Adaptive Representations for Tracking Breaking News on Twitter

Igor Brigadir and **Derek Greene** and **Pádraig Cunningham**

Insight Centre for Data Analytics
University College Dublin
igor.brigadir@ucdconnect.ie, derek.greene@ucd.ie, 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 retrieval methods often fail as stories evolve over time. In this paper we examine the effectiveness of adaptive 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 ROUGE metrics indicate that an adaptive approaches are best suited for tracking evolving stories on Twitter.

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 have difficulty with this type of content.

Recently, Twitter has introduced the ability to construct *custom timelines*[1] 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 approach for constructing *custom timelines* incorporating distributional semantic language models (DSMs) trained on tweet text. DSMs create useful representations of terms used in tweets, capturing the syntactic and semantic relationships between words.

We evaluate several retrieval approaches including a neural network language model introduced by Mikolov et al. (2013b), Random Indexing (Kanerva, Kristoferson, and Holst 2000), and a BM25 based method.

Usually, DSMs 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 adaptive approaches allow us to track events more accurately, when compared to nonadaptive models (models that rely on large, static data sets).

Problem Formulation

Custom timelines, curated tweet collections on *Storify*[2], and liveblog platforms such as *Scribblelive*[3] 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:

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 in-

---

[1]blog.twitter.com/2013/introducing-custom-timelines

[2]www.storify.com

[3]www.scribblelive.com

volves removal of redundant or duplicate information while maintaining good coverage.

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) and other distributional semantic approaches 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 (2013) 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. (2011) 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 (2013). However, the authors solely focused on generating timelines of events that are of a personal interest. *Sumblr* (Shou 2013) 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. (2009) developed a framework to identify and adaptively track the evolution of unique phrases using a graph based approach. In (Chong and Chua 2013), 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 (2012) 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 (2009), adding a temporal dimension to Random Indexing for the purpose of event detection. Our approach focuses on summarization rather then event detection, however the concept of using word co-occurance to learn word representations is similar.

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[4] as Twitter lists. Such

---

lists are commonly used 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.

Event Data

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 *Storify*. Multiple reference sources are included when available.

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.

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.

For initial model selection and tuning, timelines for six events were sourced from Twitter and other live blog sources:

- "BatKid": Make-A-Wish foundation event.
- "Iran": Follows Iranian Nuclear proliferation talks.
- "LAX": A shooting at LAX.
- "RobFord": Senator Rob Ford Council meeting.
- "Tornado": Reports of multiple tornadoes in US midwest.
- "Yale": 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 2.

"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.

In some cases the same tweets present in a human generated timeline appeared in our automatically generated timelines (see Table 1), providing an indication that our data

Event Period	Ground Truth	Retrieved Tweets (NNLM)
15:30 to 16:11	Confirmed report of a person w/ gun on/near Old Campus. SHELTER IN PLACE.	NOW: Police responding to reports of a person with a gun at Yale University. Shelter in Place issued on Central Campus (via @Yale)
17:34 to 18:15	New Haven police spokesman says there is no description of a suspect @Yale and "This investigation is in its infancy" #NHV #Yale	New Haven police spokesman says there is no description of a suspect @Yale and "This investigation is in its infancy" #NHV #Yale
18:57 to 19:38	hartman: possibility that witnesses of long guns saw instead law enforcement officers responding to the scene #Yale	RT @NBCConnecticut: Police say witnesses who saw person with long gun at @Yale could have seen law enforcement personnel. #Yalelockdown

Table 1: A manual selection of retrieved tweets for "Yale" event highlighting key developments, and how the adaptive NNLM model can handle concept drift with high recall.

id	Event Name:	Reference Sources:	Duration: (Hrs:min)	Total Updates	Tweets	Update Freq.
	BatKid	2	5:30	294	123	13.36
	Iran	3	4:15	197	190	11.59
	LAX	5	7:15	1186	944	40.90
	RobFord	4	6:45	1219	904	45.15
	Tornado	5	9:0	2224	1617	61.78
	Yale	1	7:15	124	124	4.28
1	MH17	5	7:30	554	487	18.47
2	Train	2	10:0	472	469	11.80
3	Westgate	3	18:15	73	62	1.00
4	MH370	1	7:0	42	7	1.50
5	Crimea	1	7:0	34	34	1.21
6	Bitcoin	2	4:15	157	149	9.24
7	Mandela	2	4:45	89	51	4.68
8	WHCD	2	8:0	617	440	19.28
9	P.Walker	2	5:45	152	106	6.61
10	WWDC	2	3:30	1069	81	76.36

Table 2: Details for events used for parameter fitting and evaluation. Update Frequency is average number of updates every 15 minutes.

source provides good coverage of newsworthy sources for a variety of events.

For evaluation, several new events are considered:

- "MH17": Follows shooting down of Malaysian Air Flight.

- "Train": Timeline describes a train derailment.

- "Westgate": Follows the Westgate Mall Siege.

- "MH370": Details the initial reports of the missing flight.

- "Crimea" follows an eventful day during the annexation of the Crimean peninsula.

- "Bitcoin": Reporters chase the alleged creator of Bitcoin.

- "Mandela": Reactions to illness & death.

- "P. Walker": Reactions to car accident & death.

- "WHCD": White House Correspondents Dinner.

- "WWDC": Follows the latest product launches from Apple - characterised by a very high number of updates and rapidly changing context.

Table 2 gives an overview of the reference sources, durations, content types, and update frequency for each event.

Methods

The task of realtime ad-hoc retrieval for constructing timelines is made challenging by the continuously updating collection of documents. Traditional approaches perform poorly lacking global term statistics (IDF counts for example) or become intractable as the collection of documents grows over time. The impact of "cheating" by using future term statistics in a related, but notably different realtime tweet search task is discussed in (Wang and Lin 2014). The key difference between the TREC realtime tweet search task, and the realtime retrieval task posed here is that the TREC task involves retrieving relevant tweets *before* the query time, whereas for timeline generation, the task is to retrieve tweets posted *after* the query time.

Timeline Generation

We compare three adaptive models: BM25 (with updating IDF component), Word2Vec and two Random Indexing approaches (with updating training data), and static variants.

In each case, we initialize the process with a query. For a given event, the tweet stream is then replayed 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 cosine similarity with 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. The fixed length sliding window approach used to build models of tweet representations (TF-IDF in BM25 or DSM in Word2Vec and Random Indexing models) means that new tweets arriving from the stream are analysed with trained models that are at most *refresh rate* minutes old. Parameter settings for the *window length* and refresh rates are discussed in *Parameter Selection* below.

Pre-processing. A modified stopword list was used to remove Twitter specific terms (*e.g.* "MT", "via"), together with common English stopwords. URLs and media items are removed, but mentions and hashtags are preserved. For distributional semantic models, stopwords were replaced with a placeholder token, in order to preserve relative word positions. This approach showed an improvement when compared with no stopword removal, and complete removal of stopwords. While models can be trained on any language effectively, to simplify evaluation only English tweets were considered. Language filtering was performed using Twitter metadata. On average, there are 150k-200k terms in each sliding window. 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.

Nonadaptive Approaches: The nonadaptive representation models are variants where word vectors or term frequencies are initially trained on a large number of tweets, and no further updates to the model are made as time passes.

Adaptive Approaches: The adaptive versions use 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, 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. The SumBasic(Vanderwende et al. 2007) algorithm was chosen for producing tweet summaries with high recall(Inouye and Kalita 2011). The target length for a summary is determined by the average length of the reference summaries for an event.

TF-IDF Model

BM25(Jones, Walker, and Robertson 2000) with microblog specific settings(Ferguson et al. 2011) are a family of scoring functions used to rank documents according to relevance to a query. For a query Q comprising of terms $q_1, \ldots, q_n$ the document D is scored with:

$$\text{Score}_{bm25}(D, Q) =$$

$$\sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{tf(q_i, D) \cdot (k_1 + 1)}{tf(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgdl}})}$$

Where $tf(q_i, D)$ is the term frequency of q_i in D, $|D|$ is document length, and $avgdl$ is average document length in the collection.

Parameter choices of k_1 and b are set to $k_1 = 1.2, b = 0.75$. The IDF calculation can also be substituted for alternatives, but most implementations, including *Lucene* calculate IDF of term t using $\text{IDF}(t) = \log \frac{N - n(t) + 0.5}{n(t) + 0.5}$, N being the total number of documents in the collection and $n(t)$ the number of documents containing t.

The document and term frequency counts are periodically updated as new information becomes available, using the same sliding window approach for generating training data for other models.

Skip-Gram Language Model

Recent work by (Mikolov et al. 2013a) 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.

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 (Mikolov et al. 2013b), 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 (Mikolov et al. 2013a) 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 (Mikolov et al. 2013a) 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.

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-occurance 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.

Typically, these models are trained on large, static data sets. In this case smaller sets are used, with lowered thresholds for rate terms (minimum count of 2), more training epochs and a lower learning rate. These parameters produced better performance on smaller data sets in this retrieval task, but may not be optimal for other tasks.

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. (See Figure 2)

Random Indexing

Random Indexing (RI)(Sahlgren 2005) is based on seminal work on sparse distributed memory(Kanerva 1988). While not as popular as NNLM models, it is very well suited to distributed computation, can be highly efficient and has comparable performance to more advanced techniques such as Latent Semantic Analysis/Indexing (LSA/LSI)(Deerwester et al. 1990).

The general approach to creating a word space model involves creating a matrix F where each row F_w represents a word w and each column F_c represents a context c. The context can be another co-occurring word, or a document. F is then either a word-by-word or word-by-document matrix, as in the case of LSA.

These types of word spaces suffer from efficiency and scalability problems. The number of words (vocabulary) and documents can make the matrix extremely large and difficult to use. The matrix F is also extremely sparse. LSA solves this dimensionality and sparsity problem with Singular Value Decomposition (SVD) though this creates other problems, as the SVD operation still requires the full co-occurrence matrix and is difficult to update with new information.

The Random Indexing approach to this problem is to use a random projection of the full co-occurrence matrix in a much lower dimensional space. Random indexing can then be thought of as a dimensionality reduction technique.

The standard Random indexing technique is a two step process: An *index vector* is created for each document (or word). This index vector is still high-dimensional, though in the region of several thousands, which is still much lower than an entry in a full co-occurrence matrix. The vectors are also sparse (most entries are 0) and *ternary* where randomly distributed $+1$ and -1 values ensure near-orthogonality. The near-orthogonality property is an important attribute of the index vectors(Chatterjee and Sahoo 2013) in the word space created by RI.

The second step involves an element wise sum of index vectors for each co-occurrence of a word in the text. Words are then represented as d-dimensional vectors consisting of the sum of the contexts in which a word is found. Simple co-occurrence only considers immediate surrounding words, though in practice a context window is extended to include several words.

The accumulation stage results in a d-dimensional space $F_{w \times d}$ which is an approximation of the full $F_{w \times c}$ matrix. This insight is based on the Johnson-Lindenstrauss lemma(Frankl and Maehara 1988) which states that distances between points are approximately preserved when projecting points into a randomly selected sub space of high dimensionality. The matrix F can then be approximated by projecting (multiplying) it with a random matrix R:

$$F_{w \times d} R_{d \times k} = F'_{w \times k}$$

The Random Indexing approach is incremental, easily applicable to parallel computation, and efficient, involving simple integer operations.

Variants of Random Indexing approaches involve different strategies for adding index or *elemental* vectors. Index vectors can also be initialized for terms rather than documents, in both cases this "training" step produces vectors that encode meaningful relationships between words that do not co-occur. A variant of the standard RI approach is *Reflective Random Indexing* (RRI)(Cohen, Schvaneveldt, and Widdows 2010) where vectors are built in a slightly different way.

Two variants of RI approaches are implemented as alternatives to the NNLM:

Term-term RI (TTRI)

1. Assign index vector for each term.

2. For each term, sum the index vectors for each co-occurring term in a context window.

Term based Reflective RI (TRRI)

1. Assign index vector for each term.

2. Generate document vectors by summing index vectors of all terms contained in the document.

3. For each term, sum document vectors for each document the term occurs in.

The trained model represents a word space similar to the model created by the skip-gram (word2vec) model. The same additive composition approach is used to create a vector representing a whole tweet, with an element wise sum of the individual word vectors.

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 all models, the *seed query* (either manually entered, or derived from a tweet) plays the most significant part. Overall, for the NNLM and RI models, short event specific queries with few terms perform better than longer, expanded queries which benefit term frequency (BM25) model. 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 BM25 model, this affects the IDF calculations, and for NNLM and RI 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.

Window length effectively controls how many terms are considered in each model for training or IDF calculations. While simpler to implement, this fixed window approach does not account for the number of tweets in a window, only the time range is considered. The volume of tweets is not constant over time - leading to training sets of varying sizes. However, since the refresh rate is much shorter than the window length, the natural increase and decrease in tweet volume is smoothed out. On average, there are 150k-200k unique terms in each 24 hour window. Figure 1 shows how varying window size can improve or degrade retrieval performance of different events.

Updating the sliding window every 15 minutes and re-training 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.

Figures 1 and 2 are showing the word2vec model performance. Random Indexing approaches showed a similar pattern when changing window size and vector length, though in the random indexing case, the vector size is set to 2500, larger vectors showed no increase in retrieval performance.

Evaluation

In order to evaluate the quality of generated timelines, we use the popular ROUGE set of metrics (Lin 2004), 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 (Owczarzak et al. 2012). ROUGE-1 and ROUGE-2 are widely reported and were found to have good agreement with manual evaluations in related summarization tasks. In all settings, stemming is performed, and no stopwords are removed. Text is not pre-processed to remove tweet entities
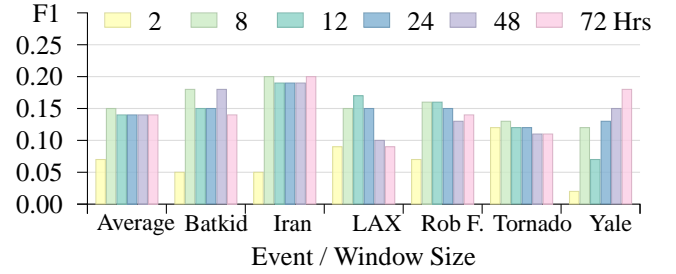

Figure 1: F1 scores for Adaptive model accuracy in response to changing window size
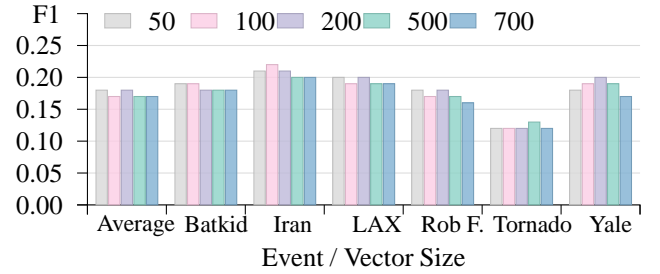

Figure 2: F1 scores for Adaptive model accuracy in response to changing vector size

such as hashtags or mentions but URLs, photos and other media items are removed. Several ROUGE variants for automatic evaluation are considered, as there is currently no manual evaluation of the generated summaries.

ROUGE-N is defined as the n-gram recall between a ground truth (reference) and system generated summary:

$$\text{ROUGE-N} =$$

$$\frac{\sum\limits_{S \in \{\text{Reference Summaries}\}} \sum\limits_{gram_n \in S} Count_{match}(gram_n)}{\sum\limits_{S \in \{\text{Reference Summaries}\}} \sum\limits_{gram_n \in S} Count(gram_n)}$$

where N is the n-gram value, and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occuring in the reference summaries and system generated summary.

ROUGE-L variant is based on the longest common subsequence (LCS) match between the reference and generated summaries. ROUGE-1 and ROUGE-L scores were highly correlated (Pearson's r > 0.99), producing the same ranking of system performance.

ROUGE-SU or Skip-bigram variant, measures the overlap of skip-bigrams between summaries. In contrast to LCS, this variant counts all matching word pairs. In the sentence *"Satoshi got free sushi"* has 6 skip-bigrams: [*"Satoshi got", "Satoshi free", "Satoshi sushi", "got free", "got sushi", "free sushi"*]. Typical settings for the maximum skip distance between two words is set to 4.

A more robust variant of ROUGE, BEwT-E: Basic Elements with Transformations (Tratz and Hovy 2008) is also reported. Basic Elements are variable sized, syntactically coherent units extracted from text. Transformations are applied to the generated and reference summaries with named entity recognition, abbreviation expansion and others. While

id	ROUGE-BEwT Scores									
	Recall					Precision				
	BM 25	w2v dyn.	RI dyn.	w2v stat.	RI stat.	BM 25	w2v dyn.	RI dyn.	w2v stat.	RI stat.
1	0.32	**0.33**	0.33	0.18	0.25	0.32	**0.33**	0.33	0.18	0.25
2	**0.19**	0.19	0.19	0.15	0.14	**0.19**	0.19	0.19	0.15	0.14
3	0.17	0.18	0.19	0.19	**0.20**	0.17	0.18	0.19	0.19	**0.20**
4	0.20	0.23	0.21	0.21	**0.24**	0.20	0.23	0.21	0.21	**0.24**
5	0.14	0.17	0.16	**0.18**	0.16	0.14	0.17	0.16	**0.18**	0.16
6	**0.19**	0.17	0.16	0.15	0.14	**0.19**	0.17	0.16	0.15	0.14
7	0.22	0.18	0.23	0.18	0.13	0.22	0.18	0.23	0.18	0.13
8	**0.08**	0.07	0.07	0.05	0.07	**0.08**	0.07	0.07	0.05	0.07
9	0.20	0.19	0.21	0.14	0.16	0.20	0.19	0.21	0.14	0.16
10	**0.16**	0.15	0.14	0.13	0.09	**0.16**	0.15	0.14	0.13	0.09

Table 3: Detailed Precision & Recall scores for ROUGE-BEwT for unseen events. Best score in bold.

id	ROUGE-SU4 Scores									
	Recall					Precision				
	BM 25	w2v dyn.	RI dyn.	w2v stat.	RI stat.	BM 25	w2v dyn.	RI dyn.	w2v stat.	RI stat.
1	0.15	0.14	**0.16**	0.10	0.13	0.23	**0.25**	0.22	0.12	0.18
2	0.17	0.17	**0.19**	0.17	0.16	0.40	**0.41**	0.35	0.25	0.33
3	0.12	**0.12**	0.11	0.11	0.10	0.09	**0.10**	0.09	0.09	0.09
4	0.14	0.17	0.15	0.16	0.17	0.20	0.25	0.23	0.25	0.27
5	0.12	**0.15**	0.13	0.15	0.13	0.12	**0.15**	0.13	0.15	0.13
6	**0.22**	0.21	0.19	0.20	0.16	0.19	0.20	0.22	0.20	0.23
7	**0.17**	0.15	0.16	0.15	0.13	0.20	0.19	0.19	**0.21**	0.19
8	**0.08**	0.07	0.08	0.05	0.07	0.20	0.31	0.22	**0.32**	0.21
9	**0.21**	0.20	0.21	0.14	0.19	0.16	0.15	0.15	0.14	0.17
10	**0.12**	0.10	0.11	0.10	0.07	0.30	**0.35**	0.30	0.33	0.30

Table 4: Detailed Precision & Recall scores for ROUGE-SU4 for unseen events. Best score in bold.

this evaluation approach more closely correlates with human judgements in other tasks, the lack of Twitter specific transformations could negatively impact performance - mapping @barackobama to "Barack Obama" for example. All default BEwT-E settings, part of speech models and named entity recognition models are used.

Performance on unseen Events In most cases, shown in Figure 3, adaptive approaches perform well on a variety of events, capturing relevant tweets as the event context changes. This is most notable in the "WWDC14" story (Event 10 in Table 4), where there were several significant changes in the timeline as new products were announced for the first time.

While adaptive approaches can follow concept drift in a news story, a notable drawback of DSMs was the lack of disambiguation between multiple meanings of some terms. Even though relevant tweets are retrieved as the news story evolves, irrelevant but semantically related tweets were also present in some timelines - mentions of other car accidents from earlier in the case of the "Paul Walker" event for example.

Overall the adaptive NNLM approach performs much more effectively in terms of recall rather than precision. A more effective summarization step could potentially im-
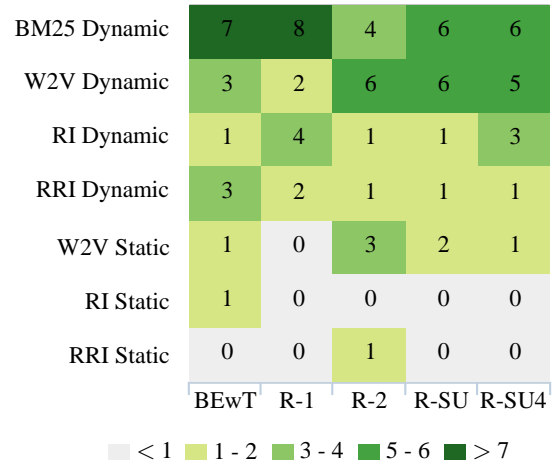

Figure 3: Methods Ranked 1st in F_1 Score, under various ROUGE settings, in all 16 events.

prove accuracy further. This property makes this model suitable for use as a supporting tool in helping journalists find the most relevant tweets for a timeline or liveblog, as the tweets retrieved tend to be much more diverse than those retrieved by the BM25 approach, which favours longer tweets with more repetitive use of terms.

Diversity of generated Timelines: The average pairwise cosine similarity of tweets in the timelines was used as a measure of diversity. Using the diversity score of the reference timelines, the diversity and redundancy can be compared relative to the available references. Scores above 1.0 indicate that timelines have less repetition and redundant information than human generated timelines. Scores below 1.0 indicate that tweets in the timeline are very similar and repetitive. Figure 4 shows events where at least 1 method produces a more diverse timeline than the reference. The Nonadaptive approach performs well in cases where the story context does not change much, tracking reactions of celebrity deaths for example. Timelines generated with this variant tend to be more general.

While the additive compositionality of learnt word representations works well in most cases, there are limits to this usefulness. Short, focused seed queries tend to yield better results. Longer queries benefit baseline term frequency models but hurt performance of the NNLM approach.

Future and Ongoing Work

Currently, there is a lack of high quality annotated Twitter timelines available for newsworthy events, as current methods provided by Twitter for creating custom timelines are limited to either manual construction, or through a private API. Other forms of liveblogs and curated collections of tweets are more readily available, but vary in quality.

As new timelines are curated, we expect that the available set of events to evaluate will grow. We make our dataset of our reference timelines and generated timelines available[5].

We adopted an automatic evaluation method for assessing timeline quality. A more qualitative evaluation involving

---
[5]http://mlg.ucd.ie/timelines

	RobFord	Tornado	Yale	MH17	MH370	WHCD	PWalker
Reference	1	1	1	1	1	1	1
BM25 Dynamic	1.5	0.7	0.8	0.7	0.9	1.1	0.9
W2V Dynamic	0.5	1.1	1.1	0.6	1.2	0.7	1
RI Dynamic	0.8	0.5	0.9	1.2	1	1	1
RRI Dynamic	0.6	0.8	0.8	1.2	1.1	0.7	1.3
W2V Static	0.4	1.1	0.7	0.6	0.9	0.5	0.6
RI Static	0.7	0.5	0.8	0.6	0.9	1	0.7
RRI Static	0.5	0.5	0.8	0.5	0.8	0.7	0.6

■ < 0.8 ■ 0.8 - 1.0 ■ > 1.0

Figure 4: Diversity relative to reference timelines for events where automatic methods performed better than human generated timelines.

potential users of this set of tools is currently in progress. There is also room for improving the model retraining approach. Rather than updating the model training data with a fixed length moving window over a tweet stream, the model could be retrained in response to tweet volume or another indicator, such as the number of "out of bag" words, *i.e.* words for which the model does not have vector representations for. Retrieval accuracy is also bound by the quality of our curated tweet stream, expanding this data set would also improve results.

The SumBasic summarization step does not make use of any information from the retrieval models, a better summarization approach that explicitly accounts for diversity and novelty could take better advantage of the DSM approaches.

Conclusion

Distributional semantic models trained on Twitter data have the ability to capture both the semantic and syntactic similarities in tweet text. Creating vector representations of all terms used in tweets enables us to effectively compare words with account mentions and hashtags, reducing the need to pre-process entities and perform query expansion to maintain high recall. The compositionality of learnt vectors lets us combine terms to arrive at a similarity measure between individual tweets.

Retraining the model using fresh data in a sliding window approach allows us to create an adaptive way of measuring tweet similarity, by generating new representations of terms in tweets and queries at each time window.

Experiments on real-world events suggest that this approach is effective at filtering relevant tweets for many types of rapidly evolving breaking news stories, offering a useful supporting tool for journalists curating liveblogs and constructing timelines of events.

Acknowledgements

References

[Chatterjee and Sahoo 2013] Chatterjee, N., and Sahoo, P. K. 2013. Effect of Near-orthogonality on Random Indexing Based Extractive Text Summarization. *International Journal of Innovation and Applied Studies* 3(3):701–713.

[Chong and Chua 2013] Chong, F., and Chua, T. 2013. Automatic Summarization of Events From Social Media. In *Proc. 7th International AAAI Conference on Weblogs and Social Media (ICWSM'13)*.

[Cohen, Schvaneveldt, and Widdows 2010] Cohen, T.; Schvaneveldt, R.; and Widdows, D. 2010. Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. *Journal of Biomedical Informatics* 43(2):240–256.

[Deerwester et al. 1990] Deerwester, S. C.; Dumais, S. T.; Landauer, T. K.; Furnas, G. W.; and Harshman, R. A. 1990. Indexing by latent semantic analysis. *JASIS* 41(6):391–407.

[Ferguson et al. 2011] Ferguson, P.; OHare, N.; Lanagan, J.; Smeaton, A. F.; Phelan, O.; McCarthy, K.; and Smyth, B. 2011. Clarity at the trec 2011 microblog track.

[Frankl and Maehara 1988] Frankl, P., and Maehara, H. 1988. The johnson-lindenstrauss lemma and the sphericity of some graphs. *Journal of Combinatorial Theory, Series B* 44(3):355–362.

[Inouye and Kalita 2011] Inouye, D., and Kalita, J. K. 2011. Comparing twitter summarization algorithms for multiple post summaries. In *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)*, 298–306. IEEE.

[Jones, Walker, and Robertson 2000] Jones, K. S.; Walker, S.; and Robertson, S. E. 2000. A probabilistic model of information retrieval: Development and comparative experiments. *Inf. Process. Manage.* 36(6):779–808.

[Jurgens and Stevens 2009] Jurgens, D., and Stevens, K. 2009. Event detection in blogs using temporal random indexing. *Proceedings of the Workshop on Events in . . .* 9–16.

[Kanerva, Kristoferson, and Holst 2000] Kanerva, P.; Kristoferson, J.; and Holst, A. 2000. Random indexing of text samples for latent semantic analysis. In *In Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, 103–6. Erlbaum.

[Kanerva 1988] Kanerva, P. 1988. *Sparse distributed memory*.

[Leskovec, Backstrom, and Kleinberg 2009] Leskovec, J.; Backstrom, L.; and Kleinberg, J. 2009. Meme-tracking and the dynamics of the news cycle. *Proc. 15th ACM SIGKDD international conference on Knowledge discovery and data mining* 497.

[Li and Cardie 2013] Li, J., and Cardie, C. 2013. Timeline Generation : Tracking individuals on Twitter. *arXiv preprint arXiv:1309.7313*.

[Lin 2004] Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In Marie-Francine Moens, S. S., ed., *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 74–81. Barcelona, Spain: Association for Computational Linguistics.

[Mikolov et al. 2013a] Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Distributed Representations of Words

and Phrases and their Compositionality. In *Proceedings of NIPS'13*, 1–9.

[Mikolov et al. 2013b] Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013b. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Owczarzak et al. 2012] Owczarzak, K.; Conroy, J. M.; Dang, H. T.; and Nenkova, A. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, 1–9. Stroudsburg, PA, USA: Association for Computational Linguistics.

[Petrović, Osborne, and Lavrenko 2012] Petrović, S.; Osborne, M.; and Lavrenko, V. 2012. Using paraphrases for improving first story detection in news and Twitter. In *Proc. Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 338–346.

[Sahlgren 2005] Sahlgren, M. 2005. An introduction to random indexing. In *Semantic Indexing Methods and applications workshop, TKE*, volume 5.

[Shou 2013] Shou, L. 2013. Sumblr: Continuous Summarization of Evolving Tweet Streams. In *Proc. 36th SIGIR conference on Research and Development in Information Retrieval*, 533–542.

[Tratz and Hovy 2008] Tratz, S., and Hovy, E. 2008. Summarization evaluation using transformed basic elements. In *In Proceedings of the 1st Text Analysis Conference (TAC*.

[Vanderwende et al. 2007] Vanderwende, L.; Suzuki, H.; Brockett, C.; and Nenkova, A. 2007. Beyond sumbasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management* 43(6):1606–1618.

[Wang and Lin 2014] Wang, Y., and Lin, J. 2014. The impact of future term statistics in real-time tweet search. In de Rijke, M.; Kenter, T.; de Vries, A.; Zhai, C.; de Jong, F.; Radinsky, K.; and Hofmann, K., eds., *Advances in Information Retrieval*, volume 8416 of *Lecture Notes in Computer Science*. Springer International Publishing. 567–572.

[Wang 2013] Wang, T. 2013. Time-dependent Hierarchical Dirichlet Model for Timeline Generation. *arXiv preprint arXiv:1312.2244*.

[Yan et al. 2011] Yan, R.; Wan, X.; Otterbacher, J.; Kong, L.; Li, X.; and Zhang, Y. 2011. Evolutionary Timeline Summarization : a Balanced Optimization Framework via Iterative Substitution. In *Proc. 34th SIGIR Conference on Research and development in Information Retrieval*, 745–754.