

Overview of the TREC-2014 Microblog Track

Jimmy Lin,¹ Miles Efron,² Yulu Wang,¹ and Garrick Sherman²

¹ University of Maryland, College Park

² University of Illinois, Urbana-Champaign

jimmylin@umd.edu, mefron@illinois.edu, ylwang@cs.umd.edu, gsherma2@illinois.edu

1. INTRODUCTION

This year represents the fourth iteration of the TREC Microblog track, which has been running since 2011. The track continued using the “evaluation as a service” model [8, 7], in which participants had access to the document collection only through an API. In addition to the temporally-anchored ad hoc retrieval task, which has been running since the inception of the track, we introduced a new task called tweet timeline generation (TTG), where the goal is to produce concise “summaries” about a particular topic for human consumption.

Although this overview covers both tasks, more emphasis is placed on the tweet timeline generation task, which necessitated the development of a new evaluation methodology. We refer the reader to previous track overview papers [8, 12, 9] for details on the setup of the ad hoc task.

2. TASK DESCRIPTION

One assumption of the Cranfield Paradigm [4], which provides the basis of most TREC evaluations, is that researchers can acquire the document collection under study. What if this were not possible? Tweets represent an example of such collections: Twitter’s terms of service forbid redistribution of tweets, and thus it would not be permissible for an organization to host a collection of tweets for download.

The evaluation-as-a-service (EaaS) model was introduced in the TREC 2013 Microblog track as one solution to this challenge [8, 7]. Under this model, we (as the track organizers) gathered a collection of tweets centrally, but, instead of distributing the tweets themselves, we provided a service API (built on the open-source Lucene search engine) through which participants could access the tweets to complete the task.

The collection and service API used in this year’s track was the same as last year’s [8]. The collection, known as Tweets2013, consists of 243 million tweets crawled from the public Twitter sample stream between February 1 and March 31, 2013 (inclusive). This level of access is available to anyone with a Twitter account and does not require any special authorization. The API provides basic search access and returns tweet content as well as various metadata fields. More details about the API specification are provided in last year’s track overview paper [8].

Experience from last year’s track suggested that the evaluation-as-a-service model provided a satisfactory solution to the collection redistribution issues. The level of participation (one of the most popular tracks at TREC) seems to indicate that the transition to API-based access was not overly

burdensome, and that the API provided sufficient flexibility for participants to pursue their own research ideas [16].

2.1 Temporally-Anchored Ad Hoc Retrieval

The putative user model for the ad hoc retrieval task is as follows: “At time T , give me the most relevant tweets about an information need expressed as query Q .” Although the task definition has not substantively changed since the first Microblog track in TREC 2011, we have in recent years gained a more refined understanding of how the task is operationalized in TREC [17].

The above user model can actually describe two slightly different scenarios: Consider a scenario where a journalist is investigating a sports scandal that has been brewing for the past several weeks. She just got news of a breaking development, and turns to searching tweets to find out more details: the scandal’s major facts, reactions from fellow athletes, commentary from analysts, etc. Since this particular news story has been developing for several weeks, any keyword search involving the athlete’s name would likely bring up results from many different points in time. In this case, the journalist specifically wants to see the most recent tweets about the event. For convenience, we refer to this as the “real-time search” scenario.

Consider another scenario where a journalist is searching an archive of tweets as part of writing a retrospective piece about the impact of social media on the Egyptian revolution. The topic is temporally-anchored in the sense that political events may not have played out fully, and the journalist is interested in perspectives at a particular point in time T . In this case, she might want to search for results that accurately reflect the volume of discourse on the topic (not necessarily biased toward more recent tweets prior to time T). For convenience, we refer to this as “archive search”.

Although the original conception of the task was closer to the real-time search scenario, NIST assessors indicate that topic development and relevance judgments followed a model closer to the archive search scenario. Given that the document collection was indeed retrospective, archive search represented a more natural operationalization of the ad hoc task.

From a technical perspective, these two alternative scenarios hold implications for the design of the search infrastructure. The search API is built on an index of the entire collection; the temporal constraint T is satisfied by retrieving from the entire collection, but then discarding tweets that occur after T . One concern with this approach is that the service is taking advantage of term statistics of tweets that occur “in the future” (if one wishes to simulate the

real-time scenario). However, we have confirmed in a recent study that such inclusion of future term statistics does not substantively affect ranking results [17].

2.2 Tweet Timeline Generation

This year, we introduced a new task called tweet timeline generation (TTG), motivated by the observation that for many queries, search systems often return many tweets that are duplicates, near-duplicates, or contain the same (or highly-similar) information—a user is unlikely to want to see an enumeration of all these tweets. Instead, it would be desirable if the system produced a “summary” timeline about the topic. In the task definition this year, a summary is operationalized as a list of non-redundant, chronologically ordered tweets. Thus, the putative user model is as follows: “At time T , I have an information need expressed by query Q , and I would like a summary that captures relevant information.” It is imagined that the user would consume the entire summary (unlike a ranked list, where the user might stop reading at any time).

The tweet timeline generation task supplements the standard challenges of ad hoc retrieval with issues from topic detection and tracking (TDT) and multi-document summarization. In our conception, systems need to address two additional challenges (beyond ad hoc retrieval):

- Detect (and eliminate) redundant tweets. This is equivalent to saying that systems must detect novelty.
- Determine how many results to return. Some topics have more relevant and non-redundant tweets than others and a system must be able to automatically infer this. Systems can make different precision/recall tradeoffs along these lines.

Redundancy is operationalized 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, and so we use them interchangeably, but in opposite contexts.

Note that because of 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. Finally, we 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 . In this task setup, redundancy boils down to the definition of the binary relation “contains substantively similar information”. This is more precisely defined as part of our evaluation methodology, which is described in Section 3.2.

We imagined that participants would tackle the TTG task in a pipelined architecture that begins with ad hoc retrieval followed by summary generation. Therefore, participants in the TTG task were also required to participate in the ad hoc retrieval task (i.e., submit runs).

The tweet timeline generation task represents a natural extension of classic ad hoc retrieval and shares similarities with previous tasks such as aspect retrieval [10], sub-topic retrieval [18], and the notion of “information nuggets” in TREC question answering evaluations [15, 3]. From the

perspective of search result diversification, Tao et al. [13] recently explored multi-aspect retrieval in the tweet context. Previous work along these lines suggest that this task is not so difficult as to preclude meaningful progress toward its solution, which is an important consideration in developing TREC tasks.

3. EVALUATION METHODOLOGY

3.1 Temporally-Anchored Ad Hoc Retrieval

For TREC 2014, NIST assessors developed a total of 55 new topics. The ad hoc retrieval task was evaluated using a standard pooling methodology by NIST assessors. Judgment pools were created using depth 100 across all submitted ad hoc runs, plus a random selection of 100 tweets per topic from each TTG run. Although we envisioned a system architecture consisting of ad hoc retrieval followed by summary generation, the inclusion of TTG results in the pool was explicitly designed to reduce missing judgments for TTG runs in cases where participants’ systems did not include an explicit ad hoc retrieval stage.

These judgment pools were further reduced by removing retweets (declared not relevant by track fiat) and then clustered so that textually similar tweets are presented to assessors in close proximity (to enhance judgment consistency). Tweets were judged on a three-way scale of “not relevant”, “relevant”, and “highly relevant”.

3.2 Tweet Timeline Generation

The TTG definition of redundancy and the assumption of transitivity means that the task can be viewed as semantic clustering—that is, we wish to group relevant tweets into clusters in which all tweets share substantively similar information. Within each cluster, the earliest tweet is novel; all other tweets in the cluster are redundant with respect to all earlier tweets.

Our annotation methodology to generate judgments for evaluation builds on exactly this idea. For a topic, we begin with the list of relevant tweets, ordered chronologically, from earliest to latest. These tweets are presented, one at a time, to a human assessor. For each tweet, the assessor can add it to an existing cluster if she thinks the tweet contains substantively similar information with respect to tweets in the existing cluster, or she can create a new cluster for the tweet. We have developed a JavaScript-based annotation interface to help assessors accomplish this task—a screenshot is shown in Figure 1.

In the interface, the next tweet to be processed is shown at the bottom of the screen. The assessor can either add the tweet to an existing cluster by clicking the “Add” button next to the cluster or create a new cluster by hitting the space bar. At any time, the assessor can expand a cluster to show all tweets contained in it, or collapse the cluster, in which only the first tweet is shown. The interface also implements an undo feature that allows the assessor to reverse the action taken and go back to the previous tweet.

The TTG evaluation methodology boils down to this central question: what exactly does “substantively similar information” mean? Like document relevance in ad hoc retrieval, assessors make the final determination and we expect natural variations among humans. However, pilot studies helped us devise a set of guidelines, which were provided as instructions to the assessors. We told them: A good rule of thumb

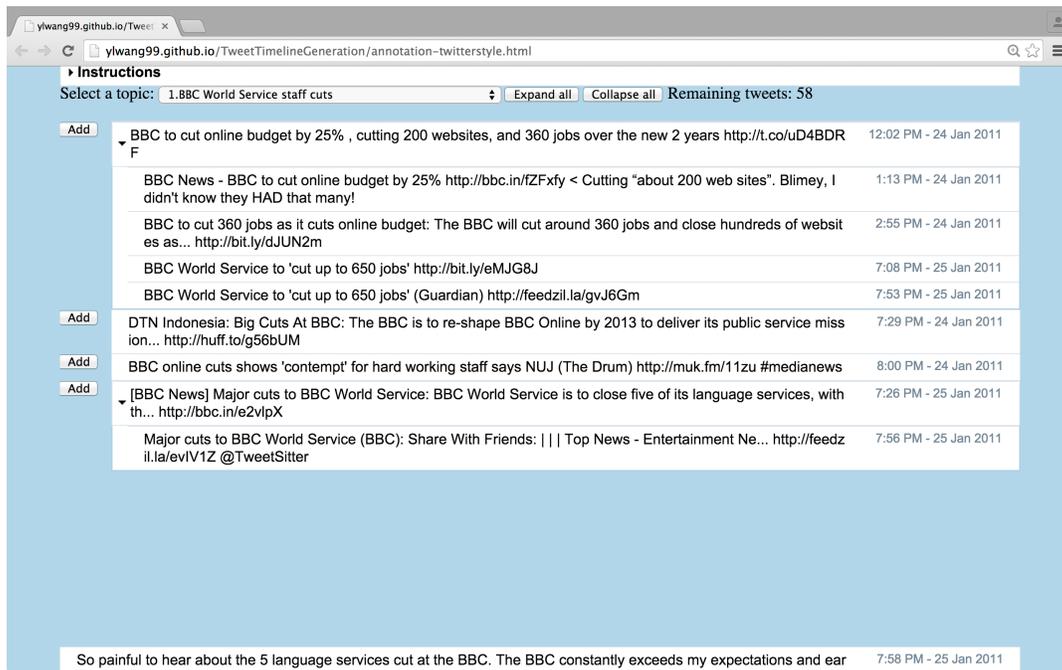


Figure 1: Screenshot of the clustering interface for assessors. Tweets are presented one at a time in chronological order. The next tweet to be processed is shown at the bottom of the screen: the assessor can either add the tweet to an existing cluster (by clicking the “Add” button next to the cluster) or create a new cluster (by hitting the space bar). At any time, the assessor can expand a cluster to show all tweets contained in it, or collapse the cluster, in which case only the first tweet is shown.

is that if two tweets “say the same thing”, then they’re substantively similar. To speed up the clustering process, the annotators were asked not to consider external content (e.g., follow links in the tweets).

To provide further guidance, we devised a number of questions that the assessor might consider in determining whether two tweets should be in the same cluster:

- If I had already seen the first tweet, would I have missed out on some information if I didn’t see the second tweet?
- If two tweets are similar but the second contains an addition to or endorsement of the first, is the addition or endorsement important enough that I would be interested in seeing both tweets?
- Sometimes two tweets look similar but actually narrate the development of an event. Are the tweets different enough from each other such that I would want to see both tweets to understand how an event develops or unfolds?

The annotation process proceeded concurrently at the University of Maryland and the University of Illinois, using relevant documents from the NIST judgment pools as the starting point. Due to resource constraints, NIST assessors were not able to perform the clustering, and thus a weakness of our setup is that the individual with the information need was not the one who created the clusters. The two assessors at the University of Maryland were graduate students in computer science (both male). The two assessors at the

University of Illinois were graduate students in library and information science (one male, one female).

Assessors were first trained in the laboratory: the session included an introduction to the task and an overview of the annotation interface. After that, assessors were free to perform annotations at their own pace on their laptops, at locations of their choosing (this was possible because the annotation interface was implemented in JavaScript and hence accessible over the web). All assessors began with a throwaway “practice topic” (although they were not aware of the throwaway nature) and then proceeded to annotate topics in batches (roughly ten topics per batch). Topics were grouped into batches of roughly equal size (in terms of the number of relevant documents). When an assessor completed a batch, he or she could request another batch to work on. In order to preserve consistency across topics, we opted to have fewer annotators each working on more topics, as opposed to many annotators each processing only a single batch.

Each site annotated the topic batches in the opposite order, and when a covering set for all topics had been obtained, we designated those clusters to be the “official” judgments. However, the annotation process continued until both sites had processed all topics, giving us alternate judgments. Thus, both the official and alternate clusters contained annotations generated from both sites.

The output of the human annotation process is an ordered list of tweet clusters. Within each cluster, the tweets are sorted by temporal order (earliest to latest). The clusters themselves are sorted by the temporal order of their earliest tweet. Following the heuristic of using the most straightforward metric when defining a new task (and then

Run	Group	MAP	P30	R-prec	Type
PKUICST3	PKUICST	0.5863	0.7224	0.5727	automatic
hltcoe3	hltcoe	0.5707	0.7121	0.5660	automatic
ECNURankLib	ECNUCS	0.5529	0.7133	0.5427	automatic
PolyURun1	POLYUCOMP	0.5402	0.6994	0.5468	automatic
ICARUN2	ecnu	0.5327	0.6909	0.5435	automatic
QUQueryExp5D25T	QU	0.5155	0.6697	0.5158	automatic
HPRF1020RR	QCRI	0.5122	0.6982	0.5086	automatic
Pris2014a	BUPT_PRIS	0.5005	0.7012	0.4995	automatic
NovaRun2	NovaSearch	0.4873	0.6709	0.4950	manual
UCASRun3	UCAS	0.4703	0.6697	0.4774	automatic
SCIAI14a	SCIAITeam	0.4407	0.6503	0.4707	automatic
ERLU	ir.cs.sfsu	0.4200	0.6291	0.4493	automatic
UDInfoQE	udel_fang	0.4154	0.6115	0.4414	automatic
ICTNETRUN4	ICTNET	0.4141	0.6248	0.4369	automatic
UWMHBUT2	UWM.HBUT	0.3427	0.5121	0.4055	automatic
RM3		0.3394	0.5176	0.3963	automatic
baseline		0.3090	0.5145	0.3793	automatic
JufeLdkeAdhoc1	LDKE	0.3090	0.5145	0.3793	automatic
NewBee	zhg15	0.2745	0.4485	0.3176	automatic
udelRunAH	udel	0.1841	0.5103	0.2485	automatic
wistudt1q	wistud	0.1730	0.3018	0.2128	automatic
OSIM	BJUT	0.1708	0.3297	0.2207	automatic
SRTLAH	HU_DB	0.0505	0.2527	0.0561	automatic

Table 1: Results of the temporally-anchored ad hoc retrieval task, showing the run with the highest mean average precision (MAP) from each group. Precision at rank 30 (P30) and R-precision are also shown. Rows are sorted by MAP.

subsequently refining the metric as needed), we decided to measure cluster-based precision and recall. The measure is cluster-based in the sense that systems only receive credit for returning one tweet from each cluster—that is, once a tweet is retrieved, all other tweets in the cluster are automatically considered not relevant. From this, we can compute precision, recall, and F-score in the usual way (lacking any basis for setting the β parameter, we simply computed F_1). Since the user model assumes that a searcher will consume the entire summary, set-based metrics seemed appropriate and straightforward.

The only additional refinement is that we computed both weighted and unweighted variants of recall. In weighted recall, each cluster is assigned a weight proportional to the sum of the relevance grades from every tweet in the cluster (relevant tweets receive a weight of one and highly-relevant tweets receive a weight of two). This weighting scheme implements the heuristic that larger clusters and those containing more highly-relevant tweets are more important, and the denominator in the weighted recall computation is the sum of clusters’ weights. In unweighted recall, all clusters are considered equally important, and the denominator is simply the total number of clusters.

Note that this setup gives equal credit to retrieving *any* tweet from a cluster. Intuitively, however, this seems overly simplistic—users would certainly prefer seeing certain tweets over others, even if they contain substantively similar information. For example, users might prefer to see the earliest tweet, a tweet from the most “authoritative” user (e.g., a verified news account), or a tweet from someone close by in their network (e.g., a tweet from someone they follow). We currently do not have sufficient understanding to accu-

rately model such preferences, and thus explicitly made the decision not to tackle this challenge.

The evaluation metrics for TTG were derived from previous work referenced in Section 2.2: aspect recall [10], subtopic recall [18], and the “nugget pyramid” approach from the TREC question answering evaluations [6]. Alternative metrics we had considered include those based on gain [2] and the extension of mean average precision to graded relevance judgments [11]. After careful consideration, for this initial evaluation we decided to stick with the simpler set-based metrics, but we will consider different metrics in the future based on lessons learned.

4. RESULTS AND DISCUSSION

4.1 Temporally-Anchored Ad Hoc Retrieval

For the temporally-anchored ad hoc retrieval task, NIST received a total of 75 runs from 21 groups. Table 1 shows the run with the highest mean average precision (MAP) from each group. Precision at rank 30 (P30) and R-precision are also shown. Rows in the table are sorted by MAP. In computing these metrics, both the “relevant” and “highly relevant” grades are considered relevant.

For reference, we provided two baselines, also shown in Table 1: The “baseline” condition is simply the raw output of the API with queries as bags of words. The “RM3” condition is our reference implementation of the RM3 variant of relevance models [5, 1].¹ For RM3, we extracted the top 20 feedback terms from the top 50 tweets, which is interpolated with the original query model with a weight of 0.5.

¹<http://twittertools.cc/>

Run	Group	recall	recall ^w	precision	F ₁	F ₁ ^w	Type
TTGPKUICST2	PKUICST	0.3698	0.5840	0.4571	0.3540	0.4575	automatic
EM50	QCRI	0.2867	0.4779	0.4150	0.2546	0.3815	automatic
hltcoeTTG1	hltcoe	0.4029	0.5915	0.3407	0.2760	0.3702	automatic
QUTmpDecayTTgCL	QU	0.3277	0.5167	0.3236	0.2440	0.3300	automatic
PrisTTG2014b	BUPT_PRIS	0.4231	0.6137	0.2730	0.2461	0.3093	automatic
SRTD	HU_DB	0.0868	0.2764	0.5798	0.1324	0.2907	automatic
3unique0	uog_twteam	0.2522	0.4374	0.2558	0.1957	0.2744	automatic
udelRunTTG1	udel	0.1873	0.3645	0.2793	0.1774	0.2669	automatic
UDInfoMMRW5	udel_fang	0.0900	0.2191	0.5709	0.1338	0.2577	automatic
wistudt2bd	wistud	0.1111	0.3075	0.2827	0.1251	0.2305	automatic
SCIAI3cm4aTTG	SCIAITeam	0.0655	0.1941	0.4992	0.0921	0.2171	manual
ICTNETAP4	ICTNET	0.2528	0.4836	0.1702	0.1559	0.2072	automatic
JufeLdkeSum2	LDKE	0.4294	0.6156	0.0861	0.1180	0.1307	automatic

Table 2: Results of the tweet timeline generation (TTG) task, showing the run with the highest weighted F₁ score from each group. Columns show recall (unweighted and weighted), precision, and F₁ (unweighted and weighted); the ^w superscript indicates the weighted variant of the metric. Rows are sorted by weighted F₁ score.

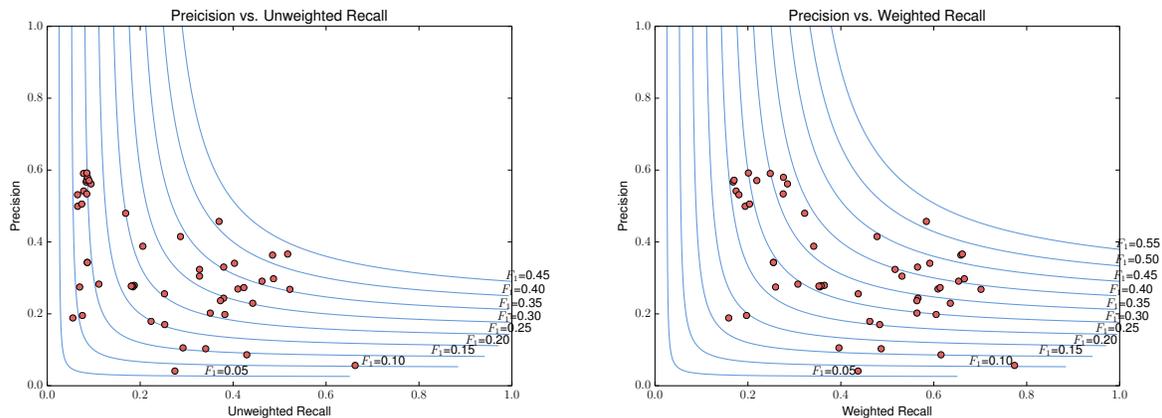


Figure 2: Scatter plots showing precision vs. unweighted recall (left) and precision vs. weighted recall (right) for all runs, overlaid with iso-F₁ contours.

Both baselines were submitted as normal runs and hence were part of the judgment pool.

These results show that participants are submitting effective runs overall. Using the baselines to calibrate, there are many more teams beating the baselines than in previous years. This suggests that the ad hoc retrieval task is perhaps becoming “too easy”.

4.2 Tweet Timeline Generation

In total, 13 groups submitted 50 runs to the tweet timeline generation task. Each topic averaged 194 relevant documents. In the official clusters, assessors formed an average of 89 clusters per topic (and each cluster averaged 2.2 tweets). Results based on these annotations are shown in Table 2, which contains the run with the highest weighted F₁ score from each group. The columns show unweighted recall, weighted recall (indicated by the ^w superscript), precision, F₁ with unweighted recall, and F₁ with weighted recall (indicated by the ^w superscript). Rows are sorted by weighted F₁ score. In computing these metrics, both the “relevant” and “highly relevant” grades are considered relevant.

Recognizing that systems make different choices with respect to balancing precision and recall, it is illustrative to visualize the tradeoffs in a scatter plot. Figure 2 shows precision vs. unweighted recall (left) and precision vs. weighted recall (right) for all runs. Iso-F₁ contours are plotted in blue; points on the same contour line have the same F₁ score, but with different precision/recall tradeoffs.

What is the effect of adding the cluster weights? Figure 3 shows a scatter plot of the weighted F₁ score vs. the unweighted F₁ score for all submitted runs (where each point represents a run). We see that for most runs, the two scores are highly correlated, but there are a number of runs where the weighted F₁ score is quite a bit higher than the unweighted F₁ score.

To verify the stability and reliability of the evaluation, we performed two sets of analyses: the first concerns the number of missing judgments in evaluating the TTG runs. Recall that our cluster annotation workflow starts with relevant tweets from the NIST judgment pools, created via the process described in Section 3.1.

In total, the TTG runs returned a combined 81,726 unique

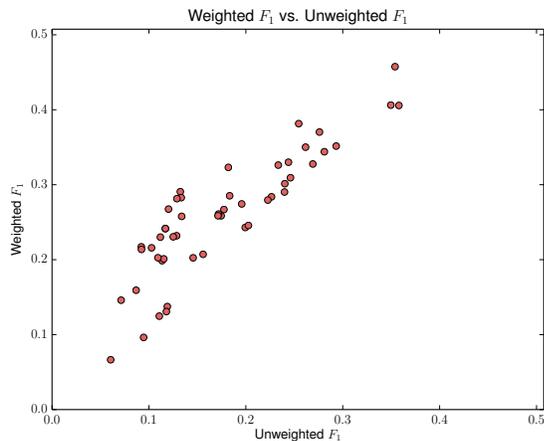


Figure 3: Scatter plot showing weighted F_1 score vs. unweighted F_1 score for all runs.

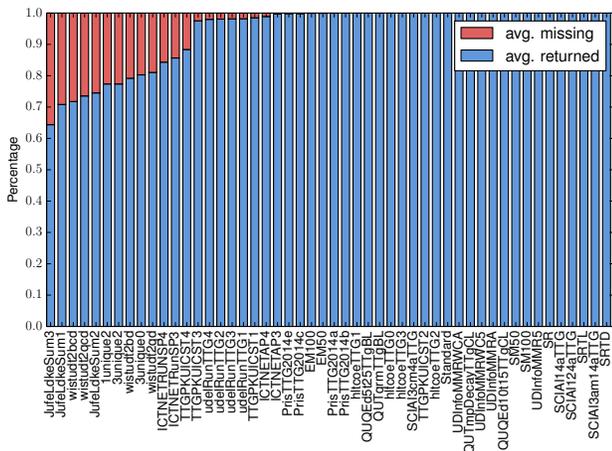


Figure 4: Visualization of missing judgments. Bars show, for each run, the fraction of returned tweets missing explicit judgments (averaged across topics).

tweets. Of these, 37,449 (45.8%) were absent from the judgment pools and therefore lacked explicit judgments. However, these missing tweets were concentrated in a relatively small number of runs: this is shown in Figure 4, which plots the fraction of tweets that are missing judgments (averaged across topics). For 18 runs (out of 50), we did not observe *any* missing judgments. Of the runs with missing judgments, 19 were missing less than 5%. Note that these bars indicate the fraction of missing judgments, which obscures the absolute number since some systems return answers that are on average longer. For example, the run that was most severely impacted had an average of 412 missing judgments per topic (out of an average of 1000 tweets returned).

There are two possible interpretations of these results: the first is that systems are building summaries without an explicit ad hoc retrieval stage (whose outputs can contribute to the ad hoc judgment pools); the second is that the systems are generating summaries that are so long that missing judgments are the result of a shallow pool depth. The TTG runs

Metric	Rank Swaps	Kendall’s τ
precision	30	0.951
unweighted recall	27	0.956
weighted recall	22	0.964
unweighted F_1	46	0.925
weighted F_1	90	0.853

Table 3: Count of rank swaps and Kendall’s τ correlation based on the official and alternate judgments for each metric.

with the largest numbers of unjudged results appear to have been generated without corresponding ad hoc runs. These runs are also very verbose for the most part. On the whole, it is the case that the number of missing judgments is positively correlated with run length, but the *fraction* of results that are unjudged does not show this relationship. From this, there appears to be evidence in favor of the hypothesis that the shallow pool depth contributed to missing judgments. However, without explicit knowledge of each team’s operational relationship between its ad hoc and TTG runs, it is hard to draw firmer conclusions.

Our second analysis focuses on the stability of the evaluation with respect to assessor differences. The judgment of whether two tweets contain “substantively similar information” is likely to have high variability across assessors, which would yield substantially different clusters. We would like to determine to what extent this impacts our ability to make system comparisons, i.e., that system X is more effective than system Y [14]. Note that as is standard in IR meta-evaluation, differences in absolute scores are not worrisome, so long as system comparisons are stable.

As part of our annotation process, we have obtained two independent sets of cluster annotations for all topics. Figure 5 shows scores based on the official judgments and the alternate judgments for each of the five metrics in Table 2. Results are sorted by scores based on the official judgments. The official set averaged 89 clusters per topic, while the alternate judgments averaged 73 clusters per topic, which indicates that humans perform the clustering task at different levels of granularity. We see that the rankings produced by both sets of judgments are highly correlated, with the exception of weighted F_1 , which shows a number of runs that yield different comparisons with respect to the two sets of judgments. Furthermore, except for the weighted variants, the absolute values of the metrics are also quite similar.

In Table 3, we tally the number of rank swaps for each metric. A rank swap is a pairwise comparison where according to one set of judgments, run A scores higher than run B , but according to another set of judgments, run B scores higher than run A . There are a total of $(50 \times 49)/2 = 1225$ pairwise comparisons. Table 3 also shows the Kendall’s τ correlation between rankings induced by the two different sets of judgments. With the exception of weighted F_1 , we do not observe many rank swaps, as confirmed by the high rank correlations.

Finally, we show histograms of the rank swaps for unweighted and weighted F_1 in Figure 6 binned by absolute score differences. Such an analysis is informative because we are less concerned with rank swaps in which the absolute score differences between the two conditions are small. For

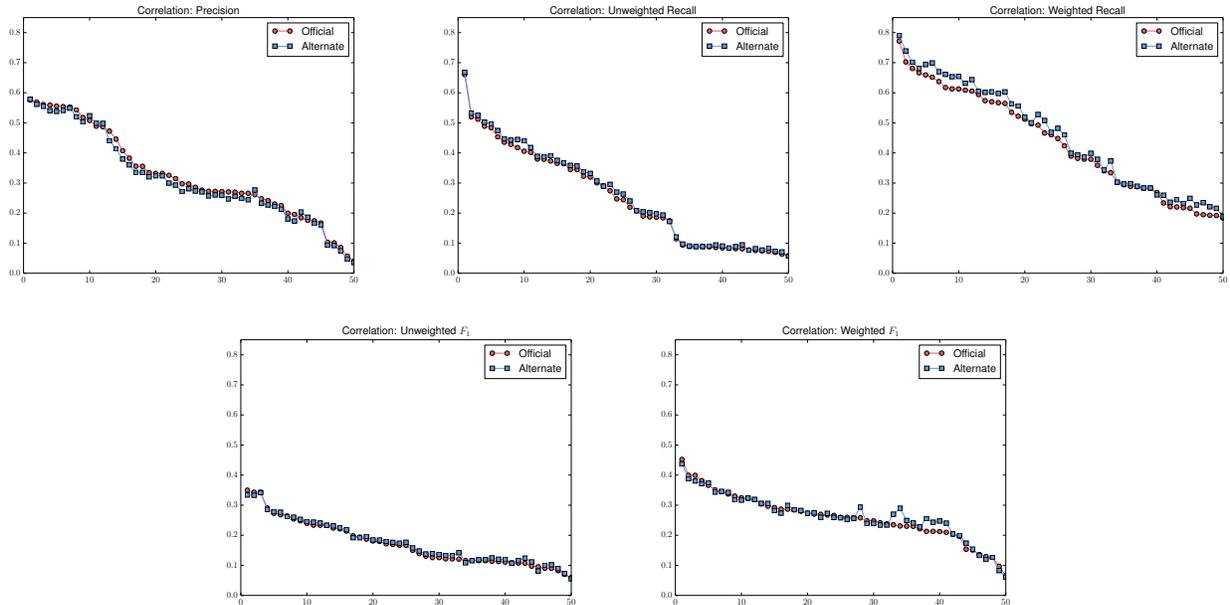


Figure 5: Comparison between scores based on the official judgments and the alternate judgments for precision, unweighted recall, weighted recall, unweighted F_1 , and weighted F_1 . Runs are sorted by score based on the official judgments in descending order.

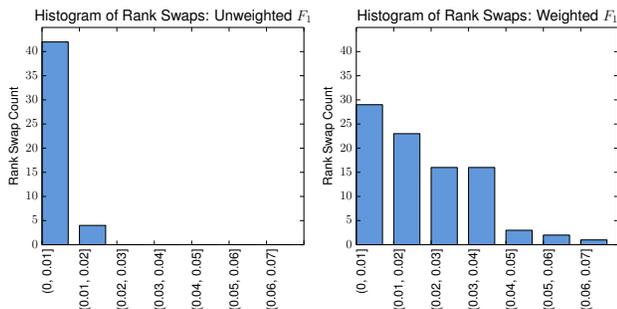


Figure 6: Histogram of rank swaps for unweighted and weighted F_1 binned by absolute score differences.

space considerations, we only show these histograms for the two metrics with the lowest Kendall’s τ correlations. We do see a number of rank swaps with large score differences, which correspond to the jagged portion of the blue line in the weighted F_1 plot in Figure 5. For the other histograms, most of the rank swaps occur with small absolute score differences (which are not of major concern).

Based on these analyses, we can conclude that the evaluation methodology for TTG is stable with respect to assessor differences. Evaluation using independent cluster annotations give rise to system comparisons that are consistent, which strengthens our confidence in the results.

5. CONCLUSION

The tweet timeline generation task this year represents a serious attempt in the TREC Microblog track to move be-

yond ad hoc retrieval. In the design of the track, we have explicitly taken a conservative approach in changing only one aspect of the evaluation at a time (last year, it was the introduction of the evaluation-as-a-service model; this year, the introduction of TTG). We believe that this approach provides continuity and allows participants to build on lessons learned in previous years in an incremental fashion. The track will continue in TREC 2015, where we look forward to continue pushing the state of the art in information retrieval techniques applied to microblogs.

6. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under IIS-1217279 and IIS-1218043. Any opinions, findings, conclusions, or recommendations expressed are those of the authors and do not necessarily reflect the views of the sponsor. We are grateful to Ellen Voorhees and the assessors at NIST for making TREC possible.

7. REFERENCES

- [1] N. Abdul-Jaleel, J. Allan, W. B. Croft, F. Diaz, L. Larkey, X. Li, D. Metzler, M. D. Smucker, T. Strohman, H. Turtle, and C. Wade. UMass at TREC 2004: Novelty and HARD. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC 2004)*, Gaithersburg, Maryland, 2004.
- [2] C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 659–666, Singapore, 2008.

- [3] H. T. Dang, J. Lin, and D. Kelly. Overview of the TREC 2006 question answering track. In *Proceedings of the Fifteenth Text REtrieval Conference (TREC 2006)*, pages 99–116, Gaithersburg, Maryland, 2006.
- [4] D. Harman. *Information Retrieval Evaluation*. Morgan & Claypool Publishers, 2011.
- [5] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 120–127, New Orleans, Louisiana, 2001.
- [6] J. Lin and D. Demner-Fushman. Will pyramids built of nuggets topple over? In *Proceedings of the 2006 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2006)*, pages 383–390, New York, New York, 2006.
- [7] J. Lin and M. Efron. Evaluation as a service for information retrieval. *SIGIR Forum*, 47(2):8–14, 2013.
- [8] J. Lin and M. Efron. Overview of the TREC-2013 Microblog Track. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, Gaithersburg, Maryland, 2013.
- [9] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff. Overview of the TREC-2011 Microblog Track. In *Proceedings of the Twentieth Text REtrieval Conference (TREC 2011)*, Gaithersburg, Maryland, 2011.
- [10] P. Over. TREC-6 interactive report. In *Proceedings of the Sixth Text REtrieval Conference (TREC 1997)*, Gaithersburg, Maryland, 1997.
- [11] S. E. Robertson, E. Kanoulas, and E. Yilmaz. Extending average precision to graded relevance judgments. In *Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*, pages 603–610, Geneva, Switzerland, 2010.
- [12] I. Soboroff, I. Ounis, C. Macdonald, and J. Lin. Overview of the TREC-2012 Microblog Track. In *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, Gaithersburg, Maryland, 2012.
- [13] K. Tao, C. Hauff, and G.-J. Houben. Building a microblog corpus for search result diversification. In *Proceedings of the 9th Asia Information Retrieval Societies Conference (AIRS 2013)*, pages 251–262, Singapore, 2013.
- [14] E. M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, pages 315–323, Melbourne, Australia, 1998.
- [15] E. M. Voorhees and H. T. Dang. Overview of the TREC 2005 question answering track. In *Proceedings of the Fifteenth Text REtrieval Conference (TREC 2005)*, Gaithersburg, Maryland, 2005.
- [16] E. M. Voorhees, J. Lin, and M. Efron. On run diversity in “evaluation as a service”. In *Proceedings of the 37th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2014)*, pages 959–962, Gold Coast, Australia, 2014.
- [17] Y. Wang and J. Lin. The impact of future term statistics in real-time tweet search. In *Proceedings of the of the 36th European Conference on Information Retrieval (ECIR 2014)*, pages 567–572, Amsterdam, The Netherlands, 2014.
- [18] C. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 10–17, Toronto, Canada, 2003.