Assorted Textual Features and Dynamic Push Strategies for Real-time Tweet Notification

Kathy Lee[*]**, Ashequl Qadir, Vivek Datla, Sadid A. Hasan,**
Joey Liu, Aaditya Prakash[†]**, Oladimeji Farri**
Artificial Intelligence Laboratory, Philips Research North America, Cambridge, MA, USA
{kathy.lee_1,ashequl.qadir,vivek.datla,sadid.hasan}@philips.com
{joey.liu,aaditya.prakash,dimeji.farri}@philips.com

Abstract

In this paper, we describe our systems and corresponding results submitted to the Real-Time Summarization (RTS) track at the 2016 Text Retrieval Conference (TREC). The task involved identifying relevant tweets based on a user's interest profile. In Scenario A of the task, tweets relevant to an interest profile were pushed to a live user in real-time. In Scenario B, a daily digest of relevant tweets was sent to a user. We submitted three automatic runs for each scenario. Our overall method for identifying relevant tweets was based on 1) automatically identifying key textual features from a set of interest profiles provided by the Track organizers, 2) expanding the textual phrases with their paraphrases, and 3) exploiting the features for message filtering and relevance measurement after novelty recognition. We experimented with different push strategies to decide when to deliver a message to a user. The evaluation results (by mobile and NIST assessors) show that our system ranked 3rd for Scenario A and 6th for Scenario B.

1 Introduction

Social media and online blogs are valuable sources of real-time information, but the volume, variety and velocity of generated data make it difficult to find the information one needs. Over the last decade, Twitter has emerged as a form of online social communication and networking platform with global popularity among people who like to write brief status messages and share information with others in these messages. Aided by the widespread adoption of smartphones, information sharing via Twitter has become convenient, instant and frequent.

Given the limited human cognitive capacity such that we can only consume a small amount of information at any instance, individuals often want to receive only information that is relevant to the topics of interest in real-time. These topics can be specific to an event, location, time, person, product, opinion, experience, etc. For example, someone may be interested in knowing opinions/reviews on a recently launched fitness watch, while another person may be interested in posts describing side effects similar to his/her experience with the same medications. With such specific interests, when new tweets or other social media contents are posted, these messages need to be identified as relevant and pushed to the individuals who wish to be updated on such topics.

Identifying relevant information is important, but is not the only challenge in this task. If many tweets are sent to a person in a day as relevant, it will likely constitute a cognitive burden for the user. So, the number of messages needs to be kept at a manageable quantity. In addition, the extent of relevance among several qualifying tweets needs to be computed before pushing the top ranked ones to the user, especially because new messages that are more relevant are sometimes generated at a later time in the day. Furthermore, relevant tweets pushed to the user should not be redundant.

We address all of these challenges in our submitted runs for the Real-Time Summarization (RTS) track at the 2016 Text Retrieval Conference (TREC). To determine the relevance of tweets to a user's interest, we automatically extract various categories of textual features (e.g., named entities, general noun

[*] The author is also affiliated with Northwestern University (kathy.lee@eecs.northwestern.edu).

[†] The author is also affiliated with Brandeis University (aprakash@brandeis.edu).

Topic/Profile ID: MB408
Title: amphetamines and ADHD
Description: Find tweets that discuss amphetamines and ADHD.
Narrative: The user's daughter has been diagnosed with attention deficit hyperactivity disorder (ADHD) and wants to follow the personal experiences of others that have used amphetamines in the treatment of ADHD. Relevant tweets discuss types of amphetamines used, adverse side effects, drug interactions, physical and psychological reactions, addiction, etc.

Topic/Profile ID: RTS2
Title: Zika in Ecuador
Description: Find updates on the current Zika crisis in the country of Ecuador.
Narrative: The user has family in Ecuador and wants to see how her family might be affected by the Zika crisis. She's interested in reports of new cases as well as measures being taken to control the outbreak.

Figure 1: Examples of interest profiles (health related topics).

phrases, phrases within quotations) from the user's interest profile, and expand the textual features using a paraphrase database. We determine the novelty of the tweets by comparing lexical and semantic similarity with those previously pushed to the user. For the push notification strategy, we present three methods that correspond to our runs in Scenario A: 1) strict threshold-based, 2) time-adjusted dynamic threshold-based, and 3) quota-restricted threshold-based message push. In scenario B, we use additional filtering strategies and relevant tweet identification methods to ensure tweet relevancy.

2 Problem Description

The task in the RTS track required participating teams to monitor the Twitter sample stream and identify relevant tweet posts with respect to a number of "interest profiles". There were two scenarios in the task with regard to when relevant messages will be delivered to a user. The following sections provide further details about the interest profiles and scenarios.

2.1 Interest Profiles

The track organizers provided a total of 203 interest profiles for this challenge. Among them, 51 profiles were assessed in the TREC-2015 Microblog Track (Lin et al., 2015), 107 profiles were retained from the same track that may still be current in terms of ongoing events and issues around the world, and 45 profiles were newly developed for the TREC-2016 RTS track. Each profile contains four fields that refer to a topic/profile id, title, description, and narrative. The topics or interest profiles are diverse and belong to many domains such as health, politics, sports, etc. Figure 1 presents some example profiles with interests in health related topics.

2.2 Scenario A: Push notifications

In this scenario, when a system identifies a relevant post, the Twitter post is immediately sent to the user's mobile phone via a push notification. The post should be relevant (on topic), timely (provide updates as close to the time of the actual event occurrence as possible), and novel (users should not be pushed multiple notifications that are essentially identical or simply retweets). For evaluation purposes, the pushed tweets are first sent to the TREC RTS evaluation broker (via a REST API), from where the messages are immediately delivered to the mobile phones of a group of assessors (users-in-the-loop) according to the platform described by Roegiest et al. (2016). The challenge ran for ten days.

Each participating team at the challenge was allowed to submit up to three runs for scenario A. Each run was only allowed to push up to ten tweets per day per interest profile. We submitted three runs for this scenario.

2.3 Scenario B: Email digest

For this scenario, a daily digest of relevant tweets is sent to the users at the end of a day. This scenario reflects the case that a user may want to receive a daily email digest that summarizes the facts and conversations that emerged during the day with respect to an interest profile. These messages also need to be relevant and novel, but timeliness is not as important as in scenario A because the digests were not sent until the end of the day. The digests are to include a batch of up to 100 ranked tweets per day per interest profile. Similar to scenario A, each participating team was allowed to submit up to three runs for scenario B. We submitted three runs for this scenario too.

2.4 Run Categories

For either scenario of the challenge, there were three categories. In the automatic run category, a system must operate without human intervention (such as, manually judging the quality of query expansion terms) before and during the evaluation period. In the manual preparation category, the system must operate without human input during the evaluation period, but human involvement is acceptable before the evaluation period (e.g., human examination of the interest profiles to add query expansion terms or manual relevance assessment on a related collection to train a classifier). And finally, in the manual intervention category, there were no limitations on human involvement before or during the evaluation period. (e.g., crowd-sourcing judgments, human-in-the-loop search, etc. were allowed). Our submitted runs for both scenarios fall under the automatic run category.

3 Run Descriptions for Scenario A

Since, for each interest profile, there is a limit imposed on the maximum number of tweets that can be sent to a user, identifying a relevant tweet is necessary but not sufficient. This is because, when a tweet is identified as relevant by the system, at that time it remains unknown whether or not a more relevant tweet will be created later in the day. So, if the daily quota of 10 tweets for an interest profile is used up early on, and later, more relevant tweets are generated, the new tweets cannot be pushed to the user. Conversely, if a relevant tweet is not instantly pushed, there may not be any more relevant tweets generated later in the day.

Due to the need for instant decision making in scenario A, we implemented the same approach to compute tweet relevancy and novelty across all three runs, but we used different push strategies for each run. Following subsections describe our methods for scenario A runs.

3.1 Learning Textual Features from Interest Profiles

We first pre-process the interest profiles to extract and expand a number of textual features from the interest profiles as follows:

3.1.1 Text Pre-processing

To normalize the content in the interest profiles, we initially transform all text to lowercase characters. We use Elasticsearch[1] as our back-end database for indexing the profiles (and also tweets) for faster retrieval.

3.1.2 Textual Feature Extraction

For each interest profile, we extract a number of textual features for measuring tweet relevancy. We use a total of seven categories of textual features. These categories are not mutually exclusive, and a phrase or word from an interest profile may belong to multiple categories. However, each of our textual feature categories captures different types of information focus in the text.

For extracting phrases, we use the NLTK chunker[2] and the NLTK interface of the Stanford Parser.[3] Tan et al. (2015) argued that texts in title tend to make more influential impact than the description and narrative. Consequently, many of our textual feature categories only consider the texts in profile title, whereas a few of them also consider other fields. Our textual feature categories are:

- **Title words**: we extract all unigrams (individual words) from the profile title after excluding stopwords and punctuations.

[1] https://www.elastic.co/products/elasticsearch

[2] http://www.nltk.org/

[3] http://www.nltk.org/api/nltk.tag.html#module-nltk.tag.stanford

- **Title phrases**: we extract all noun phrases and verb phrases that only appear in the title of an interest profile.
- **Noun phrases:** we identify all noun phrases from the title, description and narrative fields of the interest profiles.
- **Phrases within quotations**: we extract phrases from title, description, and narrative that appear within quotation marks. Intuitively, phrases within quotation carry special importance, and tweets that mention these phrases exactly, could be highly relevant to the profiles.
- **Named Entity Phrases**: we extract phrases that contain a named entity. For extracting named entities, we use the NLTK toolkit.
- **Location Named Entity Phrases**: We extract all named entity phrases that mention locations.
- **TF-IDF phrases from narrative**: We calculate TF-IDF scores for words in profile narratives, considering each narrative of an interest profile as a document. We take the top 10 words with the highest TF-IDF scores (excluding stopwords), and extract noun phrases and verb phrases that contain one of these high scoring TF-IDF words.

3.1.3 Feature Expansion with Paraphrases

A tweet may contain words or phrases that do not lexically match with the textual features that we extract from the interest profiles. Therefore, we expand the textual feature categories by including paraphrases of the extracted phrases so that phrases that are synonymous can contribute towards measuring relevance. We use the PPDB Paraphrase Database (Ganitkevitch et al., 2013) (L-size) for the paraphrase-based feature expansion. We do not expand the named Entity phrases and phrases within quotations. For the other four categories, we create four new categories with only the paraphrase terms. After feature expansion, we have a total of 11 categories of textual features.

3.2 Identifying Relevant Tweets

Once the textual features are extracted from the interest profiles and expanded with paraphrases, the next step is to monitor the Twitter feed to identify relevant messages. During this process, we filter out tweets using some constraints and measure their relevance with respect to the interest profiles as detailed in the following subsections.

3.2.1 Tweet Filtering

Twitter feed generates a massive number of tweets daily, and the majority of the tweets will have no relevance to the interest profiles. As we monitor the Twitter feed, we discard all non-English tweets using the language meta-field. For a given interest profile, we discard any tweet that has no common word with the title of the interest profile. We also require that at least half of the title phrases from an interest profile are mentioned in the tweet to give it further considerations.

Furthermore, if an interest profile has named entities in any of its fields, we require that at least one named entity phrase appears in the tweet. For example, if an interest profile wants to know about armed conflicts in Syria, and a tweet is about an armed conflict which did not take place in Syria, then there is no point in further evaluating the tweet. So for this interest profile, the system would require that the named entity 'Syria' is also mentioned in the tweet. Similarly, if there is a phrase within quotations in the interest profile, any tweet not having the exact same phrase is discarded.

3.2.2 Relevance Measurement

Once the filtering step is done, to determine tweet relevance with an interest profile, our method looks for the presence of different textual features in the tweet message. Instead of relying only on an exact match with a textual feature, our relevance measurement also takes partial matches into account.

Let, $T = \{t_i \mid 1 \leq i \leq K\}$ is the set of interest profiles. For each $t \in T$, there are a total of C categories of textual features. Let C_i be the set of textual features for the i^{th} textual feature category. For every textual feature category C_i of some interest profile $t \in T$, we first calculate a category relevance using the following:

$$relevance(x, C_i) = \sum_{c \in C_i} \frac{l_c^2}{n_c \times max_n(C_i)} \quad (1)$$

In equation (1), x is a tweet, and l_c is the maximum number of rightmost words from phrase c that appears in the tweet consecutively and in the same order. The reason for using word sequences that are the rightmost in phrase c is that the extracted textual features are most commonly noun phrases, in which case the rightmost word is typically a head noun and the words left to the head noun are typically noun or adjective modifiers. For example, for the phrase 'subway commuting problem', the word 'problem' is the head noun, and 'subway' and 'commuting' are modifiers.[4]

Here, n_c is the total number of words in c, and $max_n(C_i)$ is the maximum phrase length (in terms of words) among all of the phrases in C_i. The category relevance score for the category C_i would be 1.0 when the longest phrase in C_i would appear in a tweet. That is, for the same example, 'subway commuting problem' is rewarded more over 'commuting problem' for appearing in a tweet, and contributes with the highest score if it is also the longest phrase in its category.

Once the category relevance score is calculated, we find a weight for each category, and our final score for an interest profile $t \in T$ is the weighted sum of the category relevance scores, calculated by:

$$profile\_relevance(x) = \sum_{i=1}^{C} w_i \times relevance(x, C_i) \tag{2}$$

All profiles with the profile_relevance score above a threshold are then considered as candidate profiles for the tweet. For learning the weights w_i, we used tweets with their relevance judgements from the 2015 TREC Microblog track. For this, we use each category of textual features individually, and determine the weight that maximizes the Expected Gain (EG) (described in Section 5.2). We then normalize the weights so that they sum to 1.

3.2.3 Novelty Detection

Once the relevance of a tweet is established, we ensure that the tweet is novel, i.e., the content of the tweet has new information relative to tweets previously sent to the user for the same interest profile. Similar to last year (Hasan et al., 2015), we determine novelty by comparing ordered lexical and semantic overlap of the tweet content with previously sent tweets, using a textual similarity algorithm by Li et al. (2006). This algorithm computes pairwise text similarity between two given texts and returns a similarity score between 0 and 1. Similarity score 1 indicates the two text strings are exactly same. We used 0.65 as our threshold. As soon as the system finds a similar tweet with similarity score 0.65 or higher in the already-pushed tweets pool, it discards the tweet (because it is not novel), and, otherwise, the tweet is considered novel and pushed to the user (i.e., no similar tweet have been pushed for the interest profile). For novelty detection, we need to compare a new tweet against a pool of all pushed tweets for an interest profile. Thus the size of this pool increases as our system keeps pushing more tweets and, as a consequence, the novelty detection processing time per tweet also increases with time. To reduce this processing time, we ran several instances of the semantic similarity algorithm in parallel on the incoming tweets.

3.3 Relevant Tweet Delivery

In scenario A, it is also important to use effective push strategies because the number of tweets that can be sent to a user per day per interest profile is limited. We use three different push strategies for our three runs.

3.3.1 Strict Threshold-based Push Notification

For our run 1, we use a simple push strategy that prioritizes how relevant a tweet is to an interest profile, and only pushes a tweet when the system has determined that the message is highly relevant for an interest profile. In this method, to ensure strong relevancy, relevancy measurement threshold is set at 0.75 for all interest profiles. Any tweet for which the system measures a relevancy score of 0.75 or above is greedily pushed to the evaluation broker unless the daily limit of ten tweets per interest profile is already met. In run 1, our system pushed a total of 389 tweets.

3.3.2 Time-adjusted Dynamic Threshold-based Push Notification

One of the issues with the strict threshold-based method is that it uses a single uniform threshold for

[4]Although, some of the textual features we extract are also verb phrases, in this work we mainly focus on matching the noun phrases.

all of the interest profiles. But in reality, some topics may be more popular at the present time and many relevant tweets for these topics can be generated throughout the day. For example, tweets about *Zika virus* may get posted a lot more frequently than tweets about *ADHD*. Therefore a system can afford to wait for more relevant tweets for a popular topic, but not for a topic that is relatively less popular. Also, a uniform relevancy threshold may not be effective across all interest profiles.

To address this, in our run 2 for scenario A, we introduce *Time-adjusted Dynamic Threshold-based Push Notification* method. In this method, the system starts monitoring Twitter feed at the beginning of a day and pushes tweets above a uniform relevance threshold, but re-evaluates its expectations at mid-day for each interest profile individually.

During the first half of a day, the system runs identically as our method in run 1 that uses a strict uniform 0.75 threshold for all interest profiles. After 12 hours, the system checks what percentage of an interest profile's daily quota (i.e., 10 tweets per day per profile limit) has been met. If, for an interest profile, more than 50% of the daily quota is already met (i.e., $\#messages\_pushed \geq 5$), then the system expects that within the remaining time left for the day, it can expect to receive enough relevant tweets to fulfill its daily quota, and thus keeps the 0.75 threshold unchanged.

However, if the system discovers that, for an interest profile, 50% of the daily quota is yet to be met (i.e., $0 < \#messages\_pushed < 5$), then a medium relevance threshold (we use 0.6 as our medium relevance threshold) is set to allow for more tweets to be pushed during the remaining half of the day. If, for some interest profile, the system does not push any tweet during the first half of the day (i.e., $\#messages\_pushed = 0$), then for these interest profiles, the threshold is lowered further, and a weak relevance threshold is used for the remaining part of the day. We use 0.5 as our weak relevance threshold. In Run 2, our system pushed a total of 2,158 tweets.

3.3.3 Quota-restricted Threshold-based Push Notification

In our final push strategy in run 3 of the scenario A, we employ a *Quota-restricted Threshold-based Push* method. In this method, a tweet is pushed when its relevance score is above a weak relevance threshold (we use 0.5 in this case), but only until 50% of the quota is met. The remaining 50% is always reserved for tweets with a strong relevance threshold (we use 0.75 in this case). So, in this method, not more than five tweets can be pushed for an interest profile on a day that are within a relevance threshold range of 0.5 to 0.75, but up to 10 tweets can be pushed if they all have relevance score above 0.75. In Run 3, our system pushed a total of 1,506 tweets.

The thresholds for different push strategies were determined from the relevance scores that the system assigns on the 2015 TREC Microblog challenge data and the Expected Gain (EG) scores these thresholds obtain.

4 System Description for Scenario B

The challenges in scenario B are fundamentally different from the challenges in scenario A. In scenario B, the uncertainty that there may be more relevant tweets generated at a later time during the day is not considered, because all of the tweets for the day are already posted in Twitter. The challenge is to find tweets that are most relevant for a given interest profile, among all monitored tweets for the day. Therefore for scenario B, we shift our focus from applying different push strategies to identifying relevant tweets.

4.1 Daily Digest for Run 1

In run 1 of scenario B, we use a distinct tweet filtering strategy. For identifying messages that are relevant to an interest profile, we still require that at least one title word, or one named entity or phrase within quotation (if they are in the interest profile) must appear in the tweet, but rescind on the constraint that at least half of the title phrases must also appear in a relevant tweet. It is possible that in some cases, the number of important title phrases may account for more than half of the title phrases. On the other hand, a tweet may mention a paraphrase of a title phrase, or part of a title phrase may appear as a synonym in the tweet. For the latter, looking for exact matches would be limiting.

For each title phrase of an interest profile, we create a cluster of text alternatives (title phrase inclusive). Each text alternative can be further divided into one or more internal text blocks. These text alternatives are typically synonyms or paraphrases, but since it is not always possible to find paraphrases or synonyms of long and specific phrases, the phrases are divided into internal text blocks. We require that at least one members from each cluster of text alternatives must be present in the tweet message for the tweet to be relevant for the interest profile. A text alternative is considered present in the tweet if a member from each internal text block (that forms a text alternative) is also present in the tweet.

To illustrate this with an example, let's consider the title phrase "subway commuting problems" (from topic: MB299). Our basic idea here is that all components of this phrase must be present in a tweet for the tweet to be relevant for this profile. First, we need to find a set of alternative representations of this phrase. These may include: "issues arising from subway commute" or "problems of subway travel". We use the PPDB (large subset) paraphrase corpus to build our cluster of text alternatives. If such alternative phrases exist in the paraphrase database, then we consider these paraphrases as the members of the cluster, where each member has exactly one text block (i.e., the entire phrase).

However, being able to directly find such paraphrases is not always practical as the coverage of existing paraphrase databases are not extensive. In some cases, the original phrase may be too specific or too long to have its paraphrases in a database. Therefore, we break the phrase into smaller chunks to create a number of text blocks.

Our phrasal chunking is done by incrementally removing the leftmost words from the phrase, and building a text block for each that contains the word and its synonyms/paraphrases. This is because many of the meaningful title phrases are noun phrases, in which case the rightmost word tends to be the head noun, whereas the leftmost words typically play the role of noun or adjective modifiers. For the example above, the system would first look up the exact title phrase in the paraphrase database. If it does not exist in the database, then we create two subphrases: "subway" and "commuting problems" as text blocks. We then look for paraphrases of the two text blocks. The word subway and any paraphrase of subway (such as train, tube, etc.) forms one text block, and "commuting problems" and any paraphrase of "commuting problems" form a second text block. We require that at least one member of each text block must be present in the tweet. If "commuting problems" does not have any paraphrase in the database, then we further chunk the phrase into "commuting" and "problems" and retrieve the corresponding paraphrases. We perform the phrasal chunking for all title phrases. Together all the text blocks represent the text alternatives for "subway commuting problem".

We use the same relevance measurement method used in scenario A, and rank all tweets for a day based on the relevance score. We lower the relevance threshold to 0.2 because the filtering step already enforces a stronger relevance criteria on the tweet. The top 100 tweets are then retained to prepare a daily digest for each interest profile. In case two tweets have the same relevance score, we push the lengthier tweet based on the assumption that more words are likely to covey more description and meaning.

4.2 Daily Digest for Run 2

In run 2, we used four types of textual features: noun phrases, title phrases, named entity phrases, and location named entity phrases. We computed the optimal weights for each textual feature category that maximizes the overall EG using the TREC2015 Microblog evaluation topics, tweets and scores.

At the end of each day, for each interest profile, we search the entire tweets indexed for that day that contain phrases in at least one of the four aforementioned categories. Then the relevancy score is computed for each tweet extracted using the weights of the phrase/textual feature category that was used to filter the tweet. For each interest profile, we rank the tweets by the relevancy score and the tweet length (in case two or more tweets have the same relevancy score) and send daily digest of up to 100 tweets per topic.

4.3 Daily Digest for Run 3

In run 3, we combined all identified tweets from the previous runs in scenarios A and B. For each interest profile, we take the tweets from scenario A runs that exceed a relevance score threshold 0.3, in addition to tweets from run 1 and run 2 of scenario B. We then sort the collection by the relevance score of the tweets and keep the top 100 ranked tweets for that interest profile as the daily digest for the day.

5 Evaluation

5.1 Evaluation Methods

In the RTS track, the organizers provided two types of evaluation: 1) live user-in-the-loop assessments, and 2) post hoc batch evaluation.

In the live user-in-the-loop assessments, tweets submitted by the participating systems to the RTS evaluation broker were immediately routed to the mobile phone of an assessor. Each tweet was rendered as a push notification containing the text of the tweet and the corresponding interest profile. The assessors then could judge the tweets as relevant, relevant but redundant (on topic, but contains information conveyed previously), not relevant, or could ignore and leave unjudged. This evaluation follows the framework described in Roegiest et al. (2016).

In contrast to live user-in-the-loop assessments, the post hoc batch evaluation method evaluates tweets from scenario A and scenario B submissions at the end of the competition. The tweets are judged as not-relevant, relevant, or highly relevant. Relevant tweets are then semantically clustered into groups and on retrieving one tweet from a cluster and determining that it is relevant, all other tweets from the same cluster are considered as not relevant.

5.2 Evaluation Metrics

Tweets pushed in scenario A are evaluated with several evaluation metrics. **Expected gain (EG)** (for an interest profile on a particular day) is defined as follows:

$$EG = \frac{1}{N} \sum G(t) \qquad (3)$$

where N is the number of tweets submitted by a system and $G(t)$ is the gain of each tweet, where highly-relevant, relevant and not relevant tweets receive a gain 1.0, 0.5 and 0 respectively.

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

$$nCG = \frac{1}{Z} \sum G(t) \qquad (4)$$

where Z is the maximum possible gain (given the ten tweets per day limit).

Both metrics have two variations each. On a day when there are no relevant tweets for a particular interest profile (silent day), a system receives a score of one (i.e., perfect score) if it does not push any tweet, or zero otherwise, in the EG-1 and nCG-1 metrics. Thus, systems are rewarded for recognizing that there are no relevant tweets for an interest profile on a particular day, and remaining silent (i.e., not pushing any tweet). In the EG-0 and nCG-0 variants of the metrics, for a silent day, all systems receive a gain of zero.

The last evaluation metric is **Gain Minus Pain (GMP)**, defined as follows:

$$GMP = \alpha \times G - (1 - \alpha) \times P \qquad (5)$$

Here G (gain) is computed in the same manner as above; and P (pain) is the number of non-relevant tweets that are pushed, and controls the balance between the two. Evaluations are done at three α settings: 0.33, 0.5, and 0.66.

For scenario B runs, Normalized Discounted Cumulative Gain nDCG@10 is used as the evaluation metrics with two variants as before. nDCG@10-1 rewards a system for not pushing tweets on a silent day when there are no relevant tweets, and nDCG@10-0 does not reward for not pushing tweets on a silent day.

5.3 Results

For benchmarking the results, the organizers provided a baseline system called YoGosling, which is a simpler version of the UWaterloo system from the 2015 TREC Microblog track (Tan et al., 2016).

Table 1 presents our scenario A results with post hoc batch evaluation. Among our three runs, run 1 (*Strict Threshold-based Push Notification*) performed better than the other two runs for EG1, nCG1, and all of the GMP metrics. Run 1 also outperformed the YoGosling baseline results and was the 5th best system among all automatic runs (this includes multiple runs by the same teams as some of the top runs were from the same teams). The results for the EG0 and nCG0 metrics, that do not reward

Table 1: Scenario A batch evaluation results. EG=Expected Gain (1 = with silent day reward, 0 = no silent day reward), nCG1 = Normalized Cumulative Gain (1 = with silent day reward, 0 = no silent day reward), GMP = Gain Minus Pain (at $\alpha = 0.33$, 0.5 and 0.66).

Evaluation Metrics	EG1	EG0	nCG1	nCG0	GMP.33	GMP.5	GMP.66
YoGosling Baseline System							
YoGosling Run	0.2289	**0.0253**	0.2330	**0.0295**	-0.6000	-0.4317	-0.2733
PRNA System							
PRNA Run 1	**0.2423**	0.0119	**0.2402**	0.0098	**-0.0770**	**-0.0522**	**-0.0289**
PRNA Run 2	0.2342	**0.0253**	0.2302	0.0213	-0.4666	-0.3317	-0.2047
PRNA Run 3	0.2329	0.0240	0.2290	0.0201	-0.3365	-0.2348	-0.1391

for not pushing tweets on a silent day, were very low for all of our runs as well as for the YoGosling baseline. This indicates that for many of the interest profiles, there were very few or no relevant tweets generated for the duration of the competition. While our run 2 and 3 did not perform as good as our run 1, they still outperformed the YoGosling baseline on the EG1 and all of the GMP metrics, and were comparable to the nCG1 results of the YoGosling baseline.

Table 2: Time lantency in Scenario A

latency (seconds)	mean	median
YoGosling Baseline System		
YoGosling Run	120,908.6	8,718.0
PRNA System		
PRNA Run 1	**81,480.3**	317.0
PRNA Run 2	120,734.6	**210.0**
PRNA Run 3	172,795.8	3,321.5

Table 2 shows the mean and median time latency for our runs and the baseline. Our run 1 took significantly less time to push a tweet. The main time bottleneck in our system was the computations for novelty detection, which can be improved significantly.

Table 3 shows the evaluation results for live user-in-the-loop assessments. We report the ratio of relevant and redundant tweets pushed by our system over the judged tweets. It should be noted that this result includes tweets which had multiple judgements, as no adjudications were performed. The table also shows the percentage of tweets that had multiple judgements.

Overall estimates from the live user-in-the-loop judged samples suggest that our run 3 (*Quota-restricted Threshold-based Push Notification*) performed best among our submissions, and outperformed the Yogosling baseline results by pushing 5.65% more relevant tweets, although percentage of redundant tweets pushed in this run was relatively more. The percentages of relevant tweets for all of our runs were better than the YoGosling baseline based on the assessed samples.

Among our three runs, 293 tweets were sent for live user assessment in run 1, whereas 1,640 and

Table 3: Evaluation by human assessors in scenario A.

	#Relevant / #Judged (%)	#Redundant / #Judged (%)	#Multi / #Judged (%)
YoGosling Baseline System			
YoGosling	33.18	2.69	(4.26)
PRNA System			
PRNA R1	37.14	0.00	(7.14)
PRNA R2	37.14	2.22	(3.81)
PRNA R3	**38.83**	4.85	(3.88)

Table 4: Scenario B evaluation batch results. nDCG = Normalized Discounted Cumulative Gain (1 = with silent day reward, 0 = no silent day reward).

Evaluation Metrics	nDCG1	nDCG0
YoGosling Baseline System		
YoGosling Run	**0.2352**	0.0299
PRNA System		
PRNA Run 1	0.2334	0.0352
PRNA Run 2	0.2244	0.0226
PRNA Run 3	0.1987	**0.0665**

Table 5: Evaluation of scenario A automatic runs by the mobile assessors (Lin et al., 2016). Columns show the rank, team name, the best run by each team, "strict" and "lenient" precision. Table shows top 10 teams ranked by P(strict) from their best automatic runs.

Rank	Team	Best Run	P (strict)	P (lenient)
1	CLIP	CLIP-A-1-08	0.5028	0.5083
2	umd_hcil	UmdHcilBaseline-49	0.4762	0.4762
3	**prna**	**PRNATaskA3-36**	**0.3883**	**0.4369**
4	IRIT	iritRunBiAm-21	0.3792	0.3906
5	PKUICST	run2-32	0.3769	0.4015
6	QU	QUExpP-38	0.3689	0.3770
7	WaterlooClarke	WaterlooBaseline-50	0.3318	0.3587
8	ISIKol	MyBaseline-24	0.3189	0.3501
9	WaterlooLin	WaterlooBaseline-51	0.3180	0.3355
10	NUDTSNA	nudt_sna-29	0.3112	0.3515

Table 6: Evaluation of scenario A automatic runs by the NIST assessors (Lin et al., 2016). Table shows top 10 teams ranked by EG-1 from their best automatic runs.

Rank	Team	Best Run	EG-1	nCG-1	GMP(.50)
1	QU	QUBaseline-37	0.2643	0.2479	-0.0888
2	IRIT	iritRunBiAm-21	0.2493	0.2541	-0.3817
3	**prna**	**PRNABaseline-34**	**0.2423**	**0.2402**	**-0.0522**
4	CLIP	CLIP-A-2-09	0.2407	0.2382	-0.1656
5	NUDTSNA	nudt_sna-30	0.2392	0.2417	-0.3067
6	PKUICST	run2-32	0.2347	0.2433	-0.5183
7	DPLAB_IITBHU	iitbhu-15	0.2339	0.2339	0.0000
8	QUT_RTS	QUT_RTS-40	0.2315	0.2306	-0.0509
9	WaterlooLin	WaterlooBaseline-51	0.2298	0.2315	-0.4165
10	WaterlooClarke	WaterlooBaseline-50	0.2289	0.2330	-0.4317

Table 7: Evaluation of scenario B automatic runs by the NIST assessors (Lin et al., 2016). Table shows top 10 teams ranked by nDCG-1 from their best automatic runs.

Rank	Team	Best Run	nDCG-1	nDCG-0
1	NUDTSNA	nudt_sna	0.2708	0.0529
2	QU	QUJM16	0.2621	0.0300
3	IRIT	RunBIch	0.2481	0.0321
4	WaterlooLin	YoGoslingBSL	0.2352	0.0299
5	PKUICST	PKUICSTRunB3	0.2348	0.0151
6	**prna**	**PRNATaskB1**	**0.2334**	**0.0352**
7	ISIKol	isikol_tag	0.2213	0.0196
8	udel	udelRunBM25B	0.2151	0.0008
9	IRLAB_DA_IICT	IRLAB2	0.1972	0.0169
10	CCNU2016NLP	CCNUNLPrun1	0.1732	0.0018

1,134 tweets were sent for judgement in run 2 and run 3.[5] Among our three runs, in run 1 we pushed significantly fewer number of tweets compared to run 2 and 3. In the batch evaluation (Table 1), the metrics were averaged across the days of the competition and also across the topics (comparable to macro average). Hence it is likely that because of pushing fewer tweets, the system received more silent day rewards in run 1, and yielded better results for EG1, nCG1 and GMP; this is also supported by the lower scores that run 1 received with the EG0 and nCG0 evaluation metrics. On the other hand, with the provided statistics from the live user-in-the-loop assessment (Table 3), we calculated overall percentage of relevant tweets (among the judged tweets) across all topics and all days of the competition (comparable to micro average), and found that we pushed a slightly higher percentage of relevant tweets in run 3 when compared to our other two runs.

Finally, Table 4 shows results for our scenario B runs, using nDCG1 and nDCG0 evaluation metrics. For nDCG1, our run 1 results are comparable to the YoGosling baseline results, but did not outperform. For nDGCG0, our run 3 outperformed the YoGosling baseline results, but these metrics yield very low scores, as many of the days did not have any relevant tweet for many of the interest profiles.

Tables 5-7 highlights the results for Scenario A (Evaluation by mobile and NIST assessors) and Scenario B (Evaluation by NIST assessors) in the automatic run category (Lin et al., 2016). As shown in these tables, we ranked 3rd for Scenario A and 6th for Scenario B.

6 Conclusion

In this paper, we described our system runs for the RTS track at the TREC 2016. We submitted a total of six runs: three runs in each scenario of the challenge, under the automatic submission category. The evaluation results show that our system ranked 3rd for Scenario A and 6th for Scenario B, which demonstrates that our system using assorted textual features and dynamic push strategies is highly effective in finding relevant and novel tweets for an interest profile in real-time. For future work, we will explore identifying different sub-types of interests in the profiles (e.g., opinion, experience, news) and personalized relevance modeling of the interest profiles.

[5]The number of tweets sent for assessment is lower than the actual number of tweets pushed in each run because only a subset of the interest profiles were evaluated instead of all of 203 interest profiles

References

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, Georgia, June. Association for Computational Linguistics.

S. A. Hasan, Y. Ling, J. Liu, and O. Farri. 2015. Exploiting Neural Embeddings for Social Media Data Analysis. In *Proceedings of the Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015.*

Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering*, 18(8):1138–1150.

Jimmy Lin, Miles Efron, Yulu Wang, and Ellen Voorhees. 2015. Overview of the trec-2015 microblog track. Technical report, DTIC Document.

Jimmy Lin, Adam Roegiest, Luchen Tan, Richard McCreadie, Ellen Voorhees, and Fernando Diaz. 2016. Overview of the TREC 2016 real-time summarization track. Technical report, DTIC Document.

Adam Roegiest, Luchen Tan, Jimmy Lin, and Charles LA Clarke. 2016. A platform for streaming push notifications to mobile assessors. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1077–1080. ACM.

Luchen Tan, Adam Roegiest, and Charles LA Clarke. 2015. University of waterloo at TREC 2015 microblog track. In *Proceedings of The Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015.*

Luchen Tan, Adam Roegiest, Charles LA Clarke, and Jimmy Lin. 2016. Simple dynamic emission strategies for microblog filtering. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1009–1012. ACM.