Pooling for User-Oriented Evaluation Measures

Gaurav Baruah
Computer Science
University of Waterloo
gbaruah@uwaterloo.ca

Adam Roegiest
Computer Science
University of Waterloo
aroegies@uwaterloo.ca

Mark D. Smucker
Management Sciences
University of Waterloo
mark.smucker@uwaterloo.ca

ABSTRACT

Traditional TREC-style pooling methodology relies on using predicted relevance by systems to select documents for judgment. This coincides with typical search behaviour (e.g., web search). In the case of temporally ordered streams of documents, the order that users encounter documents is in this temporal order and not some predetermined rank order. We investigate a user oriented pooling methodology focusing on the documents that simulated users would likely read in such temporally ordered streams. Under this user model, many of the relevant documents found in the TREC 2013 Temporal Summarization Track's pooling effort would never be read. Not only does our pooling strategy focus on pooling documents that will be read by (simulated) users, the resultant pools are different from the standard TREC pools.

Categories and Subject Descriptors

H.3.4 [**Systems and Software Performance evaluation**]: Efficiency and Effectiveness

Keywords

Evaluation; Pooling; User models;

1. INTRODUCTION

Traditionally, pooling mechanisms select documents based upon predicted relevance by individual systems. This coincides with the fact that almost all search systems present their results in a Search Engine Result Page (SERP), wherein a list of documents ranked by their likelihood of relevance, is presented to the user. SERPs correspond to the Probability Ranking Principle, which states that results should be ordered by their likelihood of being relevant for optimal retrieval. Recent research has given impetus to generating user models for effective evaluation of systems [6], although most research still focuses on a ranked list over which a model of user behavior is inferred.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
ICTIR'15, September 27–30, Northampton, MA, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3833-2/15/09 ...$15.00.
DOI: http://dx.doi.org/10.1145/2808194.2809493 .

The continued growth of real-time information access platforms (e.g, Facebook, Twitter and other social media), has resulted in the need for retrieving information from large dynamic streams of documents. Increasing effort is being made to design and evaluate such systems. Primarily such efforts have focused on filtering relevant information from temporally ordered streams of documents. In particular, the TREC Temporal Summarization [2] track (retrieving sentences) and the Microblog [12] track (retrieving microblog posts), have investigated the evaluation of such filtering systems. While the Microblog track has typically used standard IR evaluation measures, the Temporal Summarization track (TST) introduces evaluation measures that incorporate notions of latency, length, as well as relevance of returned updates (sentences). Both tracks have used score-based pooling, a standard TREC practice [14, 2], where documents from each system are ordered by a system assigned score and the top-k documents (or updates) are taken from each run and evaluated by assessors, where k is the pool depth.

For many retrieval and filtering tasks standard score-based pooling techniques are effective and (generally) robust evaluation mechanisms. Score-based pooling also coincides with how users will typically use such systems. For example, a user performing a search using Google would typically expect the most likely to be relevant documents to be near the top of the result page.

A different user model may be more suitable for temporally ordered information. A user that is browsing Twitter will likely expect the posts to be ordered in roughly reverse chronological order. In fact, there was a large outcry when Facebook changed the default ranking of updates to be their own internal scoring mechanism rather than a temporal ordering [9]. Accordingly, we posit that score-based pooling is likely an unsuitable mechanism for the creation of evaluation pools for filtering tasks involving temporally ordered documents.

We employ a user model for streaming information access [3] and identify which updates were read by simulated users for the runs submitted to TST 2013. We observe that a large proportion of these simulated users may not read all the relevant updates (Figure 1). Additionally, the pool may have been too shallow as McCreadie et al. [11] found that there was little overlap between the Temporal Summarization pool and their proposed system, causing them to procure additional assessments.

We believe that it may be better to focus on those updates that can be identified as most likely to be read by (simulated) users. In this work, we present a method of

pooling that selects updates by computing the probabilities of an update being read based upon an underlying user model. It is our hypothesis that our method will allow pools to be created that consist of documents that have a higher probability of being read by (simulated) users and are more reflective of actual user experience and expectation. Accordingly, we create pools by selecting top-k updates based on their probability of being read. Furthermore, we show that the overlap between our top-k probability-based pool and the top-k score-based pool of the 2013 Temporal Summarization track, is quite low; only 60% of the relevant updates are found in the probability-based pools even when the 1000 most likely to be read updates are selected from each system. Indicating that the TST evaluation may be representative of what is being read by (simulated) users. We also show that, depending on how the probability mass is distributed over the updates, the number of judgments required to construct a test collection differs considerably (Section 4).

This investigation lays the groundwork for further exploration of pooling methods that utilize user models. Test collections based on such user-oriented pooling may benefit user-model oriented evaluation measures and move beyond the simple rank based evaluation for complex retrieval tasks.

2. USER MODELS AND POOLING

Historically, TREC has investigated retrieval as well as evaluation methods for filtering tasks [10], Topic Detection and Tracking [1] and more recently for Temporal Summarization [2]. The evaluation measures used for these tasks assume a simplified user model and use set-based (similar to Precision and Recall) evaluation criteria. They do acknowledge that criteria, like recency and novelty [2, 1] as well as verbosity [2], should be considered for evaluation of systems filtering information from a stream. Evaluation measures have attempted to subsume these evaluation criteria into a precision/recall analogous scoring system.

The Temporal Summarization Track [2] (TST) in particular, requires that the participating runs return updates (sentences) that are likely relevant to given news topics, from a time-ordered document stream. These updates are returned at any instant within a specified time period of interest (typically 10 days) for the topic. The updates have associated timestamps noting when the system released the updates. An end user presumably reads the complete stream of updates returned for a topic. The track defines measures, Expected Latency Gain and Latency Comprehensiveness (analogous to precision and recall respectively), that evaluate results from a system at the end of the interest period.

In recent work, Modeled Stream Utility [3] (MSU) is proposed as measure for evaluating information retrieval over a streaming corpus. MSU is a user focused evaluation that incorporates a model of a user browsing the results of a system producing a stream of updates. In effect, for MSU, a simulated user alternates between spending some time reading results and some time away from the system, for as long as the user is interested in the topic. A more interested user spends more time on reading and less time away from the system and vice-versa.

For each simulated user, MSU identifies the updates read, and increments gain when a relevant update is read by the user. Thus, across all simulated users, there could be some updates that have a higher likelihood of being read than others. This has interesting ramifications for pooling. For a fairer evaluation of systems, considering most frequently read updates as part of the test collection may result in a more meaningful evaluation. Figure 1 shows that for topic 10 of TST 2013, relevant updates are not necessarily the most often read updates; all other topics of TST 2013 have similar proportions. After simulating 10,000 users, only 5 relevant updates, across all runs, were read by all users (with 25 updates read by 99% of the users). It is worthy of note that 546 relevant updates (out of a total 1616 returned across all runs) were read by less than 1% of the simulated users.

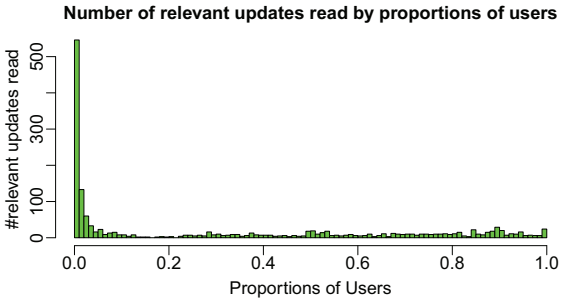

Figure 1: The number of relevant updates read by proportions of simulated users, for topic 10 from the Temporal Summarization track, at TREC 2013.

The MSU user model forms the core of our methodology. Accordingly, we use the "reasonable" parameters [3], such that a user population spends on average 2 minutes (std.dev. 1 minute), every 3 hours (std.dev. 1.5 hours), for reading updates produced by a system. Users sampled from this population may spend different durations of time for reading sessions and times spent away than the population mean. The reading speed of a user helps to determine which updates are read at each user session. We sample reading speeds from a reading speed distribution described in Clarke and Smucker [7]. By simulating 10,000 users with this user model, we construct a probability distribution over the updates indicating their likelihood of being read.

3. GENERATING UPDATE PROBABILITIES

We present two possible methods of computing the probability of an update being read. Consider a set $U = \{u_i | 1 \leq i \leq m\}$ of m simulated users, a set $D = \{d_j | 1 \leq j \leq n\}$ of n updates submitted by a given system for a particular topic, and let $read_{ij} = \{0, 1\}$ indicate if the update d_j was read by user u_i. Then $P(d_j)$, the probability of an update d_j being read can be computed as:

1. A *balanced* notion of probability: average the probabilities for d_j across all simulated users.

$$P(d_j) = \frac{1}{|U|} \sum_i \frac{1}{\sum_q read_{iq}}, (1 \leq q \leq n) \qquad (1)$$

2. An *unbalanced* notion of probability: the ratio of the number of users that have read d_j, over, the total number of documents read by all simulated users.

$$P(d_j) = \frac{\sum_i read_{ij}}{\sum_i \sum_q read_{iq}}, (1 \leq q \leq n) \qquad (2)$$

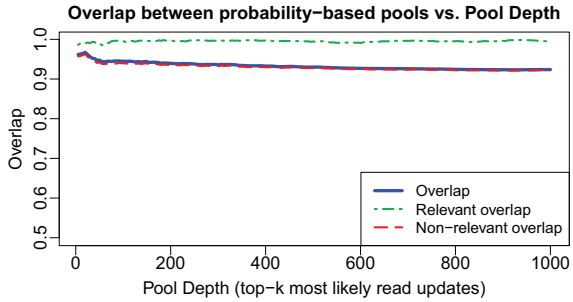

Figure 2: Overlap between the top-k probability-based pools created with balanced and unbalanced probabilities.

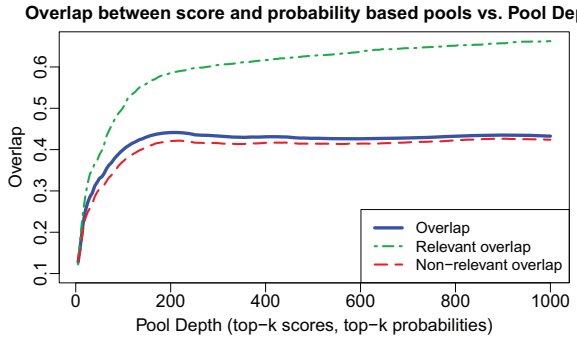

Figure 3: Overlap between the top-k score-based and top-k unbalanced probability-based pools.

Simulated users who read more are weighted more heavily in the *unbalanced* computation, whereas all simulated users are weighted equally in the *balanced* computation.

Figure 2 depicts the overlap between the two probability generation methods when depth pooling is used. Due to the high overlap between the two methods, it is likely the case that while absolute probabilities may differ, many updates are shared.

Due to the similarity between the balanced and unbalanced probability-based pools, we expect that there is little meaningful difference between them. Accordingly, the remainder of this work uses the unbalanced probability generation method due to the simplicity of the underlying theory and its implementation.

3.1 Score-based vs Probability-based Pools

Figure 3 clearly illustrates that top-k score-based pooling and top-k probability-based pooling do not produce identical pools at same depths. In general, overlap between the probability-based pool and the TREC style score-based pool, does not exceed 45%. For relevant updates, the overlap does not exceed 70%, even for extremely large k (depths).

This indicates that a non-trivial portion of the probability-based pool is different from the score-based pool. In conjunction with the proportions from Figure 1, it is clearly seen that these pools are different enough to warrant further investigation into probability-based pooling. This lack of overlap lends further credence to our hypothesis that score-based pooling does not accurately reflect user experience.

4. GLOBAL AND LOCAL PROBABILITIES

The MSU user-model induces a probability of being read on every update. Top-k probability-based pooling using balanced (or unbalanced) probabilities is one method to pool using these probabilities. Since we have probability distributions for each run, another method to construct a pool would be to select the most likely to be read updates until a specified level of probability mass over the updates is covered for each run. We call this method of pooling "local probability pooling," since the probabilities are "local" to each run. This ensures that each run has the same amount of probability mass covered. This is in contrast to depth (top-k) pooling, where, the systems may be disproportionately represented in the pool if each system returned a different number of updates. In fact, the number of updates returned by TST 2013 runs varies from 110 to 2.8 million updates per run.

An alternate method would be to average these "local" probabilities into a "global" probability distribution, such that the probability of an update not present in a run is assumed to have a local probability of 0. Pooling proceeds by ordering updates by their global probability and selecting documents in descending order until a desired probability mass is achieved. The goal of global pooling is to focus assessment effort on those updates which are most likely to be read across all runs.

As illustrated by Figure 4, the overlap between global and local pooling techniques grows in an approximately linear fashion as more probability mass is covered. The high levels of overlap likely correspond to the fact that the local pooling technique produces a much larger pool than global pooling (Figure 5). The larger pool size is likely due to the fact that runs that returned thousands of updates have a much lower local probability mass per update (requiring more updates to cover a specified local probability mass). Whereas the global probability mass per update tends to be concentrated in the updates that are most likely to be read in all runs. Figure 6, shows that different local probability masses are covered across runs for a given global probability mass cover. Specifically, the shorter runs (hundreds of updates) appear on the left hand side of the plot, while the longer runs (several thousands of updates) appear on the right hand side. This indicates that covering a specific amount of local probability mass for every run may require including a large number of updates from longer runs into the pool.

Using global probability for pooling is more likely to produce much smaller sized pools than using local probability, for a specified global probability mass cover. The effect of pool size, in the case of global and local pooling, for assessment, on the re-usability and robustness of the resulting test collection is a next step in this research.

5. DISCUSSION AND FUTURE WORK

Much of the work in this paper has revolved around exploring the space around a user-model induced probability-based pooling. Research has primarily focused on depth-based pooling using ranked retrieval to varying degrees [4, 5, 8, 10, 13, 14, 15]. We have shown two different methods of generating probabilities for updates as well as two novel methods for pooling based upon those probabilities. Further investigation into the applicability of this new pooling technique is required, especially as it relates to user-oriented evaluation measures, e.g. MSU [3].

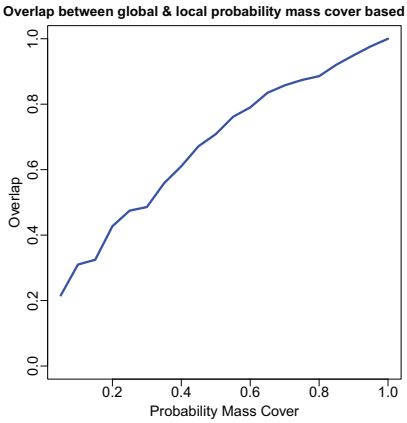

Figure 4: Overlap between pools created using global and local probability mass pooling strategies when both target the same probability mass cover.

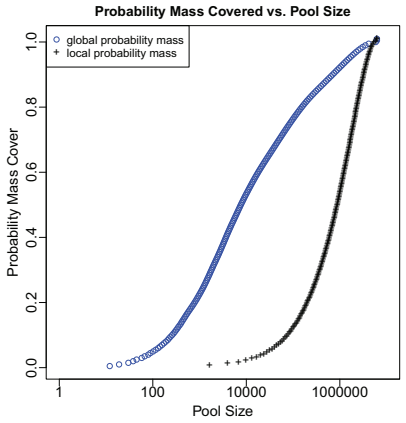

Figure 5: Comparison of pool size and probability mass covered for both local and global probability mass pooling strategies.

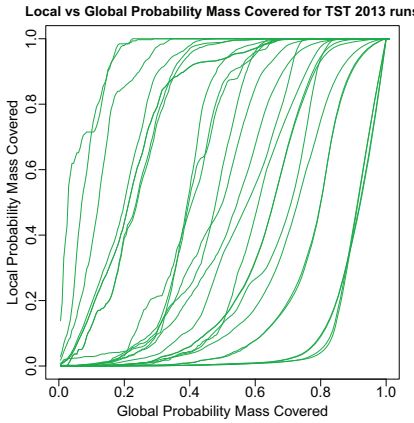

Figure 6: Average local probability mass covered for all 26 runs submitted to TST 2013, when global probability mass pooling is performed.

Additional analysis is required to determine any bias or deficiencies that may be present in this new pooling strategy. The applicability of analyses created for score-based pooling [8, 4] to this new type of pooling remains uncertain. Indeed, pooling is not an end unto itself and further work must be done to verify that the pooling methodology presented herein creates reusable test collections that provides accurate estimates for user-oriented effectiveness measures.

A major limitation of this work is that only a single set of parameters (representing so-called "reasonable users" [3]) is used for this user model. User studies are necessary to observe the behavior of real users and determine parameter settings that reflect actual practice.

6. CONCLUSIONS

We have presented a novel pooling methodology using a probabilistic approach based on a user model. Furthermore, we have shown that the pools based upon the presented methodology are different from scored-based pools constructed as per the TREC standard operating procedure. We have focused, in particular, on two forms of probabilistic pooling, such that, updates that are most likely to be read are included in the pool. By using probabilities tailored to each participant system, we can choose to cover higher (local) probability mass for each system, or a higher global probability mass overall. The trade-off becomes one of size where pools created by global probability mass sampling result in smaller pools and thus less documents to judge. Though it is typically the case that the overlap between local and global tends to be quite high. The effect of these different pooling techniques on various user-centered evaluation measures is left as an interesting avenue of future research.

7. ACKNOWLEDGMENTS

This work was made possible by the facilities of SHARCNET (www.sharcnet.ca) and Compute/Calcul Canada, and was supported in part by NSERC, in part by a Google Founders Grant, and in part by the University of Waterloo.

8. REFERENCES

[1] J. Allan, R. Gupta, and V. Khandelwal. Temporal Summaries of New Topics. In *SIGIR*, pp. 10–18, 2001.

[2] J. Aslam, F. Diaz, M. Ekstrand-Abueg, V. Pavlu, and T. Sakai. TREC 2013 Temporal Summarization. In *TREC*, 2013.

[3] G. Baruah, M. D. Smucker, and C. L. A. Clarke. Evaluating Streams of Evolving News Events. In *Proc. SIGIR*, 2015.

[4] C. Buckley, D. Dimmick, I. Soboroff, and E. Voorhees. Bias and the Limits of Pooling. In *Proc. SIGIR*, pp. 619–620, 2006.

[5] B. Carterette, J. Allan, and R. Sitaraman. Minimal Test Collections for Retrieval Evaluation. In *Proc. SIGIR*, pp. 268–275, 2006.

[6] C. L. A. Clarke, L. Freund, M. D. Smucker, and E. Yilmaz. Report on the SIGIR 2013 Workshop on Modeling User Behavior for Information Retrieval Evaluation (MUBE 2013). *SIGIR Forum*, 47(2):84–95, Jan. 2013.

[7] C. L. A. Clarke and M. D. Smucker. Time Well Spent. In *IIiX*'14, pp. 205–214, 2014.

[8] G. V. Cormack and T. R. Lynam. Power and Bias of Subset Pooling Strategies. In *Proc. SIGIR*, pp. 837–838, 2007.

[9] M. Johanson. Facebook changes: Users overwhelmingly 'dislike' new news feed and ticker, Sept. 2011. International Business Times.

[10] D. D. Lewis. The TREC-4 Filtering Track. In *Proc. TREC-4*, 1995.

[11] R. McCreadie, C. Macdonald, and I. Ounis. Incremental Update Summarization: Adaptive Sentence Selection Based on Prevalence and Novelty. In *CIKM*, pp. 301–310, 2014.

[12] I. Soboroff, I. Ounis, J. Lin, and I. Soboroff. Overview of the TREC-2012 Microblog Track. In *TREC*, 2012.

[13] K. Sparck-Jones and C. Van Rijsbergen. Report on the need for and provision of an "ideal" Information Retrieval Test Collection. 1975.

[14] E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval*. The MIT Press, 2005.

[15] J. Zobel. How Reliable are the Results of Large-scale Information Retrieval experiments? In *SIGIR*, pp. 307–314, 1998.