








SensorTree: Bursty Propagation Trees as Sensors for Protest Event Detection

Jeffery Ansa^(✉), Wei Kang, Lin Liu, Jixue Liu, and Jiuyong Li

School of Information Technology and Mathematical Sciences,
University of South Australia, Adelaide, SA 5095, Australia
{[jeffery.ansah](mailto:jeffery.ansah@unisa.edu.au),[wei.kang](mailto:wei.kang@unisa.edu.au),[lin.liu](mailto:lin.liu@unisa.edu.au),[jixue.liu](mailto:jixue.liu@unisa.edu.au),[jiuyong.li](mailto:jiuyong.li@unisa.edu.au)}@unisa.edu.au

Abstract. Protest event detection is an important task with numerous benefits to many organisations, emergency services, and other stakeholders. Existing research has presented myriad approaches relying on tweet corpus to solve the event detection problem, with notable improvements over time. Despite the plethora of research on event detection, the use of the implicit social links among users in online communities for event detection is rarely observed. In this work, we propose SensorTree, a novel event detection framework that utilizes the network structural connections among users in a community for protest event detection. SensorTree tracks information propagating among communities of Twitter users as propagation trees to detect bursts based on the sudden changes in size of these communities. Once a burst is identified, SensorTree uses a latent event topic model to extract topics from the corpus over the burst period to describe the event that triggered the burst. Extensive experiments performed on real-world Twitter datasets using qualitative and quantitative evaluations show the superiority of SensorTree over existing state-of-the-art methods. We present case studies to further show that SensorTree detects events with fine granularity descriptions.

Keywords: Burst · Event detection · Propagation trees
Social media · Twitter

1 Introduction

With the advancement of Web 2.0 technologies, social media sites such as Twitter, Facebook and Weibo have become a viable source for monitoring and analyzing the rich continuous flow of information for protest event detection. To detect events, one of the predominant approaches is to model events in text streams as bursts with keywords rising sharply in frequency as an event emerges. The basic assumption is that some related words will exhibit a sudden increase in their usage when an event is happening. An event is therefore conventionally represented by a number of keywords showing burst in appearance counts.

While these event detection approaches have gained successes, some challenges still prevail. First choosing the right set of predefined keywords to track is

a herculean task and usually requires knowledge from domain experts. Secondly, once the right keywords are chosen, another hurdle is understanding the correct context of these bursty keywords for event detection. For example, if we have two tweets of the form: *“Reduce tax and import duties or get ready for protest”* and *“Too much protest from Arsenal fans in this game”*. We observe that the keyword “protest” is used in both tweets. However, the context clearly shows that the ongoing conversation is about two separate events. Keyword based models that use the counts of keywords for event detection may not be able to distinguish different events using similar predefined keywords.

A promising solution is to leverage the social network structure and follower relationship among users for protest event detection. The intuition here is that tweets which are sent between a tightly knit group of users in a community may be more indicative of a particular event of interest than a set of tweets being propagated by a random set of users who do not have any form of social network connections. This suggests that a discussion is more likely to become active during or even before an event among Twitter users who follow each other. Such implicit relationship among users in a community can be captured using propagation trees. Propagation trees have been proven [4,8] to be effective in depicting how communities of online users connected via social links (follower relationship) discuss topics of interest. These trees are able to capture the structure and temporal growth of communities as information propagates. For example, Fig. 1 shows a plot of the size distribution of propagation trees built on the hashtag “freeport” to capture series of events on coal mining in Indonesia. These trees show bursts (represented as spikes) that correspond to increasing online user activity prior to real world protest events of interest. The spikes 1, 3 and 5 correspond to days of protest events. Spikes 2 and 4 were days of significant events leading to protests. These dynamics clearly show that propagation trees are useful in capturing the discussion of new topics, the occurrence of new events, etc. in an online community, and can be used for our event detection task.

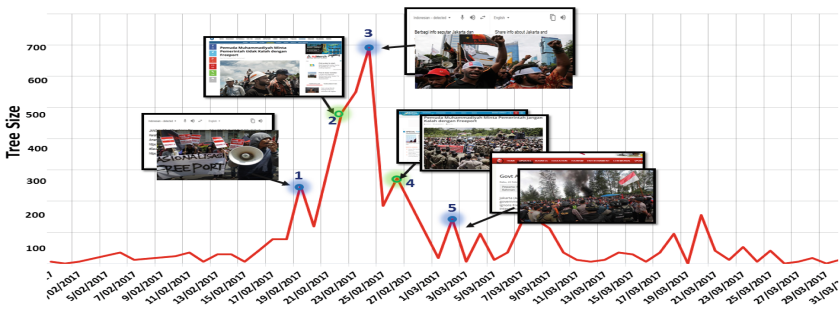


Fig. 1. Propagation tree size on the Freeport event (Blue dots (1, 3, 5) indicate main protest event days and green dots (2, 4) indicate sub-event days (sub-events are phenomena that are precursors to future protest events)). (Color figure online)

However, two main challenges exist: (1) Detecting bursts in online communities. This task requires accurately capturing the tree growth in continuous time to detect the period of sudden increase (burst) in online user communities. This problem is not trivial since there is currently no measure of quantifying bursts in propagation trees for event detection. (2) Event inference from trees. Once a burst is detected, the second challenge is to infer the protest event that has triggered the burst. Current propagation tree formulations make it impossible to infer the semantic context of the information propagation (i.e. what topic is being discussed at which time) for protest event detection. This is mainly because these information propagation studies [4, 8] are only interested in capturing the numeric structural and temporal features such as the size, depth, growth rate, etc., from information propagating through an online community.

To address the above challenges, we propose a novel event detection framework called SensorTree. SensorTree framework is developed upon the semantic propagation tree proposed in this paper. The semantic propagation tree does not only capture the numeric structural and temporal features but also has the capability of capturing the corpus (textual component) of the ongoing discussion in an online community of Twitter users. SensorTree builds semantic propagation trees and efficiently computes the changes in acceleration of the tree size. The change in acceleration is then used as a means of quantifying the period of a sudden change in growth of online communities as the burst for event detection. Once the period of a burst is detected, a tensorized latent event topic model is triggered to infer and provide extra textual information on the associated events causing the burst. SensorTree is efficient and effective in detecting protest events as well as discovering topics of discussion within an online community on a protest event.

To the best of our knowledge, this is the first work to perform event detection using the change in acceleration in the growth of online communities. Our main contributions can be summarized as:

- We propose a novel event detection method which leverages burst in information propagation from online communities to detect protest events using the change in acceleration of the size of a semantic propagation tree.
- We develop an event inference approach for describing detected protest events in bursty online communities using semantic propagation trees.
- We develop SensorTree, a novel framework based on the above proposed methods. Experiments on real world datasets show that SensorTree is language independent, does not require a predefined set of keywords and capable of detecting and describing events with fine granularity.

2 Related Work

Event Detection is a vibrant research area with evolving interest in news, blogs, and social media. This line of research has been extensively studied by the text mining community in the context of Topic Detection and Tracking [7, 14]. For the

purpose of our study, we review related works on event detection using social media, which can be categorized into two active lines of research:

1. Document clustering and semantic similarity: This line of work clusters data points on the basis of some textual similarity measures for event detection. The underlying assumption is that documents are somehow related to a number of undiscovered events. Events are identified by cluster associated word frequency or topic distribution using traditional LDA [7] topic models and its variants.

The authors of [5] explored multi-feature similarity techniques and proposed a novel framework that leverages normalized mutual information for online event clustering. To achieve a similar goal of event detection, the authors of [6] distinguished events from non-events using aggregated statistics of topically similar clusters obtained using Term Frequency-Inverse Document Frequency (tf-idf) weighting measures. EDCoW [17] combines wavelet analysis with clustering techniques to build signals for event detection. The signals are then filtered using time-series autocorrelation measures and a modularity-based graph partition technique is used to detect events. While the work in [5,6] was focused on general event detection, the authors of [9,16] studied specific events such as protest events, earthquakes, and other disasters. In [16], a tweet-based classifier with semantic analysis was used to detect earthquakes in real-time by modeling tweets as sensor information. The classifier utilized keywords in tweets as features and inferred event location using Bayesian Kalman filters. A non-parametric heterogeneous graph scan statistics was proposed by [9] to detect protest events. The authors modelled tweets, retweets, and hashtags as a graph that senses anomalous neighbourhood clusters to detect events. A notable similarity of most of the techniques in this domain is the use of textual features for event detection.

2. Event detection using bursty terms: In this line of research, a burst is defined as a sudden rise in the frequency, size, volume etc. of some keywords or data points. The intuition here is that a sudden rise in the frequency of keywords or data points can be attributed to an important event taking place.

Initial efforts by the authors of [13] was an infinite-state automaton approach to model data stream in which bursts appear as state transitions. Once a burst is detected, a nested representation of the burst is evaluated using the hierarchical structure of the overall stream. In [12], the authors presented an alternative perspective of bursts in real time as a time varying Markov modulated Poisson process. Another real-time online event topic detection framework is TopicSketch [18]. The authors proposed a bursty topic detection framework using the velocity and acceleration of words by drawing inspiration from earlier work in [10], along with hashing dimension reduction techniques to achieve scalability.

All the aforementioned works leverage textual information together with some clustering or semantic similarity measures to detect events. Table 1 shows the uniqueness of our work in this paper in comparison to closely related work. We differentiate SensorTree from these works in the sense that, SensorTree uses the growth of online communities (users connected by follower relationships) for event detection rather than predefined keywords.

Table 1. A comparison of SensorTree to existing works in social media event detection

Event Detection using Social Media		
	Keywords/Text (Tweets)	Social Context
Clustering/Document Similarity	[5, 6, 9, 16, 17]	[2, 9]
Burst Detection	[1, 10, 13, 15, 18]	SensorTree (Our approach)

3 Problem Definition

Twitter provides a functionality for users to follow other users. This enables users to receive information from those they follow on their timeline. Generally, information propagates on Twitter in the following manner: a Twitter user Alice posts a tweet on a protest event on a given day, in the form of retweet, @mentions, normal tweet etc.; Bob, a follower of Alice also posts a tweet on the same protest event after seeing the original post by Alice. As this process continues, information propagates through an online community of Twitter users. The main objective of this paper is to utilize the network structure and social links of information propagation among users in online communities for protest event detection. The intuition is that a sudden increase in activity (burst) in a community is a result of the emergence of an event of interest, hence we capture burst to detect events. Our main problem can thus be formulated as:

Problem Definition: *Given some information propagating in the form of tweets through a community of online Twitter users, our goal is to detect the periods of sudden changes (bursts) as well as the event that triggers the burst.*

To address this problem, we break it into two sub-problems as discussed below.

Sub-Problem Formulation

Propagation trees [4], have been used to effectively capture information propagating among online user communities; a feature that we need for solving the above defined research problem. Figure 2 shows a tree under construction at time t_1 using tweets and the Twitter follower network [4]. The nodes **A**, **B**, and **C** represent users in a community who have already posted tweets on some protest event of interest. A new user **D**, who is a follower of **B** posts a message on the same protest at time t_2 , we add node **D** and a directed edge from **B** to **D** to the tree. The tree grows as this process continues.

To be able to detect bursts, we need to capture the growth of these trees in continuous time. For example, Fig. 3 shows the timeline of two propagation trees. The green tree shows a sudden growth (burst) between time t_3 and t_4 while the red tree shows a gradual growth in its timeline. Studying these growth dynamics will help us to effectively capture burst for event detection. Burstiness in propagation trees may be a result of breaking news information or the emergence of compelling events which attract a lot of attention and rouse people to

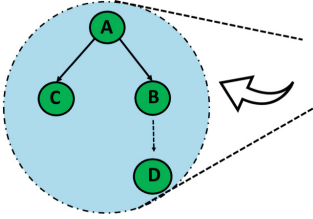


Fig. 2. Propagation tree

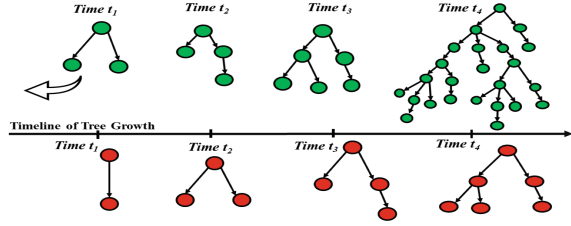


Fig. 3. The growth of two propagation trees over time

tweet about them. Therefore by making use of the propagation tree intuition, we can break our research problem into the following two sub-problems:

Sub-Problem 1: Burst Detection in Online Communities

Given tweets and the Twitter follower network, our goal is to build propagation trees and capture the period of burst W_b in continuous time as the tree grows.

This task requires modeling the increase in growth of online communities, storing of the tree size tree in continuous time and developing a method to effectively capture the period of burst.

Once we have been able to detect a burst, our second research challenge is to infer the protest event that triggers the sudden changes in the growth of the community. This leads to our next sub-problem.

Sub-Problem 2: Event Inference from Tree Bursts

Given a W_b in a tree, the goal is to infer the event which triggered the burst.

This can be formulated as a latent topic inference problem based on the tweet corpus in W_b . Existing formulations of propagation trees [4, 8] can only capture structural and temporal features of information propagation which is only useful for sub -problem 1. To address sub-problem 2, our major challenge is to redefine propagation trees to preserve both the structural-temporal links as well as the semantic context of information propagation for event detection. This formulation will help us not only detect bursts but also describe the associated latent topics in a burst period.

4 SensorTree Framework

To solve the research problem in this work, we develop SensorTree, which tracks information propagating in an online Twitter community in continuous time to capture the burst for event detection. Figure 4 gives an overview of the SensorTree framework, which contains: (1) The tree construction phase to build the propagation trees, (2) The Tree Data Gridding (TDG) phase to store the growth dynamics of a tree in sliding time windows, (3) The Burst Sensing phase to track changes in the acceleration in the TDG to identify bursty windows for event detection, (4) The Latent Event Topic Modeling phase which infers latent

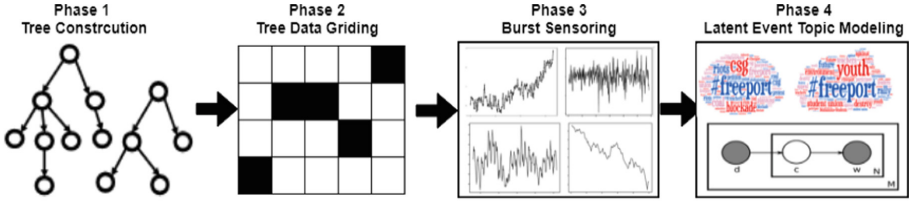


Fig. 4. Workflow of SensorTree framework

topics in a bursty time window and reports the detected event. The four phases are described in detail below.

Phase 1. Tree Construction: The first step of SensorTree is to capture information propagation among users in an online community. Propagation trees have been proven to be useful [4,8] to effectively capture how communities of online users connected via social links (follower relationship) discuss topics of interest. These trees are constructed using tweets and the Twitter follower network as described below [4].

Let the directed graph $\mathbf{G} = \langle \mathbf{V}, \mathbf{E} \rangle$ represent the Twitter follower network, where $\mathbf{V} = \{X_1, X_2, \dots, X_N\}$ is the set of N Twitter users, and $\mathbf{E} = \{X_i \rightarrow X_j | X_i, X_j \in \mathbf{V}, i \neq j\}$ is the set of directed edges representing that user X_j is a follower of user X_i on Twitter. Information therefore propagates from X_i to X_j if an edge $X_i \rightarrow X_j$ exists in \mathbf{G} . Let \mathcal{C} be the tweet corpus posted on a given day, and $p = (X, c, \tau)$ the tweet posted by user $X \in \mathbf{V}$ at time τ with content $c \in \mathcal{C}$. A time indexed tweet series is denoted by $\mathcal{P} = \langle p_1, p_2, \dots, p_K \rangle$ s.t. $p_i.\tau \leq p_j.\tau$ if $i \leq j$, where $i, j \in \{1, 2, \dots, K\}$ and K is the number of tweets posted on the current day.

Definition 1 (Propagation Tree (PT) [4]). Given a Twitter follower network \mathbf{G} and time indexed tweet series \mathcal{P} for the current day, let τ_i be the time of the first post of X_i in \mathcal{P} and τ_j be the time of the first post of X_j in \mathcal{P} , a Propagation Tree $\mathbf{PT} = \langle \mathbf{V}', \mathbf{E}' \rangle$ where $\mathbf{V}' = \{(X_i, \tau_i) | X_i \in \mathbf{V}\}$ and $\mathbf{E}' = \{(X_i, \tau_i) \rightarrow (X_j, \tau_j) | X_i \rightarrow X_j \in \mathbf{E}, \tau_i \leq \tau_j\}$ is a set of directed edges.

The nodes of a *PT* represent users and the timestamps of their posts in the community and the edges represent the follower links between these users.

Recall from Sub-problem 2, that we are also interested in inferring the event that triggers the burst, which means that we need the corpus from a burst period to describe the event. However, the above existing propagation tree representation does not retain the corpora from ongoing discussion in online communities. Also, the *PT* framework captures just the first post of a user for the tree construction. Thus we introduce Semantic Propagation Tree, which is able to capture the growth features of a tree, as well as all the posts generated by the users in an online community.

Definition 2 (Semantic Propagation Tree (SPT)). Given a Twitter follower network \mathbf{G} and time indexed tweet series \mathcal{P} for the current day, a

Semantic Propagation Tree $SPT = \langle \mathcal{X}, \mathcal{E} \rangle$, where the node-set is defined as $\mathcal{X} = \{(X_i, [(c_i, \tau_i)]) \mid X_i \in \mathbf{V}, [(c_i, \tau_i)] = [(c_{i_1}, \tau_{i_1}), (c_{i_2}, \tau_{i_2}) \cdots (c_{i_Q}, \tau_{i_Q})]\}$ to represent users, the contents and the timestamps of their posts, and the edge set is $\mathcal{E} = \{(X_i, [(c_i, \tau_i)]) \rightarrow (X_j, [(c_j, \tau_j)]) \mid X_i \rightarrow X_j \in \mathbf{E}, \tau_{i_1} \leq \tau_{j_1}\}$.

The semantic propagation tree stores the propagation information of the tweet corpora. The length of $[(c_i, \tau_i)]$ i.e. i_Q is the number of tweets posted by a user X_i . Such additions to the existing propagation tree representation provide the capability for inferring latent events that trigger bursts in the growth of communities. We use the same criteria as [4] to construct semantic propagation trees. To build a tree, the first Twitter user who posts a tweet on a protest event on a given day is selected as the source node.

Following **Criterion 1** (tree growth) from [4], assuming we $(X_m, [(c_m, \tau_m)])$ represents a new node at $\tau_m \forall (X_i, [(c_i, \tau_i)]) \in SPT, \tau_m > \tau_i$. If $(X_i \rightarrow X_m) \in \mathbf{G}$, the follower network, we grow the tree by adding the node $(X_m, [(c_m, \tau_m)])$ and a directed edge from $(X_i, [(c_i, \tau_i)]) \rightarrow (X_m, [(c_m, \tau_m)])$. As an extension to **Criterion 1**, if a user who is already in the current tree under construction posts a message, we add his or her new post to the tree by extending the sequence $[(c_i, \tau_i)]$. By this, we are able to capture all the user’s contribution to an ongoing discussion in an online community.

As information propagates in a community of online users in the form of a tree, there is a likelihood that users who belong to different communities will also share some information. This brings us to the use of **Criterion 2** (Emergence of new Propagation Trees) [4]. Given a semantic propagation tree SPT under construction with node set \mathcal{X} , let the follower list of X be $\mathcal{F}(X)$. If a user X has posted a tweet and X is not a follower of any of the users included in any of the existing SPT s, a new SPT is created with node $(X_i, [(c_i, \tau_i)])$ as the root. An SPT is terminated using (**Criterion 3 (Tree Termination)**) [4]. Assuming $(X_i, [(c_i, \tau_i)])$ is the last node added to the current SPT in a given day, the SPT is terminated if none of the followers of X_i posts a tweet after X_i ’s post.

Phase 2. Tree Data Gridding (TDG): Recall from Sub-problem 1 that being able to capture the changes in community growth in continuous time is non-trivial in order to detect bursts. We design a Tree Data Grid (TDG) to keep track of the growth of trees. With the TDG, we segment the timeline of tree growth into discrete time windows $W_{T_1}, W_{T_2}, \dots, W_{T_k}$. We use \mathcal{T} to represent the window size. \mathcal{T} is varied using fixed time windows (e.g. $\{\mathcal{T} = 15, 30, 45\}$) to measure different granularities for capturing tree growth. The TDG is synonymous to an $\mathbf{m} \times \mathbf{n}$ matrix $Z_{[\mathbf{m}, \mathbf{n}]}$, where each row is a tree and each column represents a time window. For example, a matrix element $Z_{[A, W_{T_i}]}$ in the TDG will return the value measured from tree A in time window W_{T_i} .

Phase 3. Burst Sensoring: The term burst can be viewed in different ways. We adopt a definition of burst involving kinetics and Newtonian motion [10, 18]. From a physics perspective, we define a burst as a notable change in the velocity of the tree growth. This rate of change has a natural interpretation as a kind of ‘acceleration’ or ‘force’, leading us to an intuitive physical model of bursts here as dramatic accelerations of the tree growth.

Using a basic construct in physics, a tree has an associated quantity $K(t)$ at time t . An example could be the number of tweets or the number of users in a tree. In this work, we use the size of a tree $|\mathcal{X}|$ as a basic quantity to model burst in online communities because of its ability to produce strong signals for event detection as shown earlier in Fig. 1. Generally, the velocity is obtained by finding the first derivative $\frac{dK(t)}{dt} = \frac{d|\mathcal{X}|(t)}{dt}$. We define $|\mathcal{X}|(t)$ at discrete points, thus we estimate $v(t) = \frac{\Delta|\mathcal{X}|(t)}{\Delta t}$. After obtaining the velocity, we can calculate the acceleration by finding the first derivative of $v(t)$ or the second derivative of $|\mathcal{X}|$ with respect to t . This is mathematically expressed as:

$$a(t) = \frac{d^2|\mathcal{X}|(t)}{dt^2} = \frac{v(t) - v(t-\mathcal{T})}{\mathcal{T}} \quad (1)$$

Once $a(t)$ is obtained, we can measure bursts in terms of positive accelerations beyond a parametric threshold θ . The threshold θ operates with the z-score $z_{a(t)} = \frac{a(t) - \mu}{\sigma}$, where μ is the average acceleration and σ is the standard deviation. For every time window W_{T_i} whose acceleration $a(t)$ has a z-score higher than the parametric threshold θ , we consider that window to be a bursty window.

Phase 4. Latent Event Topic Modeling (LETM): Recall from sub-problem 2 that once a burst is detected in a tree, we are interested in inferring the event associated with the burst. This requires that we utilize the conversations in the community during the burst period in order to solve our event inference problem. We extract the corpus over the bursty window and propose to employ latent event topic model using tensor decompositions [3] for event inference. Given the tweet corpus in W_b , a latent event variable $\phi = (\phi_1, \phi_2, \dots, \phi_k)$ represents the proportion of k event topics where each ϕ_i is a distribution over an exchangeable bag of words. We denote $|Voc|$ as the vocabulary size in W_b from which words on ϕ_i are sampled from a generalized Dirichlet distribution [7] with concentration parameter $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_k]$. We propose to use a single topic model over the corpus in W_b which is a special case with $\alpha_0 = 0$. Learning latent topics can be carried out efficiently via tensor-based techniques with low sample and computational complexities which achieve better performance [3, 11]. By using tensor decomposition [3], we can derive the first, second and third order moments, and reduce the model to moment forms to recover ϕ . An event is represented in the **LETM** as topics and its associated word descriptions.

A summary of SensorTree’s framework is presented in Algorithm 1. The algorithm accepts tweets and the Twitter follower network relationships as input, and then carries out the four phases sequentially. **TreeConstruct** constructs semantic propagation trees as described in Phase one. The **TDG** stores the size of each tree based on the time window size \mathcal{T} . For each tree, the **Burst-Sensor** computes $a(t)$ using Eq. (1) to detect burst. Once a burst is detected in a tree, the **LETM** extracts tweets over the burst period. The **LETM** then builds a tensorized topic model to describe the event associated with the burst period. While we focus on protest events in this work, it is worth mentioning that SensorTree can be modified and extended in the context of general event detection. SensorTree outputs detected events as topics and word descriptions.

Algorithm 1. SensorTree

-
- 1: **procedure** : **Input**: time indexed tweet series \mathcal{P} , follower relationship from \mathbf{G}
 - 2: **Output**: treeID, Event-topics, associated word descriptions,
 - 3: **TreeConstruct**: Select $p_1.X \in \mathcal{P}$ as the source node
 - 4: for every other $p \in \mathcal{P}$
 - 5: \rightarrow Use **Criterion 1 and 2** to construct trees following tree algorithm in [4]
 - 6: \rightarrow Terminate tree using **Criterion 3**
 - 7: **TDG**: Store $|\mathcal{X}|$ for each W_T vary \mathcal{T} to observe optimum threshold
 - 8: **BurstSensor**: Compute $a(t)$ using Eq. (1)
 - 9: \rightarrow Estimate $z_{a(t)} = \frac{a(t)-\mu}{\sigma} > \theta$ to detect W_b
 - 10: **LETM**: Extract Corpus in W_b for topic modelling using tensor decomposition [3]
 - 11: \rightarrow Construct and estimate empirical 2nd and 3rd order moments
 - 12: \rightarrow Whiten the data via SVD and extract the eigenvectors
 - 13: \rightarrow Use stochastic gradient descent to estimate the spectrum of whitened tensor
 - 14: \rightarrow Post-processing: Use power iteration in pairs to calculate topic-word matrix
-

5 Experiments and Evaluation

We present a detailed description of how the experiments were conducted and the results obtained in comparison with existing state-of-the-art models.

5.1 Datasets and Settings

We use two different Twitter datasets for our experiments as shown in Table 2.

(I). Freeport Dataset: The Freeport dataset contains tweets published from Jan 2017–April 2017 on the coal protests in Indonesia. During this period, there were series of protest actions towards Indonesian mining giants with protesters calling for the mines to be shut down.

(II). Newcastle Harbour Blockade Dataset (NHB): The NHB-dataset contains tweets published in Australia from Feb.-May 2016 on conversations regarding the Newcastle harbour blockade and its related protest activities.

For both datasets, we collect the follower lists of all users who posted tweets during the periods of observation using the Twitter API. We use different time window settings ($\{\mathcal{T} = 5, 10, 15, \dots, 120\}$) to empirically observe different time granularities of detecting bursts using Equation (1). We implemented and conducted all experiments using Python 3.2 and 3.6 on a Windows machine with 8 GB Memory and a 64-bit Linux virtual machine with 6 GB memory all running on an Intel(R) Core(TM) i5-4310m CPU 2.70 Ghz processor.

Table 2. Datasets description

Data	Tweets	Size of network community (# of Users)
Freeport Dataset	200,800	12,453,643
Newcastle Harbour Blockade	4,900,305	52,424,200

5.2 Comparison Methods

We compare SensorTree to **LDA** [7] **tensorLDA**, **EvenTweet** [1], and **KW-Freq**. **EvenTweet** is a state-of-the-art event detection framework that extracts bursty keywords from tweets for event detection. We follow strictly the authors’ implementation based on the published work [1]. For **LDA** and **tensorLDA**, we build the model over the entire corpus on daily basis and extract the event topics. We vary the number of topics (from $k = 1$ to $k = 5$) to achieve the most meaningful results for both models. We present the results of **tensorLDA** as it achieved better performance as compared to LDA in our experiments. As a baseline model, we also build **KW-Freq** as a naive event detection model that relies on the daily counts of keyword frequency in the entire datasets.

5.3 Evaluation Metrics

For the Freeport dataset we have ground truth data which we refer to as Gold Standard Record (GSR). The GSR contains coded protest event information extracted from major news sources, blogs and articles on real-world protest events compiled by news analysts. The event information includes the protest event date, headline description and news source with sample entries shown in Table 3. The headline description of the GSR contains a summary of protest events both in English and Indonesian.

Table 3. GSR protest event entries

Date	Event Id	News source	Event headline description
2017-02-05	9856112330	Kompas	<i>Freeport to reduce Indonesian Mining Activities. Tuntut Izin Ekspor, Kayawan Freeport Gelar Aksi di Kantor Bupati</i>
2017-02-24	996533162	Republika	<i>GMNI encourages government to take over freeport</i> <i>GMNI Dukung Pemerintah Ambil Alih</i>

We remove stopwords and manually select keywords token from the GSR event headline description. The goal is to use the keywords token to measure the performance of the various methods. Thus, for every GSR description, we expect the models to detect some keywords in its top words that match the GSR tokens. The following metrics are used in evaluating the results obtained from the Freeport dataset.

1. Topic Intrusion Score (I_T): This score measures the quality of event description generated by a model. Denoting \hat{c}_a and \hat{c}_b as the word vectors for the GSR word tokens and tokens generated an event detection method respectively. The topic intrusion score is given by: $I_T(\hat{c}_a, \hat{c}_b) = 1 - \frac{\hat{c}_a \cdot \hat{c}_b}{|\hat{c}_a|^2 + |\hat{c}_b|^2 - \hat{c}_a \cdot \hat{c}_b}$. As events are represented by words belonging to a topic, a higher I_T score implies that a model’s output has more intruding words which differ from the GSR tokens and unrelated to the event detected.

2. Topic Coherence (C_T): Cosine similarity is one of the most popular similarity measures applied to text documents. We use the cosine similarity to measure the coherence of the word tokens to the GSR word tokens. A high C_T signifies that a model is able to concisely describe events detected at a fine granularity. The topic coherence C_T is given by: $C_T(\hat{c}_a, \hat{c}_b) = \frac{\hat{c}_a \cdot \hat{c}_b}{|\hat{c}_a| \times |\hat{c}_b|}$.

3. Precision: We measure precision as: $Precision = \frac{\#EventMatches}{Total\#GSREvents}$. A model’s output is classified as a match if more than 60% of its top 20 (highest topic probabilities) word descriptions matches the word tokens of a GSR event. In our experiments, we select the 20 most representative keywords from each topic extracted daily and compare them with the GSR event descriptions.

5.4 Results and Discussion

5.4.1 Burst in Trees for Event Detection

After we build the propagation trees, we study the effect of using different time window settings in the detection of burst in propagation trees. We observe that bursts are generally very sparse and the number of detected bursts increases with increasing \mathcal{T} . We filter out trees with $|\mathcal{X}| \leq 10$ since their average accelerations over the minimum \mathcal{T} is < 1 . Using different time granularities, we show the timeline of the 10 trees with the largest size in Fig. 5 on the Freeport dataset. It can be observed that burstiness is very sparse across the timeline. We achieved similar results with varying \mathcal{T} on both datasets as shown in Fig. 6. We also

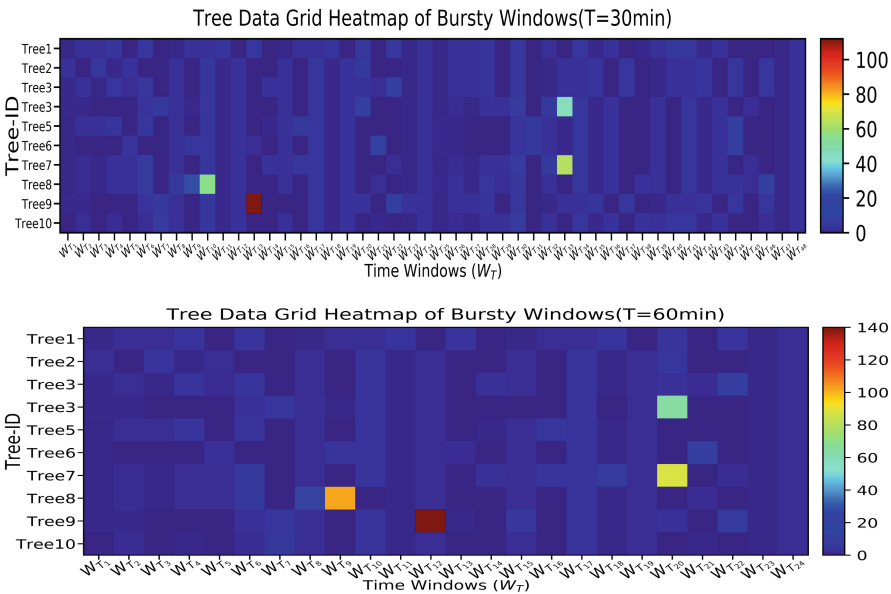


Fig. 5. Tree timeline showing growth dynamics for different time window size

observed in Fig. 6 that a reasonable window size helps in effectively capturing all the burst.

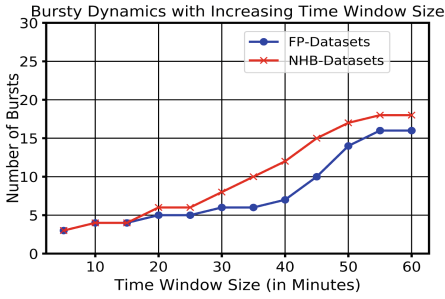


Fig. 6. Bursts dynamics on different datasets.

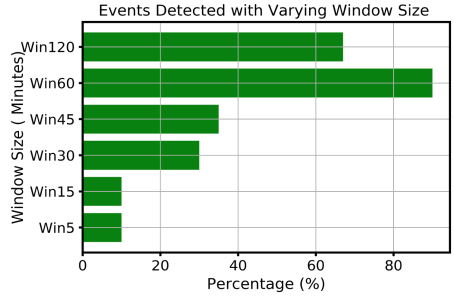


Fig. 7. Percentage of event detected using different window sizes.

We are also interested in evaluating the prowess of SensorTree in detecting real world protest events. We evaluate the percentage of real protest events detected by SensorTree using different time granularities in Fig. 7. The results show that SensorTree is able to be detect 96% of ground truth protest events with $\mathcal{T} = 60$ minutes. Hence, we set $\mathcal{T} = 60$ in the rest of our experiments.

5.4.2 Event Topic Evaluation

To compare the results of SensorTree with competing models, we show the performance of the various models on the Freeport dataset using the evaluation metrics (I_T , C_T and Precision) in Fig. 8. SensorTree outperforms comparison models in all the evaluation metrics.

On topic Intrusion, recall that a higher I_T means that the model has more intruding words and is unable to concisely describe events. It is not surprising that the **KW-Freq** recorded the highest value for I_T . This observation can be attributed to the fact that many events are discussed daily on social media and

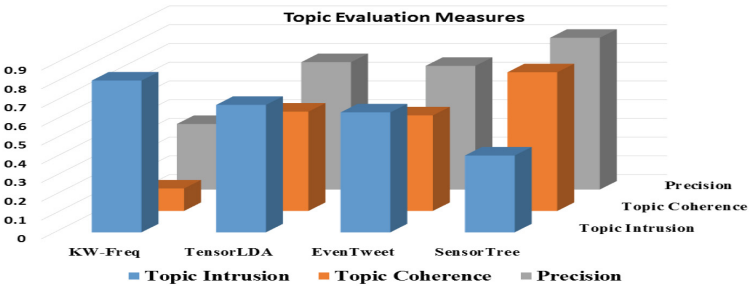


Fig. 8. 3D Bar Plots of Model Performance on Freeport dataset

the highest 20 keywords in a given day may be a distribution of words from different unrelated events. **TensorLDA** and **EventTweet** achieved similar results on topic intrusion. SensorTree records the least I_T value showing that conversations are more focused among users in a community during a protest event. Thus the word distributions have less intrusion from topics which are unrelated. A high Coherence score C_T is an indication of a concise event description. SensorTree achieves the C_T value. We also achieve the highest precision in terms of matches with GSR token, followed by the state-of-the-art **EventTweet**, and then **TensorLDA**, with **KW-Freq** achieving the lowest results.

5.4.3 Fine Granularity and Language Independent Event Detection

SensorTree has further capabilities to detect at a fine granularity without language restrictions. SensorTree relies on the network community structure which is blind to the use of a specific language. Thus we are able to capture ongoing conversations between users in non-English speaking online communities. Results on the Freeport dataset in Table 4 shows that SensorTree is able to detect events irrespective of the language at a finer granularity than comparison models. We highlight the words that match ground truth event descriptions in blue.

5.4.4 Case Studies

We performed a number of case studies on both datasets. Due to space limitations, we discuss some notable events that were detected by SensorTree which

Table 4. Comparing protest event descriptions from various models with news reports

Date	Event headline description	SensorTree	EventTweet	TLDA
16/02/17	Demanding export mine permit freeport employees stage demonstration at Regent’s office. Tuntut Izin Ekspor Karyawan Gelar Aski di Kantor Bupati	tuntut, freeport, halt, export, gelar, kantor, mine, union, Izin, destruction	freeport, googlebox, iphone, scandal, halt, mine, music lovers	freeport, Indonesia, topcharts, ranking, now playing
24/02/17	Demo di Freeport dan ESDM GMNI Minta Demonstration at Freeport and the Ministry of Energy. National Students Movement urges government to be firm	freeport, students energy, demonstration, government, destruction, Indonesia	movement, energy freeport, royalband concert, showdown trending, tickets	student, concert, freeport, energy, crowd, music, tickets, fighters
07/03/17	Berpotensi Berekor dengan Karyawan Demo anti airport bubar. Clashes with Freeport employees and Anti Freeport Protests	employees, freeport, Karyawan, berpotensi, mine, bubardengan, clash	employees, freeport movie, trailer, mine, reviews, clash, sold-out	employees, mine, freeport, trailer, sold-out, album, clash
13/03/17	PRD Unjuk Rasa di Depan Kementerian Dozens of members of the Peoples Democratic Party (PRD) held a freeport demonstration in front of the Ministry of Energy	PRD,demonstration Ministry, freeport, front,energy, destruction	demonstration, export, union, NBA, playlist, stocks, hit-maker	freeport, energy export, union, Kong, playlist, NBA, hipop

the other comparison models were not able to detect. The news¹ coverages of these events are shown in Fig. 9.

(i) **NHB Dataset:** We detect a series of events using this dataset. A notable one was the Santos Pilliga coal protest. There were hundreds of protesters on Feb. 21st, 2016 at Santos coal seam gas waste water plant in the Pilliga Forest. The protest event followed earlier protest attempts where 29 people were charged with various trespassing offences. SensorTree detected this protest with event description words such as *Pilligia, coal, csg, protest, Narrabi, arrest, police, etc.*

(ii) **Freeport Dataset:** SensorTree was able to detect events which were not actual protest events, but sub-events that led to other protest events. On the 19th of Feb. 2017, the Youth of Muhammadiyah placed a wake up call on the Indonesian government not to lose on ongoing Freeport lawsuits. This was not an actual protest event, but was a precursor to a series of protest events some days later. SensorTree accurately detected this sub-event. This added capability of SensorTree in detecting precursors is useful for forecasting future protest events.



Fig. 9. Events detected from Case Studies

6 Conclusion

In this paper, we have presented SensorTree, an event detection framework for protest event detection. SensorTree models information propagation within a community of Twitter users as a sensor to detect a period of burst as events. Once burst is detected, SensorTree infers the details (topics) of the event that triggered the burst using an event topic model. We have performed a set of experiments with real world datasets and compared SensorTree to competing event detection models using various evaluation metrics. The results show that SensorTree outperforms the comparison models. The case studies presented show the capabilities of SensorTree to accurately detect protest events with fine granularity and no language restrictions.

Acknowledgements. We acknowledge Data to Decisions CRC (D2DCRC), Cooperative Research Centres Programme, and the University of South Australia for funding this research. The work has also been partially supported by ARC Discovery project DP170101306.

¹ <https://goo.gl/LRdwd6> (left image) and <https://goo.gl/c5LCZ2> (right image).

References

1. Abdelhaq, H., Sengstock, C., Gertz, M.: EvenTweet: online localized event detection from twitter. *PVLDB* **6**(12), 1326–1329 (2013)
2. Aggarwal, C.C., Subbian, K.: Event detection in social streams. In: *Proceedings of the 2012 SDM*, pp. 624–635. SIAM (2012)
3. Anandkumar, A., Ge, R., Hsu, D., Kakade, S.M., Telgarsky, M.: Tensor decompositions for learning latent variable models. *J. Mach. Learn. Res.* **15**(1), 2773–2832 (2014)
4. Ansah, J., Kang, W., Liu, L., Liu, J., Li, J.: Information propagation trees for protest event prediction. In: Phung, D., Tseng, V.S., Webb, G.I., Ho, B., Ganji, M., Rashidi, L. (eds.) *PAKDD 2018. LNCS (LNAI)*, vol. 10939, pp. 777–789. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93040-4_61
5. Becker, H., Naaman, M., Gravano, L.: Learning similarity metrics for event identification in social media. In: *3rd ACM WSDM*, pp. 291–300. ACM (2010)
6. Becker, H., Naaman, M., Gravano, L.: Beyond trending topics: real-world event identification on twitter. *ICWSM* **11**(2011), 438–441 (2011)
7. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. *J. Mach. Learn. Res.* **3**(Jan), 993–1022 (2003)
8. Cadena, J., Korkmaz, G., Kuhlman, C.J., Marathe, A., Ramakrishnan, N., Vulikanti, A.: Forecasting social unrest using activity cascades. *PloS one* **10**(6), e0128879 (2015)
9. Chen, F., Neill, D.B.: Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs. In: *Proceedings of the 20th ACM SIGKDD*, pp. 1166–1175. ACM (2014)
10. He, D., Parker, D.S.: Topic dynamics: an alternative model of bursts in streams of topics. In: *Proceedings of 16th ACM SIGKDD*, pp. 443–452. ACM (2010)
11. Huang, F., Niranjana, U., Hakeem, M.U., Anandkumar, A.: Online tensor methods for learning latent variable models. *J. Mach. Learn. Res.* **16**(1), 2797–2835 (2015)
12. Ihler, A., Hutchins, J., Smyth, P.: Adaptive event detection with time-varying poisson processes. In: *Proceedings of 12th ACM SIGKDD*, pp. 207–216 (2006)
13. Kleinberg, J.: Bursty and hierarchical structure in streams. *Data Min. Knowl. Discov.* **7**(4), 373–397 (2003)
14. Kontostathis, A., Galitsky, L.M., Pottenger, W.M., Roy, S., Phelps, D.J.: A survey of emerging trend detection in textual data mining. In: Berry, M.W. (ed.) *Survey of Text Mining*, pp. 185–224. Springer, New York (2004). https://doi.org/10.1007/978-1-4757-4305-0_9
15. Li, J., Wen, J., Tai, Z., Zhang, R., Yu, W.: Bursty event detection from microblog: a distributed and incremental approach. *Concurr. Comput. Pract. Exp.* **28**(11), 3115–3130 (2016)
16. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: *19th WWW*, pp. 851–860. ACM (2010)
17. Weng, J., Lee, B.S.: Event detection in twitter. *ICWSM* **11**, 401–408 (2011)
18. Xie, W., Zhu, F., Jiang, J., Lim, E.P., Wang, K.: TopicSketch: real-time bursty topic detection from twitter. *IEEE TKDE* **28**(8), 2216–2229 (2016)