional Key Words and Phrases: Living labs, online evaluation

ACM Reference format:
Rolf Jagerman, Krisztian Balog, and Maarten de Rijke. 2018. OpenSearch: Lessons Learned from an Online Evaluation Campaign. *J. Data and Information Quality* 10, 3, Article 13 (September 2018), 15 pages.
https://doi.org/10.1145/3239575

1 INTRODUCTION

Information Retrieval (IR) is about connecting people to information. Users have always been central to the design and evaluation of retrieval systems. For a long time, system-oriented evaluation has primarily been performed using *offline* test collections, following the Cranfield paradigm. This rigorous methodology ensures the repeatability and reproducibility of experiments, and has been instrumental to the progress made in the field. However, it has an inherent limitation; namely, that the actual user is, to a large extent, abstracted away. Ways to overcome this include laboratory user

This research was supported by Ahold Delhaize, Amsterdam Data Science, the Bloomberg Research Grant Program, the Criteo Faculty Research Award Program, Elsevier, the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement No. 312827 (VOX-Pol), the Microsoft Research Ph.D. Program, the Netherlands Institute for Sound and Vision, the Netherlands Organisation for Scientific Research (NWO) under Projects No. CI-14-25, No. 652.002.001, No. 612.001.551, No. 652.001.003, and Yandex. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

Authors' addresses: R. Jagerman (corresponding author), University of Amsterdam, Amsterdam, The Netherlands, Science Park 904, 1098XH; email: rolf.jagerman@uva.nl; K. Balog, University of Stavanger, Stavanger, Norway, Kristine Bonnevies vei 22, 4021; email: krisztian.balog@uis.no; M. de Rijke, University of Amsterdam, Amsterdam, The Netherlands, Science Park 904, 1098XH; email: derijke@uva.nl.

1936-1955/2018/09-ART13 $15.00
https://doi.org/10.1145/3239575

studies [10], simulated users [17], and online evaluation [6]. Our focus in this article falls in the latter category.

The idea behind online evaluation is to observe users *in situ*, i.e., in a live setting in their natural task environments. Hence, the search engine operates as a "living lab." All major search engines function as living labs, but these experimental facilities are restricted to those working at the respective organizations. This means that most academic researchers do not have access to real users, and are thus required to resort to simulated users or to datasets annotated by trained assessors. Moreover, the scarce online evaluation resources available to researchers, whether academic or industrial, usually cannot be shared, making it hard to compare or replicate experimental results. The OpenSearch track at the Text Retrieval Conference (TREC) represents a recent effort that aims to address this problem by opening up live evaluation resources to the community:

> "Open Search is a new evaluation paradigm for IR. The experimentation platform is an existing search engine. Researchers have the opportunity to replace components of this search engine and evaluate these components using interactions with real, unsuspecting users of this search engine" [3].

Specifically, TREC OpenSearch focuses on the task of academic literature search, using various academic search engines as live sites from which user traffic is used. The task is set up as an *ad hoc* document retrieval task. The live sites provide a set of queries, for which a selection of candidate documents have to be ranked. The teams participating in the experiment submit rankings, which are interleaved with the production system during testing, and clicks are recorded.

Our contributions in this article are twofold. First, we present an analysis of data and results obtained from the TREC 2016–2017 OpenSearch tracks and discuss lessons learned. Second, we release a curated dataset containing observed clicks on interleaved result lists during these tracks.

The remainder of this article is structured as follows. First, we provide background material on evaluation in IR in Section 2. Next, we provide an overview of the living labs evaluation methodology in Section 3. In Section 4, we discuss the academic use-case. Our results and analysis are presented in Section 5. Finally, we conclude in Section 6.

2 BACKGROUND

The Cranfield paradigm, where several judges attempt to quantify the relevance of documents for a query [5], is a widely adopted IR evaluation paradigm, both in academia and industry. Relevance judgements collected under the Cranfield paradigm can be used to evaluate the performance of a ranking system in an *offline setting*. Although this benefits the repeatability and reproducibility of experiments, there are major caveats: (1) it is expensive to obtain relevance judgments at scale, as these have to be generated by humans; (2) it assumes that relevance is a concept that does not change over time; (3) it ignores the fact that for some scenarios, i.e., private email search, the actual user is the only credible judge; and (4) it assumes that relevance judgments produced by professional judges accurately represent the preferences of real users.

Online evaluation mitigates these problems by inferring preferences directly from the interactions of real users. There are two common ways of performing online evaluation of IR systems:

(1) **A/B Testing**
In A/B testing [12, 13], traffic to a search engine is split uniformly at random into two buckets called A and B. Bucket A is typically considered the control group, where nothing changes. Bucket B runs the alternative system we wish to evaluate. By observing statistically significant changes between the behavior of users in bucket A and those in bucket B, we can draw conclusions about the effectiveness of the alternative system.

Table 1. Overview of Living Labs Benchmarking Efforts

Name	Task	Data	Access	Evaluation
CLEF LL4IR	product search	product records	API	interleaving
	web search	feature vectors	API	interleaving
CLEF NewsREEL	news recommendation	news items	API	A/B testing
NTCIR-13 OpenLiveQ	question ranking	QA pairs	download	multileaving

(2) **Interleaving**
With interleaved comparisons [8, 14], the results of two systems *A* and *B* ("production" and "alternative") are combined into a single Search Engine Result Page (SERP), which is then shown to the user. Preferences for either system *A* or *B* can be inferred by observing which results get clicked, and attributing the clicked result to either one of the systems being evaluated. Because of their within-subject nature, interleaved comparisons are usually much more data efficient than A/B testing for studies of comparable dependent variables [4].

For online evaluation, an online interactive system is required. This naturally limits the use of this methodology to researchers at organizations with a live system. The Living Labs methodology [1] has been proposed to open up online evaluation as an service to third parties, allowing them to expose rankings generated by experimental retrieval methods to live users.

Over the past years, we have seen a number of operationalizations of this methodology at large-scale evaluation campaigns (see Table 1). One is the News Recommendation Evaluation Lab (NewsREEL) at the Conference and Labs of the Evaluation Forum (CLEF) 2015–2017, which aims at optimizing news recommender algorithms [7]. Participating systems at NewsREEL need to operate a service that responds to requests within 100ms. The recommendations of a randomly selected system are shown to the user directly.

Another instance is provided by the OpenLiveQ (Open Live Test for Question Retrieval) task at NTCIR-13 [9], in which question retrieval systems are evaluated in the production environment of Yahoo! Chiebukuro (a community Q&A service); evaluation is done using multileaving [16], a generalization of interleaving to more than two competing approaches.

Yet another instance is the Living Labs for IR Evaluation (LL4IR) Lab at CLEF 2015 and 2016. LL4IR relied on the idea of focusing on head queries, as proposed by Balog et al. [2], thereby removing the requirement of providing rankings in real time for query requests. Two use-cases were studied at LL4IR: product search and web search; see Reference [15] for an overview. TREC OpenSearch follows the same methodology of focusing on head queries, as we will explain next.

3 EVALUATION METHODOLOGY

TREC OpenSearch implements the living labs evaluation methodology [2]. It allows third parties to perform online evaluation on an existing service and enables experimentation using real users. The idea behind living labs is to share a common service for evaluation instead of having every research group attempt to build and maintain their own platform. The major benefit of sharing an experimental platform in this way is that all researchers can make use of the large and active user base of a deployed and well-maintained service without any overhead.

Before we dive into a detailed explanation of TREC OpenSearch's architecture, it is important to introduce the main concepts that will be used throughout this article:

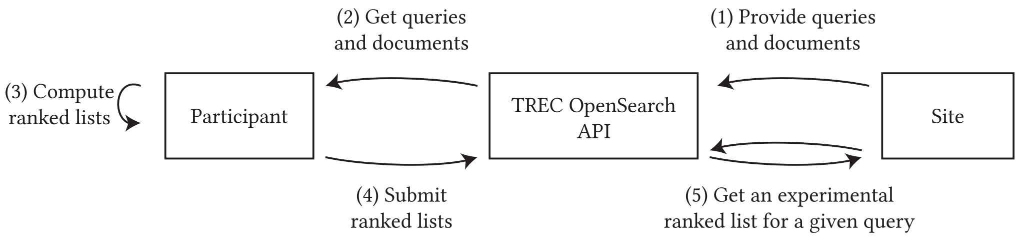
(2) Get queries and documents
(1) Provide queries and documents
(3) Compute ranked lists
Participant
TREC OpenSearch API
Site
(4) Submit ranked lists
(5) Get an experimental ranked list for a given query

Fig. 1. High-level overview of TREC OpenSearch architecture.

Site. An interactive website that makes their service available for third parties to run experiments involving their live traffic.

Participant. A third party who wants to experiment with some or all of the users that make up the traffic of a site. In our case, these third parties are participants of the TREC OpenSearch track.

API. The service that acts as a mediator between sites and participants. OpenSearch uses the Living Labs API.[1]

Experimental query. Sites make a designated set of queries available for experimentation. Following Reference [2], these are usually taken to be so-called head queries, i.e., queries that are expected to be issued frequently by the site's users. These queries are further subdivided into a training and test set by the organizers of the evaluation campaign, so that every participant uses the same splits.

Ranked list. A relevance-ordered list of documents for a single query. (The collection of ranked lists for the entire set of test queries constitutes of what is known as a *run* in TREC lingo.)

Evaluation round. TREC OpenSearch is organized in several evaluation rounds, each typically lasting between 4 and 6 weeks. During an evaluation round, the ranked lists that are submitted for test queries cannot be changed, in order to compare participating systems in fair way.

Impression. An impression occurs when a user of a site issues an experimental query and observes a ranked lists.

Click. A click is recorded whenever a user interacts with the observed ranked list and clicks on one of the results in the ranked lists.

A high-level overview of the TREC OpenSearch architecture is provided in Figure 1. The site makes available a set of experimental queries and a set of candidate documents for each query through the API. Participants need to generate a (re)ranked list of these candidate documents, and then upload the generated ranked lists to the API. When one of the experimental queries is issued by a user, the site requests a random participant's ranked lists from the API. This ranked list is interleaved with a ranked list produced by the site's production system, then presented to users. The user's click actions are recorded, and this feedback is later submitted to the API. Each of these steps will now be explained in more detail.

In the first step in Figure 1, experimental queries are selected by the site, typically based on historical log data. The selected queries are head queries, meaning that they appear very frequently and are likely to be issued again in the future. For each query, the site also provides a set of candidate documents. This also functions as a safety mechanism, to ensure that no "nonsense" is returned to users of the site. The content of the documents is made available in a semistructured

[1] The API is publicly available at https://bitbucket.org/living-labs/ll-api/.

form, encoded as JSON. This allows sites to provide additional metadata and it makes it easier to separate different fields in the content, such as author, title, and abstract. The queries are separated into *training* and *test* queries. Participants will receive interaction (click) feedback on the training queries, allowing them to tune their ranking algorithms, while test queries are used solely for evaluation purposes. That is, for test queries, only aggregated feedback statistics are made available and only at the end of the evaluation round.

In the second and third steps in Figure 1, participants download the experimental query and document collections and produce ranked lists using their experimental methods. As a fourth step, they submit their computed ranked lists back to the API. Ranking is thus done in an *offline* fashion. This removes privacy concerns and lowers the barrier of entry. A limitation of this setup, however, is that it is not feasible to include contextual information about the current user, making it impossible to experiment with personalization.

Whenever a user submits one of the candidate queries to the site, the site will ask the API for a participant's ranking, as illustrated in step 5 of Figure 1. In the 2016 track, we used a uniform random process to select users, while in 2017 we implemented a load-balancer to distribute traffic more fairly across users. The selected ranking is returned to the site. In an A/B test, we would at this point either display the participant ranking or the production ranking depending on the bucket the user falls in. Because the participant ranking comes from a third party, it is not very trustworthy and could be potentially very bad. We do not want to expose the site to such risks, so instead we use *interleaving* to ensure that the displayed ranking contains documents from both the production ranking and the participant's ranking. More specifically, we use *Team-Draft Interleaving (TDI)*, which is explained in more detail in Appendix B.

Once the interleaved ranked list is displayed to the user, the user can decide to interact with it or not. The interactions that happen are recorded in the form of clicks. From these clicks it is possible to infer whether the user prefers the production ranking or the participant ranking, producing a winner or a draw.

4 THE ACADEMIC SEARCH USE-CASE

The setup we have introduced in the previous section is very generic and is applicable to any ranking problem. Nevertheless, to make an evaluation exercise meaningful, it needs to be rooted in a specific domain and task. At OpenSearch, this domain is *academic search*, and the specific task is *ad hoc scientific literature search*: given a keyword query, return a ranked list of documents (scientific articles). We chose the academic search task for practical reasons, the search engines in this domain were very willing to participate. The *ad hoc* search task was chosen because it integrates well with the existing living-labs infrastructure and it avoids any potential problems with privacy (e.g., personalization or recommendation tasks). The *ad hoc* scientific literature search task has been evaluated with two academic search engines as our sites: CiteSeerX (in 2016) and SSOAR (in 2016 and 2017). These sites vary in terms of the scientific field as well as in the document fields/metadata that is made available, as we will detail below. Table 2 provides an overview of the rounds and the number of queries that were used for training and testing.

4.1 CiteSeerX

CiteSeerX [18] is a digital library search engine with a main focus on computer and information sciences. As of October, 2016, CiteSeerX included 8.7 million unique articles and 1.3 million unique authors. The documents uploaded to the OpenSearch API comprise two fields: (i) the document title and (ii) the full body text extracted from the PDF file. If the full document text is unavailable, the abstract is used instead. An example document entry is shown in Listing 1. The (head) queries were extracted based on access logs from 2014. The first two rounds in 2016 used a roughly even

Table 2. Overview of Evaluation Rounds at TREC OpenSearch

Year	Round	Period	CiteSeerX queries		SSOAR queries	
			training	test	training	test
2016	Round 1	June 1–July 15	100	107	57	74
	Round 2	Aug. 1–Sep. 15	100	107	57	74
	Round 3	Oct. 1–Nov. 15	100	871	57	1062
2017	Round 1	Aug. 1–Aug. 31			655	501
	Round 2	Oct. 1–Oct. 31			655	501

Table 3. Example Queries from the CiteSeerX Site

id	query string
citeseerx-q1	ontology
citeseerx-q32	journal for mathematics mobile learning
citeseerx-q261	selective fusion of heterogeneous classifiers
citeseerx-q313	recommender system
citeseerx-q442	on the evolution of random graphs
citeseerx-q534	spectral clustering
citeseerx-q729	hand tracking

```
{
  "docid": "citeseerx-d10556",
  "content": {
    "text": "035$\nMunich Personal RePEc Archive\nAn Online Recruitment..."
  },
  "creation_time": "2016-10-19T12:18:46.400+0200",
  "site_id": "citeseerx",
  "title": "An online recruitment system for economic experiments"
}
```

Listing 1. Example CiteSeerX Document.

split of training and test queries. For the third round, close to 800 additional test queries were added. Table 3 shows a selection of queries as examples. The document lists are generated by a production ranking system and have an average of 55 documents per query, where the largest list contains 100 documents.

4.2 SSOAR

The Social Science Open Access Repository (SSOAR)[2] contains about 38K full text documents from the social sciences and neighboring fields. Documents have a rich set of metadata fields, including title, abstract, authors, subject tags, publication type, year, language, and publisher. Unlike for CiteSeerX, the full document text is not available. An example document entry is shown in Listing 2. For Rounds 1 and 2 in 2016, frequent queries were selected based on access logs, and were complemented with labels of browsing categories. For Round 3 that year, another approx. thousand test queries were added, which are from the tail. As for 2017, roughly 1,200 of the most frequent queries were used as a query dataset. The first 500 were used as test queries and the rest as training queries. Unfortunately, we later found out that this query dataset was sorted by frequency, and the aforementioned split led to a test set that contains only head queries while the train set

[2] http://www.ssoar.info/en/home/about-ssoar.html.

Table 4. Example Queries from the SSOAR Site

id	query string
ssoar-q43	migration
ssoar-q115	bilateral relations
ssoar-q289	labor sozialwissenschaft
ssoar-q376	brexit
ssoar-q482	gruppendynamik
ssoar-q699	alkohol
ssoar-q803	migration und gesundheit

```
{
  "docid": "ssoar-d10466",
  "content": {
    "abstract": "Plausibilit\u00e4t spielt in allen Wissenschaftskulturen eine gewichtige Rolle...",
    "author": "Reszke, Paul",
    "available": "2015-12-14T11:20:34Z",
    "description": "Published Version",
    "identifier": "urn:nbn:de:0168-ssoar-455901",
    "issued": "2015",
    "language": "de",
    "publisher": "DEU",
    "subject": "10200",
    "type": "collection article"
  },
  "creation_time": "2017-06-15T17:04:07.403+0200",
  "site_id": "ssoar",
  "title": "Linguistic-philosophical investigations of plausibility: patterns of communication in the..."
}
```

Listing 2. Example SSOAR Document.

contains only tail queries. Table 4 lists some example queries. The document lists are generated by a production ranking system and have an average of 63 documents per query, where the largest list contains 100 documents.

5 EXPERIMENTAL RESULTS AND ANALYSIS

Using the setup described in Section 3, we organized TREC OpenSearch in collaboration with SSOAR and CiteSeerX as sites in 2016, comprising three evaluation rounds. In 2017, we only had SSOAR available as a site, and, due to time limitations, only two evaluation rounds were organized. An overview of the evaluation periods is given in Table 2. Below, we present the results we obtained, followed by an analysis of the data. Throughout this section, we focus only on test queries.

5.1 Impressions and Clicks

Table 5 presents the number of impressions and clicks, as well as the click through rate (CTR, the fraction of clicks over impressions), for each site and round. We observe that for the academic search task, clicks are extremely sparse. For the thousands of impressions that we received, only several dozen resulted in a click. A more extensive analysis of the traffic data shows that some queries are requested with extreme regularity, indicating some kind of crawler or bot. This is illustrated in Figure 2. Notice that query `ssoar-q1` is issued on an exact 5-minute interval throughout the entire day, while the other queries follow a more natural access pattern. Filtering out `ssoar-q1` is problematic, because regular users also issue that query frequently. This makes it difficult to determine the actual number of human impressions for the sites.

Recall that in 2016, the number of test queries was increased for Round 3, over 8-fold for CiteSeerX (from 107 to 871) and over 14-fold for SSOAR (from 74 to 1062). We find that the impressions

Table 5. Impressions, Clicks, and Click Through Rate (CTR) for Each Evaluation Round and Site

Evaluation round		Site	Impressions	Clicks	CTR
2016	Round 1	SSOAR	4721	25	0.0053
		CiteSeerX	359	144	0.4011
	Round 2	SSOAR	8131	14	0.0017
		CiteSeerX	571	128	0.2242
	Round 3	SSOAR	20062	210	0.0105
		CiteSeerX	4829	651	0.1802
2017	Round 1	SSOAR	10511	105	0.0100
	Round 2	SSOAR	10744	82	0.0076

Only test queries are included.

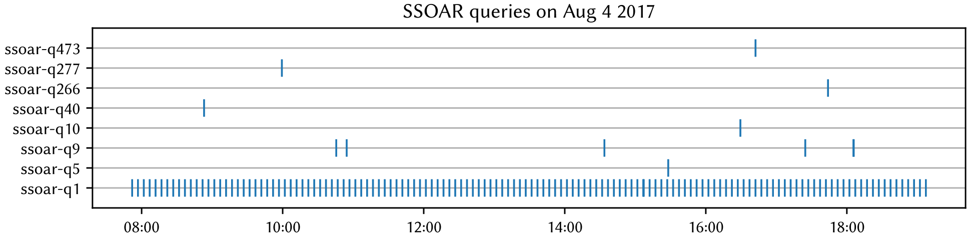

Fig. 2. SSOAR query frequency over time on August 4, 2017. Each bar indicates when a query is issued. Notice that query ssoar-q1 is issued every 5 minutes, while the others are more naturally distributed over the day.

and clicks also increased substantially, even though not to the same extent. This is explained by the fact that those additional queries are increasingly more “tail-ish.”

We also track the number of impressions over time in Figure 3. An interesting observation is that for both CiteSeerX and SSOAR traffic goes up after September 1. This may be due to the beginning of the semester in the northern hemisphere, when students are more likely to look for research material on these academic sites.

Next, in Figure 4, we look at how impressions are distributed across queries. We find that impressions follow a power law distribution, i.e., a few queries are issued a large number of times, while most queries are submitted only a handful of times. Given this, it is not surprising that many queries do not receive any clicks at all. Figure 5 displays the distribution of clicks across queries. For 2016 Round 3, out of the 871 CiteSeerX queries, only 309 got clicked. For SSOAR, this number is 65 out of 1,062 queries in 2016 (Round 3) and 52 out of 501 queries in 2017. Nevertheless, clicks seem to tail off less rapidly than impressions.

We also observe the phenomenon of rank bias in our data, see Figure 6. We see that the overall CTR is dependent on the position in the ranked list. Items near the top of the ranked list have a higher CTR. Furthermore, we see that for CiteSeerX, clicks stop after rank 10 and browsing to the second page is not observed in our data at all. For SSOAR, we observe clicks as far as the 80th rank, indicating users are more willing to “browse” deeply into the ranked list to satisfy their information needs. This indicates that the search patterns and behavior of users on the two sites is very different, possibly due to the fact that CiteSeerX is a monolingual computer science repository whereas SSOAR is a (multi-lingual) social science repository. This helps explain the difference in overall CTR between the two sites.

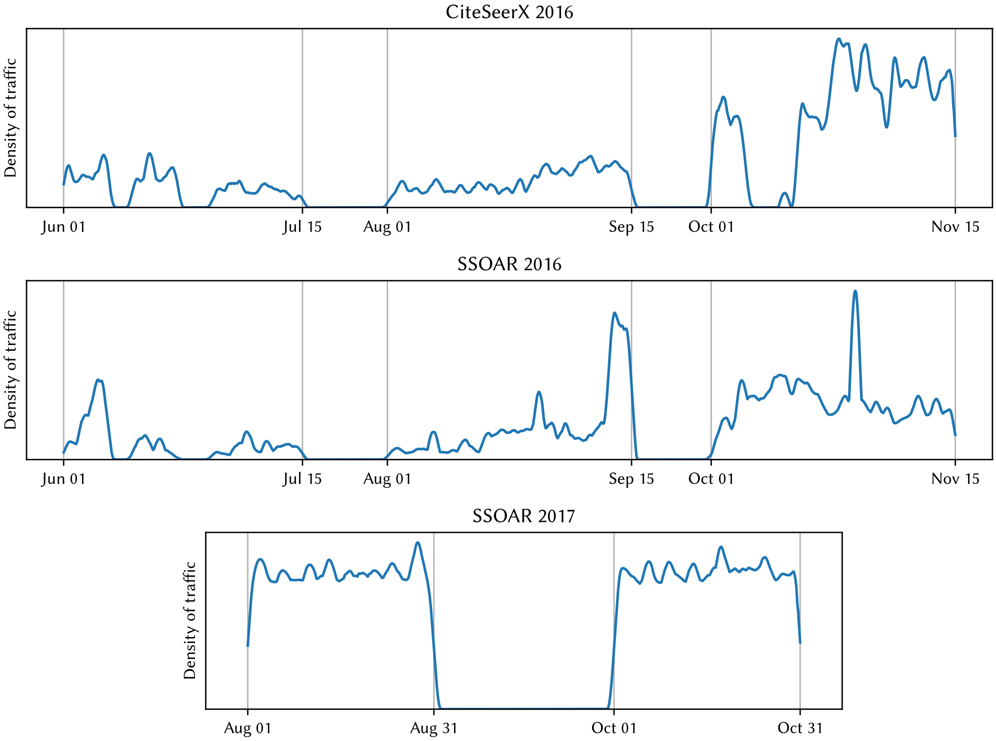

Fig. 3. Traffic on the different websites for each of the different rounds, plotted via Kernel Density Estimation. We observe an increase in traffic after September 1.

5.2 Participating Systems

Next, we turn to a comparison of the participating systems. Table 6 presents the results for CiteSeerX. For each participating system, represented by a column, we show the number of times it won, tied, or lost against the production system (i.e., the site's default ranking algorithm). A tie can occur if the experimental and production rankers had identical rankings up to the position where the click happened (cf. Appendix B). Notice that for Round 3, three participating teams, BJUT, webis, and UDEL-IRL, have numbers that are a magnitude higher than that of other teams. This is because only these teams generated ranked lists for the new test queries that have been added for Round 3. Their click counts are now in a range where we might be able to measure statistical significance. Thus, we focus on these three systems for a further analysis.

Table 7 shows the results of the three participating systems for 2016 Round 3. *Outcome* is the official evaluation measure used at TREC OpenSearch and is defined as

$$\text{Outcome} = \frac{\#\text{Wins}}{\#\text{Wins} + \#\text{Losses}}.$$

That is, an outcome greater than 0.5 means that the experimental system is outperforming the site's production system. To perform significance testing, we use the sign test. Our null-hypothesis is that there is no preference, i.e., each system has a 50% chance to win. Table 7 reports the p-values. The smallest p-value we observe is 0.3912, which is not statistically significant. Assuming the ratio of wins to losses remains the same, we would need to gather roughly 7.6 times more clicks to achieve a two-tailed p-value < 0.01 and about 4.7 times more clicks to achieve a p-value < 0.05. This is equivalent to gathering data for about a year (for $p < 0.01$) or 6 months (for $p < 0.05$).

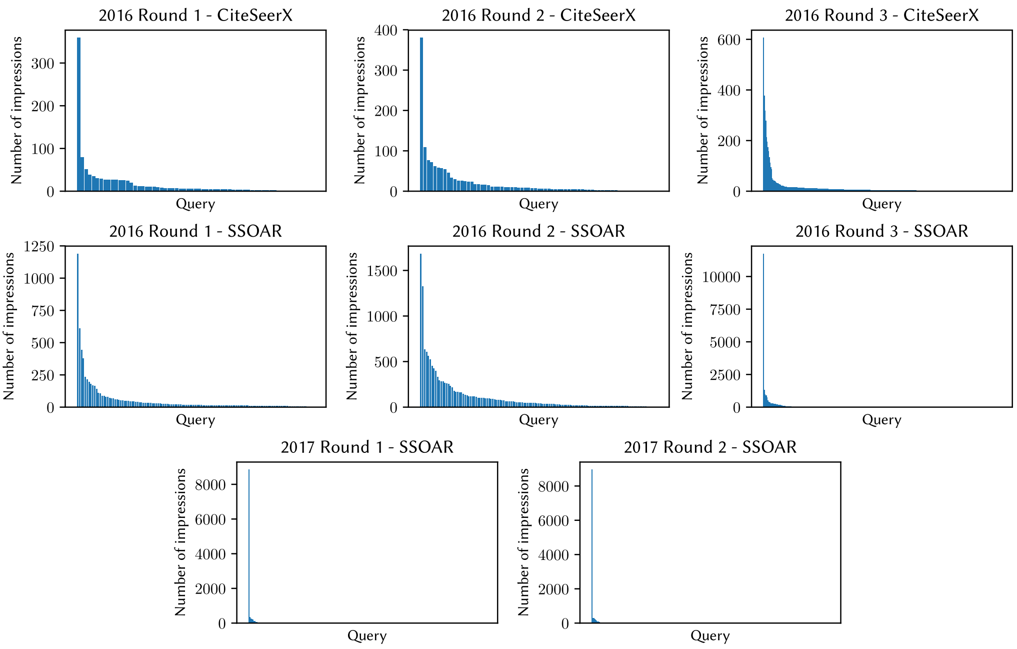

Fig. 4. The distribution of impressions follows a power law distribution with an extremely thin tail.

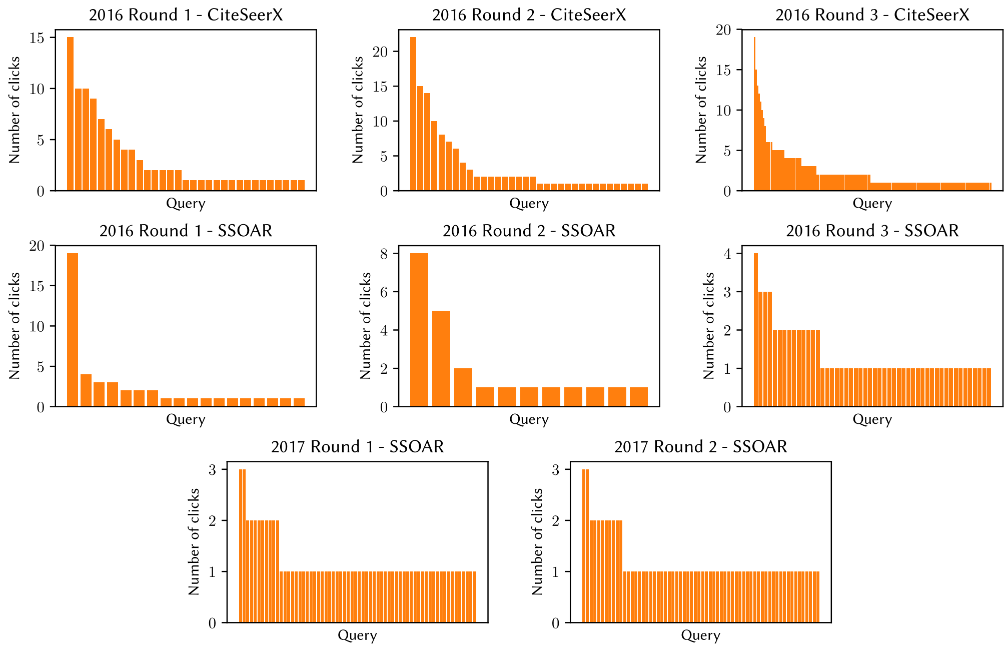

Fig. 5. The distribution of clicks follows a power law distribution, and are more heavily tailed than the distribution of impressions.

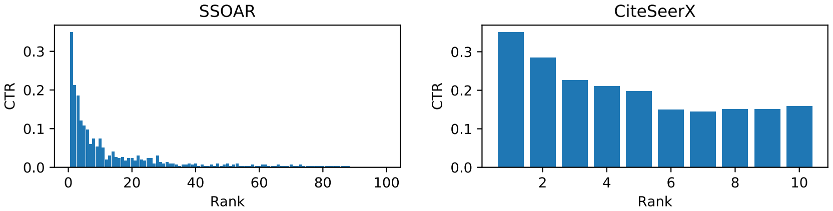

Fig. 6. The CTR per rank, indicating position bias. Note that users of SSOAR go very deep into the result list (as far as the ninth page), whereas users of CiteSeerX stop at rank 10.

Table 6. Outcome of TREC OpenSearch for the CiteSeerX Site

Participant		Gesis	IAPLab	BJUT	OpnSearch_404	QU	KarMat	UWM	webis	UDel-IRL	DaiictIr2
2016 Round 1 June 1–July 15	Wins	4	9	3	0	3	3				
	Ties	3	1	1	1	3	2				
	Losses	2	3	6	0	3	2				
2016 Round 2 Aug. 1–Sep. 15	Wins	2	3	6	4	3	4	2	3	6	
	Ties	1	1	1	1	1	0	3	1	2	
	Losses	3	2	4	4	3	5	1	1	1	
2016 Round 3 Oct. 1–Nov. 15	Wins	5	5	48	5	2	4	2	27	35	6
	Ties	0	2	15	2	2	0	0	11	14	5
	Losses	2	3	39	2	6	2	1	22	32	10

Empty cells denote non-participation.

Table 7. Evaluation Results for CiteSeerX, for 2016 Round 3

Participant	**Wins**	**Ties**	**Losses**	**Outcome**	**p-value**
BJUT	48	15	39	0.5517	0.3912
webis	27	11	22	0.5510	0.5682
UDel-IRL	35	14	32	0.5224	0.8072

Table 8 presents the results for SSOAR. For Round 3 of 2016, we can observe a similar effect that we have seen for CiteSeerX; namely, that systems that submitted ranked lists for the newly added test queries (Gesis, webis, and UDel-IRL) received more clicks than the other systems. However, the overall click counts are still below 30. This is not enough data to draw conclusions about the performance of the systems with statistical significance. Similar to our findings with CiteSeerX, we would need roughly 13 times more clicks to achieve a two-tailed p-value < 0.01 and 7.8 times more data for $p < 0.05$. To collect this amount of data, we would have to run the rounds for a little over a year (for $p < 0.01$) or about 8 months (for $p < 0.05$).

Finally, we look at position bias. Figure 6 tells us that position bias does exist for both SSOAR and CiteSeerX: clicks are more likely to occur at higher ranked documents. We compute the Spearman correlation coefficient between the outcome (1 if the participant wins, -1 if it loses, and 0 if it is a tie) and the highest position a document from the participant's ranking was placed. The hypothesis

Table 8. Outcome of TREC OpenSearch for the SSOAR Site

Participant		Gesis	UWM	QU	KarMat	webis	UDel-IRL	IAPLab	ICTNET	Webis	FEUP
2016 Round 1 June 1–July 15	Wins	1	3	1	4						
	Ties	0	1	1	0						
	Losses	1	2	2	1						
2016 Round 2 Aug. 1–Sep. 15	Wins	1	1	1	0	1	0				
	Ties	0	1	0	0	0	0				
	Losses	0	0	1	2	1	1				
2016 Round 3 Oct. 1–Nov. 15	Wins	13	0			2	2	1			
	Ties	2	1			3	7	1			
	Losses	8	1			5	19	0			
2017 Round 1 Aug. 1–Aug. 31	Wins	9							6	6	
	Ties	2							4	3	
	Losses	6							9	7	
2017 Round 2 Oct. 1–Oct. 31	Wins	5							1	6	8
	Ties	3							3	2	2
	Losses	3							9	4	11

Empty cells denote non-participation.

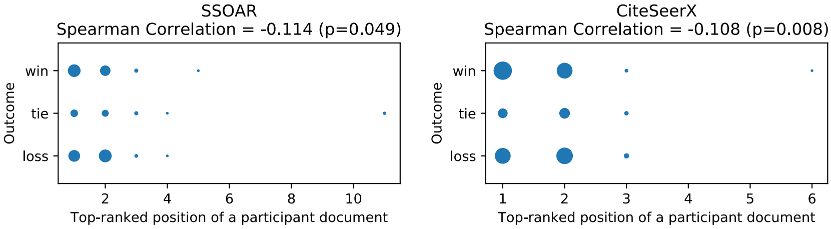

Fig. 7. Correlation between the outcome of an interleaving experiment and the top-ranked position of a document from the participants ranking. A larger circle indicates a higher number of occurrences.

is that a system is more likely to win if its documents happened to be placed at a higher position by the interleaving algorithm. We plotted the correlation in Figure 7 and observe that a weak but statistically significant correlation exists: A participant is more likely to win if one of its documents happens to be placed at a higher position by the interleaving algorithm.

6 CONCLUSIONS AND FUTURE DIRECTIONS

In this article, we have reported on our experiences with TREC OpenSearch. We conclude our work by formulating lessons learned from organizing this evaluation campaign in 2016 and 2017.

Our focus has been on the living labs evaluation methodology, which we have instantiated with academic search as a use-case. The scientific literature search task has been more successful in attracting participants than earlier attempts at CLEF (with product search and web search as use cases, which had four and zero participating systems, respectively, excluding the organizers' baselines [15]). We have also found that participating teams have managed to develop approaches that outperformed the live site's production system. Yet, we have not been able to report statistically significant results, due to the low traffic volume (clicks).

The campaign ran without any significant technical hurdles, yet there is one issue that is worth mentioning. While it sounds obvious that training and test queries should be sampled uniformly from the set of head queries, this is a mistake that is easy to make—and we indeed managed to make it on one occasion, for SSOAR in 2017. In particular, what happened was that the top frequent queries were taken as test queries, while the remaining being train queries. This led to train/test splits with very different characteristics.

One possible solution to overcome the problem of low query volume would be to use more (or all) of the query traffic for experimentation, thereby tapping into the long tail. This, however, would require rethinking the entire API architecture, as ranked lists could no longer be generated offline, but would need to be produced on-the-fly.

Independent of whether this shift to an entirely online setting happens, the main challenge we face is more of an organizational than of a technical nature. The success of a living labs setup depends heavily on a large and active set of participants, and the involvement of large industrial partners as sites is therefore critically important. However, it remains difficult to convince big search engines to allow third parties to influence the ranked lists produced for their queries.

Our participation at the TREC conference sparked several interesting discussions about the future of online evaluation. One idea that resulted from these discussions is to look into tasks where feedback data is already publicly available, instead of working with proprietary data providers. An example is combining the efforts of the TREC Real-time Summarization Track (which uses Twitter as a data source) with TREC OpenSearch. Users' feedback signals such as retweets or likes are publicly available in very large quantities. We leave the exact details of such a task as future work.

Online evaluation is an extremely important part for the future of IR. New technologies are emerging that can no longer be evaluated solely using the offline Cranfield paradigm. For instance, the evaluation of conversational assistants and dialogue systems requires incorporating all sorts of feedback signals from real users [11]. Also, the possibility of being able to run multiple rounds of experiments, without having to wait for the completion of a full annual evaluation cycle, was welcomed by participants. With our work, we have made the first steps towards an open online evaluation platform that can be used by all researchers. We strongly believe that the community needs more initiatives like this, and we hope that our experiences will encourage others to organize similar campaigns.

APPENDIXES

A RESOURCES RESULTING FROM TREC OPENSEARCH

The operation of TREC OpenSearch has resulted in several resources that are of use to anyone wanting to run a living labs style experiment in the future.

We adopted the Living Labs API for TREC OpenSearch and have made numerous contributions to its source code, which is publicly available on https://bitbucket.org/living-labs/ll-api. We have:

(1) added an interleaving API endpoint to reduce engineering overhead for sites,
(2) improved authentication by implementing HTTP Basic authentication scheme,
(3) added the ability to upload multiple runs per participant, and
(4) created a fair load-balancer to distribute traffic more evenly across participants.

All the code is written in Python and uses MongoDB[3] as a database. It is built on the web framework Flask.[4] Its implementation worked well for the purposes of our evaluation efforts, even on a small

[3] https://www.mongodb.com/.
[4] http://flask.pocoo.org/.

virtual machine. During the evaluation campaign, we ran the system on just two CPU cores (2Ghz) with 11GB RAM without any performance problems.

The second resource that we release is the full documentation belonging to the source code. It can be found at http://doc.trec-open-search.org/en/latest/. The documentation provides extensive coverage of the API, including examples on how to use it. Furthermore, there are thorough explanations for prospective participants and sites. Finally, there is a developer section, which explains how to set up a testing living labs environment locally and contribute to the code base.

Finally, our third resource is a curated dataset that can be used for *ad hoc* search. We release the raw queries and structured documents as described in Section 4. For the queries where we observed clicks, we also include the rankings that were shown to users and the documents that were clicked. For each document in these rankings, we include whether the document came from the production system or from a participant system. This makes it possible to reconstruct the baseline rankings from the host sites, so the data can be re-used to run evaluation in a lab environment with annotators. Our click data is extremely sparse, but may still be useful for training click models or for the evaluation of an *ad hoc* retrieval model. The latter is possible by treating clicks as ground-truth relevance. The full dataset (570MB compressed) is available online at https://github.com/living-labs/trec-os-data.

B TEAM-DRAFT INTERLEAVE (TDI)

The version of TDI used by OpenSearch is specified in Algorithm 1. In this variant, the interleaved list **l** is initialized with any common prefix that the ranked lists $\mathbf{l}_1$ and $\mathbf{l}_2$ may have. For this common prefix, no preferences should be inferred. The algorithm continues by flipping coins to decide which ranked list is given priority. Then, it appends the highest ranked result from the selected ranked list, that is not already in **l**, and records the assignment of that item (in **a**) to the ranker where it originates from. This repeats until all results in $\mathbf{l}_1$ and $\mathbf{l}_2$ have been consumed. Finally, the outcome is inferred based on which ranked list was credited with more clicks. We refer the reader to Chapelle et al. [4] for a more in-depth discussion of this algorithm.

ALGORITHM 1: Team Draft Interleaving, following Reference [4].

```
Input: ranked lists $\mathbf{l}_1, \mathbf{l}_2$
$\mathbf{l} = []$; $\mathbf{a} = []$; $i = 0$
while $\mathbf{l}_1[i] == \mathbf{l}_2[i]$ do
    $append(\mathbf{l}, \mathbf{l}_1[i])$
    $append(\mathbf{a}, 0)$
    $i = i + 1$
while $(\exists i : \mathbf{l}_1[i] \notin \mathbf{l}) \vee (\exists i : \mathbf{l}_2[i] \notin \mathbf{l})$ do
    if $count(\mathbf{a}, 1) < count(\mathbf{a}, 2) \vee (rand\_bit() == 1)$ then
        $k = min\,\{i : \mathbf{l}_1[i] \notin \mathbf{l}\}$
        $append(\mathbf{l}, \mathbf{l}_1[k])$
        $append(\mathbf{a}, 1)$
    else
        $k = min\,\{i : \mathbf{l}_2[i] \notin \mathbf{l}\}$
        $append(\mathbf{l}, \mathbf{l}_2[k])$
        $append(\mathbf{a}, 2)$
  // present l to user and observe clicks c, then infer outcome
$c_1 = len\,\{i : c[i] = true \wedge \mathbf{a}[i] == 1\}$
$c_2 = len\,\{i : c[i] = true \wedge \mathbf{a}[i] == 2\}$
return $-1$ if $c_1 > c_2$ else $1$ if $c_1 < c_2$ else $0$
```

ACKNOWLEDGMENTS

We want to thank Peter Dekker and Anne Schuth for their help in developing the methodology and setting up the infrastructure for TREC OpenSearch. We are thankful to CiteSeerX and SSOAR for sharing their users and for facilitating our evaluation campaign. We also want to thank all the participating teams for their participation.

REFERENCES

[1] Leif Azzopardi and Krisztian Balog. 2011. Towards a living lab for information retrieval research and development. A proposal for a living lab for product search tasks. In *CLEF 2011: Conference on Multilingual and Multimodal Information Access Evaluation*. Springer, Amsterdam, 26–37.

[2] Krisztian Balog, Liadh Kelly, and Anne Schuth. 2014. 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)*. ACM, New York, NY, 1815–1818. DOI: https://doi.org/10.1145/2661829.2661962

[3] Krisztian Balog, Anne Schuth, Peter Dekker, Narges Tavakolpoursaleh, Philipp Schaer, Po-Yu Chuang, Jian Wu, and C. Lee Giles. 2016. Overview of the TREC 2016 open search track: Academic search edition. In *Proceedings of the 25th Text REtrieval Conference (TREC'16)*. NIST.

[4] Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. 2012. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems* 30, 1 (2012), 6.

[5] Cyril W. Cleverdon. 1991. The significance of the Cranfield tests on index languages. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 3–12.

[6] Katja Hofmann, Lihong Li, and Filip Radlinski. 2016. Online evaluation for information retrieval. *Foundations and Trends in Information Retrieval* 10, 1 (2016), 1–117.

[7] Frank Hopfgartner, Torben Brodt, Jonas Seiler, Benjamin Kille, Andreas Lommatzsch, Martha Larson, Roberto Turrin, and András Serény. 2015. Benchmarking news recommendations: The CLEF NewsREEL use case. *SIGIR Forum* 49, 2 (2015), 129–136. DOI: https://doi.org/10.1145/2888422.2888443

[8] Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *KDD*. ACM, 133–142.

[9] Makoto P. Kato, Takehiro Yamamoto, Sumio Fujita, Akiomi Nishida, and Tomohiro Manabe. 2017. NTCIR-13 OpenLiveQ. Retrieved October 8, 2017 from http://www.openliveq.net/.

[10] Diane Kelly. 2009. Methods for evaluating interactive information retrieval systems with users. *Foundations and Trends in Information Retrieval* 3, 1–2 (2009), 1–224.

[11] Julia Kiseleva and Maarten de Rijke. 2017. Evaluating personal assistants on mobile devices. *arXiv:1706.04524*.

[12] Ron Kohavi, Alex Deng, Brian Frasca, Toby Walker, Ya Xu, and Nils Pohlmann. 2013. Online controlled experiments at large scale. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*. ACM, New York, NY, 1168–1176. DOI: https://doi.org/10.1145/2487575.2488217

[13] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. 2009. Controlled experiments on the web: Survey and practical guide. *Data Mining and Knowledge Discovery* 18, 1 (2009), 140–181.

[14] Filip Radlinski, Madhu Kurup, and Thorsten Joachims. 2008. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. ACM, 43–52.

[15] Anne Schuth, Krisztian Balog, and Liadh Kelly. 2015. Overview of the living labs for information retrieval evaluation (LL4IR) CLEF lab 2015. In *Proceedings of the 6th International Conference of the CLEF Association (CLEF'15)*. Lecture Notes in Computer Science, Vol. 9283. Springer Berlin Heidelberg, 484–496.

[16] Anne Schuth, Floor Sietsma, Shimon Whiteson, Damien Lefortier, and Maarten de Rijke. 2014. Multileaved comparisons for fast online evaluation. In *CIKM 2014: 23rd ACM Conference on Information and Knowledge Management*. ACM.

[17] Ryen W. White, Ian Ruthven, Joemon M. Jose, and Cornelis J. Van Rijsbergen. 2005. Evaluating implicit feedback models using searcher simulations. *ACM Transactions on Information Systems* 23, 3 (2005), 325–361.

[18] Jian Wu, Kyle Williams, Hung-Hsuan Chen, Madian Khabsa, Cornelia Caragea, Alexander Ororbia, Douglas Jordan, and C. Lee Giles. 2014. CiteSeerX: AI in a digital library search engine. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence, Innovative Applications of Artificial Intelligence*. AAAI Press, 2930–2937.

Received October 2017; revised April 2018; accepted July 2018