Evaluating Streams of Evolving News Events

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

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

Charles L. A. Clarke
Computer Science
University of Waterloo
claclark@plg.uwaterloo.ca

ABSTRACT

People track news events according to their interests and available time. For a major event of great personal interest, they might check for updates several times an hour, taking time to keep abreast of all aspects of the evolving event. For minor events of more marginal interest, they might check back once or twice a day for a few minutes to learn about the most significant developments. Systems generating streams of updates about evolving events can improve user performance by appropriately filtering these updates, making it easy for users to track events in a timely manner without undue information overload. Unfortunately, predicting user performance on these systems poses a significant challenge. Standard evaluation methodology, designed for Web search and other adhoc retrieval tasks, adapts poorly to this context. In this paper, we develop a simple model that simulates users checking the system from time to time to read updates. For each simulated user, we generate a trace of their activities alternating between away times and reading times. These traces are then applied to measure system effectiveness. We test our model using data from the TREC 2013 Temporal Summarization Track (TST) comparing it to the effectiveness measures used in that track. The primary TST measure corresponds most closely with a modeled user that checks back once a day on average for an average of one minute. Users checking more frequently for longer times may view the relative performance of participating systems quite differently. In light of this sensitivity to user behavior, we recommend that future experiments be built around clearly stated assumptions regarding user interfaces and access patterns, with effectiveness measures reflecting these assumptions.

Categories and Subject Descriptors: H.3.4 [Information Storage and Retrieval]: Systems and Software — *Performance evaluation (efficiency and effectiveness)*

Keywords: search; streams; evaluation

SIGIR'15, August 09 - 13, 2015, Santiago, Chile.

ACM 978-1-4503-3621-5/15/08 ...$15.00.
DOI: http://dx.doi.org/10.1145/2766462.2767751.

1. INTRODUCTION

On December 4, 2012, Typhoon Bopha made landfall on the eastern coast of the Philippine island of Mindanao. The strongest typhoon in the island's recorded history, the storm system had formed in the Pacific in late November, growing into a tropical storm by November 28th, and into a severe tropical storm by November 30th. On December 1st, as the storm moved west towards the island, it rapidly intensified into a category 4 typhoon. Two days later, only miles from shore, it reached category 5, a super typhoon, with winds up to 150 mph. At 4:45AM it hit. Heavy rains and waves exceeding 25 feet washed across villages and towns, destroying homes, flooding roads, and cutting communications. Initial reports limited the death toll to one, but by the next day it had increased to over 200, mostly killed by flooding and debris. Over the next week, the number of fatalities grew to a final total of over 600, with more than 200,000 people displaced or otherwise impacted.

In this paper we consider how to evaluate a search engine that aggregates a stream of information and provides a means for people to obtain the latest, most relevant information regarding an event of interest. As time permits, a user of such a search engine will check back multiple times in order to find out new information about the event.

We envision that there will be a wide variety of users of such a search engine. Government and health care workers, as well as people in the immediate vicinity, would likely need to learn of new information soon after it becomes available. People interested in the event, but not directly affected, would likely have a less urgent need for new information. Some people would be willing to consume a large amount of content in the hope of not missing any relevant information, while others may desire a minimal amount of content.

One way of designing an information retrieval system for streams of evolving events would be to allow users to visit and query the system whenever they want the latest information. In this type of system, the user *pulls* information. An alternative approach would be to create a system that *pushes* select pieces of information to users that have set up a search query or profile. In either case, users would want to control the frequency and duration of their interaction with the system. In the pull scenario, the user controls the frequency by how often they visit and controls the duration by how long they consume content during a visit's session. In the push scenario, the user would need to specify up front how often content should be sent to them as well as the amount of content to be sent.

In this paper, we propose a new effectiveness measure for the evaluation of retrieval systems that produce streams of content for evolving events. We call this measure *modeled stream utility* (MSU). This new effectiveness measure models users in terms of the time spent away from the system and the time spent with the system. In effect, we model users in terms of their frequency and duration of interaction with the system, which is a model applicable to either push or pull variants of this type of retrieval system. We measure utility in the number of relevant information nuggets the user is likely to find. Search users naturally vary, and we model this variation as well by modeling a population of users and simulating many different users' interaction with the retrieval system. A significant advantage of MSU as an effectiveness measure is that by making it user-centered, the measure has the potential to be easily calibrated using recorded user behavior.

We demonstrate MSU using the TREC 2013 Temporal Summarization Track [4]. Participating groups in the temporal summarization track (TST) take a large stream of content and attempt to produce a filtered stream of content relevant to a given search topic. For each topic, the track has determined a set of relevant nuggets of information and has judged a pool of content and determined which nuggets are present in the content.

To evaluate the runs submitted to TST, we use MSU in a scenario where users pull information from the retrieval system. MSU simulates user interaction with the retrieval system. When a simulated user visits the retrieval system and queries it for a given search topic, the user is shown the most recently filtered content in reverse chronological order. In TST, filtered content consists of sentence length material extracted from a stream of larger documents. After a simulated user has read an element of content, we measure the user's gain in terms of the number of nuggets the user is likely to have found relevant in the content. If a user has read a nugget earlier, MSU assumes that a repeated nugget will not be found relevant and has a gain value of zero. The temporal summarization track has estimated when nuggets are first known in the aggregate stream, and in the MSU simulation, nuggets delivered late to a user are less likely to be found relevant.

In addition to demonstrating MSU applied to the TREC 2013 temporal summarization track, we compare MSU to the track's existing primary measure, ELG, as well as conduct a parameter sweep over a range of the MSU user model's parameters to better understand how the measure behaves. We find that:

- When we set MSU's parameters to that of a reasonably interested user who visits the system for 2 minutes every 3 hours on average, MSU's ranking of TST submitted runs is significantly different from TST's primary measure, ELG (Kendall's $\tau = 0.47$).

- Of the parameter settings included in our sweep, ELG correlates best with a model of users who visit for only 1 minute per day (Kendall's $\tau = 0.62$). In terms of content consumed, a minute-long visit amounts to a user reading approximately 4-5 sentences.

- The ranking of systems produced by the MSU effectiveness measure appears to be most sensitive to the amount of content consumed by the simulated users.

Given that the amount of content consumed appears to affect system rankings the most, this result has implications for the design of retrieval systems that generate streams of information updates. The temporal summarization track gave participating groups the task of filtering a stream, but different users are going to be interested in different amounts of material. Rather than filter a stream, retrieval systems for evolving events may be better designed as best-match rankers that must balance recency and relevancy. No matter how such systems are designed, MSU provides a user-centered measure that allows for calibration with actual measured user behavior.

2. BACKGROUND

The news about an event, such as Typhoon Bopha, has an impact on large sections of society. The sudden onset of such events can generate related information needs among governments, aid agencies, observer groups, as well as those affected and their families. Usually such events are unexpected and can be dynamic and evolve over time.

A large number of documents will be authored in connection with significant events and made available for consumption via the Internet in near real time. Publishers of this content include social networking services such as Twitter and Facebook as well as newspapers, news feeds, and blogs. Authors can range from journalists to government agencies to persons experiencing the event first hand. We consider all newly created content in aggregate as a stream of material from which a person might want to find relevant information.

The closest analogues to this type of retrieval are microblog search [12] and temporal summarization [2, 4]. The TREC 2011 Microblog track [12] included a single adhoc retrieval task where the retrieval system receives a query at time t and must retrieve the most relevant and recent Twitter postings prior to time t. The track organizers used precision at rank 30 to evaluate submissions, but found that the task was under-specified with respect to how results should be ranked in terms of relevance and recency, making comparisons between participating groups difficult.

In the TREC 2012 Microblog track [14], the adhoc retrieval task remained and was adjusted so that participants were simply to score the microblog posts for ranking purposes. The only notion of recency was that the postings had to have occurred before the time of the query. Relevance of a posting was formulated with respect to the informativeness of the posting without considerations of recency or novelty. To evaluate a ranking, the organizers computed its precision at 30 and its receiver operating characteristic (ROC). For TREC 2013, the microblog track continued to use precision at 30 as its primary retrieval measure [11].

Temporal summarization systems attempt to extract relevant information from an aggregate stream and relay this information to the user as soon as possible. In contrast to a microblog search task, a temporal summarization task extends over a period that roughly corresponds to the period that an event is in the news. Being a summarization task, the evaluation of temporal summarization systems has focused on determining the overall quality of the resulting summary even though the nature of temporal summarization means that the summary grows over time and is likely being consumed by an end user at regular intervals, e.g. for

a multi-day event, we would expect a user to seek new information daily.

The temporal summarization work of Allan, Gupta, and Khandelwal [2] defined new precision and recall measures that incorporate notions of usefulness and novelty. A useful sentence is one that would be helpful for use in a summary and a novel sentence is one that is new and contains information not previously seen. In addition, they examine the importance of summary size and recency of information.

Guo et al. [10] primarily focus on finding updates for rapidly evolving news events for which the information need is urgent (or time-critical). They consider an event to be composed of a number of subtopics. Each update may contain one or more subtopics as well. Accordingly, the precision for an update is the fraction of the update's subtopics that are also subtopics for the event, and, the recall for an update is the the fraction of the event's subtopics contained in the update. They define measures *expected precision* and *expected recall* to be the average precision (and recall respectively) over the complete set of updates.

The TREC 2013 Temporal Summarization Track (TST) [4] follows the temporal summarization framework defined by Allan et al. [2] with two significant changes. The first change is that while Allan et al. required that summarization happen on-the-fly as each news story was received, in TST, systems are only restricted to be certain that summaries produced at time t only use content available at t or earlier. For example, if a system wanted to, it could wait to process several days of content before producing a single result. The TST directly evaluates the recency of information and thus does not have to restrict systems in when or how they produce summaries. The second change is that the TST uses a web-scale aggregate stream of world wide web content as opposed to the comparatively small collection available to Allan et al. in 2001.

The TST uses sentences (*updates* in TST terminology) as the primary retrieval unit. Groups participating in TST had the task of filtering the aggregate stream and emitting sentences containing timely and novel information regarding a news event. The TST judges the relevance of sentences by noting the presence of individual nuggets of information. Sentences are annotated such that it is known what pieces (nuggets) of relevant information the sentences contain.

The TST assessors identified nuggets about an event from Wikipedia. For example, Typhoon Bopha has its own detailed Wikipedia page[1]. TST attached a timestamp to every nugget as determined by the first occurrence of the nugget in the Wikipedia edit history for the topic's article.

The TST organizers crafted effectiveness measures that are nugget-based variants of precision and recall that incorporate several key criteria for the updates produced by a system. In addition to an update needing to contain novel nuggets, the nuggets should be emitted as early as possible (latency) and shorter updates are better than longer updates (verbosity). Each update is assigned a score based on these criteria and the track's set-based effectiveness measures are computed as averages over all updates.

Participating groups produced runs consisting of emitted updates. When an update is emitted, it is given a timestamp equal to the most recent material consumed by the system from the aggregate stream of material. Thus, a system that delayed emitting any updates until it had consumed all of the material in the period of interest, would have all of its updates given a timestamp equal to the very end of the period. If an update contains a nugget, the nugget's *latency* is the difference between the update's timestamp and the nugget's Wikipedia assigned time. The gain for a reported nugget is discounted as a function of its latency, such that, with increasing latency, the gain for a nugget drops substantially within the first 24 hours, after which its discounted gain stays under 20%. It is possible for an update to be "earlier" than the nugget's timestamp, in which case the latency discount function awards the update a bonus for reporting the nugget early. Reflecting novelty, repeated nuggets have no value, providing zero gain.

Finally, sentences are penalized for being overly verbose. To incorporate verbosity into the track's measures, an update may count as more than a single update in the computation of the average. The *verbosity* of an update is based on the extra number of nuggets the update could have contained. For example, if an update contains zero nuggets and is 60 words long, and if the average number of words per nugget is 15, then this update has a verbosity penalty of 60 / 15 = 4 and when averaged counts as 5 updates, i.e. itself (1) plus its verbosity penalty (4). If the number of words in an update equals the sum of the words in the nuggets in the update, then the update counts as a single update.

TST's evaluation has two primary metrics, expected latency gain (ELG), which measures the gain per update while discounting latency and penalizing verbosity, and latency comprehensiveness (LC), which measures coverage of nuggets relating to the topic. ELG and LC are analogous to precision and recall respectively. While both ELG and LC are important to be considered together, we consider the ELG measure to be the primary measure of the track given that runs were presented from best to worst ELG in the track overview [4]. ELG for a set $\mathcal{D}$ of system generated updates is formulated as,

$$\mathbf{ELG}_{\mathbf{V}}(\mathcal{D}) = \frac{1}{\sum_{d \in \mathcal{D}} \mathbf{V}(d)} \sum_{d \in \mathcal{D}} \mathbf{G}(d, \mathcal{D}) \qquad (1)$$

where, $\mathbf{V}(d)$ is the verbosity normalization of update d, $\mathbf{G}(d, \mathcal{D})$ is the latency discounted gain for update d, and $\mathbf{G}(d, \mathcal{D})$ is non zero when d is the earliest update from the set $\mathcal{D}$ to report one or more nuggets for the topic. Once a nugget is reported, it does not contribute to gain if it appears again in later updates. The LC metric replaces the denominator in equation (1), with $\sum_{n \in \mathcal{N}} \mathbf{R}(n)$, where $\mathcal{N}$ is the set of nuggets identified by the assessors and $\mathbf{R}(n)$ is the relevance for n based on its importance ($\mathbf{R}(n) = 1$ for binary relevance). Essentially LC computes the recall of relevant material by the system. Unjudged sentences are elided from run submissions.

It is clear that users are kept in mind in the formulation of effectiveness measures for temporal summarization. For example, an update should be timely and contain relevant, novel information without extraneous material. In addition to these qualities, we believe that it important to also consider the user's desired frequency and duration of interaction with the system. A government agency may well assemble a team of people to monitor a stream of updates 24 hours a day for the duration of a crisis. An individual in the midst of a crisis may be limited by extrinsic circumstances to only having a few moments each day to check for updates.

[1] `http://en.wikipedia.org/wiki/Typhoon_Bopha`

ID	Time	Nugget
n_9	12/05/12 15:13:56	The typhoon destroyed 70-80% of plantations, mostly bananas for export.
n_{10}	12/06/12 17:45:12	Damage to agriculture and infrastructure in Compostela Valley province could reach at least 4 billion pesos, equivalent to 75 million or $98 million U.S.
n_{11}	12/04/12 18:31:18	About 40 people were killed or missing in flash floods and landslides near a mining area on Mindanao.
n_{12}	12/04/12 03:17:18	Typhoon Bopha made landfall on Mindanao early on December 4 as a category 4.
n_{13}	12/05/12 10:31:21	Late on December 3, Bopha made landfall over Baganga, Mindanao, as a category 5 super typhoon.
n_{14}	12/05/12 13:55:42	As of 5 December, 238 deaths had been reported on Mindana with hundreds missing.

Table 1: Nuggets reported to user (read by user) in the example session (Table 2). Times for each nugget are their time of first occurrence in Wikipedia's edit history.

3. MODELED STREAM UTILITY

We have designed our new effectiveness measure, modeled stream utility (MSU), to be an easily calibrated user-centered effectiveness measure. A user-centered effectiveness measure must take into consideration both the user interface of the retrieval system and user behavior with the user interface. For an effectiveness measures to be easily calibrated to user behavior, the measure's user model must be designed such that the model's parameters can be set based on actual measurements of user behavior.

3.1 User Interface and User Behavior

Our hypothetical user interface provides a means for the user to query the system and receive a ranked list of results (i.e. updates about an event). Each result is a short piece of text. The retrieval system produces a stream of information that is consumed by the user in the order produced. In other words, after issuing a query and receiving the ranked list of results, the user reads the results in rank order. No other interaction with the retrieval system is possible.

Our simulated user is not limited to one visit with the retrieval system. Our simulated user will repeatedly visit the search engine with a frequency reflective of their interest in the evolving event or reflective of their availability to visit. On each visit, the simulated user enters the same query, and expects to receive the most recent and relevant updates concerning their event of interest. As with their frequency of visits, the simulated user will read during a visit for an amount of time reflective of their interest or time available. We will call visits *sessions*, with each having a duration.

No two users are the same. Each user will visit the search engine with different frequencies and durations. Each user will have their own reading speed. We model frequency and duration by modeling the time a user spends away from the system and by separately modeling the time a user spends with the system. We model each user with three parameters:

- A: Average time away.
- D: Average session duration.
- V: Reading speed, e.g. words per minute.

To estimate user performance with the retrieval system, we draw multiple simulated users from population distributions and average user performance over all simulated users. We model the user population's time away and their session duration with two separate log-normal distributions. Log-normal distributions are such that when one takes the natural log of the data, a normal distribution fits the resulting distribution. Usually one describes a log-normal distribution in terms of the mean, μ, and standard deviation, σ, of the normal distribution fit to the log of the data.

For both the time-away and the session-duration distributions, we will describe them in this paper in terms of the underlying distribution's mean and standard deviation. If the underlying data has mean M and standard deviation S, then, for the log-normal distribution, the variance is $\sigma^2 = log(1 + S^2/M^2)$, and the mean is $\mu = log(M) - 0.5\sigma^2$. As we describe later in the paper, we consider a wide range of user behavior by varying the time away and session duration population distributions. To model the population's reading speed, we use the log-normal parameters from Clarke and Smucker [8] wherein, the distribution of reading speed across users is described by a log-normal distribution with $\mu = 1.29$ and $\sigma = 0.558$.

Given these three population distributions, we can generate as many simulated users as needed for numerical precision in our estimate of average user performance. For example, a possible *reasonable* setting of the population parameters is:

- Time away mean, $M_A = 3$ hours, and standard deviation, $S_A = 1.5$ hours.
- Session duration mean, $M_D = 2$ minutes, and standard deviation, $S_D = 1$ minute.
- Reading speed log-normal (words per second): $\mu = 1.29$ and $\sigma = 0.558$, i.e. a mean reading speed of 255 words per minute.

A random user drawn from this distribution will spend more or less time away, have longer or shorter session durations, and read faster or slower than the above population averages. Overall, the population of simulated users will have average behavior equal to the population means. To simulate a single user's interaction with the search engine over a given period of time, we construct a *user-trace* (Figure 1d) consisting of alternating sessions with the search engine and time intervals spent away from the search engine.

It would be unusual for a user to visit on a fixed frequency and visit for the exact same amount of time on each visit. To model variation in a user's time away and session duration, we use their mean time away and mean session duration as parameters to two separate exponential distributions. An exponential distribution is commonly used to model phenomena such as the time between radioactive decay of nuclei or the time between events in a Poisson process. For example, to simulate random session duration times with a mean of D, we draw random deviates from an exponential distribution [9] with a rate parameter of $\lambda = 1/D$.

3.2 Gain

A simulated user will repeatedly visit the search engine. During a visit, the user will read the search results in rank order. Each search result has an amount of gain (possibly zero) associated with it. The simulated user will accumulate

Time	Conf.	Update Sentence Produced by System at Time Shown in Column 1	TtR	CTtR	Nuggets
9:52	0.95	Typhoon Bopha , with central winds of 75 mph and gusts of up to 93 mph, battered beach resorts and dive spots in northern Palawan on Wednesday, but there was little damage as the storm began to weaken.	10.1 s	10.1 s	
9:52	0.95	Hardest hit were the coastal, farming and mining towns in the southern Mindanao region, where Bopha made landfall on Tuesday, destroying homes, causing landslides and flash flooding and killing at least 230 people.	8.8 s	18.9 s	n_{11}, n_{12}, n_{13}, n_{14}
9:52	0.95	About 60 people died in the municipality of New Bataan alone and around 245 were still missing, Uy said, adding the area was initially cut off by road blocks.	7.7 s	26.7 s	
9:52	0.91	Damage to agriculture and infrastructure in Compostela Valley province could reach at least 4 billion pesos ($98 million), with the typhoon destroying 70-80 percentof plantations, mostly bananas for export, Uy said.	8.5 s	35.2 s	n_9, n_{10}
9:50	0.87	A man looks at the dead bodies of relatives killed by landslides after Typhoon Bopha hit Compostela town. / Getty Disaster-response agencies reported 13 other typhoon-related deaths elsewhere.	7.5 s	42.7 s	
9:15	0.91	Typhoon Bopha : Philippines Storm Kills 238 - Yahoo!	2.4 s	45.1 s	n_{14}
7:45	0.87	Most Popular Today's five most popular stories World's oldest person dies at age 116 Snake on a plane forces emergency landing What city has world's best quality of life?	7.7 s	52.8 s	
7:31	0.87	(AP Photo)</em&gt ; May 1960 A magnitude 9.5 earthquake in southern Chile and ensuing tsunami kill at least 1,716 people.< br> <em&gt ;Caption: A soldier stands guard nearrubble strewn around an electrical shop which was shattered by an earthquake in Concepcion , Chile , on May 24, 1960.	13.1 s	65.9 s	

Table 2: Example user session: User reads at 225 words per minute, spending 60 seconds reading, starting at 9:55, on Dec 06, 2012. The column heading "Time" indicates the time at which the update was emitted, "Conf." is short for "Confidence", "TtR" means "Time to Read" and "CTtR" means "Cumulative Time to Read".

gain at the end of each search result. If the simulated user does not finish reading a search result during a session, the result is considered unread and no gain is recorded. We consider unjudged updates as non-relevant, and we do not elide them.

We measure the gain of a search result as the number of nuggets of relevant information contained in the result. A nugget can be considered an atomic piece of information. Example nuggets from the TST 2013 qrels are shown in Table 1. Our simulated users consider only novel nuggets to be relevant. Previously seen nuggets provide no gain.

Users interested in evolving events want the update produced by the search engine to be recent as well as relevant. Nuggets delivered late to a user are less likely to be considered relevant by the user. Given that different users will visit the search engine at different times, our notion of nugget timeliness must be relative to the simulated user. To make nugget timeliness relative to the user, we define a nugget as being *late* if it existed in the aggregate stream at a time equal to or before the start of the previous visit. In other words, if a user reads a result that contains a nugget that the search engine could have delivered during a previous visit, the nugget is late. To measure how late a nugget is, we define a function $\alpha(n)$ that returns how many sessions ago the nugget could have been reported (Figure 1f). A nugget that is reported on time has an $\alpha(n)$ of zero.

While we do not know how the probability that a user will consider a nugget relevant changes with its lateness, an exponential decay seems reasonable. Therefore we compute the gain for every read nugget n as:

$$g(n) = 1 \times L^{\alpha(n)} \tag{2}$$

where, L is the decay parameter for late reporting of nuggets and can vary between 0 and 1. L is a pre-determined value representing how much less a user is likely to consider the nugget to be relevant if it is reported late. For our experiments, we vary L from 0 to 1, where 0 indicates that a nugget loses all value if it is reported late, and 1 indicates that the nugget does not lose any value ever regardless of how late the reporting.

The gain from each read nugget is summed to get the cumulative gain for a user over the user trace (Figure 1d). Thus a simulated user's MSU for a search topic is given by:

$$MSU = \sum_{n} g(n) \tag{3}$$

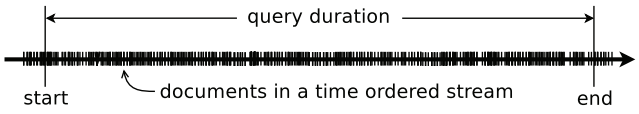
where n is the set of nuggets read by the user. We compute the MSU for a system by computing for each simulated user their mean MSU over all topics and then averaging all users' mean MSU to produce a system mean MSU. To increase the numerical precision of our system's MSU estimate, we only need to increase the number of simulated users.

3.3 MSU and TST

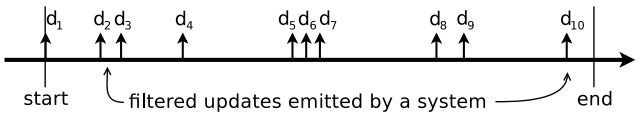
To demonstrate modeled stream utility (MSU), we use the TST 2013 test collection. The TST is a close but imperfect fit for MSU. The primary issue with using TST to demonstrate MSU is that the runs submitted by the participating groups were guided by the set-based effectiveness measures of TST.

As described in section 2, each participating group submitted runs to the TST that consisted of a set of updates. Each update has both a timestamp and a confidence associated with it. The timestamp is when the system emitted the update and the confidence is a system generated score indicating how relevant the update is.

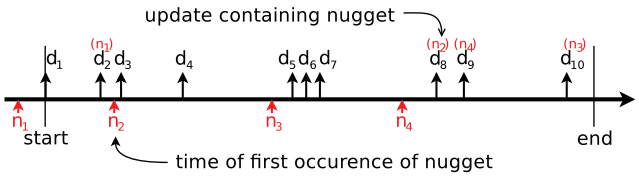
For MSU, the simulated users check the retrieval system as per their user-trace to read the system's updates. At the start of every session in the user-trace, the updates emitted between the end of the last session and the start of the current session are presented to the user in a reverse chronological order, so that most recent information is read first.



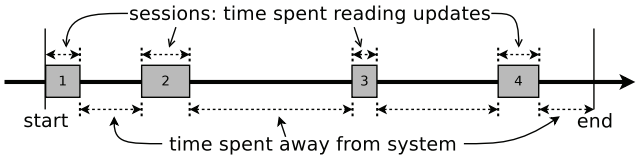
(a) Input: A time ordered document stream.



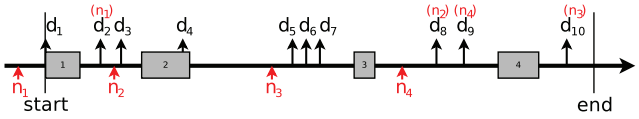
(b) Output: Stream of updates $d_1..d_{10}$ emitted at various times by a system.



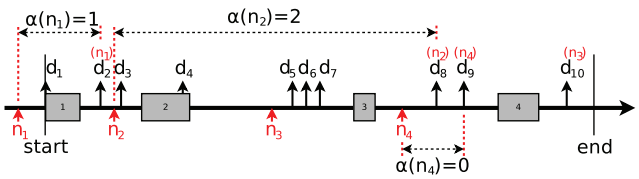
(c) Update-trace: Times of first occurrence of nuggets are identified. Updates containing nuggets are noted.



(d) User-trace: Simulated behavior of a user who reads updates from the stream from time to time.



(e) Reading-trace: Determines which updates are available to read for every session. e.g. d_3, d_2 are available to read at the start of session 2. The user's reading speed V determines which nuggets are actually read. Reading updates that contain nuggets adds to gain.



(f) $\alpha(n)$: Gain is discounted by $L^{\alpha(n)}$. $\alpha(n)$ is the number of sessions between the first occurrence of nugget n and the current session within which an update reporting n is read by the user. α is only computed only if the update containing n is read by the user.

Figure 1: Evaluation Model

Updates with the same timestamp are shown in descending order given their confidence. The simulated user starts reading the latest update and then reads the next older update, and so on, until the user runs out of time or encounters an already read update and stops reading further. In case the session ends, the last update that is partially read in the session, is considered as unread. The first user session always begins at the start of the query duration. To determine if a nugget is late, we use the TST assigned Wikipedia times as described in section 2. As with the TST, the gain of a novel, on-time nugget is 1.

3.4 MSU Example for User Gain in a Session

Figure 1 illustrates the MSU model of evaluation. Figures 1a and 1b show the input to and output of a system that generates a stream of updates. The evaluation process begins at Figure 1c, where the nuggets, their time of first occurrence and the updates containing the nuggets are identified. The user-trace in Figure 1d is generated for a modeled user by alternatively sampling the exponential distributions for the session duration and the away time respectively. With the user-trace overlaid over the update-trace we get a reading-trace (Figure 1e). The reading-trace identifies updates available to read at each user session, and by incorporating the reading speed of the user it determines which updates are actually read. Thus the reading-trace determines if any relevant updates were read by the user or not, for every user session.

For a more concrete example, consider a modeled user with A of 1 day, D of 60 seconds and a reading speed V, of 225 words per minutes. Let us also assume that this user considers late information to be half as likely to be relevant, for every session where it is unreported ($L = 0.5$). Suppose that this user checks for updates at about 10:00 am every day and had previously checked for updates at 10:02 am on Dec 4, at 10:11 am on Dec 5, and at 9:50 am on Dec 6, 2012.

On Dec 7, 2012, the user starts a session at 9:55 am. The user finds the updates listed in Table 2 at the start of the session. The times at which the updates were emitted by the system are listed in the first column. In this case, all updates were emitted on the morning of Dec 7 before the user started the session. The updates are presented to the user in reverse chronological order. In case multiple updates are emitted at the same time, the updates are ordered by their system assigned confidence (as for the 4 updates emitted at 9:52).

The user starts by reading the most recent update (delivered at 9:52 am). The user continues on to read the second update and finds 4 nuggets n_{11}, n_{12}, n_{13}, n_{14}. Table 1 lists the nuggets found in the session. These are nuggets that the user had not seen before and therefore experiences an increase in gain.

However, as per each nugget's Wikipedia time, n_{11} first occurred at 6:31 p.m. on Dec 4, 2012. It should ideally have been reported to the user at the session starting at 10:11 am on Dec 5. The system further missed reporting the nugget at the session at 9:50 on Dec 6. Therefore $\alpha(n_{11})$ is 2. By similar calculation $\alpha(n_{12}) = 3$, as it should have been reported for the user session at 10:02 on Dec 4. For nuggets n_{13} and n_{14}, $\alpha(n_{13})$ and $\alpha(n_{14})$ are 1.

As the user reads further down the list, more nuggets (n_9, n_{10}) are read and gain increases accordingly. n_{10} is reported in time at the current session and $\alpha(n_{10}) = 0$. The user gets no gain from reading the update emitted at 9:15 as

RunID (GroupID)	#Upd. / topic	ELG	Reason. MSU	ELG Rank	Reason. MSU Rank	Best Ranks achieved by Systems						
						Best Rank	MSU @Best	Parameter values for Best Rank				
								M_A	S_A	M_D	S_D	L
cluster5 (PRIS)	21.9	0.136	4.35	1	17	8	4.06	1 d	12 h	30 s	30 s	0.9
run2 (ICTNET)	93.8	0.127	9.45	2	4	1	9.27	1 d	12 h	5 m	2.5 m	0.1
run1 (ICTNET)	97.8	0.125	9.46	3	3	1	14.65	1 d	12 h	30 m	15 m	0.9
TuneExternal2 (hltcoe)	799.4	0.118	5.34	4	16	13	9.77	3 h	1.5 h	30 m	15 m	0.9
TuneBasePred2 (hltcoe)	2,696.1	0.114	5.49	5	15	11	9.89	1 h	30 m	15 m	7.5 m	0.9
cluster3 (PRIS)	42.3	0.103	5.99	6	11	5	5.83	1 d	12 h	30 s	15 s	1.0
cluster2 (PRIS)	122.1	0.074	9.31	7	5	1	12.02	3 h	1.5 h	30 s	30 s	1.0
uogTrNMTm1MM3	358.8	0.069	7.28	8	7	6	12.79	1 h	30 m	30 s	15 s	1.0
cluster1 (PRIS)	164.8	0.067	9.57	9	1	1	16.45	5 m	10 m	30 s	30 s	1.0
cluster4 (hltcoe)	163.0	0.067	9.55	10	2	1	16.15	10 m	20 m	30 s	30 s	1.0
BasePred (PRIS)	8,790.7	0.067	5.84	11	13	1	14.17	6 h	6 h	30 m	15 m	0.9
Baseline (hltcoe)	12,743.0	0.063	5.87	12	12	1	15.02	30 m	15 m	30 m	15 m	0.9
uogTrNSQ1	139.0	0.060	6.85	13	9	4	11.35	3 h	3 h	30 s	15 s	1.0
EXTERNAL (hltcoe)	22,476.1	0.055	5.60	14	14	1	20.38	30 m	1 h	30 m	15 m	1.0
uogTrNMTm3FMM4	168.3	0.049	6.33	15	10	5	10.52	3 h	1.5 h	30 s	15 s	1.0
uogTrNMM	954.7	0.045	7.63	16	6	1	19.07	30 m	15 m	15 m	7.5 m	1.0
uogTrEMMQ2	2,077.8	0.040	6.88	17	8	2	18.55	30 m	15 m	15 m	15 m	1.0
SUS1 (wim_GY_2013)	2,338.7	0.036	3.62	18	18	17	7.44	1 h	1 h	2 m	60 s	1.0
rg4 (UWaterlooMDS)	41,863.3	0.028	1.44	19	20	1	22.11	5 m	10 m	30 m	15 m	1.0
rg3 (UWaterlooMDS)	42,534.1	0.026	1.45	20	19	2	21.54	5 m	5 m	30 m	15 m	1.0
rg2 (UWaterlooMDS)	299,559.6	0.022	0.32	21	25	18	14.02	5 m	10 m	30 m	15 m	1.0
rg1 (UWaterlooMDS)	312,863.3	0.021	0.30	22	26	19	13.39	5 m	10 m	30 m	15 m	1.0
UWMDSqlec4t50	213,735.7	0.018	1.02	23	21	6	12.20	5 m	5 m	30 m	15 m	0.9
UWMDSqlec2t25	230,056.0	0.017	0.61	24	23	14	15.87	5 m	10 m	30 m	15 m	1.0
CosineEgrep (UWMDS)	11.9	0.010	0.55	25	24	19	0.30	1 d	12 h	60 s	30 s	0.3
NormEgrep (UWMDS)	151.3	0.001	0.73	26	22	19	0.52	3 h	1.5 h	30 s	15 s	0.5

Table 3: Main results. The 26 runs submitted to the TREC 2013 Temporal Summarization Track (TST) are ordered by the TST primary measure, ELG. The column "#Upd./topic" shows the average number of updates produced by the run per topic. The "Reason. MSU" column shows the score for modeled stream utility (MSU) with a reasonable set of parameters as described in section 4.2. On the right side of the table we report an experiment where we selected parameter settings from our parameter sweep that give a run the best rank possible relative to the other runs (section 4.3).

nugget n_{14} was read earlier. Since the session duration was 60 seconds, the last update was partially read and the user gets no gain on reading it. Thus the total gain (MSU) for this user, from this session on Dec 7, is 2.875.

4. EXPERIMENTS

We explore the parameter space of MSU by applying it to evaluate the runs submitted to the TST 2013. We compare MSU's ranking of the TST runs to the TST's track's primary measure, ELG. The ELG measure is described in section 2.

The 2013 TST evaluated 26 runs from 6 groups over 9 topics. Topics in the track are instances of event types from the set {accident, bombing, earthquake, shooting, storm}. Each topic had a time period (query duration) of 10 days. In other words, a run must summarize 10 days of material from the TREC KBA stream corpus [1] for each topic.

The KBA corpus is essentially a time ordered document stream spanning from October 2011 to January 2013, containing over 1 billion documents crawled from the web. The corpus contains documents with 3 categories, i.e. the documents were crawled from URLs of public newswires (news), blogs and forums (social), and the URLs submitted for shortening at `www.bitly.com`. Documents are segmented into sentences and some documents are tagged with named entities using NLP tools. There are on average 93,037 documents per hour [5] in the corpus.

4.1 Parameter Sweep

Keeping the user population's reading speed distribution the same across all parameter sets, we sweep the parameter space by setting MSU's parameters to the following values:

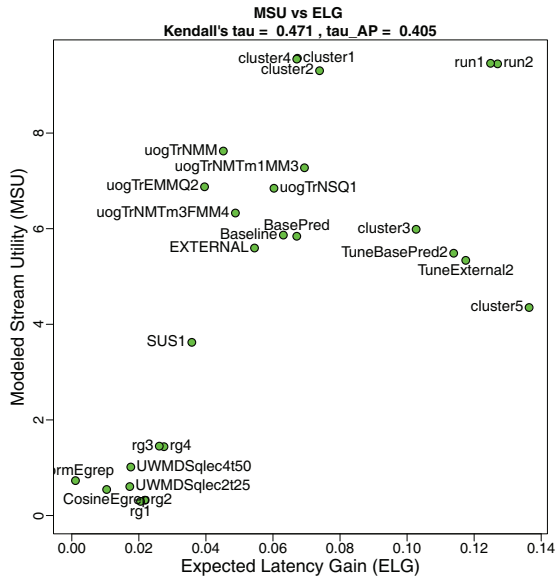
- User population mean session duration, $M_D = \{0.5, 1, 2, 5, 15, 30\}$ minutes.
- User population mean time away, $M_A = \{5, 10, 30$ minutes, 1, 3, 6, 24 hours$\}$).
- Lateness decay parameter, $L = \{0, 0.1, 0.25, 0.5, 0.75, 0.9, 1\}$.

For the standard deviations of the session duration, S_D, and the time away, S_A, we multiplied the mean by the values 0.5, 1, 2. For example, a mean session duration of 30 seconds, was associated with standard deviations of 15, 30 and 60 seconds. In total we generated 7 (M_A) x 3 (S_A) x 6 (M_D) x 3 (S_D) x 7 (L) = 2646 parameter-tuples (points in the parameter space). For instance, the parameter tuple (M_A= 6 hours, S_A= 3 hours, M_D= 5 minutes,S_D= 5 minutes, L = 1), is a point in the parameter space that represents users who spend an average of 5 minutes every 6 hours reading updates, unaffected by late reporting of nuggets.

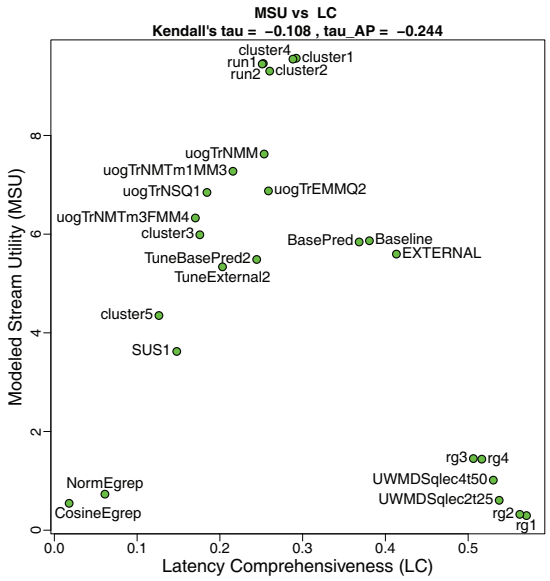
For each tuple (selected point from the parameter space) in the parameter sweep, we simulated 1000 users. For each simulated user, we generated their user-trace, determined which updates they read, and computed their average MSU across the topics for every run submitted to TST 2013.

4.2 Reasonable Parameters

We believe that a *reasonable* setting of MSU's parameters is a user population where the users visit on average for 2 minutes every 3 hours and for whom late material quickly



(a) Reasonable Users, MSU vs ELG Correlation.



(b) Reasonable Users, MSU vs LC Correlation.

Figure 2: MSU with reasonable parameter settings (M_A= 3 hours, S_A= 1.5 hours, M_D= 2 minutes, S_D= 1 minute, L= 0.5), compared to TST 2013 measures, ELG and LC.

becomes less likely to be considered relevant. This corresponds to the parameter tuple (M_D = 2 minutes, S_D = 1 minute, M_A = 3 hours, S_A = 1.5 hours, L = 0.5).

Both Figure 2a and Table 3 show the results of MSU vs. the TST measure ELG, given these reasonable parameter settings. For both MSU and ELG, larger values indicate better runs. At these parameter settings, MSU and ELG do not rank systems the same (Kendall's τ = 0.471). As Figure 2a shows, the runs in the middle-lower positions of the ELG ranking jump to the top positions. A similar observation can be made in Figure 2b for MSU vs. the LC measure. Here as well, MSU and LC rank systems differently (Kendall's $\tau = -0.108$). ELG and LC are themselves negatively correlated (Kendall's $\tau = -0.28$). The AP correlation coefficient (τ_{AP}) [16, 13], which is more sensitive to changes in higher ranks is also reported in Figure 2.

Across the entire set of swept parameter settings, the maximum Kendall's τ correlation was found to be 0.625 for the parameters: M_A= 24 hours, S_A= 12 hours, M_D= 1 minute, S_D= 2 minutes, L = 0.1. These results are shown in Figure 3. While not a high correlation, this result shows us that ELG best correlates with a simulated user population where users on average visit once a day for 1 minute. In other words, over the 10 days of the query period, an average user reads about 10 minutes of material. If we were to understand ELG in terms of the users that would prefer its top ranked runs, then ELG is tuned for highly time constrained and selective users with low tolerance for late reporting.

A correlation of 0.625 correlation is not high, and MSU and ELG are behaving quite differently. We believe that there are likely two main reasons for the different rank order of the runs. The first reason is that run performance with MSU is affected by the amount of material that the simulated user reads. If a run does not supply enough material for a session visit, then the simulated user will stop reading and also stop accumulating gain.

For example, Table 3 shows that the top ELG run `cluster5` averages only 21.9 updates per topic over a 10 day query duration. With the reasonable parameter settings, MSU simulated users spend about 2 minutes every 3 hours, i.e. about 160 minutes reading, on average, over a 10 day query duration. With an average length of each TST update being 63 words, and the average reading speed being 4.3 words per second, it takes on average just 319 seconds to read all the 21.9 updates of `cluster5`. A simulated user, who is willing to read for 160 minutes in total over 10 days, thus derives very low gain from `cluster5` because such a user's visit is cut short from a lack of material to read.

Figure 4 shows the user performance at the point (M_A= 5 minutes, S_A= 10 minutes, M_D= 30 minutes, S_D= 15 minutes, L = 1) that has the minimum correlation of MSU with ELG (Kendall's τ = -0.04), in our parameter set. This set of users seem inclined to spend almost all their time with the system reading updates taking a 5 minute break every 30 minutes on average, without any dissatisfaction for late reporting of information. Unsurprisingly, these users achieve the highest amount of MSU (22.11) with run `rg4`. While seemingly an unreasonable parameter setting for MSU, such user behavior could be achieved by a team interested in constant monitoring of a stream of updates.

This result also shows that MSU scores are related to the amount of material read by the simulated users. MSU measures gain as the number of relevant nuggets read in the total content consumed by a user, while discounting for lateness if required. The amount of material read depends on the characteristic behavior of the user (or user population) as well as the number of updates emitted by the system. MSU for a system is simply the average over all simulated users.

A second reason for differences between ELG and MSU is likely that ELG is a set-based measure that measures av-

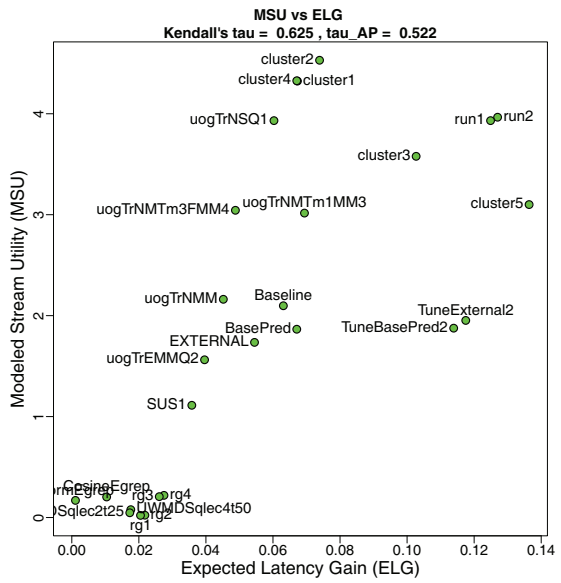


Figure 3: Maximum correlation of MSU with ELG is obtained with parameters (M_A= 24 hours, S_A= 12 hours, M_D= 1 minute, S_D= 2 minutes, L= 0.1).

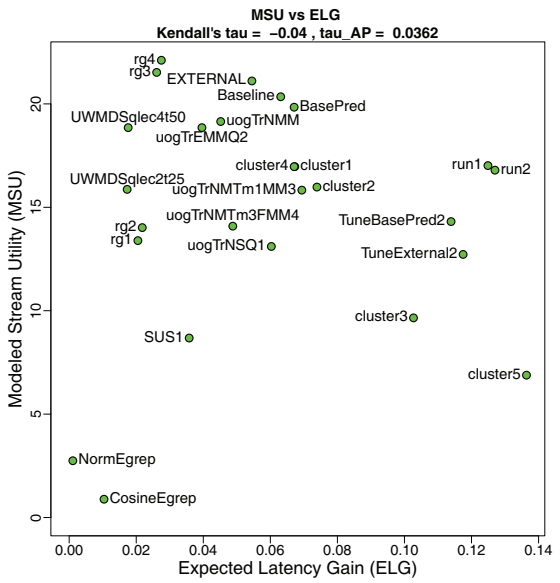


Figure 4: Minimum correlation of MSU with ELG is obtained with parameters (M_A= 5 minutes, S_A= 10 minutes, M_D= 30 minutes, S_D= 15 minutes, L= 1).

erage quality of an update. In contrast, MSU is estimating the total number of nuggets read and considered by the simulated user to be relevant. To see how much the set-based nature of ELG is causing it to be different than MSU, we transformed MSU into a similar measure by taking MSU and dividing it by the time the simulated user spent reading (MSU/second).

In our experiments, MSU/second has the highest correlation (Kendall's $\tau = 0.754$) with ELG (Figure 5), with parameters (M_A= 24 hours, S_A= 12 hours, M_D= 1 minute,S_D= 1 minutes, $L = 0.1$). The correlation between MSU/second and ELG is greater than the maximum correlation between MSU and ELG. Of note, the top ranked run for MSU/second is now the same as for ELG: `cluster5`. The MSU/second top 4 runs in Figure 5 are 4 of the 5 runs with the lowest number of updates submitted (Table 3).

4.3 Everyone's a Winner (Almost)

Noting how the ranking of systems can change greatly based on the wide range of parameter settings in our sweep, we decided to find the instances in our parameter sweep for which a particular system was ranked the highest across all parameter sets. Table 3 shows the results of this analysis on its right side. Some systems achieved their best rank for multiple parameter sets. In such cases, we chose the parameter set for which the system had the highest MSU.

The relationship between time spent and user performance can be seen as we look at the parameter settings from top to bottom of Table 3. Systems that had very few updates submitted, performed well for users who might visit a system about once a day for 30 seconds to 30 minutes on average. The run `cluster2` seems to be the best performing system for users who return to the system about every 3 hours for 30 seconds on average. As we go lower in the table, we see that spending more time reading (larger M_D) and taking shorter breaks (smaller M_A) improves performance of systems that ranked lower on ELG. These systems are typically those that submitted a large number of updates.

As Table 3 shows, almost all groups have at least one run that is able to achieve a rank 1 performance under some setting of MSU's parameters. If we want to know which system is the best, we must calibrate MSU given actual user behavior. This result may also show that submitted TST systems had very different notions of the appropriate amount of material to make available to users. TST's ELG measure is set-based, but the decision of how large of a set to return to users is left to system designers. In contrast, MSU guides system designers to return a suitable amount of material based on the model of user behavior.

5. OTHER RELATED WORK

Our work builds directly on recent research in modeling user behavior for improved evaluation of information retrieval systems [7]. Of these new effectiveness measures, Clarke and Smucker's [8] Time Well Spent (TWS) measure is the most similar. Like modeled stream utility (MSU), TWS provides a means to evaluate systems that produce a stream of content. Unlike TWS, MSU models multiple visits to the same system, and MSU models variation in each simulated user's behavior. MSU should be amenable to the same statistical analyses that TWS enables.

Other researchers have also evaluated systems by defining a hypothetical user interface and then simulating user behavior with the interface. Of the more recent work that goes beyond a single query and results list is that of Baskaya, Keskustalo, and Järvelin [6] who conducted an experiment that considered many different parameter settings for their model and investigated various query reformulation strategies that could be utilized by a user. Yang and Lad [15] also employ a user model based evaluation for information distillation systems and it would be interesting to compare MSU with their method in future work.

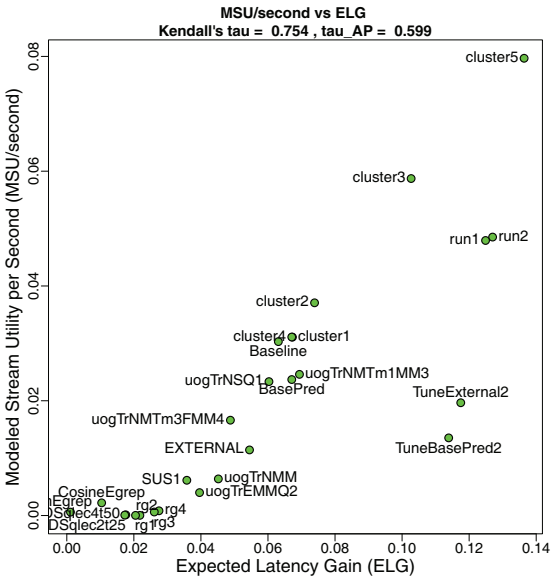


Figure 5: Maximum correlation of MSU/second with ELG is obtained with parameters (M_A= 24 hours, S_A= 12 hours, M_D= 1 minute, S_D= 1 minutes, L= 0.1).

6. DISCUSSION

ELG has aspects that try to capture characteristics of good updates, i.e. updates should be on time, not long, and have relevant material. ELG seems to be a measure oriented towards measuring the performance of systems that push highly relevant updates with low frequency to the user. ELG does not provide an easy means to be calibrated to known user behavior.

Our experiments and analysis show that it matters how much material is read. By specifying the amount of material in user terms, we have a way of then calibrating a measure once we know actual user behavior. Observing actual user behavior while evolving events are actually taking place would involve monitoring users' browsing histories. The sudden nature of news events makes a live user study difficult to organize. Search log-analysis may provide some indirect insight into user behavior when such events are running. Our analysis also shows that there may be a case for personalization of stream filtering systems for different user behaviors. As other future work, we hope to extend our understanding of MSU, both in terms of its formal properties and in terms of empirical meta-evaluation criteria, including robustness to noise, discriminativeness between systems, and strictness [3].

7. CONCLUSION

We introduced an effectiveness measure that utilizes a model of user behavior for evaluating systems producing streams of information about evolving events. Our measure is designed to be calibrated based on actual usage data. Our user model simulates a user checking back with the system to read the most recent information from time to time. Users can check back with different frequencies and for different amounts of time depending on various factors. By modeling user behavior, our effectiveness measure produces a score that is easily interpretable as the number of relevant nuggets of information read by the user. As would be expected, we found that for streams of updates, the gain is sensitive to the amount of time a user spends for reading updates. While temporal summarization systems have traditionally been evaluated in term of their precision and recall, we believe that it is important to consider both the user interface and the user behavior with this interface when evaluating such systems. Given the degree to which the rankings of systems changed as we modified the behavior of the user population, we believe that an effectiveness measure such as ours when calibrated with user data will allow system developers and researchers to build better performing systems tuned to user behavior.

8. ACKNOWLEDGMENTS

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

9. REFERENCES

[1] KBA Stream Corpus 2013. `http://trec-kba.org/kba-stream-corpus-2013.shtml`.
[2] J. Allan, R. Gupta, and V. Khandelwal. Temporal Summaries of New Topics. In *SIGIR*, pp. 10–18, 2001.
[3] E. Amigó, J. Gonzalo, and S. Mizzaro. A Formal Approach to Effectiveness Metrics for Information Access: Retrieval, Filtering, and Clustering. In *ECIR*, pp. 817–821. 2015.
[4] J. Aslam, F. Diaz, M. Ekstrand-Abueg, V. Pavlu, and T. Sakai. TREC 2013 Temporal Summarization. In *TREC*, 2013.
[5] G. Baruah, A. Roegiest, and M. D. Smucker. The Effect of Expanding Relevance Judgements with Duplicates. In *SIGIR*, pp. 1159–1162, 2014.
[6] F. Baskaya, H. Keskustalo, and K. Järvelin. Modeling Behavioral Factors in Interactive Information Retrieval. In *CIKM*, pp. 2297–2302, 2013.
[7] C. L. 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.
[8] C. L. A. Clarke and M. D. Smucker. Time Well Spent. In *IIiX*, pp. 205–214, 2014.
[9] B. P. Flannery, W. H. Press, S. A. Teukolsky, and W. T. Vetterling. Numerical Recipes in C, pp. 214–215, Cambridge University Press, 1988.
[10] Q. Guo, F. Diaz, and E. Yom-Tov. Updating Users About Time Critical Events. In *ECIR*, pp. 483–494, 2013.
[11] J. Lin and M. Efron. Overview of the TREC-2013 Microblog Track. In *TREC*, 2013.
[12] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff. Overview of the TREC-2011 Microblog Track. In *TREC*, 2011.
[13] M. D. Smucker, G. Kazai, and M. Lease. Overview of the TREC 2013 Crowdsourcing Track. In *TREC*, 2013.
[14] I. Soboroff, I. Ounis, J. Lin, and I. Soboroff. Overview of the TREC-2012 Microblog Track. In *TREC*, 2012.
[15] Y. Yang and A. Lad. Modeling Expected Utility of Multi-Session Information Distillation. In *ICTIR*, pp. 164–175, 2009.
[16] E. Yilmaz, J. A. Aslam, and S. Robertson. A new Rank Correlation Coefficient for Information Retrieval. In *SIGIR*, pp. 587–594, 2008.