Overview of the TREC 2017 Real-Time Summarization Track

Jimmy Lin,[1] Salman Mohammed,[1] Royal Sequiera,[1] Luchen Tan,[1] Nimesh Ghelani,[1]
Mustafa Abualsaud,[1] Richard McCreadie,[2] Dmitrijs Milajevs,[3] and Ellen Voorhees[3]

[1] David R. Cheriton School of Computer Science, University of Waterloo, Ontario, Canada
[2] School of Computing Science, University of Glasgow, Scotland, the United Kingdom
[3] National Institute for Standards and Technology, Maryland, USA

1 INTRODUCTION

The TREC 2017 Real-Time Summarization (RTS) Track is the second iteration of a community effort to explore techniques, algorithms, and systems that automatically monitor streams of social media posts such as tweets on Twitter to address users' prospective information needs. These needs are articulated as "interest profiles", akin to topics in *ad hoc* retrieval. In real-time summarization, the goal is for a system to deliver interesting and novel content to users in a timely fashion. We refer to these messages generically as "updates". For example, the user might be concerned about tensions on the Korean Peninsula and wishes to be notified whenever there are new developments.

Real-Time Summarization was introduced at TREC 2016 [8] and represented the merger of the Microblog (MB) Track, which ran from 2010 to 2015, and the Temporal Summarization (TS) Track, which ran from 2013 to 2015 [2]. The creation of RTS was designed to leverage synergies between the two tracks in exploring prospective information needs over document streams. The evaluation design is largely based on the real-time filtering task in the TREC 2015 Microblog Track [7].

Following the setup of the track in 2016, we originally considered two methods for disseminating updates, as outlined in the published track guidelines:[1]

- **Scenario A: "Push notifications".** As soon as the system identifies a relevant post, it is immediately sent to the user's mobile device as a push notification. At a high level, push notifications should be relevant (on topic), novel (users should not be delivered multiple notifications that say the same thing), and timely (updates should be provided as soon after the actual event occurrence as possible).
- **Scenario B: Email digests.** Alternatively, a user might wish to receive a daily email digest that summarizes "what happened" on that day with respect to the interest profiles. One might think of these emails as supplying "personalized headlines". These results should be relevant and novel, but timeliness is not particularly important provided that the posts were all written on the day for which the digest was produced.

For expository convenience and to adopt standard information retrieval parlance, we describe tweets that are desirable to the user as *relevant*, even though "relevant" in our context might be more accurately operationalized as a combination of interesting, novel, and timely.

As with the evaluation last year, we recruited a number of mobile assessors who evaluated output from scenario A systems *in situ* on their mobile devices during the evaluation period. Despite our initial intentions, there were last minute technical issues with the implementation of the evaluation infrastructure: we were able to deploy a mobile web-based interface for the assessors, but it *lacked push notification functionality*. In other words, posts were "delivered" to the mobile devices of assessors, but without an accompanying notification signal—this setup is analogous to email inboxes into which relevant content is being continuously deposited, from which the assessors could "pull" new content as they desired. Thus, to be more accurate we refer to scenario A as "mobile delivery" in the remainder of this paper.

Overall, the evaluation design of the RTS Track in TREC 2017 remained unchanged from the 2016 iteration, with the exception of two substantive improvements:

- Participants in scenario A (mobile delivery) were able to obtain the mobile assessors' relevance judgments as they were being generated during the live evaluation period. This allowed systems to experiment, for the first time, with relevance feedback and techniques based on active learning.
- In addition to interest profiles developed by NIST assessors, the mobile assessors this year were also invited to contribute interest profiles of their own. This increased the realism of the task, since the mobile assessors were considering posts retrieved for their own information needs.

2 EVALUATION DESIGN

2.1 General Setup

The overall design of the TREC 2017 Real-Time Summarization Track followed the iteration of the track in TREC 2016 [8], which was itself adapted from the real-time filtering task in the TREC 2015 Microblog Track [7]. Although we are interested in exploring filtering techniques over streams of social media posts in general, we restricted the content under consideration to tweets due to their widespread availability. In particular, Twitter provides a streaming API through which clients can obtain a sample (approximately 1%) of public tweets, colloquially known as the "spritzer". This level of access is available to anyone who signs up for an account.

During the official evaluation period, which began Saturday, July 29, 2017 00:00:00 UTC and lasted until Saturday, August 5, 2017 23:59:59 UTC, participants' systems "listened" to Twitter's live tweet sample stream to identify relevant posts with respect to users' interest profiles, under the mobile delivery (scenario A) or email digests (scenario B) setups. This design required participants to maintain running systems that continuously monitor the tweet sample stream during the evaluation period. The track organizers provided boilerplate code and reference implementations from previous years, but it was the responsibility of each individual team to run its own system(s), connect with the RTS evaluation broker to submit results (more details below), all the while coping

[1] http://trecrts.github.io/TREC2017-RTS-guidelines.html

with crashes, network glitches, power disruptions, etc. A number of recent tracks at TREC have required participants to deploy live systems, which demonstrates that the increased software engineering demands do not present an onerous barrier to entry for participating teams.

An important consequence of the evaluation design is that, unlike in most previous TREC evaluations, no collection or corpus was distributed ahead of time. Since each participant "listened" to tweets from Twitter's streaming API, the collection was generated in real time and delivered to each participant independently. In a previous pilot study [9], we verified that multiple listeners to the public Twitter sample stream receive effectively the same tweets. A more recent study by Sequiera and Lin [15] confirmed the same finding with respect to the Tweets2013 collection gathered for the TREC 2013 Microblog Track [5]. Differences due to, for example, network glitches, do not appear to have a significant impact on evaluation results. Working directly on live data began in the TREC 2015 Microblog Track and continued through last year's iteration of RTS; thus, we consider this design fairly mature. Due to the transient nature of the collection, for archival purposes, both the University of Waterloo and NIST separately collected the live Twitter stream.

Despite superficial similarities, our task is very different from document filtering in the context of earlier TREC Filtering Tracks, which ran from 1995 [4] to 2002 [11], and the general research program known as Topic Detection and Tracking (TDT) [1]. The TREC Filtering Tracks are best understood as binary classification on *every* document in the streaming collection with respect to standing queries, and TDT is similarly concerned with identifying *all* documents related to a particular event—with an intelligence analyst in mind. In contrast, we are focused on identifying a small set of the most relevant updates to deliver to users. Furthermore, in both TREC Filtering and TDT, systems must make online decisions as soon as documents arrive. In our case, for scenario A, systems can choose to deliver older posts (latency is one aspect of the evaluation), thus giving rise to the possibility of algorithms operating on bounded buffers, trading off latency for quality. Finally, previous evaluations, including TDT, TREC Filtering, and Temporal Summarization, merely *simulated* the streaming nature of the document collection, whereas participants in our evaluation were actually required to process tweets posted in real time.

2.2 Interest Profiles

Interest profiles for real-time summarization are difficult to develop because of their prospective nature—this was one of the key lessons learned from previous iterations of the evaluation. For retrospective *ad hoc* topics over a static collection, it is possible for topic developers to explore the document collection to get a sense of the amount of relevant material, range of topical facets, etc. for a particular information need. Typically, topic developers prefer information needs that have neither too many nor too few relevant documents. This is not possible for RTS interest profiles, since they essentially require profile authors to "predict the future".

For this year's evaluation, a total of 188 new interest profiles were created: 148 interest profiles were developed by NIST assessors and 40 additional interest profiles were contributed by the mobile assessors (i.e., assessors who were recruited for the *in situ* evaluation; see Section 2.3 for more details). The latter set of profiles meant that mobile assessors interacted with tweets retrieved for their own information needs.

NIST contracted six assessors to develop interest profiles around topics that were likely to be discussed around the time of the evaluation. During the first week of June 2017, they used a web interface[2] to search a collection of tweets from the public Twitter sample stream, the same source the participants used in the evaluation. The collection consisted of tweets from August 2016, March 2017, and May 2017. Tweets from August 2016 were provided so that the assessors could get a sense of topics that are typically discussed in August. The tweets from March and May 2017 were provided so that the assessors could examine more recent content on Twitter. The time gap allowed the assessors to examine the temporal characteristics of topics being considered.

Using an interactive interface, the assessors were able to explore the collection by:

(1) issuing a query,
(2) clustering the search results, and
(3) hiding or showing media associated with the tweets (images, videos, link previews, etc.).

Once a query was issued to the system, the assessor could select the month from which results were shown. Apart from the top 100 ranked tweets, the total number of retrieved tweets was also displayed, which provided an indication of the size of the underlying topic. The assessors were asked to develop interest profiles that were not too big (less than 1000 tweets in a given month) but also not too small (at least 50 tweets in a given month). Assessors were additionally asked to provide relevance judgments for some of the tweets, to obtain a more reliable indication of the topic size. If the result of a query was too big, assessors could cluster the tweets. Clustering might surface additional search terms that retrieve topically similar tweets, and in this way the assessor could examine subtopics of the initial information need.

Following last year's RTS evaluation and the TREC 2015 Microblog Track before that, we adopted the "standard" TREC *ad hoc* topic format of "title", "description", and "narrative" for the interest profiles. The so-called title consists of a few keywords that provide the gist of the information need, akin to something a user might type into the query box of a search engine. The description is a one-sentence statement of the information need, and the narrative is a paragraph-length chunk of prose that sets the context of the need and expands on what makes a tweet relevant. By necessity, these interest profiles are more generic than the needs expressed in typical retrospective topics because the assessor does not know what future events will occur. Thus, despite superficial similarities in format, we believe that interest profiles are qualitatively different from *ad hoc* topics.

The initial set of NIST-created interest profiles were publicly posted on the track website on July 13, 2017. The mobile assessors were asked to select the profiles they were interested in assessing, along with an option of supplying their own. They were not given specific instructions other than pointers to the NIST-developed interest profiles as reference. After gathering interest profiles from

[2] https://github.com/dimazest/flock

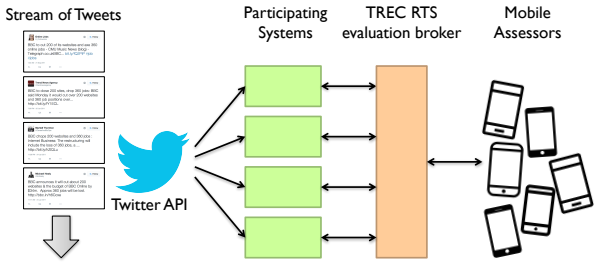

Figure 1: Evaluation setup for scenario A. Systems processed the Twitter sample stream in real time and submitted relevant tweets to the RTS evaluation broker, which immediately delivered the tweets to the mobile devices of assessors who had subscribed to those interest profiles.

the mobile assessors, the organizers checked them for appropriateness and lightly edited the submitted prose for formatting and standardization, but without changing the original intents. For example, by convention, terms from the title should also appear in the description section of the interest profile, and so we modified the profiles accordingly. The final set of interest profiles (including the ones created by the mobile assessors) were publicly posted on July 21, 2017.[3]

2.3 Scenario A: Mobile Delivery

As in the RTS Track in TREC 2016, scenario A implemented a user evaluation whereby system outputs are delivered to the mobile devices of assessors in real time, who examine the tweets *in situ*. This general approach builds on growing interest in so-called "Living Labs" [14] and related Evaluation-as-a-Service (EaaS) [3] approaches that attempt to better align evaluation methodologies with user task models and real-world constraints to increase the fidelity of research experiments.

Our evaluation architecture is shown in Figure 1. All participating systems "listened" to the live Twitter sample stream during the evaluation period, and as the systems identified relevant tweets, they were submitted to the RTS evaluation broker, which immediately recorded each tweet and delivered it to mobile assessors who provided judgments *in situ*—i.e., they were going about their daily lives and could choose to evaluate as many or as few tweets as they wished, whenever they wanted. We have, in effect, built an A/B testing infrastructure for real-time summarization. This evaluation architecture was first described in Roegiest et al. [13] and all code is available on GitHub.[4]

The entire evaluation was framed as a user study (with appropriate ethics approval). A few weeks prior to the beginning of the evaluation period, we recruited assessors from two sources: the undergraduate and graduate student population at the University of Waterloo (via posts on various email lists as well as personal contacts) and RTS participants on the track mailing list. We specifically targeted the track participants so that system developers could gain a better intuition for the types of output that RTS systems produced. All assessors were compensated $1 CAD per 20 judgments.

[3]http://trecrts.github.io/TREC2017-RTS-topics1.json
[4]https://github.com/trecrts/trecrts-eval/

As part of the onboarding process, assessors selected (i.e., "subscribed to") interest profiles (from the NIST-developed set) they were interested in judging. To encourage diversity, we did not allow more than four assessors to select the same profile (on a first come, first served basis). The assignment of interest profiles was interwoven with the solicitation of additional interest profiles from the mobile assessors themselves. However, before the evaluation period began, we arrived at a fixed and static mapping between interest profiles and assessors, which determined which mobile assessor saw which tweets.

From an RTS participant's perspective, prior to the beginning of the evaluation period, each participant's system "registered" with the RTS evaluation broker via a REST API call to request a unique token, which was used in all subsequent interactions with the broker to associate all submitted tweets with that system.[5] Each system was allowed to submit at most ten tweets per interest profile per day. This tweet delivery limit represents a crude attempt to model user fatigue.

During the evaluation period, whenever a system identified a relevant tweet with respect to an interest profile, the system submitted the tweet id to the RTS evaluation broker via a REST API. The broker recorded the submission time, saved the tweet to a database, and immediately delivered the tweet to the LIFO (last in, first out) assessment queues of all mobile assessors who had subscribed to the interest profile. For convenience, we refer to each of these queues as the assessor's "inbox". Note that each tweet was delivered only once, even if it was submitted by multiple systems at different times. This design operationalizes the temporal interleaving strategy proposed by Qian et al. [10]. Note that, critically, unlike last year, the delivery of a tweet was *not* accompanied by a push notification. That is, there was no explicit cue (notification message, chime, vibration, etc.) that a new tweet had been added to the assessor's inbox.

Mobile assessors provided judgments via a webapp that was specifically designed for mobile devices; a screenshot is shown in Figure 2. The assessment interface was derived from software built by the University of Waterloo's team that participated in the TREC 2017 Core Track [18]. Substantial effort was devoted to refining the user experience and making the interface as responsive as possible.

The interface is divided into three sections: the interest profile, the tweet, and the judgment buttons. The top section shows the profile title and description for which a system posted the tweet. The widget in the middle shows the tweet, rendered using Twitter's API, which meant that the tweet appeared exactly as it would on Twitter's own clients (with proper preview of embedded content such as links and videos). The assessor can further interact with the embedded content, e.g., click on a link, watch a video, etc. Finally, there are three buttons at the bottom of the screen for the assessor to render a judgment:

- *relevant*, if the tweet contains relevant and novel information;
- *redundant* (i.e., duplicate), if the tweet contains relevant information, but is substantively similar to another tweet that the assessor had already seen;
- *not relevant*, if the tweet does not contain relevant information.

[5]In this discussion, each participant run is considered a separate system.

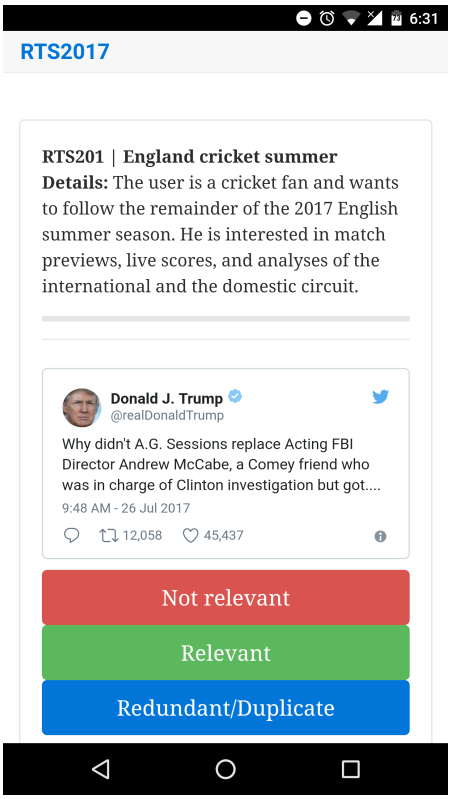

Figure 2: Screenshot of the mobile assessment interface.

Once the assessor taps one of the buttons, the judgment is registered by the server. The page disappears and the next tweet in the queue is displayed. There is no way to modify a judgment once it has been provided.

Previously mentioned but worth emphasizing: the mobile assessors provided judgments *in situ*, i.e., as they were going about their daily lives. In contrast to the push notification setup from last year, this year's design can be characterized as *pull-based*: that is, the assessors, on their own initiative, *pulled* relevant content to examine from their inboxes. We had no control over how frequently they visited the assessment interface, how many tweets they assessed, or any other aspect of assessment behavior.

Finally, to close the loop, the RTS broker provided an API for each participant's system to retrieve relevance judgments for tweets that it had posted (specifically, a system did not have access to relevance judgments for tweets posted by another system). For rate limiting purposes, we asked participants not to call this API more than once per hour, but the constraint was not enforced. This feature was a major addition to the RTS evaluation this year, and the design allowed participants to, for the first time, experiment with relevance feedback and active learning techniques.

2.4 Scenario B: Email Digests

The overall evaluation setup for scenario B is shown in Figure 3. As with scenario A, participants "listened" to the live Twitter sample stream to identify relevant tweets with respect to the interest

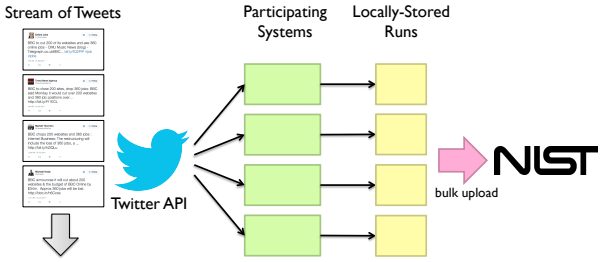

Figure 3: Evaluation setup for scenario B. Systems processed the Twitter sample stream in real time and stored their results locally during the evaluation period. After the evaluation ended, the runs were uploaded to NIST in batch.

profiles. Each system was tasked with identifying up to 100 tweets per day per interest profile, which are putatively delivered to the user daily. For simplicity, all tweets from 00:00:00 to 23:59:59 UTC are valid candidates for that particular day. It was expected that systems would compute the results in a relatively short amount of time after the day ends (e.g., at most a few hours), but this constraint was not enforced. Each system recorded its own results (i.e., ranked lists) for each day, which were then uploaded to NIST servers in batch shortly after the evaluation period ended.

The per-day limit of 100 tweets was arbitrarily set, but at a value that is larger than what one might expect from a daily email digest, primarily to enrich the judgment pool (more details in Section 4). As with scenario A, we neglected to model real-world constraints in favor of simplicity, since defining a "day" in terms of UTC does not take into account the reading habits of users in different time zones around the world.

2.5 Submissions Types

For both scenarios, systems were asked to only consider tweets in English. Each team was allowed to submit up to three runs for scenario A and three runs for scenario B. Runs were categorized into three different types based on the level of human involvement:

- **Automatic Runs:** In this condition, system development (including all training, system tuning, etc.) must conclude prior to downloading the interest profiles from the track homepage (which were made available before the evaluation period). The system must operate without human input before and during the evaluation period. Note that it is acceptable for a system to perform processing on the profiles (for example, query expansion) before the evaluation period, but such processing cannot involve human input.
- **Manual Preparation:** In this condition, the system must operate without human input during the evaluation period, but human involvement is acceptable before the evaluation period begins (i.e., after downloading the interest profile). Examples of manual preparation include human examination of the interest profiles to add query expansion terms or manual relevance assessment on a related collection to train a classifier. However, once the evaluation period begins, no further human involvement is permissible.

- **Manual Intervention:** In this condition, there are no limitations on human involvement before or during the evaluation period. Crowd-sourced judgments, human-in-the-loop search, etc. are all acceptable.

Note that judgments provided by the mobile assessors did not count as manual intervention for determining the run type category. For example, a run that exploited relevance feedback using mobile judgments could still be classified as an automatic run as long as there was no additional human input from the system's developers.

Participants were asked to designate the run type at submission time for the scenario B runs when they uploaded their results to NIST. For scenario A runs, we asked each team about the type of each of their runs over email after the evaluation period.

All types of systems were welcomed; in particular, manual preparation and manual intervention runs are helpful for understanding human performance and for enriching the judgment pool.

2.6 Runs Postprocessing

A detail worth discussing is postprocessing performed by the organizers to create the "official" scenario A runs. Due to the nature of a live evaluation, there might be minor differences between records of posted tweets at the RTS broker and from the perspective of each individual system, for example, due to incomplete API requests. For the purposes of the evaluation, the record of activity at the RTS broker constituted the "ground truth".

The postprocessing had another specific purpose: after the evaluation began, we discovered a clock synchronization issue between the RTS broker and its backend database that allowed clients to submit more tweets than the ten-per-day limit. This bug was fixed and the RTS broker was restarted on July 31, 2017 around 1pm EDT. To better conform to the evaluation guidelines, we truncated runs that submitted more than ten tweets on any day to the first ten tweets submitted on that day.

The postprocessed "official" runs were provided back to the RTS participants, and these runs served as input to the evaluation scripts whose results we report in this paper.

3 IN-SITU EVALUATION METRICS

In this section we describe how judgments from the mobile assessors in scenario A (see Section 2.3) are aggregated into evaluation metrics to quantify the effectiveness of each run. At a high level, the RTS broker interleaved submitted tweets from participating systems, delivered them to the mobile devices of assessors, and gathered a stream of judgments: whether a tweet is *relevant*, *redundant*, or *not relevant* with respect to an interest profile. Because each tweet was delivered to *all* assessors who had subscribed to the profile, the broker might have received more than one judgment per tweet.

Another implication of the interleaved evaluation setup is that an assessor may have encountered tweets from different systems, which makes proper interpretation of *redundant* judgments difficult. A tweet might only appear redundant because the same information was contained in a tweet delivered earlier by another system (and thus it was not the "fault" of the system that submitted the tweet). In other words, the interleaving of outputs from different systems was responsible for introducing the redundancy. Furthermore, since assessors were always examining the most recent tweet first, a more recent tweet might have caused an older tweet to appear redundant. These are, unfortunately, unavoidable consequences of "messy" user evaluations, and systems must be designed with the ability to interpret noisy relevance signals.

To measure the effectiveness of a run, we computed two aggregate metrics based on user judgments:

Online Precision. A simple and intuitive metric is to compute precision, or the fraction of relevant judgments:

$$\frac{\text{relevant}}{\text{relevant} + \text{redundant} + \text{not relevant}} \tag{1}$$

We term this "strict" precision because systems don't get credit for redundant judgments. Also, we can compute "lenient" precision, where systems do receive credit for redundant judgments:

$$\frac{\text{relevant} + \text{redundant}}{\text{relevant} + \text{redundant} + \text{not relevant}} \tag{2}$$

Two additional details are necessary for the proper interpretation of these metrics: First, tweets may be judged multiple times since each tweet was delivered to all users who had subscribed to the profile. For simplicity, all judgments were included in our calculation. Second, our precision computations represent a micro-average (and *not* an average of per-profile precision). This choice was made because different profiles received different numbers of judgments, and thus macro-averaging would magnify the effects of interest profiles with few judgments.

Online Utility. As an alternative to online precision, we can take a utility-based perspective and measure the total gain received by the user. The simplest method would be to compute the following:

$$\text{relevant} - \text{redundant} - \text{not relevant} \tag{3}$$

which we refer to as the "strict" variant of online utility. Paralleling the precision variants above, we define a "lenient" version of the metric as follows:

$$(\text{relevant} + \text{redundant}) - \text{not relevant} \tag{4}$$

Of course, we could further generalize online utility with weights for each type of judgment. However, we lacked the empirical basis for setting the weights and thus did not choose to do so.

To summarize: from user judgments, we computed two aggregate metrics—online precision and online utility. Note that there is no good way to compute a recall-oriented metric since we have no control over when and how frequently user judgments are provided. Finally, following last year's RTS evaluation, we made strict precision the primary metric for assessing scenario A runs using mobile assessors.

4 BATCH EVALUATION METRICS

In this section we describe the batch evaluation methodology and metrics used to evaluate both scenario A and scenario B runs. Note that scenario A runs were assessed using both the mobile assessor judgments (described in the previous section) as well as the batch methodology described here. Scenario B runs were evaluated with the batch methodology only.

At a high level, we adopted the Tweet Timeline Generation (TTG) evaluation methodology that was originally developed for the TREC 2014 Microblog Track [6] and was used both in the TREC 2015 Microblog Track [7] and the RTS Track last year [8]. The methodology is mature, in that it has been externally validated [17] and similar approaches have been deployed in evaluations dating back at least a decade. The assessment workflow proceeded in two stages: relevance assessment and semantic clustering. Both were performed by NIST assessors.

Relevance assessments were performed using pooling with a single pool across both scenario A and scenario B runs. The pools were built using all scenario A tweets (after postprocessing, see Section 2.6) and a maximum of 90 tweets per profile for each scenario B run, the same as last year. To select the final set of interest profiles to assess, we removed from consideration profiles that had fewer than ten relevant judgments (i.e., were too sparse and/or too difficult) or had greater than 60% precision (i.e., were too easy) in the mobile judgments. This still left too many profiles to judge, so NIST eliminated additional profiles by hand, discarding those whose pools were too large and culling profiles that were topically similar. Each pool was judged by the assessor who authored the profile, although some assessors were given other interest profiles to judge as well. NIST staff also contributed some judgments, including profiles for which they were the author. In total, 96 interest profiles were judged. One additional interest profile (RTS107) had only one tweet, marked as missing (see below).

The pools contained 94,307 tweets in total. The maximum number of tweets for an interest profile was 1585, the minimum was one (RTS107), and on average there were 972 tweets per profile.

These pools were then judged by NIST assessors for relevance. To facilitate consistent judgments, tweets were first clustered and presented in order of lexical similarity. Each tweet was independently assessed on a three-way scale of "not relevant", "relevant", and "highly relevant". Non-English tweets were marked as not relevant by fiat. If a tweet contained a mixture of English and non-English content, discretion was left to the assessor. As with previous TREC Microblog evaluations, assessors examined links embedded in tweets, but did not explore any additional external content. Retweets did not receive any special treatment and were assessed just like any other tweet.

The assessment interface rendered tweets using Twitter's official API, which meant that content appeared exactly as it would on the Twitter official site (for example, with previews of embedded content such as links and videos). Because of this, however, tweets submitted by systems but were deleted prior to assessment could not be shown. These tweet are specifically marked as missing in the final qrels.

In rendering judgments, the NIST assessors tried to maintain consistency with judgments made by the mobile assessors. In the assessment interface, next to each tweet, there are indicators of the number of mobile assessors who judged the tweet as relevant and not relevant. Since the tweets were presented in cluster order, the effect of this interface design is that the temporal sequencing of mobile judgments was lost to the assessors—potentially, for example, reflecting evolving notions of relevance regarding an interest profile. Specifically, the assessors were provided the following guidance in writing:

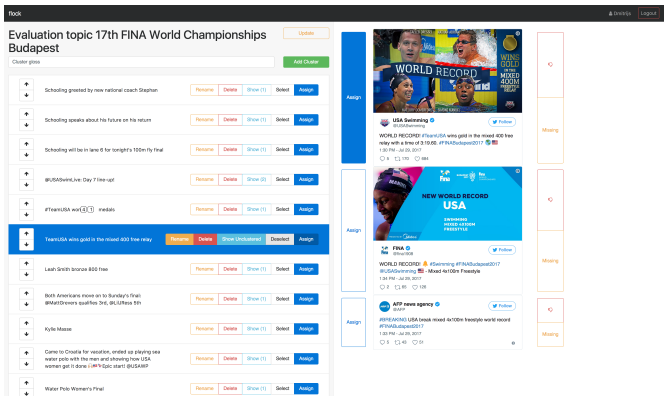

Figure 4: Screenshot of the clustering interface.

> Because the systems might have adapted to the judgments they saw, we want you to judge tweets consistently with the existing judgments, to the extent possible. You are *not* required to always take the majority judgment, and you are not required to agree with a single judge if you are convinced that the [mobile] judgment is just plain wrong (because it might be). But if the tweet represents a gray area that you could legitimately assess in either direction, please opt to go with the [mobile assessor].

Anecdotally, most assessors independently reflected that the mobile assessments were poor in quality.

After the relevance assessment process, the NIST assessors proceeded to perform semantic clustering on the relevant tweets using the Tweet Timeline Generation (TTG) protocol, originally developed for the TREC 2014 Microblog Track [6, 17].

The TTG protocol was designed to reward novelty (or equivalently, to penalize redundancy) in system output. In both scenario A and scenario B, we assume that users would not want to see multiple tweets that "say the same thing", and thus the evaluation methodology should reward systems that eliminate redundant output. Following the TTG protocol, we operationalized redundancy as follows: for every pair of tweets, if the chronologically later tweet contains substantive information that is not present in the earlier tweet, the later tweet is considered novel; otherwise, the later tweet is redundant with respect to the earlier one. In our definition, redundancy and novelty are antonyms, so we use them interchangeably but in opposite contexts.

Due to the temporal constraint, redundancy is *not* symmetric. If tweet A precedes tweet B and tweet B contains substantively similar information found in tweet A, then B is redundant with respect to A, but not the other way around. We also assume transitivity. Suppose A precedes B and B precedes C: if B is redundant with respect to A and C is redundant with respect to B, then by definition C is redundant with respect to A.

For semantic clustering, the assessors were shown all the relevant tweets for an interest profile (from the previous stage) in a custom assessment interface (see Figure 4 for a screenshot). The

tweets were shown in the left pane in the same order as during the relevance assessment process, such that lexically similar tweets were displayed next to each other, and the list of current clusters were shown in a pane on the right side. Tweets were also rendered with Twitter's official API, and so it was possible for assessors to encounter a deleted tweet during the clustering stage. These tweet ids were also marked missing in the final qrels. For each tweet in the left pane, the assessor could either use that tweet as the basis for a new cluster or add it to one of the existing clusters. In this way, clusters representing important pieces of information, comprised of semantically similar tweets, were constructed incrementally. To aid in the clustering process, assessors could enter a short textual description for each cluster. Assessors could also move tweets between clusters and mark a tweet as not relevant, in case they changed their mind. The instructions given to the assessors did not specify a particular target number of clusters to form. Instead, they were asked to use their best judgment, considering both the scope of the interest profiles and the actual tweets.

The final output of the batch assessment process (for each interest profile) is a list of clusters, where tweets in each cluster represent a particular "facet" of information that addresses the user's need.

4.1 Scenario A Metrics

For scenario A, we computed a number of metrics from the relevance judgments and clusters provided by NIST assessors, detailed below. At a high level, mobile delivery of content should be relevant (on topic), novel (users should not be shown multiple tweets that say the same thing), and timely (provide updates as soon after the actual event occurrence as possible). Following the 2016 iteration of the track—instead of devising single-point metrics that attempt to incorporate relevance, novelty, and timeliness, the official metrics separately quantify output quality (relevance and novelty) and latency (timeliness).

We envision that systems might trade off latency with output quality: For example, a system might wait to accumulate evidence before submitting tweets, thus producing high-quality output at the cost of high latency. Alternatively, a low-latency system might aggressively submit results that it might "regret" later. Computing metrics of output quality separately from latency allows us to understand the potential tradeoffs. Additionally, we believe this approach is appropriate because we have no empirical evidence as to what the "human response curve" to latency looks like—that is, how much should we discount a quality metric based on tardiness? Attempting to formulate a single-point metric collapses meaningful distinctions in what users may be looking for in systems.

Expected Gain (EG) for an interest profile on a particular day is defined as follows:

$$\frac{1}{N} \sum \mathrm{G}(t) \tag{5}$$

where N is the number of tweets returned and $\mathrm{G}(t)$ is the gain of each tweet:

- Not relevant tweets receive a gain of 0.
- Relevant tweets receive a gain of 0.5.
- Highly-relevant tweets receive a gain of 1.0.

Once a tweet from a cluster is retrieved, all other tweets from the same cluster automatically become not relevant. This penalizes systems for returning redundant information.

Normalized Cumulative Gain (nCG) for an interest profile on a particular day is defined as follows:

$$\frac{1}{\mathcal{Z}} \sum \mathrm{G}(t) \tag{6}$$

where $\mathcal{Z}$ is the maximum possible gain (given the ten tweet per day limit). The gain of each individual tweet is computed as above. Note that gain is not discounted (as in nDCG) because the notion of document ranks is not meaningful in this context.

The score for a run is the mean of scores for each day over all the interest profiles. Since each profile contains the same number of days, there is no distinction between micro- and macro-averages.

An interesting question, which has only recently been empirically "resolved", is how scores should be computed for days in which there are no relevant tweets: for rhetorical convenience, we call days in which there are no relevant tweets for a particular interest profile (in the pool) "silent days", in contrast to "eventful days" (where there are relevant tweets). Tan et al. [16] examined this issue and proposed two metric variants, which were adopted in 2016 [8]: In the EG-1 and nCG-1 variants of the metrics, on a "silent day", a system receives a score of one (i.e., a perfect score) if it does not submit any tweets to the broker, or zero otherwise. In the EG-0 and nCG-0 variants of the metrics, for a silent day, all systems receive a gain of zero no matter what they do.

Therefore, under EG-1 and nCG-1, systems are rewarded for recognizing that there are no relevant tweets for an interest profile on a particular day and remaining silent (i.e., does not submit any tweets to the broker). The EG-0 and nCG-0 variants of the metrics do not reward recognizing silent days: that is, it never hurts to submit tweets. Recently, Roegiest et al. [12] concluded that EG-0 and nCG-0 are flawed metrics precisely for this reason. These metrics correlate with volume (number of tweets submitted to the broker), and only weakly to sensible metrics of quality.

EG-1 and nCG-1, however, are both binary on a silent day (i.e., either zero or one), which makes optimization difficult because of a discontinuity. As a remedy, this year we introduced EG-p and nCG-p (p for *proportional*), where on a silent day, the score is one minus the fraction of the ten-tweet daily quota that is used. For example, if a system submits zero tweets, it receives a score of 1.0; if it submits one tweet, a score of 0.9; two tweets, 0.8; etc., such that if a system uses up its entire quota of ten tweets for a day, it receives a score of zero. EG-p and nCG-P still reward systems for recognizing silent days, but with a penalty that is proportional to how "quiet" the system is. EG-p was adopted as the primary metric (i.e., the sort key in the results table).

Gain Minus Pain (GMP) is defined as follows:

$$\alpha \cdot \sum G - (1 - \alpha) \cdot P \tag{7}$$

The G (gain) is computed in the same manner as above. Pain P is the number of non-relevant tweets that the system submitted, and α controls the balance between the two. We investigated three settings: 0.33, 0.50, and 0.66. Note that this metric is the same as the linear utility metric used in the TREC Filtering Tracks [4, 11],

albeit with a different mathematical form. Thus, our metric builds squarely on previous work.

Latency. In addition to the quality metrics above, we report, for tweets that contribute to gain, the mean and median difference between the time the tweet was delivered and the first tweet in the semantic cluster that the tweet belongs to (based on the NIST assessors). For example, suppose tweets *A*, *B*, and *C* are in the same semantic cluster, and were authored at 09:00, 10:00, and 11:30, respectively. No matter which of the three tweets was submitted by a system, latency is computed with respect to the creation time of *A* (09:00). Therefore, posting tweet *C* at 11:30 and posting tweet *A* at 11:30 yields the same latency.

4.2 Scenario B Metrics

Scenario B runs were evaluated in terms of nDCG as follows: for each interest profile, the list of tweets returned per day is treated as a ranked list, and from this nDCG@10 is computed. Note that in this scenario, the evaluation metric *does* include gain discounting because email digests can be interpreted as ranked lists of tweets. Gain is computed in the same way as in scenario A with respect to the semantic clusters. Systems only receive credit for the first relevant tweet they retrieve from a cluster.

The score of an interest profile is the mean of the nDCG scores across all days in the evaluation period, and the score of a run is the mean of scores for each profile. Once again, the micro- vs. macro-average distinction is not applicable here. As with scenario A, we computed two variants of the metric: With nDCG-1, on a "silent day", the system receives a score of one (i.e., a perfect score) if it does not submit any tweets, or zero otherwise. With nDCG-p (p for *proportional*), the definition is the same as in scenario A: on a silent day, the score is $1 - \min(n, 10)/10$ where n is the number of tweets submitted for that day.

5 RESULTS

5.1 Scenario A

For scenario A, we received a total of 41 runs from 15 groups. These runs submitted a total of 78,556 tweets, or 50,124 unique tweets after de-duplicating within each interest profile (but not across interest profiles).

For the *in situ* mobile evaluation of scenario A systems, we recruited 42 assessors (six of whom were from participating teams). Over the entire evaluation period, we received 85,525 judgments, with a minimum of 16 and a maximum of 14,441 by an individual assessor. We found that 17,140 tweets received a single judgment, 18,306 tweets received two judgments, 8,671 tweet received three judgments, and 1,440 tweets received four judgments. All 188 interest profiles received at least one judgment; one profile received 81 judgments; 129 received $(100, 500]$ judgments; 50 received $(500, 1000]$ judgments; seven received $(1000, 2000]$ judgments; one received 2074 judgments. On average, the mobile assessors submitted 455 judgments per profile.

The distribution of judgments by assessor is shown in Table 1. The columns list: assessor id, the number of judgments provided, the number of profiles subscribed to, and the number of tweets delivered to that assessor. The final column shows the response rate, computed as the ratio between the second and fourth columns. Note that these statistics include judgments of tweets that may have been subsequently removed in postprocessing, as described in Section 2.6. We see that there were quite a few highly-motivated assessors who judged nearly all the tweets that were delivered to them for the profiles they subscribed to; in one case, a particularly "diligent" assessor provided over 14k judgments.

Results of the *in situ* evaluation by the mobile assessors are shown in Table 2. The first two columns show the participating team and run. The next columns show the number of tweets that were judged relevant (R), redundant (D), and not relevant (N); the number of unjudged tweets (U); the length of each run (L), defined as the total number of messages delivered by the system. The next column shows coverage (C), defined as the fraction of *unique* tweets that were judged. Following that, the columns report the mean ($\bar{t}$) and median ($\tilde{t}$) latency of submitted tweets in seconds, measured with respect to the time the original tweet was posted. The next sets of columns provide metrics of quality: strict and lenient precision, strict and lenient utility. The final column shows the run type: 'A' denotes automatic and 'P' manual preparation; '?' indicates unknown (we did not receive a response from one team, despite repeated inquiries). The rows in the table are sorted by strict precision.

Results of the batch evaluation by NIST assessors are shown in Table 3. The columns list the various metrics discussed in Section 4 and also the mean and median latency in seconds. Note that latency here is computed with respect to the first tweet in each cluster (which is different from how latency is computed with respect to the mobile assessors' judgments), and thus a system may have a high latency even if it submits a tweet immediately after it is posted. The second to last column shows the length of each run, defined as the number of tweets posted for the interest profiles that were assessed. The final column shows the run type: 'A' denotes automatic and 'P' manual preparation; '?' indicates unknown. The rows in the table are sorted by EG-p. For reference, an empty run (i.e., a system that does not submit any tweets) would receive a score of 0.1765 for EG-p/EG-1 and nCG-p/nCG-1 (with all other scores being zero).

We examined the correlations between strict precision (mobile metric) and expected gain variants (batch metric) in Figure 5, which shows scatterplots with EG-p (left) and EG-1 (right). Each blue square represents an individual run. For ease of comparison, both plots have the same scales. We observe a higher correlation between EG-1 and strict precision than between EG-p and strict precision; this is confirmed by the R^2 values from applying linear regression. In fact, for systems with roughly the same strict precision—a vertical band with many runs between 0.35 and 0.40—there is quite a big range in EG-p scores.

In Figure 6, we plot mobile quality metrics against latency: each scenario A run is represented by a blue square. For ease of comparison, corresponding plots have the same scales. We note that most systems have very low latency—they appear to submit tweets almost immediately after they are posted. However, there are a number of runs that exhibit much higher latency. These runs do not appear to be able to achieve better quality as measured by online precision or online utility. In other words, at least according to metrics derived from the mobile judgments, systems were not able to

effectively exploit the additional relevance signals that accumulate over time if tweets are not immediately submitted.

In Figure 7, we plot batch quality metrics against latency: each scenario A run is represented by a blue square. In contrast to the mobile metrics, the runs that achieved the highest EG and nCG scores (but not GMP) are those with high latency. It seems that, from the perspective of the batch evaluation metrics (unlike metrics derived from the mobile assessors), systems were successful in exploiting signals that only become available if tweets are not submitted immediately. That is, waiting to accumulate evidence before deciding to submit tweets affords an opportunity to achieve higher quality—but of course, at the cost of incurring higher latency. Interestingly, this seems to be a new development in this year's evaluation. That is, systems this year were able to trade latency for higher quality (most pronounced in EG-p). From last year's evaluation, in contrast, the best high-latency system scored no higher in EG-1 than the best system that pushed tweets immediately.

5.2 Scenario B

For scenario B, we received a total of 40 runs from 15 groups. Evaluation results based on NIST assessors are shown in Table 4. Runs are sorted by nDCG-p. For reference, the empty run would have received nDCG-p and nDCG-1 scores of 0.1765.

The separation of quality metrics from latency allows us to unify the evaluation of scenario A and scenario B runs—we can simply convert scenario B runs into scenario A runs by pretending that up to ten tweets per day were submitted at 23:59:59, and then running the evaluation scripts for scenario A exactly as before. Table 5 shows the results of such an evaluation setup by the mobile assessors, and Table 6 shows the results of such an evaluation based on NIST judgments. In Figures 5, 6, and 7, all scenario B runs treated as scenario A runs are shown as empty black squares. In particular, such a treatment allows us to compare high-latency scenario A runs against scenario B runs. Interestingly, we find that the best scenario B runs can achieve higher online precision than any "true" scenario A run. In terms of nCG, with the exception of an outlier, scenario B runs are quite effective, which makes sense since delayed submission of tweets allows a system to better accumulate evidence and achieve higher recall.

6 CONCLUSIONS

The TREC 2017 RTS Track built on last year's evaluation to introduce additional novel elements. Continuing with live *in situ* evaluation using mobile assessors, we added a feedback mechanism that allowed systems to obtain judgments during the evaluation period and adapt their algorithms accordingly. This feature, coupled with information needs submitted by the mobile assessors, further enhanced the realism of the evaluation setup. Healthy participation suggests continued interested in this problem, and our efforts will continue with another iteration of the track in TREC 2018.

7 ACKNOWLEDGMENTS

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada. We'd like to thank all the mobile assessors who participated in our user study and the NIST assessors.

REFERENCES

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

[2] Javed Aslam, Fernando Diaz, Matthew Ekstrand-Abueg, Richard McCreadie, Virgil Pavlu, and Tetsuya Sakai. 2015. TREC 2015 Temporal Summarization Track Overview. *Proceedings of the Twenty-Fourth Text REtrieval Conference (TREC 2015).* Gaithersburg, Maryland.

[3] Allan Hanbury, Henning Müller, Krisztian Balog, Torben Brodt, Gordon V. Cormack, Ivan Eggel, Tim Gollub, Frank Hopfgartner, Jayashree Kalpathy-Cramer, Noriko Kando, Anastasia Krithara, Jimmy Lin, Simon Mercer, and Martin Potthast. 2015. Evaluation-as-a-Service: Overview and Outlook. *arXiv:1512.07454.*

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

[5] Jimmy Lin and Miles Efron. 2013. Overview of the TREC-2013 Microblog Track. *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013).* Gaithersburg, Maryland.

[6] Jimmy Lin, Miles Efron, Yulu Wang, and Garrick Sherman. 2014. Overview of the TREC-2014 Microblog Track. *Proceedings of the Twenty-Third Text REtrieval Conference (TREC 2014).* Gaithersburg, Maryland.

[7] Jimmy Lin, Miles Efron, Yulu Wang, Garrick Sherman, and Ellen Voorhees. 2015. Overview of the TREC-2015 Microblog Track. *Proceedings of the Twenty-Fourth Text REtrieval Conference (TREC 2015).* Gaithersburg, Maryland.

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

[9] Jiaul H. Paik and Jimmy Lin. 2015. Do Multiple Listeners to the Public Twitter Sample Stream Receive the Same Tweets? *Proceedings of the SIGIR 2015 Workshop on Temporal, Social and Spatially-Aware Information Access.* Santiago, Chile.

[10] Xin Qian, Jimmy Lin, and Adam Roegiest. 2016. Interleaved Evaluation for Retrospective Summarization and Prospective Notification on Document Streams. In *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2016).* Pisa, Italy, 175–184.

[11] Stephen Robertson and Ian Soboroff. 2002. The TREC 2002 Filtering Track Report. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002).* Gaithersburg, Maryland.

[12] Adam Roegiest, Luchen Tan, and Jimmy Lin. 2017. Online In-Situ Interleaved Evaluation of Real-Time Push Notification Systems. *Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017).* Tokyo, Japan, 415–424.

[13] Adam Roegiest, Luchen Tan, Jimmy Lin, and Charles L. A. Clarke. 2016. A Platform for Streaming Push Notifications to Mobile Assessors. *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2016).* Pisa, Italy, 1077–1080.

[14] 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).*

[15] Royal Sequiera and Jimmy Lin. 2017. Finally, a Downloadable Test Collection of Tweets. *Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017).* Tokyo, Japan, 1225–1228.

[16] Luchen Tan, Adam Roegiest, Jimmy Lin, and Charles L. A. Clarke. 2016. An Exploration of Evaluation Metrics for Mobile Push Notifications. *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2016).* Pisa, Italy, 741–744.

[17] Yulu Wang, Garrick Sherman, Jimmy Lin, and Miles Efron. 2015. Assessor Differences and User Preferences in Tweet Timeline Generation. *Proceedings of the 38th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2015).* Santiago, Chile, 615–624.

[18] Haotian Zhang, Mustafa Abualsaud, Nimesh Ghelani, Angshuman Ghosh, Mark D. Smucker, Gordon V. Cormack, , and Maura R. Grossman. 2017. UWaterlooMDS at the TREC Core Track 2017 (Notebook). *Proceedings of the Twenty-Six Text REtrieval Conference (TREC 2017).* Gaithersburg, Maryland.

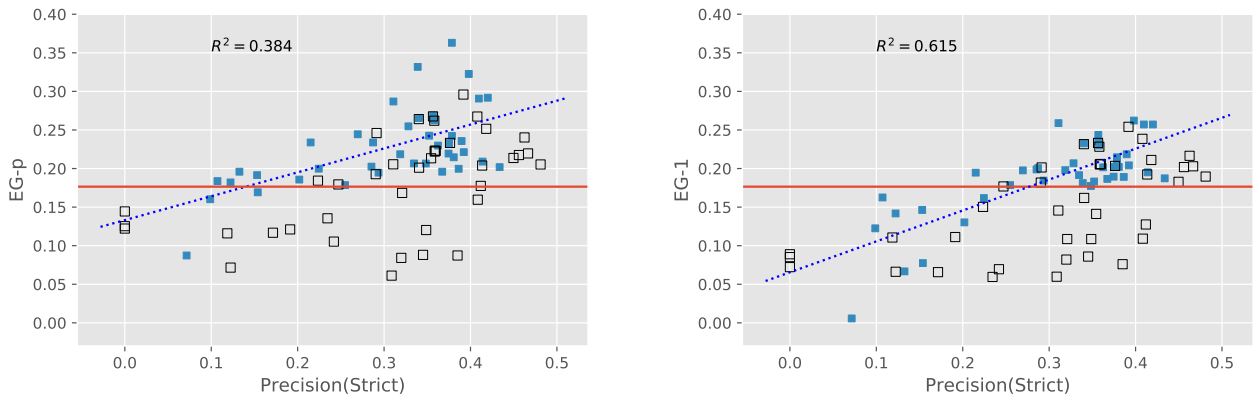

Figure 5: Scatterplots showing correlations between strict precision against EG-p (left) and EG-1 (right). Each blue square represents a scenario A run and each empty square represents a truncated scenario B run treated as if it were a scenario A run. The horizontal red lines indicate the score of an empty run. Results of linear regression include scenario A runs only.

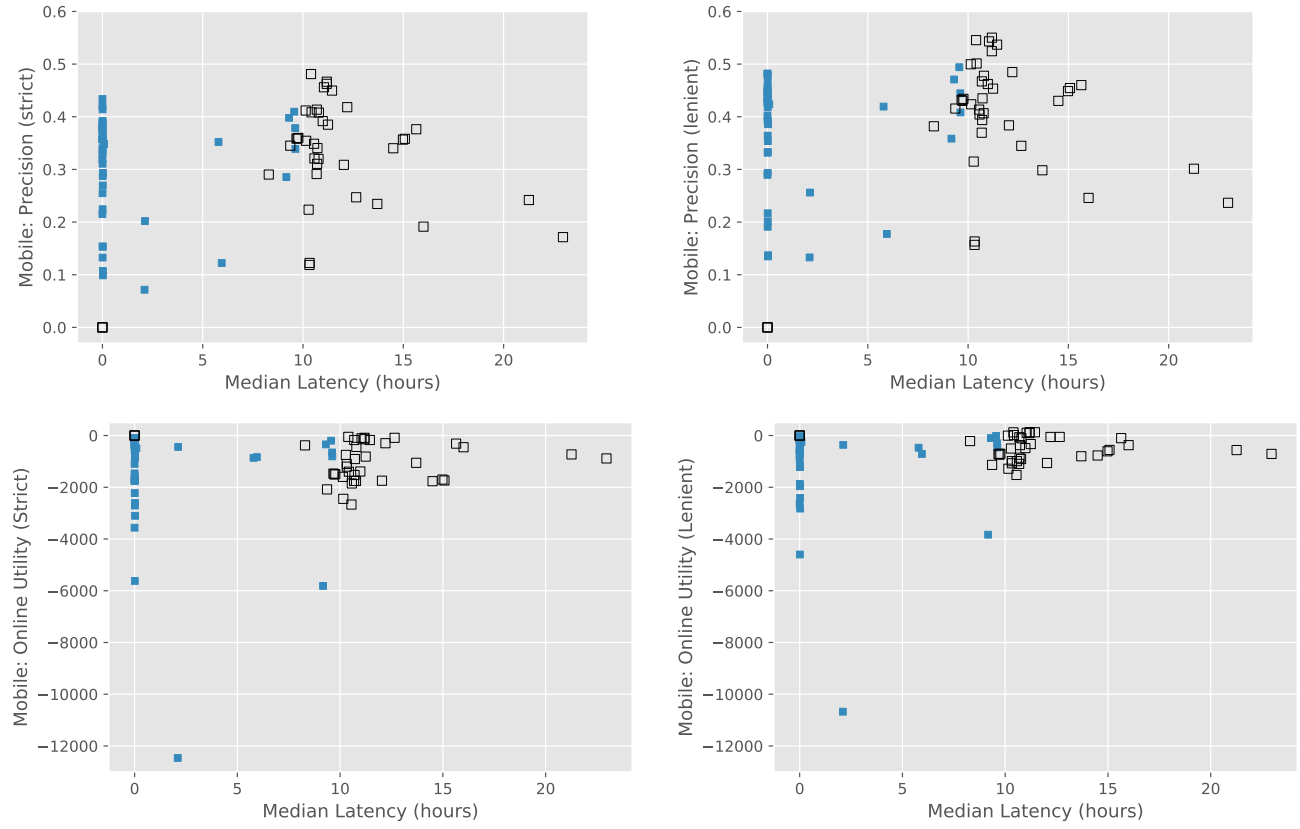

Figure 6: Scatterplots relating different mobile evaluation metrics to median latency. Each blue square represents a scenario A run and each empty square represents a truncated scenario B run treated as if it were a scenario A run. Top row: strict and lenient precision; Bottom row: strict and lenient online utility.

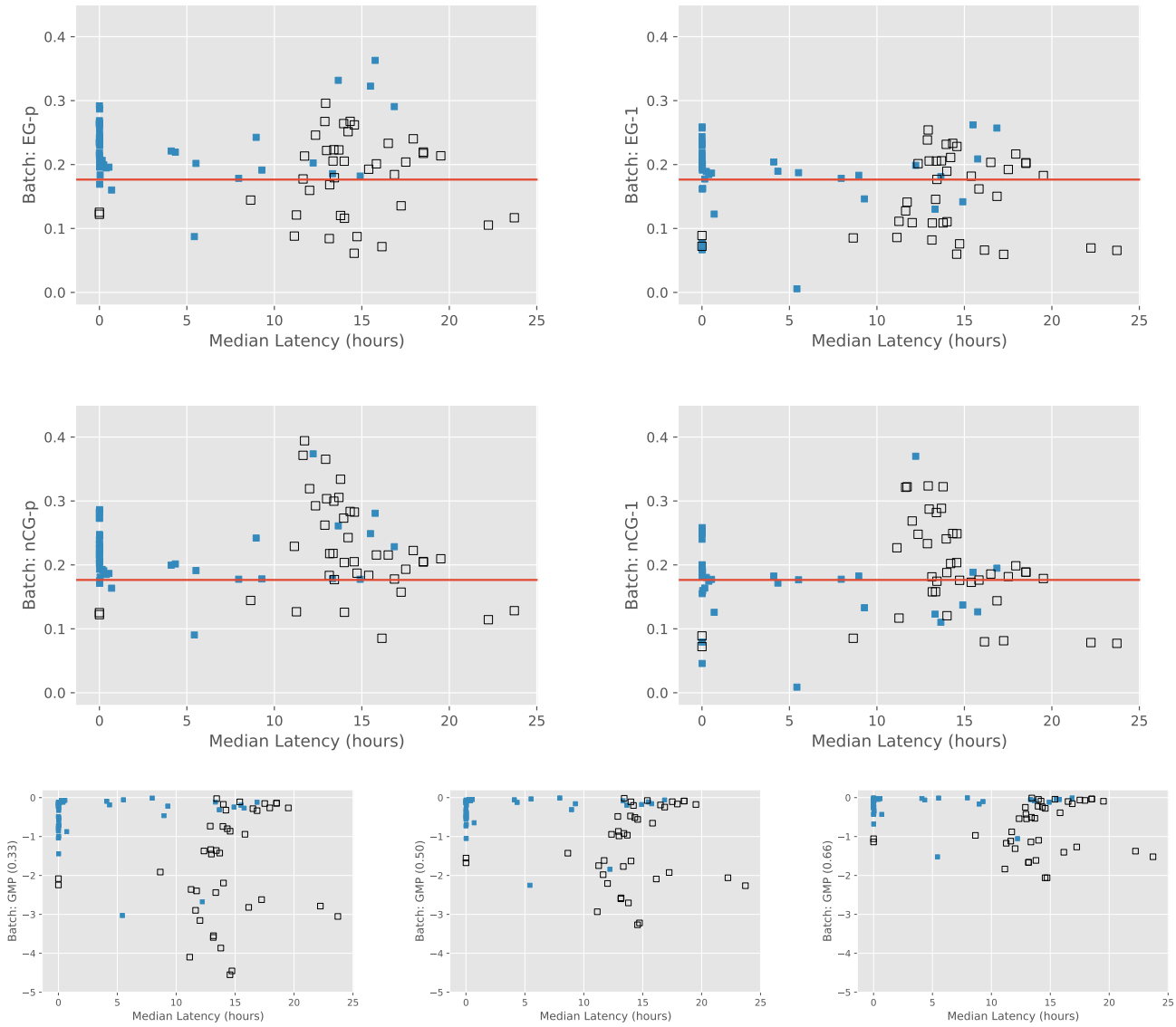

Figure 7: Scatterplots relating different batch evaluation metrics to median latency. Each blue square represents a scenario A run and each empty square represents a truncated scenario B run treated as if it were a scenario A run. Top row: EG-p (left) and EG-1 (right); Middle row: nCG-p (left) and nCG-p (right); Bottom row: GMP with $\alpha = \{0.33, 0.50, 0.66\}$. The horizontal red lines indicate the score of an empty run.

Assessor	Judgments	Profiles	Messages	Response
1	7442	29	7565	98.37%
2	14441	68	14477	99.75%
3	1181	5	2100	56.24%
4	213	3	888	23.99%
5	575	3	624	92.15%
6	539	4	766	70.37%
7	1407	16	5469	25.73%
8	6120	59	13906	44.01%
9	2488	48	12987	19.16%
10	16	2	760	2.11%
11	281	16	5045	5.57%
12	2318	9	2490	93.09%
13	119	1	198	60.1%
14	193	4	1178	16.38%
15	2923	12	2929	99.8%
16	905	5	1068	84.74%
17	157	5	789	19.9%
18	3800	35	10063	37.76%
19	8407	71	19326	43.5%
20	49	5	1208	4.06%
21	8092	24	8487	95.35%
22	91	5	1401	6.5%
23	283	7	3463	8.17%
24	495	3	539	91.84%
25	83	3	1033	8.03%
26	43	4	1409	3.05%
27	194	7	1276	15.2%
28	530	3	1209	43.84%
29	706	6	1946	36.28%
30	675	16	3856	17.51%
31	152	3	1081	14.06%
32	899	25	8560	10.5%
33	1879	8	2225	84.45%
34	26	4	1277	2.04%
35	181	33	8317	2.18%
36	74	16	4835	1.53%
37	161	2	434	37.1%
38	220	3	857	25.67%
39	4404	14	4515	97.54%
40	9979	79	20640	48.35%
41	100	8	2549	3.92%
42	2684	15	2749	97.64%

Table 1: Assessor statistics. For each assessor, columns show the number of judgments provided, the number of interest profiles subscribed to, the number of tweets delivered to that assessor, and the response rate.

team	run	R	D	N	U	L	C	$\bar{\tau}$	$\tilde{\tau}$	P_s	P_l	Util_s	Util_l	type
WUWien	WuWien-Run1	304	34	363	14	364	0.962	1	1	0.4337	0.4822	-93	-25	A
IRIT	IRIT-Run1	520	76	642	48	677	0.929	1	1	0.4200	0.4814	-198	-46	A
prna	PRNA-A1	631	98	795	46	844	0.945	296	31	0.4140	0.4783	-262	-66	A
udel_fang	UDInfoSDWR-run1	451	93	557	67	640	0.895	38722	34421	0.4096	0.4941	-199	-13	A
udel_fang	UDInfoBL-run2	667	122	887	92	954	0.904	37725	33495	0.3980	0.4708	-342	-98	A
IRIT	IRIT-Run2	404	75	551	48	607	0.921	115	32	0.3922	0.4650	-222	-72	A
QU	QUExp-run2	664	100	940	79	1014	0.922	232	1	0.3897	0.4484	-376	-176	A
PKUICST	PKUICSTRunA3	243	30	356	39	397	0.902	35	35	0.3863	0.4340	-143	-83	A
WUWien	WuWien-Run3	411	70	599	42	600	0.930	1	1	0.3806	0.4454	-258	-118	A
QU	QUBaseline-run1	875	139	1298	111	1397	0.921	213	1	0.3785	0.4386	-562	-284	A
HLJIT	testRun2	1018	178	1494	106	1470	0.928	39361	34603	0.3784	0.4446	-654	-298	A
udel	udelRun081HT-run1	501	80	754	99	874	0.887	0	0	0.3753	0.4352	-333	-173	A
IRIT	IRIT-Run3	554	113	812	62	875	0.929	131	29	0.3746	0.4510	-371	-145	A
PKUICST	PKUICSTRunA2	249	34	395	46	437	0.895	35	35	0.3673	0.4174	-180	-112	A
prna	PRNA-A3	1123	198	1777	95	1733	0.945	385	50	0.3625	0.4264	-852	-456	A
advanse	advanse_lirmm-Run3	2190	588	3335	364	3671	0.901	1	1	0.3583	0.4544	-1733	-557	A
WUWien	WuWien-Run2	1364	267	2191	231	2227	0.896	1	1	0.3569	0.4267	-1094	-560	A
advanse	advanse_lirmm-Run1	2114	543	3268	355	3563	0.900	1	1	0.3568	0.4484	-1697	-611	A
HLJIT	testRun3	1027	196	1694	168	1711	0.902	21377	20836	0.3521	0.4193	-863	-471	P
irlab	irlab-Run1	565	122	935	57	900	0.937	2285	317	0.3483	0.4236	-492	-248	P
advanse	advanse_lirmm-Run2	1876	494	3142	323	3297	0.902	1	1	0.3403	0.4300	-1760	-772	A
HLJIT	testRun1	847	173	1479	153	1464	0.895	38811	34628	0.3389	0.4082	-805	-459	A
prna	PRNA-A2	686	116	1248	68	1161	0.941	400	60	0.3346	0.3912	-678	-446	A
QU	QUExpDyn-run3	1399	274	2589	175	2388	0.927	310	1	0.3282	0.3925	-1464	-916	A
udel	udelRun081HTD-run3	334	87	627	60	661	0.909	0	0	0.3187	0.4017	-380	-206	A
PKUICST	PKUICSTRunA1	1401	241	2866	317	2864	0.889	32	32	0.3108	0.3642	-1706	-1224	A
umd-hcil	pertopicburst-run01	315	99	659	60	617	0.903	102	93	0.2936	0.3858	-443	-245	A
ICTNET	ICTNET-run3	1176	272	2645	249	2538	0.902	59	58	0.2873	0.3538	-1741	-1197	?
udel_fang	UDInfoEXP-run3	3865	990	8688	1401	9250	0.849	37822	33015	0.2854	0.3585	-5813	-3833	A
ICTNET	ICTNET-run2	1582	373	3915	378	3678	0.897	59	59	0.2695	0.3330	-2706	-1960	?
umd-hcil	retweet-run02	53	16	139	0	89	1.000	20	19	0.2548	0.3317	-102	-70	A
irlab	ldrp-Run2	640	197	2015	231	1798	0.872	37	31	0.2244	0.2935	-1572	-1178	A
udel	udelRun081D-run2	1344	466	4440	275	3528	0.922	13	0	0.2150	0.2896	-3562	-2630	A
ST	SHNU_run2	149	40	549	29	423	0.931	13616	7632	0.2019	0.2561	-440	-360	P
BJUT	BL1	1248	511	6358	225	4353	0.948	86	36	0.1538	0.2167	-5621	-4599	P
ST	SHNU_run1	181	57	944	53	687	0.923	593	49	0.1531	0.2014	-820	-706	P
ICTNET	ICTNET-run1	401	177	2447	143	1743	0.918	7470	34	0.1326	0.1911	-2223	-1869	?
ST	SHNU_run3	135	61	908	49	657	0.925	28211	21411	0.1223	0.1775	-834	-712	P
BJUT	BL2	356	99	2865	102	1823	0.944	82	83	0.1072	0.1370	-2608	-2410	P
BJUT	BL3	382	139	3345	88	2129	0.959	82	82	0.0988	0.1348	-3102	-2824	P
SOIC	SOIC-Run1	1039	894	12608	698	8453	0.917	7679	7558	0.0715	0.1329	-12463	-10675	A

Table 2: Evaluation of scenario A runs by the mobile assessors. The first two columns show the participating team and run. The next columns show the number of tweets that were judged relevant (R), redundant (D), and not relevant (N); the number of unjudged tweets (U); the length of each run (L), defined as the total number of messages delivered by the system. The next columns show coverage (C), defined the fraction of unique tweets that were judged; the mean ($\bar{t}$) and median ($\tilde{t}$) latency of submitted tweets in seconds, measured with respect to the time the original tweet was posted; strict and lenient precision; strict and lenient utility. The final column shows the run type: 'A' denotes automatic, 'P' manual preparation, and '?' indicates unknown. Rows are sorted by strict precision.

team	run	EG-p	EG-1	nCG-p	nCG-1	$GMP_{.33}$	$GMP_{.50}$	$GMP_{.66}$	mean	median	length	type
HLJIT	testRun2	0.3630	0.2088	0.2808	0.1266	-0.2720	-0.1566	-0.0479	119374	56744	621	A
HLJIT	testRun1	0.3318	0.1811	0.2610	0.1102	-0.3118	-0.1936	-0.0824	116649	49154	618	A
udel_fang	UDInfoBL-run2	0.3226	0.2622	0.2489	0.1886	-0.1952	-0.1105	-0.0308	118653	55781	452	A
IRIT	IRIT-Run1	0.2918	0.2571	0.2321	0.1974	-0.1195	-0.0615	-0.0070	67555	1	320	A
udel_fang	UDInfoSDWR-run1	0.2907	0.2571	0.2285	0.1949	-0.1190	-0.0622	-0.0087	126484	60685	308	A
PKUICST	PKUICSTRunA1	0.2869	0.2588	0.2864	0.2583	-0.7308	-0.4700	-0.2246	35761	36	1344	A
advanse	advanse_lirmm-Run1	0.2686	0.2352	0.2835	0.2501	-0.7895	-0.5045	-0.2363	38389	1	1468	A
advanse	advanse_lirmm-Run2	0.2653	0.2327	0.2728	0.2402	-0.7279	-0.4642	-0.2161	37570	1	1362	A
WUWien	WuWien-Run2	0.2640	0.2436	0.2737	0.2532	-0.4887	-0.3003	-0.1229	38666	1	996	A
advanse	advanse_lirmm-Run3	0.2626	0.2298	0.2825	0.2498	-0.8527	-0.5532	-0.2712	38112	1	1532	A
QU	QUExpDyn-run3	0.2547	0.2068	0.2475	0.1996	-0.5182	-0.3457	-0.1833	77033	19	879	A
ICTNET	ICTNET-run2	0.2444	0.1976	0.2455	0.1987	-0.9911	-0.6891	-0.4049	47988	74	1473	?
HLJIT	testRun3	0.2426	0.1832	0.2420	0.1826	-0.4618	-0.3086	-0.1645	101708	32255	773	P
QU	QUBaseline-run1	0.2422	0.2146	0.2260	0.1984	-0.2326	-0.1459	-0.0644	64813	1	446	A
QU	QUExp-run2	0.2356	0.2185	0.2159	0.1987	-0.1498	-0.0909	-0.0354	63944	1	306	A
ICTNET	ICTNET-run3	0.2338	0.2005	0.2227	0.1893	-0.5869	-0.4040	-0.2318	66596	80	892	?
udel	udelRun081D-run2	0.2338	0.1947	0.2393	0.2002	-1.0364	-0.7252	-0.4323	61905	1	1521	A
udel	udelRun081HT-run1	0.2330	0.2023	0.2193	0.1886	-0.2165	-0.1401	-0.0683	31787	1	393	A
prna	PRNA-A3	0.2298	0.2016	0.2280	0.1998	-0.3278	-0.2052	-0.0899	39366	74	636	A
IRIT	IRIT-Run2	0.2212	0.2041	0.1996	0.1825	-0.0942	-0.0557	-0.0195	96894	14768	201	A
IRIT	IRIT-Run3	0.2194	0.1895	0.2015	0.1716	-0.1853	-0.1221	-0.0626	98865	15623	320	A
udel	udelRun081HTD-run3	0.2185	0.1979	0.2022	0.1816	-0.1891	-0.1279	-0.0703	37468	0	303	A
WUWien	WuWien-Run3	0.2146	0.2021	0.2095	0.1970	-0.1421	-0.0931	-0.0470	70499	2	245	A
prna	PRNA-A1	0.2090	0.1951	0.2052	0.1913	-0.1330	-0.0780	-0.0262	50613	69	295	A
prna	PRNA-A2	0.2066	0.1914	0.2058	0.1906	-0.2630	-0.1707	-0.0839	29994	78	470	A
irlab	irlab-Run1	0.2065	0.1774	0.1929	0.1638	-0.1156	-0.0696	-0.0263	72250	561	242	P
udel_fang	UDInfoEXP-run3	0.2025	0.1988	0.3737	0.3700	-2.6753	-1.8402	-1.0542	66577	43980	4140	A
WUWien	WuWien-Run1	0.2018	0.1873	0.1912	0.1767	-0.0567	-0.0335	-0.0116	122571	19872	122	A
irlab	ldrp-Run2	0.1998	0.1617	0.1932	0.1551	-0.5084	-0.3634	-0.2269	71463	58	697	A
PKUICST	PKUICSTRunA3	0.1997	0.1892	0.1908	0.1804	-0.0657	-0.0438	-0.0232	106453	892	111	A
ICTNET	ICTNET-run1	0.1959	0.0667	0.1751	0.0458	-0.5525	-0.4037	-0.2636	58681	38	700	?
PKUICST	PKUICSTRunA2	0.1959	0.1866	0.1866	0.1774	-0.0705	-0.0477	-0.0262	135577	1957	114	A
umd-hcil	pertopicburst-run01	0.1947	0.1844	0.1850	0.1746	-0.1165	-0.0789	-0.0436	97337	1404	183	A
ST	SHNU_run1	0.1914	0.1463	0.1781	0.1330	-0.2181	-0.1579	-0.1012	106013	33428	283	P
ST	SHNU_run2	0.1857	0.1302	0.1785	0.1229	-0.1127	-0.0812	-0.0515	96782	47966	151	P
BJUT	BL2	0.1837	0.1625	0.1809	0.1598	-0.7058	-0.5184	-0.3420	88013	147	869	P
ST	SHNU_run3	0.1820	0.1418	0.1775	0.1373	-0.2420	-0.1782	-0.1181	155030	53664	296	P
umd-hcil	retweet-run02	0.1785	0.1785	0.1776	0.1776	-0.0130	-0.0090	-0.0053	67657	28658	20	A
Empty run		0.1765	0.1765	0.1765	0.1765	0.0000	0.0000	0.0000	-	-	0	-
BJUT	BL1	0.1692	0.0774	0.1711	0.0793	-1.4431	-1.0493	-0.6787	74216	52	1859	P
BJUT	BL3	0.1602	0.1225	0.1636	0.1258	-0.8745	-0.6466	-0.4320	103187	2513	1058	P
SOIC	SOIC-Run1	0.0873	0.0057	0.0903	0.0088	-3.0285	-2.2526	-1.5223	95850	19535	3554	A

Table 3: Evaluation of scenario A runs by NIST assessors. The columns marked "mean" and "median" show the mean and median latency with respect to the first tweet in each cluster. The second to last column shows the length of each run, defined as the number of tweets delivered for the interest profiles that were assessed. The final column shows the run type: 'A' denotes automatic, 'P' manual preparation, and '?' indicates unknown. Rows are sorted by EG-p.

team	run	nDCG-p	nDCG-1	type
HLJIT	qFB_url	0.3656	0.2910	A
PKUICST	PKUICSTRunB1	0.3483	0.3003	A
HLJIT	HLJIT_l2r	0.3274	0.2778	P
udel_fang	UDInfoW2VPre	0.2933	0.2775	A
udel_fang	UDInfoW2VTWT	0.2906	0.2759	A
udel_fang	UDInfoJac	0.2886	0.2723	A
HLJIT	HLJIT_rank_svm	0.2865	0.2376	P
udel	udelRun081D-B	0.2808	0.2329	I
PRNA	PRNA-B2	0.2752	0.2400	A
NOVASearch	NOVASearchB3	0.2710	0.2587	A
advanse_lirmm	adv_lirmm-Run1	0.2669	0.2289	A
advanse_lirmm	adv_lirmm-Run3	0.2656	0.2285	A
advanse_lirmm	adv_lirmm-Run2	0.2601	0.2227	A
udel	udelRun081HT-B	0.2552	0.2124	I
PKUICST	PKUICSTRunB3	0.2306	0.2024	A
udel	udelRun081HTD-B	0.2242	0.1933	I
ICTNET	ICTNET-Run3	0.2185	0.1527	A
PRNA	PRNA-B3	0.2143	0.1686	A
IRIT	IRIT-RunB2	0.2142	0.1833	A
IRIT	IRIT-RunB1	0.2130	0.1962	I
IRIT	IRIT-RunB3	0.2117	0.1961	I
PRNA	PRNA-B1	0.2071	0.1914	A
ICTNET	ICTNET-Run2	0.2047	0.1381	A
PKUICST	PKUICSTRunB2	0.1968	0.1809	A
NOVASearch	NOVASearchB1	0.1896	0.1896	A
umd-hcil	umc_hcil_ptbv1	0.1863	0.1747	A
BJUT	bjut_tmg	0.1796	0.1456	A
umd-hcil	umc_hcil_rtv1	0.1778	0.1753	A
Empty run		0.1765	0.1765	-
ISIKol	lm-jm-lambda0.5	0.1725	0.1725	A
ST	SHNU_run1	0.1551	0.0741	P
SOIC	IUB	0.1442	0.1442	A
NOVASearch	NOVASearchB2	0.1440	0.1333	A
IRLAB_DAIICT	IRLAB-DAIICT	0.1324	0.0697	A
ICTNET	ICTNET-Run1	0.1208	0.1143	A
BJUT	bjutg	0.1169	0.1169	A
ST	SHNU_run3	0.1166	0.0689	P
ST	SHNU_run2	0.1135	0.0729	P
IRLAB_DAIICT	IRLAB_LDRP	0.1099	0.0773	A
IRLAB_DAIICT	IRLAB-LDRP2	0.0995	0.0619	A
BJUT	bjutgs	0.0746	0.0746	A

Table 4: Evaluation of scenario B runs by NIST assessors. The final column shows the run type: 'A' denotes automatic, 'P' manual preparation, and 'I' manual intervention. Rows are sorted by nDCG-p.

team	run	R	D	N	U	L	C	$\bar{\tau}$	$\tilde{\tau}$	P_s	P_l	Util$_s$	Util$_l$	type
PRNA	PRNA-B1	650	87	614	276	982	0.719	41934	37436	0.4811	0.5455	-51	123	A
IRIT	IRIT-RunB1	614	110	592	312	1033	0.698	43355	40275	0.4666	0.5502	-88	132	I
PKUICST	PKUICSTRunB3	887	119	912	588	1641	0.642	43904	40259	0.4625	0.5245	-144	94	A
IRIT	IRIT-RunB3	554	106	555	283	946	0.701	42685	39757	0.4560	0.5432	-107	105	I
IRIT	IRIT-RunB2	773	150	796	464	1404	0.670	44401	41206	0.4497	0.5369	-173	127	A
udel	udelRun081HT-B	759	121	935	393	1434	0.726	47018	43938	0.4182	0.4848	-297	-55	I
PKUICST	PKUICSTRunB2	416	54	536	383	964	0.603	43110	38474	0.4135	0.4672	-174	-66	A
HLJIT	HLJIT_l2r	3735	796	4536	6574	11272	0.417	40959	36482	0.4119	0.4997	-1597	-5	P
HLJIT	HLJIT_rank_svm	3117	708	3807	7184	11154	0.356	41746	37532	0.4084	0.5012	-1398	18	P
PRNA	PRNA-B2	975	167	1248	2088	3327	0.372	42175	38846	0.4079	0.4778	-440	-106	A
PKUICST	PKUICSTRunB1	2518	452	3456	1698	5141	0.670	43171	39538	0.3918	0.4622	-1390	-486	A
NOVASearch	NOVASearchB2	1363	242	1935	12379	14261	0.132	44002	40481	0.3850	0.4534	-814	-330	A
udel	udelRun081HTD-B	476	106	683	379	1075	0.647	53107	56322	0.3763	0.4601	-313	-101	I
udel_fang	UDInfoW2VPre	1918	395	3017	1732	4694	0.631	40155	35163	0.3598	0.4340	-1494	-704	A
udel_fang	UDInfoJac	1866	380	2956	1615	4517	0.642	39934	34831	0.3587	0.4318	-1470	-710	A
udel_fang	UDInfoW2VTWT	1908	380	3031	1721	4680	0.632	40049	34996	0.3587	0.4302	-1503	-743	A
advanse_lirmm	adv_lirmm-Run3	2192	590	3339	460	3769	0.878	52952	54271	0.3581	0.4545	-1737	-557	A
advanse_lirmm	adv_lirmm-Run1	2116	546	3271	449	3657	0.877	52573	53930	0.3566	0.4487	-1701	-609	A
HLJIT	qFB_url	2979	582	4844	5334	9791	0.455	40957	36560	0.3544	0.4237	-2447	-1283	A
NOVASearch	NOVASearchB3	3074	572	5170	9498	14231	0.333	42827	37981	0.3487	0.4136	-2668	-1524	A
NOVASearch	NOVASearchB1	2319	474	3927	10176	13790	0.262	38422	33712	0.3451	0.4156	-2082	-1134	A
PRNA	PRNA-B3	969	269	1608	2156	3686	0.415	42269	38588	0.3405	0.4350	-908	-370	A
advanse_lirmm	adv_lirmm-Run2	1878	497	3146	412	3388	0.878	52223	52198	0.3402	0.4302	-1765	-771	A
ICTNET	ICTNET-Run2	1659	431	3080	7650	10540	0.274	43584	38054	0.3209	0.4043	-1852	-990	A
SOIC	IUB	1558	423	2890	9173	11840	0.225	43256	38784	0.3199	0.4067	-1755	-909	A
ICTNET	ICTNET-Run3	1250	335	2440	4896	7145	0.315	43752	38528	0.3106	0.3938	-1525	-855	A
ISIKol	lm-jm-lambda0.5	1409	342	2813	12580	15040	0.164	45996	43334	0.3087	0.3837	-1746	-1062	A
udel	udelRun081D-B	1225	330	2651	2626	4840	0.457	42683	38446	0.2913	0.3697	-1756	-1096	I
umd-hcil	umc_hcil_ptbv1	263	83	560	182	649	0.720	33269	29852	0.2903	0.3819	-380	-214	A
umd-hcil	umc_hcil_rtv1	43	17	114	119	199	0.402	43190	45544	0.2471	0.3448	-88	-54	A
ST	SHNU_run2	342	84	988	7253	8034	0.097	74267	76535	0.2419	0.3013	-730	-562	P
ST	SHNU_run1	466	127	1394	6891	7967	0.135	50202	49330	0.2345	0.2984	-1055	-801	P
BJUT	bjut_tmg	303	124	929	517	1239	0.583	42435	37007	0.2235	0.3149	-750	-502	A
ICTNET	ICTNET-Run1	140	40	552	6170	6527	0.055	51994	57620	0.1913	0.2459	-452	-372	A
ST	SHNU_run3	231	88	1029	7555	8289	0.089	80368	82656	0.1714	0.2366	-886	-710	P
BJUT	bjutgs	195	65	1333	7151	8010	0.107	41301	37180	0.1224	0.1632	-1203	-1073	A
BJUT	bjutg	171	55	1216	5296	6070	0.128	41230	37158	0.1186	0.1567	-1100	-990	A
IRLAB_DAIICT	IRLAB-DAIICT	0	0	0	6256	6256	0.000	0	0	0.0000	0.0000	0	0	A
IRLAB_DAIICT	IRLAB-LDRP2	0	0	0	7416	7416	0.000	0	0	0.0000	0.0000	0	0	A
IRLAB_DAIICT	IRLAB_LDRP	0	0	0	6821	6821	0.000	0	0	0.0000	0.0000	0	0	A

Table 5: Evaluation of scenario B runs as scenario A runs by the mobile assessors. The first two columns show the participating team and run. The next columns show the number of tweets that were judged relevant (R), redundant (D), and not relevant (N); the number of unjudged tweets (U); the length of each run (L), defined as the total number of messages delivered by the system. The next columns show coverage (C), defined the fraction of unique tweets that were judged; the mean ($\bar{t}$) and median ($\tilde{t}$) latency of submitted tweets in seconds, measured with respect to the time the original tweet was posted; strict and lenient precision; strict and lenient utility. The final column shows the run type: 'A' denotes automatic, 'P' manual preparation, and 'I' manual intervention. Rows are sorted by strict precision.

team	run	EG-p	EG-1	nCG-p	nCG-1	$GMP_{.33}$	$GMP_{.50}$	$GMP_{.66}$	mean	median	length	type
PKUICST	PKUICSTRunB1	0.2959	0.2541	0.3653	0.3236	-1.3363	-0.8676	-0.4265	73387	46551	2409	A
advanse_lirmm	adv_lirmm-Run1	0.2676	0.2332	0.2836	0.2492	-0.7992	-0.5113	-0.2402	85809	51599	1483	A
PRNA	PRNA-B2	0.2674	0.2385	0.2622	0.2333	-0.7328	-0.4836	-0.2490	80632	46389	1272	A
advanse_lirmm	adv_lirmm-Run2	0.2641	0.2316	0.2732	0.2407	-0.7400	-0.4726	-0.2210	84177	50264	1381	A
advanse_lirmm	adv_lirmm-Run3	0.2620	0.2283	0.2826	0.2488	-0.8625	-0.5599	-0.2752	85436	52466	1547	A
udel	udelRun081HT-B	0.2515	0.2111	0.2427	0.2022	-0.3211	-0.2017	-0.0893	68787	51170	627	I
udel	udelRun081D-B	0.2460	0.2014	0.2925	0.2479	-1.3709	-0.9433	-0.5408	84751	44471	2121	I
PKUICST	PKUICSTRunB3	0.2403	0.2165	0.2225	0.1986	-0.2633	-0.1646	-0.0717	105965	64582	508	A
udel	udelRun081HTD-B	0.2332	0.2034	0.2155	0.1857	-0.2836	-0.1888	-0.0995	76252	59434	477	I
udel_fang	UDInfoJac	0.2232	0.2054	0.2997	0.2819	-1.3598	-0.9198	-0.5057	81574	48242	2190	A
udel_fang	UDInfoW2VPre	0.2229	0.2059	0.3055	0.2885	-1.4253	-0.9652	-0.5322	82742	49278	2289	A
udel_fang	UDInfoW2VTWT	0.2220	0.2056	0.3036	0.2873	-1.4505	-0.9836	-0.5441	81348	46754	2320	A
IRIT	IRIT-RunB1	0.2196	0.2029	0.2053	0.1885	-0.1459	-0.0889	-0.0353	121242	66657	298	I
IRIT	IRIT-RunB3	0.2175	0.2019	0.2045	0.1889	-0.1405	-0.0860	-0.0348	126789	66657	286	I
IRIT	IRIT-RunB2	0.2137	0.1828	0.2096	0.1787	-0.2648	-0.1762	-0.0928	124420	70250	449	A
HLJIT	qFB_url	0.2133	0.1412	0.3942	0.3222	-2.3972	-1.6153	-0.8795	67876	42215	3946	A
ICTNET	ICTNET-Run3	0.2055	0.1456	0.2180	0.1581	-2.4390	-1.7697	-1.1397	84868	48076	3149	A
PRNA	PRNA-B1	0.2053	0.1896	0.2037	0.1880	-0.1729	-0.1073	-0.0455	97621	50383	344	A
PKUICST	PKUICSTRunB2	0.2039	0.1923	0.1932	0.1816	-0.1510	-0.1047	-0.0611	140193	63037	228	A
PRNA	PRNA-B3	0.2011	0.1619	0.2155	0.1763	-0.9414	-0.6569	-0.3892	104941	56998	1389	A
umd-hcil	umc_hcil_ptbv1	0.1927	0.1818	0.1836	0.1728	-0.1089	-0.0741	-0.0413	128880	55415	170	A
BJUT	bjut_tmg	0.1843	0.1503	0.1779	0.1439	-0.3349	-0.2432	-0.1569	129274	60711	426	A
umd-hcil	umc_hcil_rtv1	0.1794	0.1768	0.1770	0.1745	-0.0264	-0.0180	-0.0102	69261	48350	41	A
HLJIT	HLJIT_l2r	0.1775	0.1275	0.3714	0.3214	-2.8964	-1.9800	-1.1176	65680	41908	4574	P
Empty run		0.1765	0.1765	0.1765	0.1765	0.0000	0.0000	0.0000	-	-	0	-
ICTNET	ICTNET-Run2	0.1684	0.1086	0.2177	0.1579	-3.5487	-2.5760	-1.6606	87481	47392	4586	A
HLJIT	HLJIT_rank_svm	0.1596	0.1091	0.3192	0.2687	-3.1576	-2.2065	-1.3114	70265	43251	4660	P
IRLAB_DAIICT	IRLAB-DAIICT	0.1444	0.0852	0.1444	0.0852	-1.9129	-1.4272	-0.9701	31121	31121	2217	A
ST	SHNU_run1	0.1355	0.0595	0.1574	0.0814	-2.6213	-1.9243	-1.2683	106742	62079	3251	P
IRLAB_DAIICT	IRLAB_LDRP	0.1253	0.0889	0.1253	0.0889	-2.0825	-1.5541	-1.0568	0	0	2412	A
IRLAB_DAIICT	IRLAB-LDRP2	0.1222	0.0722	0.1222	0.0722	-2.2440	-1.6746	-1.1387	0	0	2599	A
ICTNET	ICTNET-Run1	0.1211	0.1112	0.1267	0.1168	-2.3595	-1.7481	-1.1726	131465	40543	2821	A
NOVASearch	NOVASearchB3	0.1203	0.1086	0.3340	0.3223	-3.8660	-2.7055	-1.6134	79407	49619	5684	A
ST	SHNU_run3	0.1168	0.0657	0.1284	0.0773	-3.0533	-2.2635	-1.5202	157904	85379	3630	P
BJUT	bjutg	0.1160	0.1106	0.1259	0.1205	-2.1947	-1.6285	-1.0957	102827	50425	2610	A
ST	SHNU_run2	0.1053	0.0695	0.1143	0.0785	-2.7875	-2.0599	-1.3752	130636	80036	3356	P
NOVASearch	NOVASearchB1	0.0881	0.0859	0.2290	0.2268	-4.0991	-2.9327	-1.8348	80135	40115	5580	A
NOVASearch	NOVASearchB2	0.0873	0.0760	0.1871	0.1759	-4.4537	-3.2204	-2.0595	94941	53018	5862	A
SOIC	IUB	0.0842	0.0820	0.1835	0.1813	-3.5923	-2.6002	-1.6665	85217	47307	4680	A
BJUT	bjutgs	0.0716	0.0662	0.0852	0.0798	-2.8204	-2.0905	-1.4036	105516	58134	3370	A
ISIKol	lm-jm-lambda0.5	0.0612	0.0598	0.2050	0.2036	-4.5514	-3.2680	-2.0602	86089	52416	6144	A

Table 6: Evaluation of scenario B runs as scenario A runs by NIST assessors. The columns marked "mean" and "median" show the mean and median latency with respect to the first tweet in each cluster. The second to last column shows the length of each run, defined as the number of tweets delivered for the interest profiles that were assessed. The final column shows the run type: 'A' denotes automatic, 'P' manual preparation, and 'I' manual intervention. Rows are sorted by EG-p.