Finding Short Lived Events on Social Media

David Kilroy[1], Simon Caton[1], and Graham Healy[2]

[1] School of Computer Science, University College Dublin, Ireland
david.kilroy1@ucdconnect.ie, simon.caton@ucd.ie
[2] School of Computing, Dublin City University, Ireland
graham.healy@dcu.ie

Abstract. Companies have been looking to Social Media for event and trend detection for a number of years to assist in business decision making. Contemporary approaches typically consider specific time windows within which to search for evidence of emerging topics, events or trends. Yet, in doing so, short(er) term events can be crowded out. In this paper, we show how subdividing time windows and recombining their results can significantly improve detection. To do this, we use three historical Twitter datasets that are prevalent in the literature. We evaluate 6 different methods common in the literature, and in most cases cases, observe a significant improvement through our approach. However, we also note that picking the correct subdivision of time is key to this improvement.

Keywords: Twitter · Event Detection · Temporal Windowing

1 Introduction

Social media services such as Twitter, Reddit, Sina Weibo and Facebook have gained a lot of traction in recent years allowing users to spread information at the click of a finger. Users of these services can therefore be seen as *social sensors* [1] reporting real-world experiences and opinions about hot topics in real time. In order to make sense of this massive amount of data, event detection techniques can be used in order to identify the latent events in the continuous stream of posts. Similarly to defining a topic in topic modelling, an *event* can be defined as a "number of keywords" [2] adhering to the same subject.

In the literature, it is common to run an event detection algorithm over a time window of data and then extract events [3–5] e.g. extract 10 events every hour. A problem with this approach is that if the time window of interest is too large or has too high of an arrival rate of data, many event detection algorithms cannot capture the "short lived" events if they are not prevalent enough. This is due to the fact that many event detection algorithms do not have any intrinsic notion of time in them, thus are not able to identify a group of posts occurring in close proximity. Fig. 1 details this problem. If an algorithm is run over a single time window (as in Fig. 1) it will not be able to differentiate the burst of posts (in orange) compared to the steady stream of posts (in blue).

In order to deal with this problem we propose to subdivide the data into shorter windows (in grey) from the original time window, thus allowing an event detection algorithm to distinguish "short lived" events in shorter time windows. At the end of the final subdivided time window (when a subset of events have

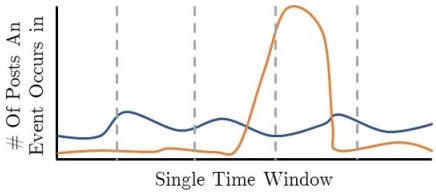

Fig. 1. Challenges of Finding Short Lived Events within Fixed Time Windows

been collected) we propose an ending *event clustering and ranking mechanism*. This is used to group similar events between the *short time windows*, so that detected events are not repeated in the final ranking by an *importance score*.

In our evaluation we benchmark various algorithms used in topic modelling to more classical families of topic detection algorithms with and without the presence of *short time windows* and an ending *event clustering and ranking mechanism*. We run these algorithms on three different datasets [3] which contain ground truth event labels that allows the calculation of specific key event detection metrics and comparison with key approaches in the literature.

2 Related Work

We first discuss the three categories of event detection loosely grouped into (a) type of event (*specified* or *unspecified*) (b) detection task (*retrospective* or *new*) and (c) detection algorithms [6, 7]. We then briefly discuss dynamic topic modelling (DTM), with our approach being closely related to it. Finally we discuss our approach and which categories it aligns to.

Specified event detection differs from *unspecified* event detection in that some prior information is known about the event before performing detection. This may be in the form of only searching for posts with specific content/terms [8] or in a specific geolocation [4]. In contrast, *Unspecified* event detection uses no prior information and instead finds events from unlabelled posts.

Retrospective (or offline) event detection (*RED*) tries to find events that have already happened from historical collections of posts. New (or online) event detection (*NED*) tries to find events from a continuous stream of posts. As pointed out in [9], *RED* provides better results than *NED*. This is due to the fact that *RED* techniques can make better decisions than *NED* by having a view of the corpus in its entirety. However, when dealing with a continuous stream of documents in real-time only *NED* techniques can be performed.

Generally speaking "the detection of unknown events" [10] can be solved in three ways: the documents themselves are clustered (document-pivot) [4, 11, 12], the terms from the documents are selected then clustered (feature-pivot) [3, 5, 13] or events are seen as a probability distribution over documents/terms (topic modelling) [14]. In document-pivot approaches, the documents themselves are represented as a bag-of-words which may be weighted by term frequency-inverse document frequency. These documents are then grouped together using clustering techniques with some similarity metric e.g. cosine. If an incoming document is considered dissimilar to any existing cluster by a predefined threshold, it is considered a new event. Early approaches of document-pivot techniques were

proposed in [15], where the objective was to find the first story (i.e. first report of an event) within an incoming stream of stories. However, this approach did not scale to the large volume of data on twitter. In response to this problem [12] proposed an approach using locality sensitive hashing, allowing for a considerable speed-up. In feature-pivot techniques there are two main stages in the event detection process: (a) have some way to relate terms to each other, and (b) group terms together to form events. Early approaches related terms based off their signal cross correlation [16], with other approaches using co-occurrence [17] or pairwise similarity from word embeddings [18, 19]. Grouping terms is usually followed by clustering, with works such as [19] and BnGrams in [3] using hierarchical clustering. Other approaches group terms by using modularity-based graph partitioning [16]. It is also common for feature-pivot techniques to identify *bursty* terms [3, 4, 13] (i.e. terms occurring at an unusually high rate) and then only consider terms which meet this criteria. Topic models can be seen as a dimensionality reduction technique by splitting a document-term matrix into a document-topic matrix and a term-topic matrix. Some of the most well known topic models include Latent Dirichlet Allocation (LDA) [20], Non-Negative Matrix Factorization [21] and Latent Semantic Analysis (LSA) [22]. Much work on topic modelling in social media focuses on variants of popular algorithms such as LDA for short text mining [23], while other works focus purely on real-time over online (NED) topic models [14] with the subtle difference that "in real-time detection, time is crucial, so much so that no fixed time window for detection should be assumed" [14].

Our work is closely related to DTM in the sense that topics are obtained by dividing the data into *shorter time windows* of equal length. [24] proposed a DTM based on two layers of NMF. The first layer extracts the document-topic matrix for each window topic model and stacks it onto a matrix B. The second layer of NMF is then applied to B to obtain dynamic topics.

Our approach uses all three families of detection algorithms mentioned, is of unspecified detection type (although the datasets in our evaluation search for posts with the presence of specific lists of terms and hence is specified) and only uses techniques that deal with an incoming stream of posts (NED). Many algorithms miss out on many "short lived" events due to having no intrinsic notion of time in them. To overcome this problem, our approach uses ideas from DTM by subdividing the data into *short(er) time windows* and thus being able to find "short lived" events. Our approach differs from DTM in the fact that we recombine the events found at *short(er) time windows* in a way that focuses on the effectively ranking events rather than trying to combine events found at *short time windows* to make dynamic topics.

3 Implementation

Fig. 2 outlines our approach to retrieving a number of events for each time window. The processes of "Data Pre-processing", "Event Detection Algorithms" and "Event Clustering and Ranking" make up the following subsections of this section. The code for this approach can be found at github.com/davidkilroy/EDetect.

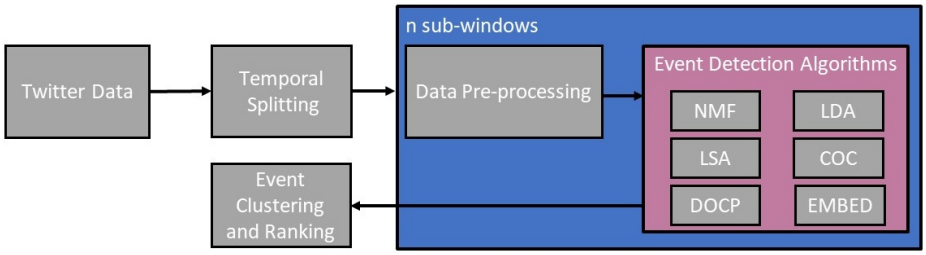

Fig. 2. Implementation Overview

3.1 Data Pre-processing

The data pre-processing stage is split into three main stages: (i) general pre-processing, (ii) term importance and (iii) aggressively filtering documents.

General Pre-processing: For each post, the raw text is tokenized and then each token is reduced to its base form using lemmatization, so to avoid the poor readability of events as can happen with stemming [3, 11]. When tokenizing the text we only consider unigrams, which are then converted to lowercase. Terms with certain part-of-speech (POS) tags such as determiners, conjunctions, subordinating conjunctions, particles and punctuation are removed due to providing little meaning into the documents. Twitter specific *#hashtags* and *@usermentions* are removed due to producing more "interesting" events [25]. All words which are in the Terrier list of stopwords as well as all non-English words are also removed.[1] The python library spaCy was used to carry out the data pre-processing.

Term Importance: A term importance score is generated for each term following on from [2, 13, 26] and many others as we only consider a subset of *bursty* terms. The term importance score is calculated based on two criteria (a) how *bursty* the term is in a time window and (b) term POS tag which acts as a scaling factor. As used in [14, 25] to evaluate the significance of a term, we also make use of z-scoring, however in our case we use it to define how *bursty* a term is in a particular time window. We do this by comparing an individual term's frequency to its frequency at previous time windows. We also normalize each term's window frequency by the total term frequency for a window to arrive at the "relative popularity" for each term [25], ensuring seasonal patterns don't affect the term's score. Similarly to how BnGram defines how *bursty* a term is [3], we also do not compare a term's frequency in a particular window to its frequency in all other windows but only to its frequency in the window's behind it. Failing to comply with this would not make our approach compatible for NED tasks. To increase the likelihood that more "event worthy" POS tags appear in events, we also scale each z-score by an additional *boost factor* based on its POS tag, like in [3, 11]. We assign this *boost factor* a value of 2.5 as in [11] and apply it to the z-scores of all terms with nouns and proper noun POS tags to arrival at the final *term importance score* for each term. In order to comply with our original

[1] https://github.com/terrier-org/terrier-desktop/blob/master/share/stopword-list.txt - last accessed 14/11/2020

objective of only considering a subset of *bursty* terms, we remove all terms from the documents which don't surpass a *term importance score* of 0 i.e. a term must be occurring more than usual for it to be considered. Similarly to [11], we also set a minimum threshold frequency in order for a term to be considered in a particular time window. In our tests a minimum frequency of 0.005 times the number of total posts for nouns and pronouns and a minimum frequency of 0.015 times the number of total posts for all other POS tags worked well.

Aggressively Filtering Documents: [11] seeks to remove documents with a low term count to improve performance. A minimum document length of 4 was used in the experiments of [11], and was also used in our experiments.

3.2 Event Detection Algorithms

With each of the following algorithms producing events, we can quantify how important an event is (*event importance*) by finding its summed *term importance score* divided by the total number of terms in the event. This *event importance score* is used in the *event clustering and ranking mechanism* when the final ranking of events is calculated.

In event detection, one important hyper-parameter is the number of events to detect. There are many approaches for automatically finding this value [27], however in our evaluation the number of events for each full time window is specified in advance [3]. With the following algorithms all using topic models, another hyper-parameter to be considered is the number of terms to extract for each event. We choose this number to be 10, as in preliminary experimentation this yielded the best performance; further optimisation is left for future work.

Non-Negative Matrix Factorisation (NMF): NMF [21] is used in topic modelling to reduce the dimensions of a non-negative matrix $\mathbf{X}$ (document-term matrix) by splitting it into a document-topic matrix $\mathbf{W}$ and a term-topic matrix $\mathbf{H}$ whose product approximates the original non-negative matrix $\mathbf{X}$. When applying NMF, [28] shows that the use of tf-idf vectors over a raw document-term matrix can provide more coherent topics and thus was included in our implementation.

Latent Dirichlet Allocation (LDA): LDA is a "generative probabilistic model for collections of discrete data such as text corpora" [20]. Similarly to NMF, it can be seen as a dimensionality reduction technique. When applying LDA, [28] also shows the use of a raw document-term matrix providing more coherent topics than its tf-idf weighted counterpart and hence was used in our implementation.

Latent Semantic (LSA): LSA uses a singular value decomposition (SVD) to approximate a matrix [22] (dimensionality reduction). In our implementation the document-term matrix inputted into LSA is tf-idf weighted.

NMF Applied to a Co-occurrence Matrix (COC): In this feature-pivot/topic modelling approach we apply NMF to a co-occurrence matrix. A co-occurrence matrix is a term-by-term matrix which signifies how many times two terms co-occurred. In our experiments, co-occurrence is defined as two terms co-occurring

in the same post rather than within some window length. It was defined like this due to the short character limit on Twitter (280), with words in the same post likely being related to one another. So long as the number of documents is larger than the number of terms in the corpus of interest, the first step of this approach can be seen as an additional dimensionality reduction step before running NMF and therefore may provide the following benefits: (a) more stable results as approximating a small matrix leads to less varied results b) computational speed-up as approximating a smaller matrix takes less time.

NMF Applied to a Term Embeddings Matrix (EMBED): Word embeddings try to relate words based on their semantic similarity to one another. It would be useful to know if word embeddings could capture the relationships between similar events in order to provide accurate event detection as in [18, 19]. We implement this by constructing a term embeddings matrix by first training a word2vec model [29] on the time window corpus. We then match each word to one another using the cosine similarity between the vectors to arrive at the final term embeddings matrix. NMF is then applied to this matrix in order to extract events.

Document-pivot Approach Using NMF (DOCP): This topic modelling/document-pivot approach uses the document-topic matrix (tf-idf weighted) returned by NMF to construct events. It does this by retrieving the most activated document for a particular given topic and then retrieves the terms in that document. It does this until 10 terms are found.

3.3 Event Clustering and Ranking Mechanism

Given that overlapping and similar events may be detected in different *short time windows*, a way to effectively rank and combine these events is needed. A simple ranking of events by the defined *event importance score* would lead to similar events with similar terms and thus similar *event importance scores* being grouped together in the final ranking of events (lacking diversity). In order to overcome this problem we first cluster similar events together that - along with some other processing - eliminates them being grouped together (event clustering). This is followed by a final ranking of events (event ranking), with the aim of achieving a diverse set of ranked events.

Event Clustering: The event clustering stage is an incremental process which groups together events emitted from each *short time window* at increasingly higher distance thresholds (cosine) in order to group the most similar events first, followed by less similar events. In each iteration of the event clustering stage, the events are first represented as a document-term matrix. The events (now considered documents) are then clustered using agglomerative clustering using an incrementing distance threshold. The events which cluster below a specific distance threshold are grouped into *event clusters*. The events in each of the grouped *event clusters* are also ranked between themselves by their *event importance score* (defined in Section 3.2). Each of the grouped *event clusters* are also given an *event cluster importance score* which can be defined as the

mean *event importance scores* in the *event cluster*. The events which aren't clustered are passed onto the next stage of clustering with an incrementing distance threshold. Events which never cluster are considered single *event clusters*. The values used for the distance thresholds are 0.2 (most similar), 0.4 (less similar) and 0.7 (just about similar). These values work well in our experiments however other values which mimic a similar upwards progression could also be tried.

Event Ranking: The first stage of the event ranking phase is to rank the grouped *event clusters* between themselves. The *event clusters* are ranked by a) whatever distance threshold they were found at (ascending), b) the number of events in each *event cluster* (descending) and c) *event cluster importance score* (descending) in that order. Thus, events which are highly similar to other events are themselves important (ranking by distance threshold) and large numbers of similar events represent an overall important event theme (ranking by number of events). In order to obtain a final ranking of events, the ranked *event clusters* are iterated through one by one popping off the top event in each of them into the final ranking of events. This aims to achieve a diverse (obtaining events at different event clusters) and popular (sorting event clusters by popularity) ranking of events.

4 Evaluation

In order to demonstrate the effectiveness of our approach, we first compare the algorithms discussed in Section 3.2 with and without the presence of subdividing the data into *short time windows* followed by an ending *event clustering and ranking mechanism*. This is followed by an experiment showing the importance of picking a good value for the *number of short time windows* variable.

4.1 Datasets

While other proposed approaches evaluating event detection algorithms make use of human evaluators [16, 30], this approach leverages an existing automated evaluation solution [3] that compares lists of submitted events to a subset of ground truth labels for a predefined full time window, allowing specific event detection metrics to be calculated. The datasets proposed in [3] each consist of a collection of tweets from three major events in 2012; a) FA Cup Final, b) Super Tuesday Primaries and c) US Elections. The fact that the three datasets themselves have different characteristics, it makes it hard to achieve high accuracy across all three as seen in [5]. These characteristics include a) length of full time window to submit events (1 minute for FA Cup, 1 hour for Super Tuesdays and 10 minutes for US Elections), b) number of events to submit at each full time window (20 for FA Cup and 100 for both Super Tuesdays and US Elections) and c) dataset size/arrival rate of data at each full time window (148,652 for FA Cup, 474,109 for Super Tuesdays and 1,247,483 for US Elections). These three datasets are also used in the evaluation of many popular event detection algorithms [3–5] (useful for benchmarking purposes).

As there is a number of events to be submitted at each full time window for these datasets, it is common practice to see the performance of algorithms when

using a reduced number of submitted events (assuming the submitted events are ranked) [3–5] e.g. Super Tuesdays on 10, 20, 50 and 100 rather than 100 events.

As described in the paper of [3] and their evaluation script,[2] in each ground truth label there are three sets of terms: a) mandatory, b) optional and c) prohibited. From these sets of terms it is possible to calculate "event recall", which can be defined as the number of ground truths "detected" by the algorithm where in this case "detected" refers to all mandatory terms being matched by the algorithms submitted terms (so long as no prohibited terms are matched). In our evaluation we don't record term precision and term recall [3] instead focusing on event recall. Also term precision and term recall can also be considered misrepresentation metrics due to the fact that they're only recorded after an event is considered "detected".

4.2 Assessing the General Impact of Short Time Windows

In order to assess the general impact of *short time windows* we define two approaches: approach A which subdivides data into *short(er) time windows* followed by the *event clustering and ranking mechanism*; and approach B (i.e. baseline approach) which runs an algorithm over a full time window with no post-clustering or ranking of events. Both of these approaches use the same data pre-processing as described in Section 3. With an important hyper-parameter for approach A being the *number of short time windows* to use when extracting events for a full time window, it includes a range of values for the *number of short time windows* variable, specifically 2, 3, 4, 5, 6, 10, 12 and 15. These values were chosen as they result in equal time windows (all factors of 60) and the amount of data at any *short time window* is not too small e.g with their being an average of 492 posts in each full time window for the FA Cups dataset, dividing by more than 15 would result in too little data at each *short time window*.

We test the hypothesis **H**, that the mean ranks of pooled event recall results generated by approach A are greater than the mean ranks of event recall results generated by approach B. In order to do this, for approach A, 72 event recall scores were calculated for each different value of the *number of short time windows* variable (totalling 576 event recall scores). Similarly, 576 event recall scores were calculated for approach B. For each event recall score which was calculated a different random seed was used.[3] A Mann-U-Whitney Rank Test was run comparing approach A and approach B for each baseline algorithm at a range of different submitted events across three different datasets. Specifically, the range of submitted events was 2, 5, 10 and 20 for FA Cup and 10, 20, 50 and 100 for Super Tuesdays and US Elections. The reason this test was used instead of a t-test was because the results data was not normally distributed. Table 1 shows the p-value (rounded to 3 decimal places) of this test for the specified number of events (in columns). If the test was in favour of approach B, a $+$ was post-fixed to the result.

[2] http://socialsensor.iti.gr/use-cases/evaluation - last accessed 14/11/2020

[3] Multiple runs of the same algorithm were required due to the random element present in each of the algorithms

One possible reason for the embeddings model showing evidence of a statistically significant difference in favour of approach B for the FA Cups dataset might be that the word2vec model wasn't able to capture accurate relationships between words when run on a small subset of the data. The FA Cups dataset has the lowest tweet count out of the three datasets, furthering this point. A possible reason for the results being varied between models in the US Elections dataset might be due to the fact that the ground truth events picked span the full time period of interest, and not being considered "short lived" (something which the *short time windows* in approach A try to capture). On the other hand, our hypothesis was almost always supported by statistically significant results for the FA Cups and Super Tuesdays datasets, perhaps due to the fact that they contain many "short lived" events as per their ground truths.

Table 1. P-value of Approach A vs Approach B Mann-U-Whitney Rank Test

	FA CUP				SUPER TUESDAYS				US ELECTIONS			
	2	**5**	**10**	**20**	**10**	**20**	**50**	**100**	**10**	**20**	**50**	**100**
NMF	0.072^{+}	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
LDA	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000^{+}	0.000^{+}	0.000^{+}	0.084
LSA	0.000	0.000	0.161	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
COC	0.000^{+}	0.089^{+}	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.002^{+}	0.000^{+}	0.000
DOCP	0.000	0.000	0.000	0.000	0.079^{+}	0.009	0.260	0.000	0.000^{+}	0.000^{+}	0.000^{+}	0.000^{+}
EMBED	0.054	0.000^{+}	0.000^{+}	0.000^{+}	0.001	0.214	0.032^{+}	0.002	0.000^{+}	0.000^{+}	0.000^{+}	0.000^{+}

+ test was in favour of the baseline

4.3 Picking a "Reasonably" Good Value for the Number of Short Time Windows Variable and Comparisons with Other Methods

Within the range of values for the *number of short time windows* used in testing approach A (described in Section 4.2) it is important to pick a value for this variable which yields "reasonably" high results (due to high variance of results with different values of the *number of short time windows* variable). Table 2 shows the high variance of approach A's results compared to approach B's results (described in section 4.2) in a *(maximum - minimum, median)* tuple using the same range of submitted events used in Table 1. A "_b" was post-fixed to the model name where approach B was used, otherwise approach A was used.

Table 2. Spread of Event Recall Results (Max-Min, Median)

	FA CUP				SUPER TUESDAYS				US ELECTIONS			
	2	**5**	**10**	**20**	**10**	**20**	**50**	**100**	**10**	**20**	**50**	**100**
NMF	.15,.77	.08,.85	.08,.92	.08,1.0	.50,.36	.64,.50	.64,.55	.68,.73	.44,.42	.39,.55	.30,.66	.20,.72
NMF_b	.08,.77	.08,.77	0.0,.85	0.0,.92	.18,.04	.27,.14	.23,.18	.41,.27	.27,.17	.25,.22	.17,.41	.16,.50
LDA	.31,.77	.23,.92	.15,.92	.15,.92	.32,.41	.41,.46	.36,.64	.36,.77	.30,.47	.25,.58	.19,.67	.14,.77
LDA_b	.23,.77	.08,.85	.08,.85	0.0,.92	.23,.23	.23,.27	.27,.46	.27,.55	.14,.55	.14,.66	.08,.74	.05,.75
LSA	.15,.85	.08,.92	.08,.92	.08,.92	.32,.23	.45,.32	.41,.41	.36,.59	.14,.31	.17,.46	.22,.59	.16,.69
LSA_b	0.0,.62	.08,.85	0.0,.92	0.0,.92	.05,.11	0.0,.23	.05,.23	0.0,.27	0.0,.19	0.0,.27	0.0,.47	0.0,.55
COC	.38,.65	.23,.85	.15,.92	.08,.92	.27,.36	.32,.48	.36,.59	.36,.86	.22,.52	.14,.59	.11,.70	.05,.75
COC_b	.08,.77	0.0,.85	0.0,.85	0.0,.92	.18,.18	.23,.32	.27,.50	.27,.55	.20,.48	.13,.62	.05,.73	.05,.73
DOCP	.08,.85	.15,.92	.08,.92	.08,1.0	.41,.23	.41,.27	.41,.46	.46,.59	.27,.43	.20,.52	.17,.64	.12,.72
DOCP_b	.08,.77	0.0,.85	0.0,.85	0.0,.92	.09,.23	.18,.27	.23,.46	.05,.55	.12,.52	.09,.67	.05,.73	.02,.75
EMBED	.38,.31	.23,.38	.15,.46	.15,.46	.27,.14	.36,.18	.36,.32	.41,.46	.27,.11	.31,.16	.38,.24	.34,.39
EMBED_b	.15,.31	.15,.38	.15,.54	.15,.62	.23,.09	.32,.18	.32,.32	.32,.36	.27,.23	.31,.34	.24,.48	.27,.53

"_b" - Approach B (baseline)

Interestingly, if a value for the *number of short time windows* for approach A is picked within a reduced consecutive range of values for each dataset, it can yield better and less varied results (when compared to approach A). Table 3 shows the same information as Table 2 however only showing the results for the *number of short time* windows within a reduced consecutive range for each dataset. Note that these values were derived by looking at the best results for each dataset. The range of values used for each dataset were: 2, 3 and 4 for the FA Cup; 10, 12 and 15 for Super Tuesdays; and 3, 4 and 5 for the US Elections.

It would be useful to automatically pick a value in a "reasonably" good range for the *number of short time windows* variable (Table 3). One factor to consider might be the granularity of the events required. The more or less *short time windows* there are the more chance there is to capture "short" or "broad" lived events retrospectively. Another factor to consider is the size of the full time window, with more "short lived" events occurring in large full time windows (e.g. a day). Similarly, if there is a high arrival rate of data at a particular full time window, the more chance there is for "short lived" events.

As seen in Table 3, it is clear to see the advantages *short time windows*. Using *short time windows* seems to work especially well when the respective ground truths contain "short lived" events, as in the Super Tuesdays dataset which has a large full time windows (1 hour) allowing for more "short lived" events.

The algorithms exhibit very competitive results in comparison to other methods [3–5] in the presence of subdividing data into short time windows (when picked in an optimal range of the *number of short time windows* variable) followed by an ending *event clustering and ranking mechanism* as show in the median "event recall" results in 3. However, the algorithms show poor results when run over a full time window with no post-ranking of events, as seen in the median "event recall" results in 2. This may allude to the fact that presence of the *short time windows* along with the *event clustering and ranking mechanism* contributes to the improved results rather than the algorithms themselves. With this being said, it would be interesting to see how algorithms which already perform well when run on full time windows [5] perform in the presence of *short time windows* along with an ending *event clustering and ranking mechanism*.

Table 3. Spread of Event Recall Results Within A Consecutive Range of the Number Short Time Windows Variable (Max-Min, Median)

	FA CUP				SUPER TUESDAYS				US ELECTIONS			
	2	**5**	**10**	**20**	**10**	**20**	**50**	**100**	**10**	**20**	**50**	**100**
NMF	.08,.77	.08,.85	.08,1.0	0.0,1.0	.23,.41	.27,.59	.41,.77	.27,.86	.19,.44	.12,.57	.09,.67	.08,.75
LDA	.15,.77	.23,.92	.08,.92	.08,1.0	.32,.41	.41,.50	.23,.64	.27,.82	.24,.54	.16,.61	.11,.72	.09,.78
LSA	.15,.85	.08,.92	.08,.92	.08,1.0	.18,.23	.32,.36	.27,.46	.18,.59	.09,.33	.14,.45	.19,.59	.09,.73
COC	.08,.69	0.0,.92	.08,.92	0.0,1.0	.09,.46	.18,.55	.14,.73	.18,.86	.22,.45	.14,.58	.11,.70	.05,.75
DOCP	0.0,.85	.15,.92	.08,.92	.08,1.0	.18,.27	.27,.36	.32,.48	.36,.77	.27,.48	.20,.55	.14,.66	.08,.73
EMBED	.15,.38	.15,.46	.15,.54	.08,.54	.27,.14	.27,.18	.27,.27	.23,.36	.19,.13	.23,.20	.23,.31	.22,.42

5 Conclusion

Many "short lived" events on social media go undetected due to the nature of many popular algorithms not considering their full time window(s) when return-

ing ranked lists of events. In this paper, we proposed how subdividing time into shorter time windows, extracting events and then recombining the event results using an *event clustering and ranking mechanism* can provide statistically significant improvements compared to when no time-based subdivisions are used.

We note, however, that to find a "reasonable" value for this *number of short time windows* variable in order to yield improved results is not always straightforward. Picking this variable could be a function of a) granularity of the events required, b) size of full time window and c) arrival rate of data. More research is needed to identify robust methodologies for how to determine this value.

In our approach, we proposed ranking events based on a "event clustering and ranking mechansim". However in reality, any other way of effectively ranking events could be explored and thus would constitute an area of future work.

In this paper, we have explored a selection of event detection methods. It would be interesting to see how some algorithms perform (which already achieve high results when not considering short(er) time windows perform) when run on multiple subdivisions of the data before having their events recombined together.

Acknowledgements

This work was funded by Science Foundation Ireland through the SFI Centre for Research Training in Machine Learning (18/CRT/6183).

References

1. Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860, 2010.
2. Jon Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003.
3. Luca Maria Aiello, Georgios Petkos, et al. Sensing trending topics in Twitter. *IEEE Transactions on Multimedia*, 15(6):1268–1282, 2013.
4. Carmela Comito et al. Bursty event detection in Twitter streams. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(4):1–28, 2019.
5. Zafar Saeed, Rabeeh Ayaz Abbasi, Imran Razzak, Onaiza Maqbool, Abida Sadaf, et al. Enhanced heartbeat graph for emerging event detection on Twitter using time series networks. *Expert Systems with Applications*, 136:115–132, 2019.
6. Farzindar Atefeh and Wael Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.
7. Mário Cordeiro et al. Online social networks event detection: a survey. In *Solving Large Scale Learning Tasks. Challenges and Algorithms*, pages 1–41. Springer, 2016.
8. Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. Tedas: A twitter-based event detection and analysis system. In *2012 IEEE 28th International Conference on Data Engineering*, pages 1273–1276. IEEE, 2012.
9. Yiming Yang, Tom Pierce, et al. A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 28–36, 1998.
10. James Allan et al. Topic detection and tracking pilot study final report. 1998.
11. Georgiana Ifrim, Bichen Shi, and Igor Brigadir. Event detection in twitter using aggressive filtering and hierarchical tweet clustering. In *Second Workshop on Social News on the Web (SNOW), Seoul, Korea, 8 April 2014*. ACM, 2014.

12. Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics*, pages 181–189. Association for Computational Linguistics, 2010.
13. Yu Zhang and Zhiyi Qu. A novel method for online bursty event detection on Twitter. In *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pages 284–288. IEEE, 2015.
14. Wei Xie et al. Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2216–2229, 2016.
15. James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–45, 1998.
16. Jianshu Weng and Bu-Sung Lee. Event detection in twitter. In *Fifth international AAAI conference on weblogs and social media*, 2011.
17. Michael Mathioudakis and Nick Koudas. TwitterMonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158, 2010.
18. Carmela Comito, Agostino Forestiero, and Clara Pizzuti. Word Embedding based Clustering to Detect Topics in Social Media. In *2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 192–199. IEEE, 2019.
19. Ali Mert Ertugrul, Burak Velioglu, and Pinar Karagoz. Word embedding based event detection on social media. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 3–14. Springer, 2017.
20. David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
21. Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.
22. Scott Deerwester, Susan T Dumais, et al. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.
23. Xueqi Cheng, Xiaohui Yan, et al. Btm: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2928–2941, 2014.
24. Derek Greene et al. Exploring the political agenda of the european parliament using a dynamic topic modeling approach. *arXiv preprint arXiv:1607.03055*, 2016.
25. Erich Schubert, Michael Weiler, and Hans-Peter Kriegel. Signitrend: scalable detection of emerging topics in textual streams by hashed significance thresholds. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 871–880, 2014.
26. David A Shamma, Lyndon Kennedy, and Elizabeth F Churchill. Peaks and persistence: modeling the shape of microblog conversations. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 355–358, 2011.
27. Jonathan Chang et al. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296, 2009.
28. Derek O'Callaghan et al. An analysis of the coherence of descriptors in topic modeling. *Expert Systems with Applications*, 42(13):5645–5657, 2015.
29. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
30. Symeon Papadopoulos, David Corney, and Luca Maria Aiello. SNOW 2014 Data Challenge: Assessing the Performance of News Topic Detection Methods in Social Media. In *SNOW-DC@ WWW*, pages 1–8, 2014.