Adaptive Representations for Tracking Breaking News on Twitter

Igor Brigadir  
Insight  
Centre for Data Analytics  
University College Dublin  
igor.brigadir@ucdconnect.ie

Derek Greene  
Insight  
Centre for Data Analytics  
University College Dublin  
derek.greene@ucd.ie

Pádraig Cunningham  
Insight  
Centre for Data Analytics  
University College Dublin  
padraig.cunningham@ucd.ie

ABSTRACT
Twitter is often the most up-to-date source for finding and tracking breaking news stories. Therefore, there is considerable interest in developing filters for tweet streams in order to track and summarize stories. This is a non-trivial text analytics task as tweets are short, and standard text similarity metrics often fail as stories evolve over time. In this paper we examine the effectiveness of adaptive text similarity mechanisms for tracking and summarizing breaking news stories. We evaluate the effectiveness of these mechanisms on a number of recent news events for which manually curated timelines are available. Assessments based on the ROUGE metric indicate that an adaptive similarity mechanism is best suited for tracking evolving stories on Twitter.

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

General Terms
Twitter, Topic Tracking, Summarization

Keywords
Neural Network Language Models, Distributional Semantics, Microblog Retrieval, Representation Learning

1. INTRODUCTION
Manually constructing timelines of events is a time-consuming task that requires considerable human effort. Twitter has been shown to be a reliable platform for breaking news coverage, and is widely used by established news wire services. While it can provide an invaluable source of user generated content and eyewitness accounts, the terse and unstructured language style of tweets often means that traditional information retrieval techniques perform poorly on this type of content.

Recently, Twitter has introduced the ability to construct custom timelines or collections from arbitrary tweets. The intended use case for this feature is the ability to curate relevant and noteworthy tweets about an event or topic.

We propose an adaptive approach for constructing custom timelines - i.e. collections of tweets tracking a particular news event, arranged in chronological order. Our approach incorporates the skip-gram neural network language model introduced by Mikolov et al. [7] for the purpose of creating useful representations of terms used in tweets. This model has been shown to capture the syntactic and semantic relationships between words. Usually, these models are trained on large static data sets. In contrast, our approach trains models on relatively smaller sets, updated at frequent intervals. Regularly retraining using recent tweets allows our proposed approach to adapt to temporal drifts in content.

This retraining strategy allows us to track a news event as it evolves, since the vocabulary used to describe it will naturally change as it develops over time. Given a seed query, our approach can automatically generate chronological timelines of events from a stream of tweets, while continuously learning new representations of relevant words and entities as the story changes. Evaluations performed in relation to a set of real-world news events indicate that this approach allows us to track events more accurately, when compared to nonadaptive models and traditional “bag-of-words” representations.

2. PROBLEM FORMULATION
Custom timelines, curated tweet collections on Storify, and liveblog platforms such as Scribblelive3 are conceptually similar and are popular with many major news outlets.

For the most part, liveblogs and timelines of events are manually constructed by journalists. Rather than automating construction of timelines entirely, our proposed approach offers editorial support for this task, allowing smaller news teams with limited budgets to use resources more effectively. Our contribution focuses on retrieval and tracking rather than new event detection or verification.

We define a timeline of an event as a timestamped set of tweets relevant to a query, presented in chronological order. The problem of adaptively generating timelines for breaking news events is cast as a topic tracking problem, comprising of two tasks:

1. Identify relevant tweets from a stream of tweets.
2. Organize these tweets into a chronological timeline.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

NewsKDD ’14 New York, United States
Copyright 20XX ACM X-XXXX-XX-X/XX/XX ...$15.00.

1blog.twitter.com/2013/introducing-custom-timelines
2www.storify.com
3www.scribblelive.com
Realtime ad-hoc retrieval:
For each target query (some keywords of interest), retrieve all relevant tweets from a stream posted after the query. Retrieval should maximize recall for all topics (retrieving as many possibly relevant tweets as available).

Timeline Summarization:
Given all retrieved tweets relating to a topic, construct a timeline of an event that includes all detected aspects of a story. Summarization involves removal of redundant or duplicate information while maintaining good coverage.

3. RELATED WORK
The problem of generating news event timelines is related to topic detection and tracking, and multi-document summarization, where probabilistic topic modelling approaches are popular. Our contribution attempts to utilise a state-of-the-art neural network language model (NNLM) in order to capitalise on the vast amount of microblog data, where semantic concepts between words and phrases can be captured by learning new representations in an unsupervised manner.

Timeline Generation.
An approach by Wang [11] that deals with longer news articles, employed a Time-Dependent Hierarchical Dirichlet Model (HDM) for generating timelines using topics mined from HDM for sentence selection, optimising coverage, relevance, and coherence. Yan et al. [13] proposed a similar approach, framing the problem of timeline generation as an optimisation problem solved with an iterative substitution approach, optimising for diversity as well as coherence, coverage, and relevance. Generating timelines using tweets was explored by Li & Cardie [4]. However, the authors solely focused on generating timelines of events that are of a personal interest. Stumblr [10] uses an online tweet stream clustering algorithm, which can produce summaries over arbitrary time durations, by maintaining snapshots of tweet clusters at differing levels of granularity.

Tracking News Stories.
To examine the propagation of variations of phrases in news articles, Leskovec et al. [3] developed a framework to identify and adaptively track the evolution of unique phrases using a graph based approach. In [1], a search and summarization framework was proposed to construct summaries of events of interest. A Decay Topic Model (DTM) that exploits temporal correlations between tweets was used to generate summaries covering different aspects of events. Osborne & Lavrenko [9] showed that incorporating paraphrases can lead to a marked improvement on retrieval accuracy in the task of First Story Detection.

Semantic Representations.
There are several popular ways of representing individual words or documents in a semantic space. Most do not address the temporal nature of documents but a notable method that does is described by Jurgens and Stevens [2], adding a temporal dimension to Random Indexing for the purpose of event detection. Our approach focuses on summarization rather than event detection, however the concept of using word co-occurrence to learn word representations is similar.

4. SOURCE DATA
The corpus of tweets used in our experiments consists of a stream originating from a set of manually curated “newsworthy” accounts created by journalists as Twitter lists. Such lists are commonly used by journalists for monitoring activity and extracting eyewitness accounts around specific news stories or regions. Our stream collects tweets from a total of 16,971 unique users, segmented into 347 geographical and topical lists. This sample of users offers a reasonable coverage of potentially newsworthy tweets, while reducing the need to filter spam and personal updates from accounts that are not focused on disseminating breaking news events. While these lists of users have natural groupings (by country, or topic), we do not segment the stream or attempt to classify events by type or topic.

As ground truth for our experiments, we use a set of publicly available custom timelines from Twitter, relevant content from Scribblelive liveblogs, and collections of tweets from Storyful. Each event has multiple reference sources. (See Appendix C).

It is not known what kind of approach was used to construct these timelines, but as our stream includes many major news outlets, we expect some overlap with our sources, although other accounts may be missing. Our task involves identifying similar content to event timelines posted during the same time periods.

5. METHODS
Short documents like tweets present a challenge for traditional retrieval models that rely on “bag-of-words” representations. We propose to use an alternative representation of short documents that takes advantage of structure and context, as well as content of tweets.

Recent work by [6] introduced an efficient way of training a Neural Network Language Model (NNLM) on large volumes of text using stochastic gradient descent. This language model represents words as dense vectors of real values. Unique properties of these representations of words make this approach a good fit for our problem.

The high number of duplicate and near-duplicate tweets in the stream benefits training by providing additional training examples. For example: the vector for the term “LAX” is most similar to vectors representing “#LAX”, “airport”, and “tsa agent” — either syntactically or semantically related terms. Moreover, retraining the model on new tweets create entirely new representations that reflect the most recent view of the world. In our case, it is extremely useful to have representations of terms where “#IranTalks” and “nuclear talks” are highly similar at a time when there are many reports of nuclear proliferation agreements with Iran.

Additive compositionality is another useful property of the these vectors. It is possible to combine several words via an element-wise sum of several vectors. There are limits to this, in that summation of multiple words will produce an increasingly noisy result. Combined with standard stopword removal, and URL filtering, and removal of rare terms, each tweet can be reduced to a few representative words. The NNLM vocabulary also treats mentions and hashtags as words, requiring no further processing or query expansion. Combining these words allows us to compare similarities between whole tweets.

Tweet data provided by Storyful (www.storyful.com)
5.1 Timeline Generation

We compare three alternative models to generate timelines from a tweet stream. In each case, we initialize the process with a query. For a given event, the tweet stream is then retracted from the event’s beginning to end, with the exact dates defined by tweets in the corresponding human generated timelines. Inclusion of a tweet in the timeline is controlled by a fixed similarity threshold. The stream is processed using a fixed length sliding window updated at regular intervals in order to accommodate model training time.

Pre-processing.

A modified stopword list was used to remove Twitter specific terms (e.g. “MT”, “via”), together with common English stopwords. In the case of NNLM models, stopwords were replaced with a placeholder token, in order to preserve word context. This approach showed an improvement when compared with no stopword removal, and complete removal of stopwords. While the model can be trained on any language effectively, to simplify evaluation only English tweets were considered. Language filtering was performed using Twitter metadata.

Bag-of-Words (tf) Model.

A standard term frequency-inverse document frequency model is included as a baseline in our experiments, which uses the cosine similarity of a bag-of-words representation of tweets. We use the same pre-processing steps as applied to the other models. Inverse document frequency counts for terms are derived from the same window of tweets used to train the NNLM approaches. The addition of inverse document frequencies did not offer a significant improvement, as most tweets are short and use terms only once. The term frequencies did not offer a significant improvement, as most tweets are short and use terms only once. The term frequency model is moderately adaptive in the sense that the seed query can change as the stream evolves. The seed query is updated if it is similar to the current query, while introducing a number of new terms.

Nonadaptive NNLM.

The nonadaptive version of the NNLM model is a static variant where word vectors are initially trained on a large number of tweets, and no further updates to the model are made as time passes.

Adaptive NNLM.

The adaptive version uses a sliding window approach to continuously build new models at a fixed interval. The trade-off between recency and accuracy is controlled by altering two parameters: window length (i.e. limiting the number of tweets to learn from) and refresh rate (i.e. controlling how frequently a model is retrained). No updates are made to the seed query in both NNLM approaches, only the representation of the words changes after retraining the model.

Post-processing

For all retrieval models, to optimise for diversity and reduce timeline length the same summarization step was applied to remove duplicate and near duplicate tweets. Tweets are considered duplicate or near duplicate if all terms excluding stopwords, mentions and hashtags are identical to a tweet previously included in the timeline.

6. EVALUATION

In order to evaluate the quality of generated timelines, we use the popular ROUGE set of metrics [5], which measure the overlap of ngrams, word pairs and sequences between the ground truth timelines, and the automatically generated timelines. ROUGE parameters are selected based on [8]. ROUGE-1 and ROUGE-2 are widely reported and were found to have good agreement with manual evaluations. In all settings, stemming is performed, and no stopwords are removed. Text is not pre-processed to remove tweet entities such as hashtags or mentions but URLs, photos and other media items are removed.

To take into account the temporal nature of an event timeline, we average scores across a number of event periods for each variant of the model. This ensures that scores are penalised if the generated timeline fails to find relevant tweets for different time periods as a story evolves. The number of evaluation periods is dependent on the event duration, and selected refresh rate parameter. (See Appendix B).

“Max” Baseline

The “Max” baseline is an illustrative retrieval model, having perfect information about the ground truth and source data. It is designed to represent the maximum achievable score on a metric, given our limited data set and ground truth. For every evaluation period, for each ground truth update, this baseline will select the highest scoring tweet from our stream. This method gives an upper bound on performance for each test event, as it will find the set of tweets that maximise the target ROUGE score directly.

Performance on unseen Events

For initial parameter selection, a number of representative events were selected. (See Appendix C). We evaluate the system on several new events, briefly described here.

Table 1 gives an overview of the durations, total length, number of reference sources and average number of updates per evaluation period for each event. “Train” timeline describes a Metronorth train derailment, “Floodes” deals with flooding in the Solomon Islands, and is characterised by having a low number of potential sources, and sparse updates. “Westgate” follows the Westgate Mall Siege, MH370 details the initial reports of the missing flight, “Crimea” follows an evenful day during the annexation of the Crimean peninsula, “Bitcoin” follows reporters chasing after the alleged creator of Bitcoin, “Mandela” and “P. Walker” are reactions to celebrity deaths, “WHCD” follows updates from the White House Correspondents Dinner, and “WWDC” follows the latest product launches from Apple - characterised by a very high number of updates and rapidly changing context.

In most cases, shown in Figure 1, our adaptive approach performs well on a variety of events, capturing relevant tweets as the event context changes. This is most notable in the “WWDC14” story, where there were several significant changes in the timeline as new products were announced for the first time. While the adaptive approach can follow concept drift in a news story, it cannot understand or disambiguate between verified and unverified developments, even though relevant tweets are retrieved as the news story evolves, incorrect or previously debunked facts are still seen as relevant, and included in the generated timeline.

Overall the adaptive NNLM approach performs much more effectively in terms of recall rather than precision. A more
### Table 1: Details for events used for evaluation. Update Frequency is average number of updates every 15 minutes.

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Reference Sources</th>
<th>Duration (Hrs:min)</th>
<th>Total Updates</th>
<th>Tweets</th>
<th>Update Freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>WWDC</td>
<td></td>
<td>10:30</td>
<td>432</td>
<td>465</td>
<td>8</td>
</tr>
<tr>
<td>P.Walker</td>
<td></td>
<td>8:00</td>
<td>617</td>
<td>449</td>
<td>19.28</td>
</tr>
<tr>
<td>Mandela</td>
<td></td>
<td>7:00</td>
<td>480</td>
<td>51</td>
<td>1.68</td>
</tr>
<tr>
<td>Crimea</td>
<td></td>
<td>7:00</td>
<td>34</td>
<td>34</td>
<td>1.21</td>
</tr>
<tr>
<td>MH370</td>
<td></td>
<td>6:00</td>
<td>40</td>
<td>51</td>
<td>1.68</td>
</tr>
<tr>
<td>MH370</td>
<td></td>
<td>24:15</td>
<td>2</td>
<td>3</td>
<td>0.19</td>
</tr>
<tr>
<td>MH370</td>
<td></td>
<td>4:15</td>
<td>433</td>
<td>449</td>
<td>19.28</td>
</tr>
<tr>
<td>Crimea</td>
<td></td>
<td>4:15</td>
<td>2</td>
<td>3</td>
<td>0.19</td>
</tr>
<tr>
<td>Crimea</td>
<td></td>
<td>18:15</td>
<td>73</td>
<td>62</td>
<td>1.00</td>
</tr>
<tr>
<td>Crimea</td>
<td></td>
<td>10:30</td>
<td>433</td>
<td>449</td>
<td>19.28</td>
</tr>
<tr>
<td>Crimea</td>
<td></td>
<td>2:00</td>
<td>2</td>
<td>3</td>
<td>0.19</td>
</tr>
<tr>
<td>Identity</td>
<td></td>
<td>2:00</td>
<td>2</td>
<td>3</td>
<td>0.19</td>
</tr>
</tbody>
</table>

### Table 2: ROUGE-1 Scores for evaluation events.

<table>
<thead>
<tr>
<th>Event</th>
<th>ROUGE-1 Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>Max</td>
</tr>
<tr>
<td>WWDC</td>
<td>0.40</td>
</tr>
<tr>
<td>P.Walker</td>
<td>0.43</td>
</tr>
<tr>
<td>WWDC</td>
<td>0.09</td>
</tr>
</tbody>
</table>

### Table 3: ROUGE-2 Scores for evaluation events.

<table>
<thead>
<tr>
<th>Event</th>
<th>ROUGE-2 Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>Max</td>
</tr>
<tr>
<td>WWDC</td>
<td>0.40</td>
</tr>
<tr>
<td>P.Walker</td>
<td>0.43</td>
</tr>
<tr>
<td>WWDC</td>
<td>0.09</td>
</tr>
</tbody>
</table>

### Figure 1: ROUGE-1 F1 Scores for each event. See link\(^5\) to view generated timelines for all events.

---

5http://mlg.ucd.ie/timelines
9. REFERENCES


9.1 Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

We thank Storyful for providing access to data, and early adopters of custom timelines who unknowingly contributed ground truth used in the evaluation.

APPENDIX

A. SKIP-GRAM LANGUAGE MODEL

The skip-gram model described in methods section 5 has a number of hyper parameters. Choices for these are discussed here.

A.1 Training:

The computational complexity of the skip-gram model is dependent on the number of training epochs $E$, total number of words in the training set $T$, maximum number of nearby words $C$, dimensionality of vectors $D$ and the vocabulary size $V$, and is proportional to:

$$O = E \times T \times C \times (D + D \times \log_2(V))$$

The training objective of the skip-gram model, revisited in [7], is to learn word representations that are optimised for predicting nearby words. Formally, given a sequence of words $w_1, w_2, \ldots, w_T$ the objective is to maximize the average log probability:

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t)$$

In effect, word context plays an important part in training the model.

Pre Processing:

For a term to be included in the training set, it must occur at least twice in the set. These words are removed before training the model.

Filtering stopwords entirely had a negative impact on overall accuracy. Alternatively, we filter stopwords while maintaining relative word positions.

Extracting potential phrases before training the model, as described in [6] did not improve overall accuracy. In this pre-processing step, frequently occurring bigrams are concatenated into single terms, so that phrases like “trade agreement” become a single term when training a model.

Training Objective:

An alternative to the skip-gram model, the continuous bag of words (CBOW) approach was considered. The skip-gram model learns to predict words within a certain range (the context window) before and after a given word. In contrast, CBOW predicts a given word given a range of words before and after. While CBOW can train faster, skip-gram performs better on semantic tasks. Given that our training sets are relatively small, CBOW did not offer any advantage in terms of improving training time. Negative sampling from [6] was not used. The context window size was set to 5. During training however, this window size is dynamic. For each word, a context window size is sampled uniformly from 1,...,k. As tweets are relatively short, larger context sizes did not improve retrieval accuracy.

A.2 Vector Representations:

The model produces continuous distributed representations of words, in the form of dense, real valued vectors.
These vectors can be efficiently added, subtracted, or compared with a cosine similarity metric.

The vector representations do not represent any intuitive quantity like word co-occurrence counts or topics. Their magnitude though, is related to word frequency. The vectors can be thought of as representing the distribution of the contexts in which a word appears.

Vector size is also a tunable parameter. While larger vector sizes can help build more accurate models in some cases, in our retrieval task, vectors larger than 200 did not show a significant improvement in scores.

**B. PARAMETER SELECTION**

Our system has a number of tuneable parameters that suit different types of events. When generating timelines of events retrospectively, these parameters can be adapted to improve accuracy. For generating timelines in real-time, parameters are not adapted to individual event types. For initial parameter selection, a number of representative events were chosen, detailed in Table 5.

For all models, the seed query (either manually entered, or derived from a tweet) plays the most significant part. Overall, for the NNLM models, short event specific queries with few key words perform better than long or expanded queries which benefit term frequency (TF) models. In our evaluation, the same queries were used while modifying other parameters. Queries were adapted from the first tweet included in an event timeline to simulate a lack of information at the beginning of an event.

The refresh rate parameter controls how old the training set of tweets can be for a given model. In the case of TF models, this affects the IDF calculations, and for NNLM models, the window contains the preprocessed text used for training. As such, when the system is replaying the stream of tweets for a given event, the model used for similarity calculations is refresh rate minutes old.

Updating the sliding window every 15 minutes and retraining on tweets posted in the previous 24 hours was found to provide a good balance between adaptivity and quality of resulting representations. Larger window sizes encompassing more tweets were less sensitive to rapidly developing stories, while smaller window sizes produced noisier timelines for most events.

**C. TUNING EVENT EVALUATION**

**Ground Truth Data**

Since evaluation is based on content, reference sources may contain information not in our dataset and vice versa. Where there were no quoted tweets in ground truth, the text was extracted as a sentence update instead. Photo captions and other descriptions were also included in ground truth. Advertisements and other promotional updates were removed.

**Events used for Parameter Selection**

For initial model selection and tuning, timelines for six events were sourced from Twitter and other live blog sources: the “BatKid” Make-A-Wish foundation event, Iranian Nuclear proliferation talks, a shooting at LAX, Senator Rob Ford speaking at a Council meeting, multiple tornadoes in US midwest, and an alert regarding a possible gunman at Yale University.

These events were chosen to represent an array of different event types and information needs. Timelines range in length and verbosity as well as content type. See Table 5.

“Batkid” can be characterised as a rapidly developing event, but without contradictory reports. “Yale” is also a rapidly developing event, but one where confirmed facts were slow to emerge. “Lax” is a media heavy event spanning just over 7 hours while “Tornado” spans 9 hours and is an extremely rapidly developing story, comprised mostly of photos and video of damaged property. “Iran” and “Robford” differ in update frequency but are similar in that related stories are widely discussed before the evaluation period.