Update Delivery Mechanisms for Prospective Information Needs: A Reproducibility Study

Royal Sequiera, Luchen Tan, Yinan Zhang, and Jimmy Lin
David R. Cheriton School of Computer Science
University of Waterloo

ABSTRACT

Real-time summarization systems monitor continuous streams of documents with the goal of delivering relevant, novel, and timely updates to users. These updates can either be sent to users’ mobile devices as push notifications or be silently deposited in an inbox to be consumed—the important difference is whether the user is interrupted by the delivery. Previously, a two-year study examining user attention under these different mechanisms revealed interesting findings about users’ information consumption behavior, but the conclusions were marred by a few methodological shortcomings. We present a reproducibility study that follows the same design as the original evaluation, but corrects its flaws. We find that most conclusions from the original study are confirmed, although there are some surprising differences as well. Overall, the magnitude of the observed effects are not as strong as in the original study.

ACM Reference Format:
Royal Sequiera, Luchen Tan, Yinan Zhang, and Jimmy Lin. 2020. Update Delivery Mechanisms for Prospective Information Needs: A Reproducibility Study. In *2020 Conference on Human Information Interaction and Retrieval (CHIIR’20), March 14–18, 2020, Vancouver, BC, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3343413.3377988

1 INTRODUCTION

Our work focuses on a class of real-time summarization systems that monitors continuous streams of documents with respect to users’ prospective information needs and delivers *updates* that are relevant in real time. Although similar in spirit to the “standard” document filtering task [1, 8], there are a few key differences: “summarization” means that the user does not want *all* topically-relevant documents, only *novel* ones. Furthermore, the system is given flexibility in *when* to deliver an update: for example, it can deliver a document created an hour ago, using the elapsed time to accumulate evidence about its value. Thus, relevance in this context is operationalized as on-topic, non-redundant, and timely.

Previous work has explored two mechanisms for delivering updates to users: In the so-called “push” approach, an update is delivered to a user’s mobile device as a push notification, designed to attract the user’s attention with an alert [14]. In the so-called “pull” approach, an update is deposited into an inbox without interrupting the user; the setup is very much like email, where the updates are examined on the user’s own initiative. Lin et al. [10] compared these two approaches with data drawn from a two-year study in the context of the TREC Real-Time Summarization (RTS) Tracks [11, 12]. The study involved over 50 users who evaluated live system updates on their mobile devices *in situ*, i.e., as they were going about their daily lives, using a mobile app that implemented either the push- or pull-based delivery mechanism described above. In their paper, Lin et al. noted a number of interesting findings about user attention and information consumption behavior, providing concrete guidance to system designers. However, the study was marred by a few methodological shortcomings (see Section 2), which raises questions about the veracity of their conclusions.

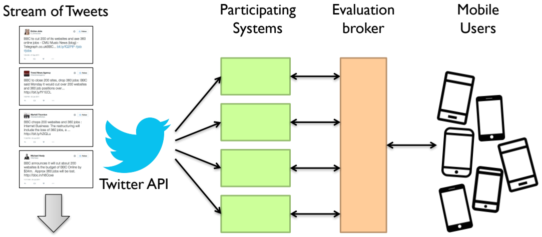

Figure 1: The RTS evaluation setup: systems “listen” to the live Twitter sample stream and send results to the evaluation broker, which then delivers results to users, either via push notifications or silent deposit into users’ inboxes.

This paper describes a reproducibility study following the same basic design as Lin et al. [10], but correcting for the methodological shortcomings. Overall, our results largely confirm the findings of the original study, although we note some surprising differences as well. Although our evaluation provides evidence supporting the veracity of the original conclusions, we also observed that the magnitude of the effects are not as strong as in the original study.

2 EVALUATION METHODOLOGY

The context of experiments by Lin et al. was the Real-Time Summarization (RTS) Tracks at TREC 2016 [12] and TREC 2017 [11] (RTS16 and RTS17, for short), whose setup is shown in Figure 1. Twitter was used as the source of the live document stream, which participating systems “listened” to during a live evaluation period, sending their updates (i.e., tweets identified as relevant) to an evaluation broker. After deduplicating, the evaluation broker then delivered the updates to a cohort of users who subscribed to interest profiles (i.e., topics), received the updates, and provided relevance judgments *in situ* on their mobile devices, i.e., they were going about their daily business and were free to ignore or engage with the updates as

CHIIR ’20, March 14–18, 2020, Vancouver, BC, Canada

ACM ISBN 978-1-4503-6892-6/20/03…$15.00
https://doi.org/10.1145/3343413.3377988

they wished. This "living labs" setup [7, 13, 15, 17] attempts to faithfully mimic the real-world deployment of real-time summarization systems. These evaluations were framed as user studies (with appropriate ethics approval), where university students were recruited as paid human subjects to assess the delivered notifications.

Ideally, in order to examine the impact of the delivery mechanism on information consumption behavior, the delivery mechanism should be the only interface manipulation, with all other variables controlled for. However, due to the realities of organizing large-scale evaluations at TREC, this ideal was not achieved. In 2016, updates were delivered to the users' mobile devices via a custom app, where each update was accompanied by a push notification [12]. In 2017, updates were delivered to the users' mobile devices via a completely redesigned mobile web app, but each update was silently deposited into users' inboxes and *not* accompanied by alerts [11]. Thus, Lin et al. [10] analyzed push vs. pull differences across two evaluations, where delivery differences were conflated with interface changes. There were other methodological flaws as well:

- *Cohort size.* In total, 13 users participated in the 2016 evaluation (push condition) and 42 in 2017 (pull condition). As the 2016 evaluation employed the "living labs" approach for the first time, the organizers had little experience executing such studies.
- *Participating teams.* Different teams participated in the two evaluations, although there was some overlap. While the study focused on user behavior and not system effectiveness, effects of the latter could not be ruled out as contributing to the observed differences.
- *Interest profiles.* Since the results came from separate TREC evaluations, they used different information needs.

To be fair, Lin et al. discussed all of these shortcomings in their paper and couched their findings with appropriate caveats. Nevertheless, questions remain about the veracity of their conclusions.

For the RTS Track in TREC 2018 [16], the organizers followed the same basic study design as in RTS16 and RTS17, but attempted to correct the methodological flaws. In this respect, these efforts can be viewed as a reproducibility study. The three issues in the bulleted list above were addressed by virtue of studying both mechanisms in a single TREC evaluation. The only remaining issue was to isolate the push and pull mechanisms as the *only* experimental manipulation. This required rebuilding the update delivery infrastructure: After careful initial study, we decided to use the Telegram messaging platform. Specifically, mobile assessors installed the Telegram messaging app on their mobile devices and updates were delivered via a Telegram bot we built called the `RTS_bot`. The assessors were asked to subscribe to the bot. Updates from participating systems, along with the associated interest profile (i.e., the statement of information need), were delivered as messages in the Telegram app from our bot. Assessors provided judgments by clicking on buttons indicating that the tweet was relevant, not relevant, or redundant, which triggered corresponding API invocations that recorded the decision in a backend database.

The mobile assessors recruited for RTS18 were randomly divided into the push and pull conditions. Prior to the evaluation period, as part of the assessor onboarding process, we ensured that all participants in the pull condition switched their push notifications off in the app settings. That is, the delivery of a system update did not trigger a notification on their mobile devices. Instead, the assessors had to visit the messaging app on their own initiative to examine the delivered updates. For users in the push condition, we verified that push notifications did indeed accompany update delivery. In this way, we carefully manipulated the mobile interface such that the presence or absence of push notifications was the only difference, corresponding to our variable of interest.

3 RESULTS AND DISCUSSION

The most interesting findings from RTS16 and RTS17 relate to the information consumption behavior of users under push- and pull-based delivery. Specifically, Lin et al. [10] examined in detail *response delay* (or simply *delay*), defined as the difference in time from when an update was delivered to when the user consumed it (as measured by when a judgment was registered in our mobile app). With respect to the delivery mechanism, this intuitively seems like the dependent variable of greatest interest. It is important to note that participant compensation in our study was not tied to how quickly users provided judgments, and therefore these delays are not the result of experimental manipulation.

In Figure 2, we visualize the probability distribution of these delays as heatmaps. Each column represents a specific user, denoted with an anonymized id. For simplicity, we bucket delays into one hour blocks, i.e., the fraction of updates that the user attended to (i.e., judged) during that hour. Each row represents a particular hour: the first 12 represent individual hours, while the last row captures the remaining probability mass (longer than 12 hours). In the heatmap, color is used to encode the amount of probability mass—darker reds indicate more mass. In the top of the figure, we show results from RTS16 (the push condition) and RTS17 (the pull condition), copied from Lin et al. [10]. In the bottom of the figure, we show the corresponding conditions from RTS18: the push condition on the left and the pull condition on the right. Within each condition, the columns (users) are sorted in ascending order of the amount of probability mass in the first row.

The heatmaps show, as expected, that the delivery mechanism has a big impact on information consumption behavior. Looking at the band of dark red in the first row in the RTS16 (Push) heatmap, we see that many users consume updates within an hour of delivery, which is of course not surprising. Compare this to RTS17 (Pull), where far fewer users consume far fewer updates within the first hour. Looking at the RTS18 analysis, we see that the results are largely consistent—more users consume more updates within the first hour under the push condition than the pull condition. However, the differences do not appear as dramatic as in RTS16/17 (the shades of red in the push condition are not as dark as in RTS16).

Lin et al. [10] attempted to generalize patterns of user behavior by taking the probability distributions encoded in the heatmaps, treating them as 13-dimensional vectors, and applying hierarchical clustering using cosine similarity as the distance metric (with average link merging). The results are shown on the left in Figure 3, taken from their paper. Each column in the heatmap is now a row in the cluster visualization. We repeated exactly the same procedure, with results shown on the right in Figure 3. From this analysis, Lin et al. identified three different types of users:

- *Early-heavy distributions.* For these users (coded orange), the probability mass is concentrated in the early hours, with short

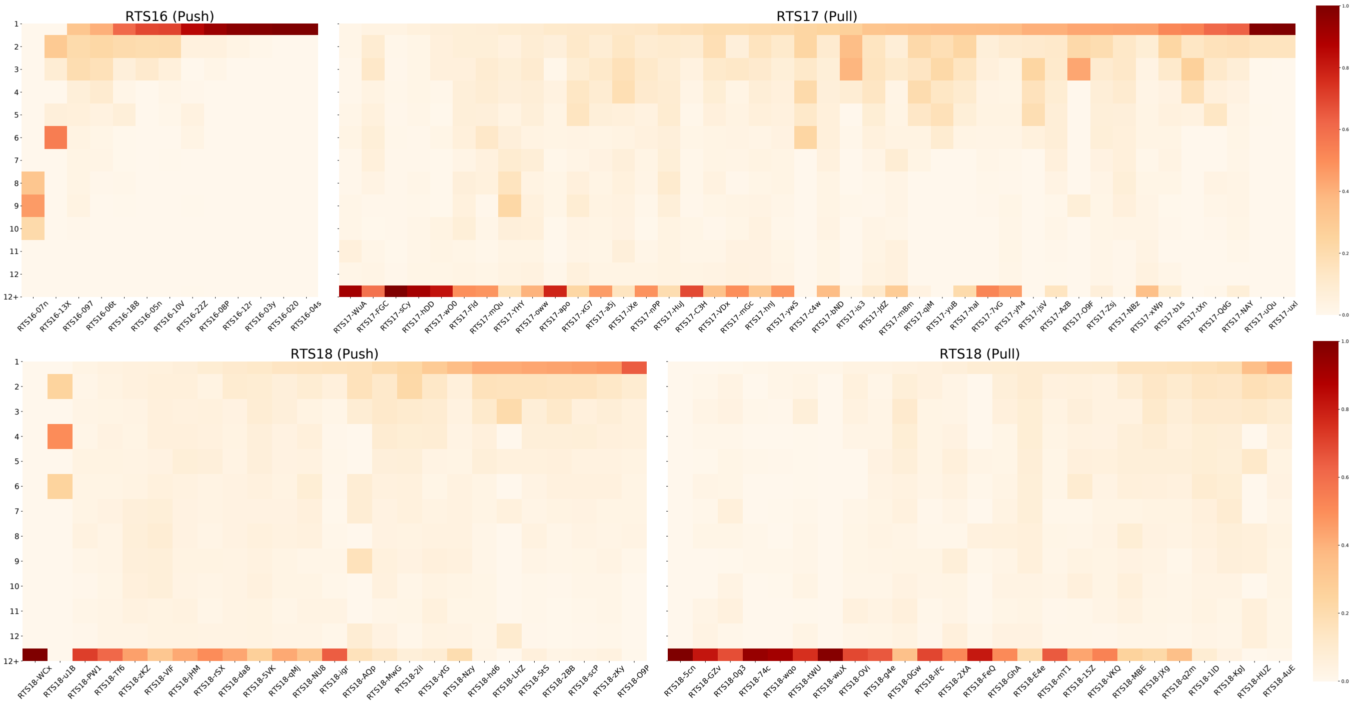

Figure 2: Heatmaps visualizing response delays: for each user (column), each cell shows the fraction of updates consumed in the n-th hour (row) after the system update was delivered; the last row represents delays more than 12 hours. RTS16 (Push) and RTS17 (Pull) shown on the top, from Lin et al. [10]; RTS18 (Push and Pull) shown on the bottom.

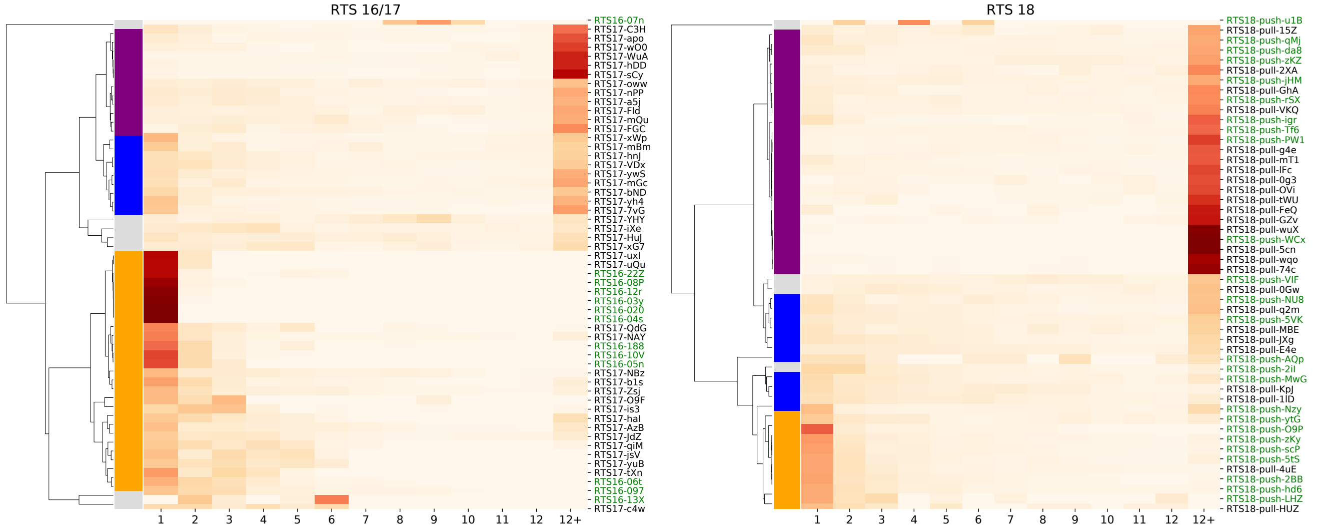

Figure 3: Hierarchical clustering of user delays, showing early-heavy (orange), late-heavy (purple), and bimodal (blue) distributions. RTS16/RTS17 shown on the left, from Lin et al. [10]; RTS18 shown on the right. Push users are labeled in green.

response delays. For the most part, these users examine the updates within a short time of delivery—which means that they were active throughout the day engaging with our app.

- *Late-heavy distributions.* For these users (coded purple), the probability mass is concentrated in the 12+ bucket. These users typically consume large batches of updates over relatively few sessions, usually with large temporal gaps between sessions.
- *Bimodal distributions.* Interestingly, a third cluster of users (coded blue) exhibits a bimodal distribution. That is, there is a non-negligible amount of probability mass in the first and last buckets—which means that they consume some of the tweets with short response delays, but also engage in long sessions with our app that cover many updates.

Although we do see the same three types of users in RTS18 (Figure 3, right), the bimodal distributions do not appear as clearly as in RTS16/17. Once again, the findings from the previous study are largely confirmed, but the effects do not appear as pronounced. Lin et al. [10] make two additional observations:

(1) Some users in the pull condition behave as if they were in the push condition.
(2) No users in the push condition exhibit late-heavy and bimodal delay distributions.

We find that observation (1) holds based on the RTS18 results. That is, at least some users under the pull condition visit our app with such frequency that they consume updates with delays that are similar to the most avid users under the push condition. We imagine that these are akin to users who compulsively check their email very frequently. However, the RTS18 data suggest a refinement: some users in the push condition behave as if they were in the pull condition. That is, they appear to simply ignore the push notifications and (presumably) consume the updates mostly on their own initiative. This can be seen by the existence of users under the push condition that exhibit late-heavy distributions.

In contrast, observation (2) does not appear to hold based on the RTS18 data. The explanation offered by Lin et al. is that push notifications exhaust users' initiative—that is, spending all day responding to push notifications causes fatigue, and thus the users don't have the energy to engage further. In RTS18 data, however, we observe users who *both* attended to updates with short delays (i.e., they frequently checked the mobile app), and then later "caught up" with other updates in longer sessions.

We can think of at least two reasons for this inconsistency: The findings from RTS16 may simply be an artifact of the interface, which was substantially less refined than the RTS18 app. A more difficult-to-use app may have rendered long sessions sufficiently awkward as to dissuade deep engagement with many updates. Another possible explanation might be changing user attitudes to push notifications—over the past few years, interruptions caused by these messages could have become more "normalized" such that users do not think much of the disruptions they cause.

Beyond response delay, *volume* and *response rate* are other dependent variables of interest. Volume refers to the number of updates that a user may have received, which is dependent on the number of profiles that the user subscribed to, the nature of those information needs, as well as system characteristics. The response rate refers to the proportion of updates that a user consumed, as measured in terms of relevance judgments provided. Figure 4 shows the volume and response rates for RTS16/17/18, where the first two are taken directly from Lin et al. [10]. From the RTS16/17 data, we see that response rates are far lower for the push condition than the pull condition: this seems like an intuitive result, since push notifications interrupt the user and are likely to be ignored. However, for the RTS18 users, it is hard to spot a difference in response rates between the two conditions; given that we also observe similar volumes across both conditions, the response rates are directly comparable. This is a surprising and counter-intuitive finding that is inconsistent with the previous results.

There are a number of possible explanations for these differences. The RTS18 users were exposed to a far smaller volume of updates (note the differences in the scales of the y-axes), due to better control over the experimental design that came with experience. For one, it could simply be the case that, in RTS18, the users were less exhausted by the sheer volume of updates. Two additional plausible explanations are already mentioned above: the RTS18 app was more refined and supported more fluid interactions, as well as the normalization of smartphone interruptions over time. Finally, note that our user compensation scheme was tied to the volume of assessments, and thus, combined with an easier-to-use mobile app, might have incentivized users to assess as many updates as possible (regardless of delivery mechanism).

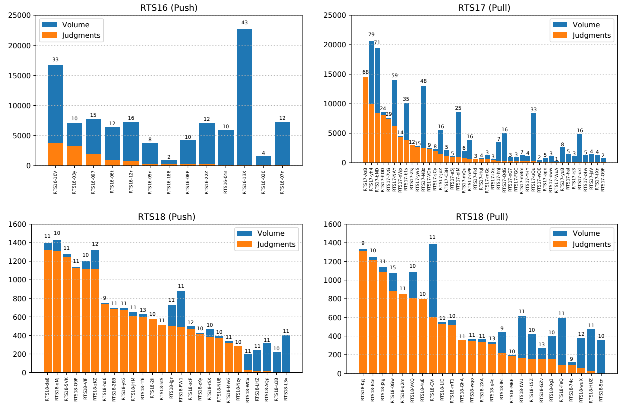

Figure 4: Volume and number of judgments received for all evaluations. Bars are sorted by the number of judgments and annotated with the number of profiles the user subscribed to. The RTS16/17 figures are taken from Lin et al. [10].

Whatever the underlying explanation, it seems that users under both conditions consumed roughly the same fraction of updates, although with shorter delays under push-based delivery. One might easily believe that the constant barrage of push notifications would annoy users, but the RTS18 results suggest greater receptivity to such interruptions compared to RTS16.

4 CONCLUSIONS

There have been many discussions about reproducibility in recent years, not only across many scientific disciplines, but also in information retrieval specifically [2–6, 9]. Much of the discourse has centered around systems-oriented research, as user studies are far more challenging to reproduce, with even fewer incentives for the effort required. Our reproducibility study was only possible with the structure provided by TREC, which allowed organizers to refine their evaluation methodology on a year-over-year basis. However, this paper illustrates the importance of such studies—while the original conclusions of Lin et al. [10] are largely confirmed, we report a few inconsistent results that enrich the literature and suggest future avenues of exploration.

5 ACKNOWLEDGMENTS

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada. We would like to thank all the assessors who participated in our studies as well as additional support by NIST.

REFERENCES

[1] James Allan. 2002. *Topic Detection and Tracking: Event-Based Information Organization.* Kluwer Academic Publishers, Dordrecht, The Netherlands.

[2] Jaime Arguello, Matt Crane, Fernando Diaz, Jimmy Lin, and Andrew Trotman. 2015. Report on the SIGIR 2015 Workshop on Reproducibility, Inexplicability, and Generalizability of Results (RIGOR). *SIGIR Forum* 49, 2 (2015), 107–116.

[3] Ryan Clancy, Nicola Ferro, Claudia Hauff, Jimmy Lin, Tetsuya Sakai, and Ze Zhong Wu. 2019. Overview of the 2019 Open-Source IR Replicability Challenge (OSIRRC 2019). In *Proceedings of the Open-Source IR Replicability Challenge at SIGIR 2019: CEUR Workshop Proceedings Vol-2409.* Paris, France, 1–7.

[4] Nicola Ferro. 2017. Reproducibility Challenges in Information Retrieval Evaluation. *Journal of Data and Information Quality* 8, 2 (2017), Article 8.

[5] Nicola Ferro, Norbert Fuhr, Kalervo Järvelin, Noriko Kando, Matthias Lippold, and Justin Zobel. 2016. Increasing Reproducibility in IR: Findings from the Dagstuhl Seminar on "Reproducibility of Data-Oriented Experiments in e-Science". *SIGIR Forum* 50, 1 (2016), 68–82.

[6] Nicola Ferro and Diane Kelly. 2018. SIGIR Initiative to Implement ACM Artifact Review and Badging. *SIGIR Forum* 52, 1 (2018), 4–10.

[7] Frank Hopfgartner, Allan Hanbury, Henning Müller, Ivan Eggel, Krisztian Balog, Torben Brodt, Gordon V. Cormack, Jimmy Lin, Jayashree Kalpathy-Cramer, Noriko Kando, Makoto P. Kato, Anastasia Krithara, Tim Gollub, Martin Potthast, Evelyne Viegas, and Simon Mercer. 2018. Evaluation-as-a-Service for the Computational Sciences: Overview and Outlook. *Journal of Data and Information Quality* 10, 4 (2018), Article 15.

[8] David D. Lewis. 1995. The TREC-4 Filtering Track. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4).* Gaithersburg, Maryland, 165–180.

[9] Jimmy Lin, Matt Crane, Andrew Trotman, Jamie Callan, Ishan Chattopadhyaya, John Foley, Grant Ingersoll, Craig Macdonald, and Sebastiano Vigna. 2016. Toward Reproducible Baselines: The Open-Source IR Reproducibility Challenge. In *Proceedings of the 38th European Conference on Information Retrieval (ECIR 2016).* Padua, Italy, 408–420.

[10] Jimmy Lin, Salman Mohammed, Royal Sequiera, and Luchen Tan. 2018. Update Delivery Mechanisms for Prospective Information Needs: An Analysis of Attention in Mobile Users. In *Proceedings of the 41st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2018).* Ann Arbor, Michigan, 785–794.

[11] Jimmy Lin, Salman Mohammed, Royal Sequiera, Luchen Tan, Nimesh Ghelani, Mustafa Abualsaud, Richard McCreadie, Dmitrijs Milajevs, and Ellen Voorhees. 2017. Overview of the TREC 2017 Real-Time Summarization Track. In *Proceedings of the Twenty-Sixth Text REtrieval Conference (TREC 2017).* Gaithersburg, Maryland.

[12] Jimmy Lin, Adam Roegiest, Luchen Tan, Richard McCreadie, Ellen Voorhees, and Fernando Diaz. 2016. Overview of the TREC 2016 Real-Time Summarization Track. In *Proceedings of the Twenty-Fifth Text REtrieval Conference (TREC 2016).* Gaithersburg, Maryland.

[13] Abhinav Mehrotra, Robert Hendley, and Mirco Musolesi. 2016. PrefMiner: Mining User's Preferences for Intelligent Mobile Notification Management. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2016).* Heidelberg, Germany, 1223–1234.

[14] Abhinav Mehrotra, Veljko Pejovic, Jo Vermeulen, Robert Hendley, and Mirco Musolesi. 2016. My Phone and Me: Understanding People's Receptivity to Mobile Notifications. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI 2016).* Santa Clara, California, 1021–1032.

[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 CLEF 2015.* Toulouse, France, 484–496.

[16] Royal Sequiera, Luchen Tan, and Jimmy Lin. 2018. Overview of the TREC 2018 Real-Time Summarization Track. In *Proceedings of the Twenty-Seventh Text REtrieval Conference (TREC 2018).* Gaithersburg, Maryland.

[17] Luchen Tan, Gaurav Baruah, and Jimmy Lin. 2017. On the Reusability of "Living Labs" Test Collections: A Case Study of Real-Time Summarization. In *Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017).* Tokyo, Japan, 793–796.