Incremental Update Summarization: Adaptive Sentence Selection based on Prevalence and Novelty

Richard McCreadie, Craig Macdonald, and Iadh Ounis
{firstname.lastname}@glasgow.ac.uk

University of Glasgow
G12 8QQ, Glasgow, UK

ABSTRACT

The automatic summarization of long-running events from news steams is a challenging problem. A long-running event can contain hundreds of unique 'nuggets' of information to summarize, spread-out over its lifetime. Meanwhile, information reported about it can rapidly become outdated and is often highly redundant. Incremental update summarization (IUS) aims to select sentences from news streams to issue as updates to the user, summarising that event over time. The updates issued should cover all of the key nuggets concisely and before the information contained in those nuggets becomes outdated. Prior summarization approaches when applied to IUS can fail, since they define a fixed summary length that cannot effectively account for the different magnitudes and varying rate of development of such events. In this paper, we propose a novel IUS approach that adaptively alters the volume of content issued as updates over time with respect to the prevalence and novelty of discussions about the event. It incorporates existing state-of-the-art summarization techniques to rank candidate sentences, followed by a supervised regression model that balances novelty, nugget coverage and timeliness when selecting sentences from the top ranks. We empirically evaluate our approach using the TREC 2013 Temporal Summarization dataset extended with additional assessments. Our results show that by adaptively adjusting the number of sentences to select over time, our approach can nearly double the performance of effective summarization baselines.

Categories and Subject Descriptors: H.3.3 [Information Storage & Retrieval]: Information Search & Retrieval

Keywords: Temporal Summarization, Adaptive Models, Machine Learning

1. INTRODUCTION

When a significant event occurs, it is reported in a variety of streams such as newswire, microblogs/blogs and forums. However, given the large number of these media streams and the vast quantities of articles/posts produced each day, monitoring an event poses a challenging problem for end-users. Multi-document summarization (MDS) techniques have been proposed to tackle event summarization. These techniques generate a concise, on-topic summary from a set of related documents [21], often by extracting informative sentences from those documents [5, 7, 16, 20, 22, 27]. However, these approaches are designed to retrospectively summarize an event given a (high-quality) set of on-topic newswire articles about it and hence are not suitable to summarize an ongoing event over time. Some MDS approaches were later extended to the task of update summarization [10], where the aim is to produce a fixed-length 'update' summary about the event given an initial summary and a set of documents containing new information about the event. However, due to the fixed-length nature of the update summary produced, these approaches are not well equipped to summarize long-running events, particularly for use-cases that require updates to be issued in a timely manner.

For instance, consider a user that had heard about the garment factory fires that took place in Pakistan[1] and wanted to access a summary of that event, which automatically updates over time with new content. For this event, we consider there to be a set of unique pieces of information that the user would like to know about e.g. that it was 'Pakistan's deadliest industrial disaster', or that the 'total killed

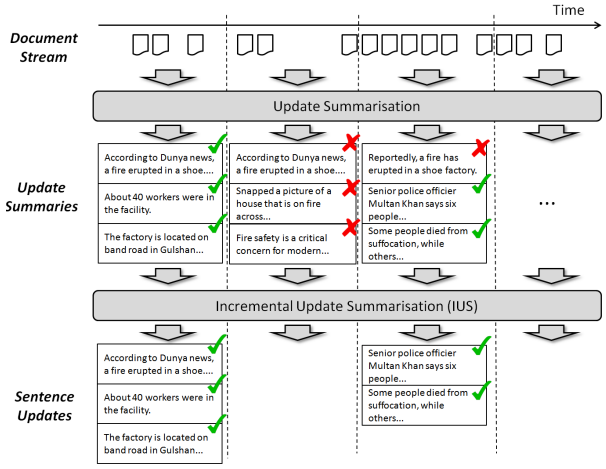


Figure 1: An illustration of Incremental Update Summarization (IUS) task over a time-ordered document stream.

CIKM'2014, November 03 - 07 2014, Shanghai, China

http://dx.doi.org/10.1145/2661829.2661951

[1] http://en.wikipedia.org/wiki/2012_Pakistan_garment_factory_fires

was 315'. However, this event ran over a 6-day period, with many important nuggets emerging after the initial event, e.g. that a 'fake safety certificate had been obtained [by the factory owners] to fulfil developed countries demands'. Deploying a traditional MDS system using only the first articles would result in many missed nuggets, since they will occur much later in the event timeline. Meanwhile, iteratively applying an update summarization system will result in update summaries that contain irrelevant and/or redundant content during periods when no new information emerged. Hence, we need an alternative summarization approach that can automatically extract novel updates from news streams in a timely manner, while minimising irrelevant, uninformative or redundant content.

In this paper, we introduce the task of Incremental Update Summarization (IUS), which aims to produce a series of sentence updates about an event over time to issue to an end-user tracking the event. Figure 1 gives an example of an IUS system. As we can see from Figure 1, documents published over time relating to an event are first processed by a traditional update summarization system at regular time intervals, producing fixed-length update summaries comprised of sentences. However, during periods when no new pieces of information emerge, the update summary produced may contain irrelevant or redundant sentences. The aim of an IUS system therefore is to take the update summaries and extract only those sentences from them that are on-topic and provide new previously unseen information, such that they can be issued to an end-user as updates about the event. IUS is highly useful for users wanting short timely updates pushed to them, for example in the form of an RSS feed or updating timeline.

To tackle IUS, we propose a supervised approach, where we treat sentence selection from the update summary as a rank-cutoff problem. Here the aim is to predict – based upon the current update summary and any sentences previously issued as updates – how many sentences from the current update summary to issue to the end-user. We train a regression model that combines over 300 features, describing the prevalence of the event, the novelty of content contained, and overall sentence quality across the sentences in the input update summary. These features are used to balance the cost of selecting a deeper cutoff in terms of returning redundant content against the risk of missing important information by selecting a shallower cutoff.

The contributions of this paper are three-fold. First, we propose a new model for IUS that can adaptively adjust the amount of summary content to select over time. Second, through experimentation using the TREC 2013 Temporal Summarization dataset – expanded with over 22,000 new assessments to combat incompleteness [3] in the original dataset – we show that our proposed approach can improve IUS performance by up-to 47% in comparison to the update summarization baselines tested. Third, we provide an in-depth analysis, identifying the most informative features and highlighting directions for future enhancement.

The remainder of this paper is structured as follows. Section 2 provides a background into prior works examining multi-document summarization and summarization over time. In Section 3, we define the IUS task, while Section 4 details our proposed approach to tackle it. Section 5 describes our experimental setup in terms of the dataset, the development of additional assessments, training regime and evaluation measures, while in Section 6, we detail our results and analysis. Concluding remarks are provided in Section 8.

2. RELATED WORK

As described above, the IUS task that we introduce builds upon traditional multi-document/update summarization systems from the literature, using them to produce intermediate summaries over time. In this section, we describe prior works in the field of multi-document summarization (Section 2.1) and update/temporal summarization (Section 2.2).

2.1 Multi-Document Summarization

Multi-document summarization (MDS) approaches take as input a set of documents about a topic to be summarized and produce a (typically fixed length) summary of those documents. The MDS task was investigated at both the Document Understanding Conference (DUC)[2] and Text Analysis Conference (TAC) [10]. MDS approaches can be categorised as either *extractive* – where unmodified sentences from the input documents are selected for inclusion into the summary [5, 7, 16, 20, 22, 31, 34, 37], or *abstractive* – where new sentences are generated from the input documents, e.g. via sentence compression or information fusion techniques [16, 33, 37]. Summarization approaches that also take as input a query (in a similar way to a Web search engine) relating to the topic are referred to as 'query-focused' approaches. The IUS task that we introduce takes as input, summaries produced by a query-focused multi-document summarization system and extracts sentences from them to produce updates for long-running events, issuing them to users who wish to track those events.

Extractive summarization techniques can be defined in terms of three stages [21]: the generation of an intermediate representation of the input documents, e.g. the identification of key terms [18]; the scoring of each sentence with respect to its preference for inclusion into the summary, e.g. by topicality [30], text quality [27] or aspect coverage [20]; and the selection of sentences from the ranked list, e.g. selecting the top n, or applying a redundancy sensitive technique such as MMR [4]. One of the most widely used approaches to score sentences for inclusion into a summary is clustering with respect to the centroid of the sentences within the input documents [19, 24], thereby selecting those sentences most central to the topic first. Early systems that take this approach include NeATS [19] and MEAD [24]. Other approaches score sentences with respect to only highly informative terms [7, 18, 22] or based upon their coverage of latent sub-topics extracted from the input documents [20]. More recent approaches have used graph-based methods that model textual/aspect overlap between sentences, and in some cases infer sentence hierarchies from input documents [5, 11, 23, 34]. For example, LexPageRank [11] builds upon the well known PageRank algorithm to identify the most important sentences in the graph. Finally, machine-learned sentence ranking approaches have been shown to be effective for MDS [16, 17, 27, 33]. For instance, the Pythy summarization system proposed by Toutanova et al. [27] linearly combines a series of weighted summarization features to score each sentence, while Li et al. proposed a structural learner that jointly optimises diversity and coverage when scoring [16]. Most recently, Wang et al. [33] used a learning-to-rank technique, combining sentence features and the output scores of other summarization approaches for sentence scoring. In our later experiments, we leverage some of the extractive summarization approaches described above to produce the summaries that our IUS approach takes as input, namely the learning-

[2] http://www-nlpir.nist.gov/projects/duc/

to-rank approach by Wang et al. [33] and the classical SumBasic [22], SumFocus [27] and Classy [7] approaches.

2.2 Update Summarization and Timeline Generation

One limitation of MDS is that it is retrospective in nature, in that there is an assumption that all of the relevant documents about the event are available as input beforehand [1]. The task of update summarization was designed to address this issue through the generation of further fixed-length 'update' summaries from documents published later in time [10]. An update summarization system takes as input the original summary of the event and a new set of documents, with the aim to produce an update summary that contains only new information not already contained within the original summary. Prior approaches to tackle update summarization generally apply an existing MDS technique over the new documents to produce a candidate ranking of summary sentences [10]. They then use a redundancy removal technique to remove sentences that are similar to those already contained within the original summary. Effective redundancy removal techniques include using an upper-bound on the pair-wise sentence similarity [9] or using MMR [29] to identify and filter out excessively similar sentences. In our later experiments, we use a learning-to-rank approach to produce an initial summary at periodic intervals. Then, following [9], we employ an upper bound υ on permissible similarity between sentences, removing any that exceed this upper bound, forming the update summaries that our IUS approach takes as input.

An alternative approach to event summarization over time is timeline generation, where the aim is to produce a timestamped list of updates covering the most important nuggets about that event. Swan and Allan [26] proposed an early approach to timeline generation by extracting clusters of named entities and noun phrases relating to events over time. This approach was later extended to select related sentences for each date [1]. Chieu et al. [6] also examined the creation of sentence timelines for news events. They studied the effect of using different centrality and burstiness estimates to rank sentences for inclusion into a timeline. Recently, Yan et al. [36] investigated evolutionary timeline generation (ETS) for an event. They extracted sentences from newswire articles crawled from the Web at 24 hour intervals, using a combined utility function that incorporates relevance, coverage, coherence and diversity (with respect to sentences selected previously) for sentence scoring. A timeline is iteratively updated over time by selecting sentences from each interval based upon local and global optimisations aiming to maximise the summary utility. IUS is similar to ETS in that both aim to produce a timeline of sentences summarising an event. However, rather than producing a high precision summary with respect to only the most important updates at the end of each day, IUS focuses on timely updating about the event throughout each day.

3. IUS TASK DEFINITION

In this paper, we tackle the problem of how to extract sentences about a long-running event from news streams to issue to a user wishing to track that event. However, avoiding the selection of redundant or irrelevant content is challenging, since the amount of new content relating to an event varies over time. For example, Hurricane Sandy[3] started out as a tropical storm in the Caribbean but grew into a huge event as it strengthened and passed over densely populated areas along the U.S. east coast. To illustrate, Figure 2 shows the distribution of manually identified information nuggets about Hurricane Sandy over the lifetime of the event. As we can see, there is a large burst of updates around the 29th of October, when Sandy made landfall on the U.S. east coast. Hence, to summarize long-running events effectively, we need an approach that can adaptively alter the amount of new sentences extracted over time.

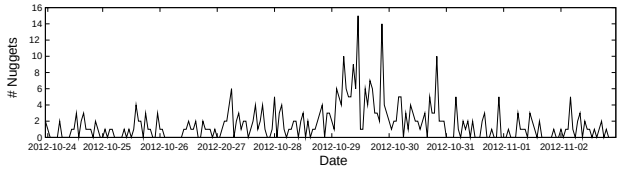


Figure 2: The number of manually identified nuggets about the Hurricane Sandy over time.

To tackle this issue, we introduce the task of incremental update summarization (IUS), which aims to issue periodic updates about an event to a user in a timely manner. In particular, we use the following definitions:

Long-running Event e**:** A newsworthy happening in the world that was significant enough to be reported on over multiple days.
Event representation (Query) Q**:** A short textual representation of the long-running event provided by an end-user. For example, a user wanting to track the progress of Hurricane Sandy might issue the query 'hurricane sandy'.
Time Interval t**:** A period of time during the lifetime of an event e. An event is split into a series of equal size time intervals $t = \{t_0, t_1, t_2...\}$.
Update Summary S_t**:** A fixed length query-focused update summary for the time interval t and event representation Q, produced by an update summarization system like those described in Section 2.2. Each update summary S_t is comprised of a ranked list of sentences $s \in S_t$ published during time interval t. The ranking criteria for sentences in S_t is relevance with respect to Q and the intrinsic informativeness/quality of those sentences.
Sentence Update u**:** A sentence to be issued to the end-user about the event from an update summary S_t.
Issued Updates U_t**:** The timeline of all sentence updates issued to the user before t.

Using these definitions, we formalise the IUS task as follows. Given a query Q issued by a user about a long-running event e to be tracked, for each time interval t over the lifetime of e, the IUS system takes as input an update summary S_t about e and selects sentences from it to issue as a series of sentence updates u (given the previous selections U_t).

4. UPDATE SENTENCE SELECTION

We propose a new approach that treats IUS as a *rank-cutoff problem*, inspired by prior work in the Web search domain on determining when to stop reading a ranked list [2]. Assuming that we are at time interval t within an event e. If there is new information available in the current update summary S_t, then an IUS system should select multiple sentences to issue as updates, achieved by selecting a deep rank-cutoff for sentences within S_t. However, during time intervals with low activity, the update summary S_t will mostly contain redundant or irrelevant sentences, particularly in the

[3] `http://en.wikipedia.org/wiki/Hurricane_Sandy`

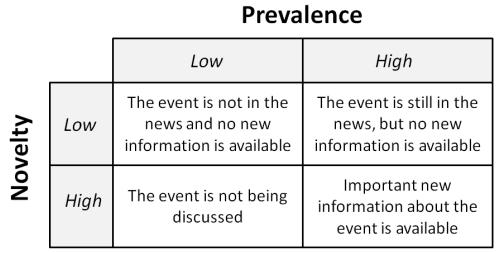


Figure 3: Illustration of how prevalence and novelty relates to different event states.

lower ranks. Hence, in this case, an IUS system should select a shallow (or in the extreme a zero-rank) cutoff. The aim of our proposed approach therefore is to predict the optimal rank-cutoff for S_t, denoted as θ_t.

To accurately predict the optimal θ_t for a given update summary S_t, we leverage two concepts: *prevalence* – the proportion of a population found to belong to a class; and the *novelty* – the degree to which a population is similar to a second population. In the context of IUS, we define prevalence to be the proportion of the update summary S_t to a rank r (in S_t) that relates to the event. Meanwhile, novelty is the proportion of sentences in the update summary S_t to a rank r that are similar to sentences previously issued as updates, i.e. in U_t. The idea is that a large (deep) rank-cutoff should only be selected if the update summary S_t exhibits a high degree of both prevalence and novelty. To illustrate, Figure 3 shows how prevalence and novelty map to different states of an event. As we can see from Figure 3, only in the case of high prevalence and novelty is there likely to be new information available that we would want to issue as updates to the user.

From a user perspective, θ_t is a prediction of where to stop reading the update summary S_t produced for time interval t given the sentence updates read previously (U_t). A non-zero θ_t value indicates that there is new content the user would like to read. In contrast, a predicted θ_t value of 0 indicates that no new information is available and hence, no sentence updates should be issued. To illustrate, consider the following update summary S_t containing four sentences about the aforementioned Pakistan factory fire disaster, assuming that we have not issued any sentence updates so far:

Rank	Sentence	Prevalence	Novelty
1	*The fire is being described as the deadliest industrial accident in Pakistan's 65-year history, and highlighted the woeful safety conditions that exist at many factories around the country.*	High	High
2	*Buildings regularly lack fire exits and basic safety equipment like alarms and sprinklers.*	High	High
3	*Fire safety induction for new staff running at 10:30am.*	Medium	High
4	*Such safety issues are common through Pakistan, where buildings also lack emergency equipment*	Medium	Medium

The first sentence in this ranking contains two useful pieces of information, namely that the event was Pakistan's deadliest industrial accident and that Pakistan has poor safety conditions. Hence, up-to rank 1, the event is prevalent (1/1 sentences are on-topic) and the summary is novel (1/1 sentences contain new information). The second sentence also provides more new on-topic information about the safety conditions, i.e. up-to rank 2, the event is prevalent (2/2 sentences) and the summary is novel (2/2 sentences). On the other hand, the third sentence, while containing novel information, is not on-topic. As a result, up-to rank 3, we can characterise the update summary as having medium prevalence (2/3 sentences) and high novelty (3/3 sentences). Finally, the fourth sentence is on-topic but redundant given sentence 2. Hence, up-to rank 4, we describe the summary as having medium prevalence (3/4 sentences) and medium novelty (3/4 sentences). Given that we want high prevalence and novelty, the optimal θ_t for this ranking would therefore be 2. Of course, during subsequent time intervals when we have already issued updates to the user, these would also need to be considered when estimating the novelty of an update summary.

4.1 Methodology

To perform the actual prediction of θ_t we use a supervised regression model that combines multiple features extracted from the update summary S_t and the previously issued sentence updates U_t, describing the event prevalence and novelty within S_t. Under this approach, a series of training instances – each comprised of an update summary S_t, previously issued sentence updates U_t and a pre-calculated optimal θ_t value – are used to learn a model that predicts θ_t on a continuous scale. This model can then be applied to predict θ_t values for each time interval t for unseen events.

To produce this supervised model, there are two prerequisites. First, a set of *prediction features* about each update summary S_t, used to distinguish between intervals containing new information (for which we should predict a large θ_t value) and those without (where a low or zero θ_t should be predicted). Second, a *loss function* that balances the redundancy cost of issuing the top r sentences from an update summary as sentence updates, against the added value in terms of new information contained (the best r value found then becomes θ_t). We describe each prerequisite in the following two sections.

4.2 Prediction Features

To facilitate accurate predictions of θ_t, we define 333 prediction features. For clarity, we describe each feature with respect to three distinct properties of that feature, namely: the aspect modelled; the input source; and the feature depth. We summarize these three properties below.

Aspect Modelled: Represents the property of the input that the feature aims to capture. Our features each belong to one of three broad aspects, namely: *prevalence features*; *novelty features*; or *quality features*. Prevalence features measure the degree to which the event (as specified in the query) is represented in the input, while novelty features estimate the extent to which the input contains previously unseen information. On the other hand, quality features measure the writing quality and cohesiveness of the input. The idea of using quality features for rank-cutoff prediction is that should the sentences within the input be of poor quality, then they are less valuable for inclusion and hence a smaller θ_t should be selected.

Input Source: Denotes whether the feature is calculated using the update summary S_t, the previously issued sentence updates U_t, or both.

Feature Depth: Each feature is defined at the level of the update summary S_t to a rank r. However, there are two ways that a feature can be calculated: via aggregation of sentence-level evidence; or when considering all terms from sentences between rank 1 and r as belonging to a single document. For instance, consider the tf-idf similarity of a sentence with respect to the user query. The average of

Aspect	Input	Depth	Description	Count
Prevalence	S_t	AVG[1-10]	Sentence contains entire topic	10
	S_t	AVG[1-10]	Unigram/Bigram/Skip Bigram (4-word window) overlap	30
	S_t	AVG[1-10]	Unigram/Bigram tf/tf-idf similarity	40
	S_t	AVG[1-10]	Topic overlap with sentence subject/object	20
	S_t	AVG[1-10]	Topic term overlap that are Foreign words/Adjectives/Nouns/Verbs/Cardinal Numbers	50
	S_t	VD[$\|U_t\|$]	Overlap (exact/unigram/bigram) between the query Q and topic models [12] extracted from S_t (count/weight/max)	9
Novelty	S_t and U_t	AVG[1-10]	Levenshtein/Cosine/Monge Elkan pairwise distance to sentences in U_t (max/min/avg.)	90
	U_t	VD[$\|U_t\|$]	Overlap (exact/unigram/bigram) between the query Q and topic models [12] extracted from U_t (count/weight/max)	9
Quality	S_t or U_t	AVG[10]	Relative/Absolute position within the document	4
	S_t or U_t	AVG[10]	Is among the first 1/3/5 sentences	6
	S_t or U_t	AVG[10]	The sentence length with/without stopwords	4
	S_t or U_t	AVG[10]	The number of sentences in excess of 5/10 with/without stopwords	8
	S_t or U_t	AVG[10]	Unique unigram/bigram count	4
	S_t or U_t	AVG[10]	Average/total unigram/bigram tf/tf-idf	16
	S_t or U_t	AVG[10]	Cosine similarity with batch centroid	2
	S_t or U_t	AVG[10]	Average/sum of SumBasic scores using all/content-only terms [22]	8
	S_t or U_t	AVG[10]	Average/sum of SumFocus scores [27]	4
	S_t or U_t	AVG[10]	Total/Proportion/Coverage of the number of topic signature words contained [18]	6
	S_t or U_t	AVG[10]	tf-idf Mutual information between the document and sentence	2
	S_t or U_t	AVG[10]	Basic/improved sentences scorers from [7]	4
	S_t or U_t	AVG[10]	Contains URL/Verb/Phone Number	6
	S_t or U_t	AVG[10]	Contains/Proportion of content in brackets	4
Total				333

Table 1: Features used to predict θ_t.

these similarities over all sentences in the update summary S_t yields an estimate of the relatedness of that summary as a whole to the event (in this case, producing a feature similar to a post-retrieval query performance predictor [14]). Feature depth indicates whether a feature is calculated as the average of the scores for each sentence to a rank depth r (denoted 'AVG[1..r]'), or calculated from the virtual document produced by combining the sentences to rank depth r (denoted 'VD[1..r]').

Table 1 summarizes the features that we use to predict θ_t, separated by the feature aspects. Note that during learning, features calculated from different events and time periods will be compared. Hence, to make these features comparable, we apply a standard feature normalisation at the granularity of each time interval as follows:

$$norm(f, S_t) = \frac{f - min(f, S_t)}{max(f, S_t) - min(f, S_t)} \quad (1)$$

where f is a feature and $min(f, S_t)$ and $max(f, S_t)$ are the minimum and maximum observed scores for f across the sentences within S_t, respectively.

1: **Input**
   Q, $S_t = \{S^{t_0}, S^{t_1}, S^{t_2}...\}$
2: **Output**
   Training instances to use for learning I^e
define sentences previously selected, U_t=$\{\emptyset\}$
define training instance set, I^e=$\{\emptyset\}$
for each time interval t loop
   $\theta_t \leftarrow \text{argmax}_{r \in \{0..|S_t|\}}\, \ell(S_t, U, r)$
   $S_t[0..\theta_t] \leftarrow$ Crop S_t to rank θ_t
   $I^e \leftarrow$ Add predictionFeatures(S_t,U_t,Q) and optimal θ_t
   $U \leftarrow$ Add sentences $S_t[0..\theta_t]$
return I^e

Figure 4: Training instance generation pseudo-code.

4.3 Loss Function

To train the regression model using the features described above, we define a loss function ℓ that balances the redundancy cost of issuing the top r sentences from an update summary as sentence updates, against the added value in terms of new information contained. ℓ takes as input the update summary S_t, the previously issued updates U_t and a r value to try. It estimates the expected loss for issuing the top r sentences from S_t (denoted $S_t[1..r]$) as follows:

$$\ell(S_t, U_t, r) = H_{mean}\left(Gain(U \cup S_t[1..r]), Coverage(U \cup S_t[1..r])\right) \quad (2)$$

where *Gain* measures the amount of new information added by each sentence in $S_t[1..r]$ and U_t. *Coverage* measures the total number of information nuggets that are covered by $S_t[1..r]$ and U_t together out of all the information nuggets that an effective summary of the event should cover. H_{mean} is the harmonic mean between these two performance measures. Notably, *Gain* and *Coverage* can be considered analogous to the information retrieval precision and recall metrics, respectively within the context of summarization. *Gain* measures the number of 'relevant' updates (where relevance incorporates additional novelty, quality and timeliness components) issued over all updates, while *Coverage* measures the number of 'relevant' updates issued over the total number of 'relevant' updates available. Hence, the cost function can be seen as playing a similar role to the popular F_1 measure [28], penalising systems that place too much focus on either *Gain* or *Coverage*. For our later experiments, we use the official TREC 2013 Temporal Summarization track [13] measures, namely Expected Latency Gain (ELG) for the *Gain* component and Latency Comprehensiveness (LC) for the *Coverage* component. The optimal r value for the current update summary can then be calculated as follows:

$$\theta_t = \underset{r \in \{0..|S_t|\}}{\text{argmax}}\ \ell(S_t, U_t, r) \quad (3)$$

Having defined the feature set and loss function that we use, given an event e we can generate a set of training instances I^e, where each instance represents a time interval within e. Importantly, since the loss function is dependant upon the previously selected sentences U_t to calculate the optimal θ_t, we generate training instances in a greedy manner. In this case, we assume that at each time interval, the optimal θ_t is selected. Figure 4 illustrates the training instance generation process, while in the next section we define the dataset, training data and measures we use within our evaluation.

Event e		2012 Aurora shooting
Representation (Query) Q		'colorado shooting'
Time Range		Start: 20 Jul 2012 06:38:00 End: Mon, 30 Jul 2012 06:38:00
Nuggets	Nugget Time	Nugget Text
1	20 Jul 2012 13:06:05	At least 12 people were shot in the city of Aurora near Denver, Colorado.
2	20 Jul 2012 13:17:46	One person has been arrested but police did not have any other immediate details.
3	20 Jul 2012 13:39:19	The gunman is believed to have killed 14 people and injured 50.
4	20 Jul 2012 15:13:39	the suspect identified as a 24-year old male is believed to have acted alone
5	21 Jul 2012 06:09:03	Holmes bought approximately 6,000 rounds of Ammunition in last 60 Days.
6	22 Jul 2012 13:50:02	Many police departments and theatres across the country increased security after the attack.

Table 2: Example event '2012 Aurora shooting' (6/205 nuggets shown).

5. EXPERIMENTAL SETUP

Dataset: To evaluate our approach for IUS, we use the TREC 2013 Temporal Summarization sequential update summarization (SUS) task dataset. Notably, SUS is a stream processing task, where the aim is to select sentences from the stream. This contrasts with IUS, which focuses on selecting sentences from update summaries already produced from the underlying stream. The SUS 2013 dataset uses the 2013 Knowledge-Based Acceleration (KBA) stream corpus – containing over 1 billion timestamped Web documents (e.g. news articles, blogs and forum posts) spanning the period of Oct 2011 to Feb 2013.

Events: The dataset also contains 10 topics representing long-running events from during this period, along with a user query Q representing each. Each event has a pre-defined 10-day timespan. A typical 10-day period contains around 8-9 million documents. Each event also has a predefined set of information nuggets (describing key pieces of information that a summary about the event should cover) and nugget timestamps (indicating approximately when that information emerged) that were manually extracted from the updates made to the Wikipedia page about each event. Table 2 illustrates an example event.

Update Summary Generation: In contrast to the SUS task, our IUS approach takes as input 'update' summaries produced over time intervals from the underlying document stream. These update summaries are produced as follows. First, we cluster the 8-9 million documents published from each 10-day event period into 1 hour intervals, i.e. our intervals $t = \{t_0, t_1, t_2...\}$. This results in 240 intervals t per event. We index each document cluster, creating 240 document indices. Stopword removal and Porter stemming is applied. For each document index, we use the event representation Q as a query, retrieving the top 10 documents relating to the event using the BM25 document weighting model, and applying query expansion with the Kullback-Leibler (KL) term weighting model to improve search effectiveness. For each set of 10 documents retrieved, we process them using an update summarization approach to produce an update summary comprised of 10 sentences. As they have previously been shown to be effective [27], we adopt a state-of-the-art learning-to-rank approach for update summarization. In particular, we train a LambdaMART [35] list-wise learning-to-rank model separately from our IUS approach, using the 63 events from the Document Understanding Conference (DUC) datasets from 2005/2006/2007 that have Semantic Content Unit (SCU) marked sentences [8].[4] We use 57 sentence features based on those reported in [33], using NDCG as the target measure. Finally, to remove redundancy and following [9], we then employ an upper bound υ on the permissible similarity between sentences, removing any sentences that exceed this upper bound. We calculate the pairwise similarity between each sentence in a greedy time-ordered manner against those ranked above it. Similarity is calculated via matching terms, weighted using the tf-idf term weighting model. A Wikipedia snapshot predating the events is used to obtain the background term frequencies. Sentences with similarity scores exceeding υ are discarded. υ was optimised for NDCG on a separate corpus using 0.1 increments, resulting in an upper bound tf-idf similarity of 0.5. We then take the top 10 remaining sentences to form each update summary (i.e. summary length is 10 sentences).

Sequential Update Summarization Evaluation: For the SUS task, participants performed an extractive summarization for each event over that event's timespan, returning a timeline of sentences from the corpus summarising that event with confidence scores for each. Unlike prior summarization tasks such as MDS [10], the identification of on-topic documents from the stream was left to the participants. The top 60 sentences (by confidence score) returned by each participating system were then sampled to form an assessment pool. TREC assessors manually compared each sentence against the information nuggets extracted for the event, creating a matching between each sentence and zero or more nuggets. Assessors were also allowed to define new nuggets from sentences they assessed, if a sentence was on-topic but no existing nugget matched. A sentence returned by a participant system is only considered relevant if it covers information nuggets not already covered by previously returned sentences, i.e. the update contains new information.

Tackling Assessment Incompleteness: Importantly, we observed little (<1%) overlap between the top sentences identified by our system and those assessed as matches during the TREC task, i.e. assessment completeness [3] was low. To counteract this, we used crowdsourcing to generate an additional set of assessments to augment those created by the TREC assessors. In particular, for each of the 10 events, we pooled the 10 sentences contained within the update summaries we take as input for each 1 hour interval over the 10 days, resulting in 22,424 sentences (10 events * 240 hours * 10 sentences[5]). Next, we assessed the 22,424 sentences in three stages. First, an IR expert manually labelled each sentence as relevant or not to the event that they summarised, based upon the information nuggets defined by the TREC task. 6,073 of the 22,424 sentences were labelled as relevant (27.1%). Second, the 6,073 relevant sentences were subject to duplicate detection. If multiple sentences were textually identical, only the earliest was kept, resulting in 4,463 unique sentences. For later evaluation, the assessment labels obtained for each unique sentence are propagated to its duplicates. Third, the remaining 4,463 relevant and unique sentences were matched against the information nuggets for their respective events (as per the TREC task) using crowdsourcing. For example, for the event illustrated in Table 2, the sentence 'theatres across the country are increasing security after 12 people were shot in Denver' would be labelled as covering nugget 1 and 6.

Crowdsourced Assessments: For each of the 4,463 sentences, three crowdsourced assessors matched that sentence

[4]These DUC datasets total 45,683 labelled sentences.

[5]Not all hours returned 10 sentences, hence the total number of sentences is less than 24,000.

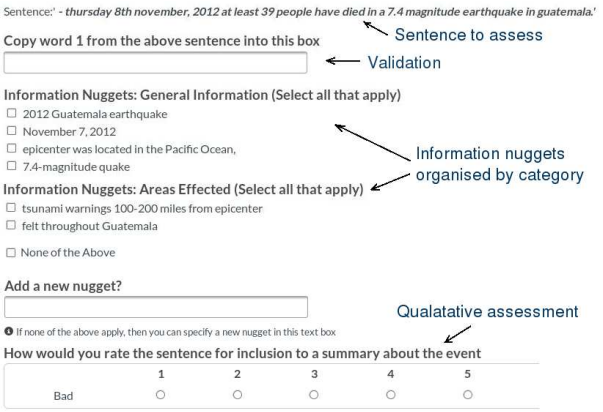


Figure 5: Sentence assessment interface (only a subset of the information nugget options are shown).

against the list of information nuggets for the event, in a similar manner to the TREC assessors. Crowdsourced assessors were also able to specify new information nuggets, when the existing nuggets did not provide a good match to the information provided in a sentence. Finally, each assessor also rated the quality of each sentence for inclusion into a summary about the event on a 5-point Likert scale. A sentence is considered to match a nugget if two or more assessors select that nugget. A sentence's quality rating is the average of the ratings assigned. Figure 5 illustrates the sentence assessment interface used.

Crowdsourcing Setup and Statistics: We use the CrowdFlower crowdsourcing service to recruit workers for our evaluation. Following best practices in crowdsourcing [25], we have three individual assessors match a sentence against the information nuggets. Each assignment involves the assessment of 5 sentences. We paid US $0.08 for each assignment. To detect bots and spammers, for each sentence, the assessor enters a word from a specified position within that sentence into a pre-provided text box (acting as a form of captcha). Workers who fail this test more than three times are barred from completing more assignments. Workers are also subject to an entry test where they perform a single assignment, their performance is measured against a gold standard [15]. A worker's agreement with the gold standard must exceed 70% to qualify for the task. The total crowdsourcing cost was US $305.64. Automatic validation resulted in the rejection of 2.45% of assignments. Inter-worker agreement was 50.4% under Fleiss κ, indicating that the resultant work was of reasonably good quality.

Measures: We report the IUS performance using the official TREC 2013 Temporal summarization track (TREC-TS) measures – Expected Latency Gain (ELG) and Latency Comprehensiveness (LC). ELG measures for each update issued to the user, the sum of latency-discounted relevance of the nuggets for which that update is the earliest issued, in doing so it accounts for the proportion of updates that contain new information (nuggets) and the timeliness of that content with respect to when those nuggets emerged. LC measures the coverage of an event by the updates issued with an additional timeliness component, i.e. it measures nugget recall over all updates issued, where the score for a nugget is discounted if it is reported late. Finally, we also report the macro harmonic mean of ELG and LC, denoted ELG/LC Macro F_1, as a combined measure.

Retrospective vs. Live: Since IUS is a time-oriented task, there are two settings under which the above measures can be calculated. First, in a retrospective manner, where performance is calculated at the end of each 10-day period based upon the final aggregated output, denoted *Retrospective*. This provides a measure of summary effectiveness at the close of the event. However, retrospective effectiveness may not reflect the performance that the end-user sees, since they will access the summaries produced while the event is ongoing. Hence, we also report summary effectiveness as an average of the performance calculated incrementally each hour across the 10-days, denoted *Live* and examine how summary performance varies over time in more detail in Section 6.3.

Training Regime: We train our IUS approach using a 10-fold-cross validation.[6] We test both the model produced by a classical linear regression learner and that produced by a learner based on Model Trees [32]. When reporting the most influential features using a feature ablation study, we average the performance loss observed across all 10 folds.

Baselines: We compare our IUS approach with traditional update summarization approaches that produce fixed-length summaries. In particular, we compare against SumBasic [22], SumFocus [27], and the Classy [7] sentence scorers configured as per their original papers, in addition to the update summaries produced by the learning-to-rank model used as input for our proposed IUS approach, denoted LambdaMART. For these baselines, we report performance when selecting one or three sentences from each time interval.[7] For reference, the TREC 2013 best system under the ELG/LC Macro F_1 measure has a reported performance of 0.1673. However, in contrast to the TREC systems, it is of note that performance under our approach is reported using the expanded assessment set. Hence, it is not possible to directly compare performances.

6. RESULTS

In this section, we investigate whether by adaptively altering the number of updates to issue about an event over time we can enhance the effectiveness of IUS over the baselines described earlier. In particular, we examine the following three research questions, each in a separate (sub)section:

- How does our adaptive IUS approach compare with baseline update summarization approaches that produce a fixed length summary? (Section 6.1)
- What are the most effective types of feature for predicting the rank-cutoff? (Section 6.2)
- How does summary effectiveness vary over time? (Section 6.3)

6.1 Adaptation using Prevalence and Novelty

We begin by evaluating the performance of our proposed IUS approach against a series of traditional update summarization systems from the literature. If our proposed IUS approach outperforms these baselines, then we will be able to conclude that predicting the number of sentences to select from each update summary is more effective than selecting a fixed number of top sentences only. Furthermore, we will have shown that our machine learned approach is able to learn how to predict an effective cutoff based upon the event prevalence and novelty within the update summaries produced over time.

Table 3 reports the performance of the SumBasic [22]; SumFocus [27]; Classy [7] and learning-to-rank-based [33]

[6]Equivalent to a leave-one-out setting over the 10 events.
[7]Selecting more than 3 sentences reduces overall performance for this task due to decreased ELG.

Row	Approach					TREC-TS 2013 Dataset+					
	Update summarization		IUS		Statistics	Retrospective			Live		
	Ranking	Training	Cut-off Selection	Training	Avg. Summary Length	ELG	LC	ELG/LC Macro F_1	ELG	LC	ELG/LC Macro F_1
4	SumBasic	None	Top 1	None	240	0.0260	0.0683	0.0376	0.0246	0.0489	0.0327
5	SumBasic	None	Top 3	None	685	0.0215	0.1050	0.0356	0.0381	0.0797	0.0516
6	SumFocus	None	Top 1	None	240	0.0267	0.0721	0.0390	0.0255	0.0519	0.0342
7	SumFocus	None	Top 3	None	685	0.0176	0.1134	0.0304	0.0315	0.0878	0.0464
8	Classy	None	Top 1	None	240	0.0665	0.1891	0.0984	**0.1420**	0.1608	0.1508
9	Classy	None	Top 3	None	685	0.0417	0.3606	0.0748	0.0960	0.3132	0.1470
10	LambdaMART	DUC	Top 1	None	240	0.0479	0.0590	0.0529	0.0928	0.0450	0.0607
11	LambdaMART	DUC	Top 3	None	685	0.0245	0.1317	0.0414	0.0516	0.1071	0.0696
12	LambdaMART	DUC	IUS Adaptive	Linear Regression (10-folds)	591	0.0224	**0.5163**△	0.0429	0.0412	**0.3888**△ ▲	0.0745△
13	LambdaMART	DUC	IUS Adaptive	Model Trees (10-folds)	260	**0.1292**	0.3289△	**0.1856**△	0.1321△	0.2643△	**0.1762**△ ▲
14	LambdaMART	DUC	Oracle	N/A	60	0.3113	0.6231	0.3867	0.3440	0.5970	0.4365

Table 3: Temporal summarization performance for the baseline approaches and our proposed adaptive models under Expected Latency Gain (ELG), Latency Comprehensiveness (LC) and the ELG/LC mean. Statistically significant improvements over LambdaMART (Top 3) and the best baseline (for each measure) under the paired t-test p<0.05 are denoted △ and ▲, respectively.

extractive MDS approaches with redundancy removal based upon a per-sentence upper-bound similarity threshold, where the top 1 or 3 sentences are selected during each time interval to issue as updates. Table 3 also reports the performance of our proposed IUS approach (denoted IUS Adaptive) when selecting sentences from the learning-to-rank-based update summaries and trained using the Linear Regression and Model Trees learners when trained using 10-fold cross validation. The oracle performance (where the optimal rank-cutoff was selected) is also reported. Performance is measured in terms of Expected Latency Gain (ELG), Latency Comprehensiveness (LC), and the combined ELG/LC Macro F_1 measure under both retrospective/live settings.

From Table 3, we observe the following three points of interest. First, comparing the baseline approaches that select a fixed number of sentences to issue as updates (rows 4-11), we see that the most effective underlying update summarization approach under ELG/LC Macro F_1 is Classy [7]. Indeed, it is interesting to note that the generative Classy model outperforms the LambdaMART model trained on DUC data (when using a fixed cutoff). This would indicate that the effective sentence ranking features differ between the DUC newswire documents and the Web documents retrieved from the KBA corpus being summarized here. Second, comparing the LambdaMART baseline (rows 10-11) with our proposed IUS approach when trained using Linear regression (row 12) under ELG/LC Macro F_1, we see that our approach marginally under-performs the baseline when evaluating retrospectively, but outperforms that baseline by a small (but statistically significant) margin under the live setting. This lack of substantial improvement over the baseline indicates that a simple linear function is not sophisticated enough to predict an effective rank-cutoff for IUS. However, comparing the LambdaMART baseline (rows 10-11) with our approach when the Model Trees learner is used (row 13), we see that our approach outperforms LambdaMART by a large margin under both retrospective (13.7% absolute ELG/LC Macro F_1) and live (11.6% absolute ELG/LC Macro F_1) settings. Indeed, the performance of the Model Trees learner is over 47%/14.4% more effective under retrospective/live ELG/LC Macro F_1 than the best baseline (Classy Top 1 - row 8). Third, comparing the baselines (rows 4-11) with the oracle performance (row 14), we observe that improvements over the best baseline of up-to 28.8%/30.6% absolute ELG/LC Macro F_1 are possible - highlighting the promise of the approach and the scope for future improvement.

To answer our first research question, as illustrated by the performance of our proposed IUS adaptive approach, by adaptively altering the number of sentences to select from the update summaries, we are able to outperform the best baseline by up-to 47% (under ELG/LC Macro F_1). This shows that it is possible to predict the number of sentences to issue as updates to the end-user over time. In the next section, we analyse the most influential feature subsets.

6.2 Top Prediction Features

We next examine the features subsets that contribute most to the prediction of an effective rank-cutoff. Identifying the most influential features can provide insights into what properties an update summary should have if we are going to select sentences from it to issue to the user as updates. To identify the most influential features, we perform an ablation study, where we remove a subset of the features and re-evaluate the performance of the model produced. When an influential feature subset is removed, we would expect performance to decrease. In contrast, when an uninfluential feature subset is removed, we would expect performance to remain approximately unchanged.

Table 4 reports the ELG/LC Macro F_1 performance of our IUS approach using the Model Trees learner when leveraging the feature subsets described previously in Table 1. In particular, we report performance when we ablate each feature subset for both the live and retrospective settings. From Table 4, we observe the following. First, as a sanity check, comparing the performance of the models produced under the two evaluation settings, we observe that performance decreases when features are removed in all cases, following our expectations. Indeed, the performance decreases observed as features are ablated are statistically significant (paired t-test p<0.01) with respect to using all features under the live evaluation setting.

In terms of the most influential feature subsets divided by aspect, we see that the quality features contribute the most, followed by the novelty and prevalence features. To illustrate why this is the case, Figure 6 shows the number of sentences selected by the ablated models produced for the Pakistan factory fires event during the first 24 hours of that event. As we can see from Figure 6, in comparison to using all features (front-most curve), using a model without quality features (rear-most curve) follows the same pattern of selection (sentences are selected during the same hour intervals). However, it selects many more sentences during each interval. This results in irrelevant or redundant sentences being issued as updates from deeper within the update summaries (see the example from earlier in Section 4).

To answer our second research question, all of the feature groups that we proposed in Section 4.2 are influential, since all reduce prediction performance when removed. The most influential of these were the quality features, followed by

Property	Feature Sub-set	Retrospective ELG/LC Macro F_1	Live ELG/LC Macro F_1
All Features		0.1856	0.1762
Aspect	Prevalence	0.1694	0.1457▼
	Novelty	0.1249	0.1279▼
	Quality	0.0977	0.1085▼
Input	Features from S_t	0.1469	0.1338▼
	Features from U_t	0.0856	0.1021▼
Depth	AVG Features	0.0917	0.0957▼
	VD Features	0.1287	0.1218▼

Table 4: Influential IUS features identified. ▼ indicates a statistically significant decrease (paired t-test p<0.01) with respect to using all features.

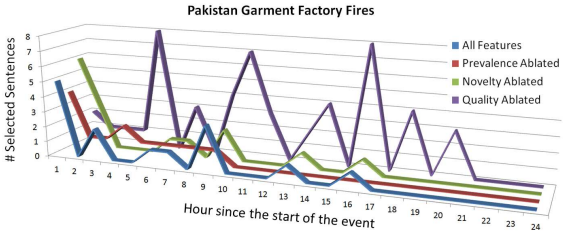


Figure 6: The number of selected sentences over time using our approach when Prevalence, Novelty and Quality features are ablated.

the prevalence and novelty features. In the next section, we analyse how IUS performance varies as an event evolves.

6.3 Evaluating Performance Over Time

In Section 6.1, we reported the combined performance of our IUS system under both retrospective and live settings. However, it is also important to examine how summary performance varies over time. To evaluate this, we report the ELG/LC Macro F_1 performance of our proposed approach in comparison to the oracle system at each hour interval t, on a per-event basis. Importantly, ELG/LC Macro F_1 for a time interval t measures summarization performance based upon sentences emitted both prior to and during t. Performance will increase as sentences covering new relevant information nuggets are emitted, but will decrease if off-topic or redundant sentences are emitted. Due to space limitations, we report only two of the ten events that illustrate two common performance distributions observed.

Figure 7 (a) and (b) report the performance distribution of our approach (Model Trees with leave-one-out training) and the oracle under ELG/LC Macro F_1 for each hour interval. The events reported are the Wisconsin Sikh Temple shooting and the Buenos Aires Rail Disaster. From Figure 7 (a) and (b), we make the following observations. Considering the oracle performance, for both events, we see a sharp increase in performance within the first few hours of the event. This shows that a large proportion of the important information appears during the first few hours. Next, examining the performance of our approach, we see that for the Wisconsin Sikh Temple shooting example (Figure 7 (a)), starting from the second day, IUS performance begins to degrade. This shows that the model is continuing to issue more sentence updates over time even though little new information is available. From this, we can make two conclusions. First, the predictor appears to make the most prediction errors when the optimal rank-cutoff θ_t is 0. Second, the learned model is more effective early-on in an event's lifetime than later in that lifetime. The performance distribution of Figure 7 (a) is common to 6 of the 10 events in our dataset. Hence, assuming that users are more interested in tracking an event early in that event's lifetime, the performance of our approach reported earlier in Table 3 – that considers all updates issued over the 10 days – may be an under-estimate. In contrast, from Figure 7 (b), we see a different performance distribution. In this case, performance increases early as initial sentence updates are issued. However, after around 12 hours, no new updates are issued. Contrasting this distribution to the oracle for the same event, we see that new information is still emerging for this event after the 12 hour period, albeit at a slow rate. This indicates that the learned model lacks sensitivity when new information arrives slowly, i.e. when rank-cutoff θ_t values in the range 0-1 are predicted. In answer to our third research question, we conclude that our IUS approach can be effective over time, as illustrated by its overall effective performance of the Buenos Aires Rail Disaster event and its early performance for the Wisconsin Sikh Temple shooting event. However, we also observed that IUS performance can degrade once an event ends, as for some events, redundant sentences continue to be selected. To illustrate the quality of the summaries produced by our approach over time, Table 5 shows an extract from the summaries produced for the Wisconsin Sikh Temple shooting and Buenos Aires Rail Disaster events discussed above.

7. CONCLUSIONS

In this paper, we introduced the task of incremental update summarization (IUS), which aims to select sentences from update summaries about an event over time to be issued to a user tracking that event. We proposed a new approach to this task that treats IUS as a rank-cutoff selection problem from the update summaries. We trained a regression model comprised of over 330 features that balances the cost of selecting a deeper cutoff in terms of returning redundant content, against the risk of missing important information by selecting a shallower cutoff. Through empirical evaluation using the TREC 2013 Temporal Summarization dataset expanded with over 22,000 additional assessments, we show that our proposed approach can improve the IUS performance by 47% in comparison to effective update summarization baselines from the literature. Hence, we conclude that adaptively altering the number of top sentences to select from the update summaries over time is critical to achieve an effective IUS performance.

From the results of this evaluation as a whole, we believe that the generation of timely updates about long-running events is still a challenging problem. To effectively summarize events over time, approaches need to adapt to changes at the rate at which events are reported, accounting for periods when no new information emerges. For future work, we aim to investigate how to increase the sensitivity of the supervised prediction model when small amounts of new information arrive over time, as well as to further examine IUS from an information filtering perspective.

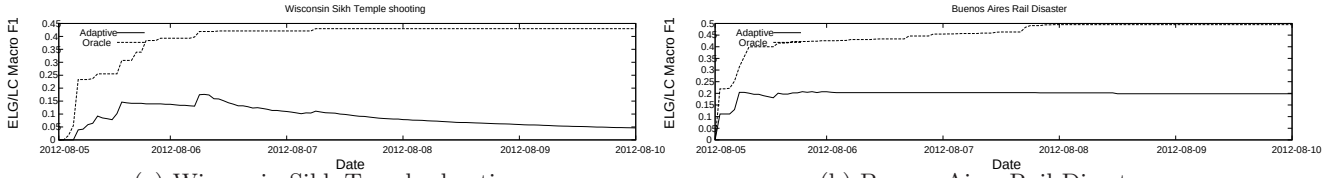
8. ACKNOWLEDGEMENTS

This work has been carried out in the scope of the EC cofunded projects SMART (FP7-287583) and SUPER (FP7-606853).

9. REFERENCES

[1] J. Allan, R. Gupta, and V. Khandelwal. Temporal summaries of new topics. In *Proc. of SIGIR*, 2001.

[2] A. Arampatzis, J. Kamps, and S. Robertson. Where to stop reading a ranked list?: Threshold optimization using truncated score distributions. In *Proc. of SIGIR* , 2009.



(a) Wisconsin Sikh Temple shooting (b) Buenos Aires Rail Disaster

Figure 7: IUS performance of our proposed approach and the oracle over time for the Wisconsin Sikh Temple shooting and Buenos Aires Rail Disaster events.

Wisconsin Sikh Temple shooting		Buenos Aires Rail Disaster	
Time Issued	Sentence Update	Time Issued	Sentence Update
2012-08-05 19:00	Recent related news breaking: multiple people shot at Wisconsin Sikh temple Milwaukee, wisc.	2012-02-22 16:00	Some 550 others were injured the breaking news dashboard officials fear up to 40 dead following commuter train crash in Buenos Aires, #Argentina.
2012-08-05 19:00	CNN reports that a gunman was killed by a police officer.	2012-02-22 18:00	Argentina train crash in Buenos Aires 'kills dozens'
2012-08-05 21:00	Hours of uncertainty followed as police in tactical gear and carrying assault rifles surrounded the temple with armoured vehicles and ambulances.	2012-02-22 19:00	They said the train was unable to stop, presumably due to faulty brakes, and it slammed into the buffers inside the centrally located Once station.
2012-08-05 22:00	The shootings happened before 10:30 a.m., when witnesses said several dozen people were gathering at the Sikh temple of Wisconsin for a service.	2012-02-22 23:00	The derailment of an inter-city train has resulted in the death of at least 40 people and the injury of at least 550 people in Buenos Aires early Wednesday.
2012-08-05 22:00	"we never thought this could happen to our community," said Devendar Nagra, 48, mount pleasant, whose sister escaped injury by hiding in the temple's kitchen.	2012-02-23 00:00	The crash, the second-worst in Argentina's history, may have been caused by brake failure, transport secretary Juan Pablo Schiavi told local media shortly after the crash.

Table 5: Sentence updates issued by our IUS approach for the Wisconsin Sikh Temple shooting and Buenos Aires Rail Disaster events

[3] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proc. of SIGIR*, 2004.

[4] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. of SIGIR*, 1998.

[5] A. Celikyilmaz and D. Hakkani-Tur. A hybrid hierarchical model for multi-document summarization. In *Proc. of ACL*, 2010.

[6] H. L. Chieu and Y. K. Lee. Query based event extraction along a timeline. In *Proc. of SIGIR*, 2004.

[7] J. M. Conroy, J. D. Schlesinger, and J. Goldstein. Classy tasked based summarization: Back to basics. In *Proc. of DUC*, 2005.

[8] T. Copeck, D. Inkpen, A. Kazantseva, A. Kennedy, D. Kipp, V. Nastase, and S. Szpakowicz. Leveraging DUC. In *Proc. of DUC*, 2006.

[9] T. Copeck, A. Kazantseva, A. Kennedy, A. Kunadze, D. Inkpen, and S. Szpakowicz. Update Summary Update. In *Proc. of TAC*, 2008.

[10] H. T. Dang and K. Owczarzak. Overview of the TAC 2008 Update Summarization task. In *Proc. of TAC*, 2008.

[11] G. Erkan and D. R. Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proc of EMNLP*, 2004.

[12] A. E. Gelfand and A. F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

[13] Q. Guo, F. Diaz and E. Yom-Tov. Updating users about time critical events. In *Proc of ECIR*, 2013.

[14] B. He and I. Ounis. Query performance prediction. *Information Systems Journal*, 31(7):585–594, 2006.

[15] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proc. of CHI*, 2008.

[16] C. Li, X. Qian, and Y. Liu. Using supervised bigram-based ilp for extractive summarization. In *Proc. of ACL*, 2013.

[17] L. Li, K. Zhou, G-R. Xue, H. Zha, and Y. Yu. Enhancing diversity, coverage and balance for summarization through structure learning. In *Proc. of WWW*, 2009.

[18] C.-Y. Lin and E. Hovy. The automated acquisition of topic signatures for text summarization. In *Proc. of ACL*, 2000.

[19] C.-Y. Lin and E. Hovy. From single to multi-document summarization: A prototype system and its evaluation. In *Proc. of ACL*, 2002.

[20] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *Proc. of ACL*, 2011.

[21] A. Nenkova and K. McKeown. A survey of text summarization techniques. *Mining Text Data*, Springer, 2012.

[22] A. Nenkova, L. Vanderwende, and K. McKeown. A compositional context sensitive multi-document summarizer: Exploring the factors that influence summarization. In *Proc. of SIGIR*, 2006.

[23] Y. Ouyang, W. Li, and Q. Lu. An integrated multi-document summarization approach based on word hierarchical representation. In *Proc. of IJCNLP*, 2009.

[24] D. R Radev, H. Jing, M. Styś, and D. Tam. Centroid-based summarization of multiple documents. *Information Processing & Management Journal*, 40(6):919–938, 2004.

[25] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proc. of EMNLP*, 2008.

[26] R. Swan and J. Allan. Automatic generation of overview timelines. In *Proc. of SIGIR*, 2000.

[27] K. Toutanova, C. Brockett, M. Gamon, J. Jagarlamudi, H. Suzuki, and L. Vanderwende. The Pythy summarization system: Microsoft Research at DUC 2007. In *Proc. of DUC*, 2007.

[28] C.J. van Rijsbergen. *Information Retrieval, 2nd edition*. Butterworths, London, 1979.

[29] S. Wan and C. Paris. Experimenting with clause segmentation for text summarization. In *Proc. of TAC*, 2008.

[30] D. Wang and T. Li. Document update summarization using incremental hierarchical clustering. In *Proc. of CIKM*, 2010.

[31] D. Wang, T. Li, S. Zhu, and C. Ding. Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proc. of SIGIR*, 2008.

[32] I. Wang and H. Witten. Introduction of model trees for predicting continuous classes. In *Proc. of ECML*, 1997.

[33] L. Wang, H. Raghavan, V. Castelli, R. Florian, and C. Cardie. A sentence compression based framework to query-focused multi-document summarization. In *Proc. of ACL*, 2013.

[34] F. Wei, W. Li, Q. Lu, and Y. He. Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization. In *Proc. of SIGIR*, 2008.

[35] Q. Wu, C. Burges, K. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Information Retrieval Journal*, 13:254–270, 2010.

[36] R. Yan, X. Wan, J. Otterbacher, L. Kong, X. Li, and Y. Zhang. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *Proc. of SIGIR*, 2011.

[37] J. Zhang, X. Cheng, G. Wu, and H. Xu. AdaSUM: An adaptive model for summarization. In *Proc. of CIKM*, 2008.