Detection and Delineation of Events and Sub-Events in Social Networks

Antonia Saravanou [#1], Ioannis Katakis [+2], George Valkanas [*3], Dimitrios Gunopulos [#4]

[#] *National and Kapodistrian University of Athens,* [+] *University of Nicosia,* [*] *Detectica*

[1] antoniasar@di.uoa.gr, [2] katakis.i@unic.ac.cy, [3] george@detectica.com, [4] dg@di.uoa.gr

***Abstract*—We define and solve the problem of *event detection and delineation* as a task of identifying events and decomposing them to their major sub-events, with a description and a timeline. We propose DeLi, an algorithm that focuses on providing such an understanding of events and sub-events. DeLi, to the best of our knowledge, is the first method that addresses the problem in a generic stream of text, and in an online fashion. Extensive evaluation on social streaming data demonstrates that, by combining the structure of a social network with content attributes, our method outperforms the state-of-the-art techniques.**

***Index Terms*—Sub-Event Detection, Event Detection, Anomaly Detection, Social Network Analysis, Graph Mining, Text Mining**

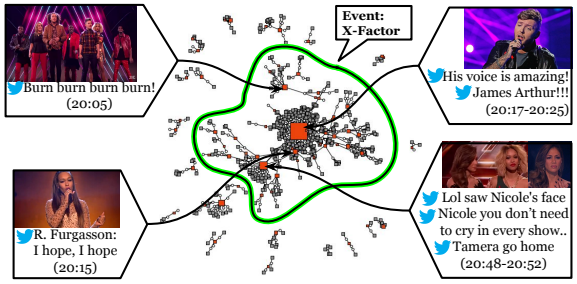

Fig. 1. An example of the graph DeLi builds at the time window of an event, the X-Factor show. The very large connected component, highlighted in green, contains the event detected by DeLi, and its sub-events (noted by orange squares) that are described by their summary and duration in a timeline.

I. Introduction

Event detection techniques in social networks have been proposed as a way to sense and understand what is happening in the offline and online world. Despite being a challenging task, event detection has been utilized in applications that target critical aspects of peoples' life, like health [1], [2] and natural hazards [3], [4], [5], [6].

A fundamental problem is that event detection techniques capture and treat events as singletons, as if they are all disconnected and irrelevant to each other. Clearly, this is not the case in real life, where we abstract a series of highlights or turning points – a *sub-event* – into groups, which collectively constitute the *event*. This grouping makes it easier for us to remember, analyze, understand and narrate what happened. For instance, a football match is better described as a sequence of highlights, e.g., goals, penalties, red and yellow cards, that occur during the 90 minutes of the game. Another example is a music festival that lasts for several days and during which a large number of artists appear on stage. Finally, city protection units may utilize sub-event detection to track individual failures during a natural disaster. In all of these examples, sub-event delineation offers a better understanding through a more detailed breakdown of the main event.

Related Work Given the potential benefits, numerous techniques have been proposed, that fall under two main categories: (a) content-based methods [7], [8], [4], [9], [10], that detect new topics in a collection of text documents, and (b) structure-based methods [11], [12], [13], [14], [15], that search the social network for either significant structural changes over time or for highly active sub-graphs. Techniques from these categories have shortcomings and, therefore, are unable to detect specific types of events. For instance, content-based techniques perform poorly in cases of events that generate high network activity. On the other hand, structure-based methods ignore content altogether and lose useful information, such as opinions. Recently, we introduced a novel technique for event detection in social networks [16], that combines both the textual and the structural information. We showed that this *hybrid* method achieves better results than the existing ones.

Surprisingly, event delineation has not yet been adequately discussed in the literature, despite how natural this decomposition seems in retrospect. Most importantly, algorithms that claim sub-event detection qualities [17], [18], [19], [20], [21], [22], [23], [24] assume they are given as input high quality information, associated to the main event (and only that) and they focus on identifying its main elements. It is easy to see that this assumption breaks the moment we deploy these techniques in real world conditions.

Motivated by these shortcomings, in this paper, we consider the problem of *event delineation*. We present DeLi (Detection and deLineation), an efficient algorithm for event detection in social media with automated timeline construction. DeLi addresses the event delineation problem at its core: it identifies the main event and simultaneously captures highlights (sub-events) that describe it over time, without any prior knowledge of the main event. The technique is unsupervised and is able to distinguish between distinct main events that take place concurrently. It achieves that by modeling the social network as a dynamic, heterogeneous graph with both *user* nodes (with links representing interactions) and *content* nodes (with links representing similarity). Such a representation contains

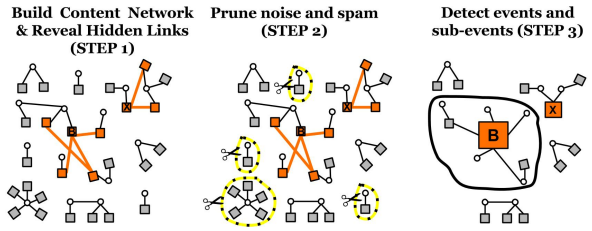

Fig. 2. An overview of the main steps of the DeLi approach.

information about the structure and the content of the network, thereby bringing together the two method categories. Within this graph, (sub-)events are large connected components that DeLi uses in order to provide effective event detection and delineation (see an example in Fig. 1).

II. Problem Definition

Definition I: A *Snapshot Network* is a dynamic, heterogeneous graph $G_t = (V_t, E_t)$ with V_t and E_t being the nodes and the edges of the graph at time point t, respectively. The nodes in V_t are from two disjoint groups; the "*user*" nodes $V_{u,t}$ and the "*content*" nodes $V_{c,t}$. At any t it holds that $V_t = V_{u,t} \cup V_{c,t}$ and $V_{u,t} \cap V_{c,t} = \emptyset$. The edges in E_t are from three categories; the "*user-user*" edges $E_{u,u,t}$, the "*user-content*" edges $E_{u,c,t}$ and the "*content-content*" edges $E_{c,c,t}$. At any t it holds that $E_t \subseteq V_t \times V_t$, $E_t = E_{u,u,t} \cup E_{u,c,t} \cup E_{c,c,t}$ and the pairwise intersections of $E_{u,u,t}, E_{u,c,t}$ and $E_{c,c,t}$ are empty.

Definition II: An *Event* Γ is a set of three main parts:

- A representative summary R_Γ, describing what happened,
- A time window of duration $T_\Gamma = (T_\Gamma^{start}, T_\Gamma^{end})$, and
- A set of sub-events $S_\Gamma = \{\gamma_1, \ldots, \gamma_N\}$ that are *significant*, *distinct* parts of the main event.

A *Sub-Event* γ is similarly defined, but it does not contain lower level events. A sub-event corresponds to a topic that emerges from the data stream, being intensively discussed during a short period, and then gradually fades away. Therefore, it consists of *two* parts: a representative summary r_γ and a duration t_γ. This makes an event Γ contain the sequence of its sub-events $\gamma_i = (r_{\gamma_i}, t_{\gamma_i})$. It holds that $T_\Gamma^{start} \leq t_{\gamma_i}^{start} < t_{\gamma_i}^{end} \leq T_\Gamma^{end}$, $\forall \gamma_i \in S_\Gamma$. We note that sub-events are not necessarily sequential, i.e., a sub-event can start while another one is in progress.

Problem Definition [*Event Detection and Delineation*]: *Find all events* Γ *of content network* $\mathcal{G} = \{G_t | t = 1, \ldots, t_{max}\}$, *where* G_t *is the snapshot graph at time* t.

According to this definition, the goal is to identify all discussed events. For each event, we capture: i) a representative summary, ii) the duration (start-end time) and, iii) its sub-events. For each sub-event, we also need description/duration.

III. Event Detection and Delineation

Figure 2 provides a graphic overview of our method. At first, we build the snapshot network G_t (**step 1**). *User* and *content* nodes are added, followed by "*user-user*" edges between users that interact and "*user-content*" edges between user and content nodes associated with an action. This network is quite sparse and there may be conversations about the same topic in disconnected regions. To overcome this issue, we link similar content nodes by computing their pairwise cosine similarity [16]. If a pair exceeds a threshold ϕ, we include the respective "content-content" edge. Experiments for the threshold ϕ are shown in Sec.V. We also speed up the cosine similarity computations via the SimHash algorithm [25].

Much of the content posted in social networks, is either *spam*, e.g., "please, check out my fashion blog", or *repetitions*, such as "happy birthday" wishes. Upon experimentation, we observed that connected components related to such topics include very few user nodes and many more content nodes, typically in a star-like formation. In **step 2**, we filter out all star-like subgraphs to avoid uninformative conversations.

In **step 3**, we check whether we have an actual event in the snapshot network. G_t contains a number of connected components, each of which signifies a discussion over a topic. A *very large* connected component is a good event indicator [16], because it implies an increased interest. Consequently, the problem of event detection maps to the identification of such connected components. Note that this method can detect multiple events in a single time window.

We distinguish event-related conversations from the rest via an anomaly detection approach. In particular, given two thresholds l and L, $l < L$, for every time window, we iterate over all of the connected components of G_t. If the size of a connected component is above L, it is immediately marked as an event. If its size is below l, the component is ignored. When the size is between l and L, the component has the potential to become an event at the next time window and it is considered as an *event candidate*. With this feature, our algorithm is independent of the way the social network's stream is split.

Sizes l and L are automatically computed from the connected components that appear in the streaming content network. Here, we employ a Gaussian-based anomaly detection technique, akin to Grubb's test [26], that is widely used in anomaly and event detection [27], [28]. For a variable X, with average avg(X) and standard deviation std(X), the anomaly function is:

$$h_p(x) = \begin{cases} 1, \text{if } x > p, \text{ where } p = avg(X) + \theta\, std(X) \\ 0, \text{otherwise.} \end{cases}$$

Here, 1 is anomaly, while 0 is the norm. Threshold l, and h_l, is given by considering the sizes of all connected components of G_t. Threshold L, and h_L, is given by the sizes of connected components that are considered anomalous by h_l.

So far, we have addressed event detection, but not event delineation. For this task, we only consider *content* nodes. We take advantage of the graph structure and utilize the content of the most *central* nodes for the representative summary. Network centrality is a widely used metric that identifies central individuals, key hubs or super-spreaders of diseases. We employ the *betweenness centrality* [29] that ranks nodes according to the number of shortest paths they participate in.

Complexity The worst-case complexity of DeLi is $\mathcal{O}(n_c^2 + n_c * m_c)$, where n_c is the number of content nodes in

TABLE I
EVALUATION RESULTS FOR ALL METHODS ON TWITTER DATA

Method	Precision	Recall	F-Score
Act	0.33	0.70	0.45
Struct	0.28	0.87	0.42
K-Cores	0.21	0.43	0.28
DeLi$_{Con}$	0.44	**0.90**	0.59
DeLi$^{\#}_{Con}$	0.48	0.32	0.38
DeLi$_{Ter}$	0.39	0.95	0.55
DeLi	**0.53**	0.78	**0.63**
DeLi$^{\#}$	0.56	0.69	0.62

$\mathcal{G}$. The pairwise similarities yield the first quadratic factor. Betweenness centrality – used for sub-event extraction – runs on content nodes alone, and has a complexity of $\mathcal{O}(n_c * m_c)$, where m_c is the number of edges between content nodes. In practice, SimHash has a better average-case runtime; similarities are only run on event candidates and the connected components for sub-events are sparse.

IV. EVALUATION: GOALS AND SETUP

We evaluated *DeLi* on a real-world Content Network. We collected geotagged tweets from London through the Twitter Streaming API [1], posted between 11/29-12/09/2013. This resulted in 556K content nodes from 69K user nodes. We extracted ground truth events from Wikipedia[2] and enriched this list with others that were discovered manually.

Evaluation Metrics and Techniques Our time span is split into k time windows. As a result, our ground truth and methods are represented by binary vectors $M = \{m_1, \ldots, m_k\}$, where $m_i = 1$ if there is an event at time i, or $m_i = 0$ otherwise. Based on that, we calculate *Precision, Recall* and *F1-Score* to compare the following techniques:

Act: This widely used approach identifies events when an unexpected volume of messages appears. We consider 3-hour segments, as traffic in Twitter differs throughout the day.

Struct: Similar to DeLi, this baseline tracks very large connected components on the network's structure alone.

DeLi: This is our approach described in Section III.

DeLi$_{Con}$: We apply DeLi to a graph of *only* content nodes. This will show the value of including users nodes.

DeLi$_{Ter}$: We use DeLi on a popular text representation in event detection [22], [13]; the *Term Graph*. It contains terms as nodes and edges between terms denote a co-occurrence.

DeLi$^{\#}$ and **DeLi$^{\#}_{Con}$** are alternative versions of DeLi and DeLi$_{Con}$, respectively, that employ SimHash (Sec. III).

K-Cores: Current state-of-the-art sub-event detection technique [22] that utilizes the text's structure. We compare with the best performing k-core metric.

Experimental Setup All methods are implemented in Python 2.7 using NetworkX[3]. K-Cores code was kindly provided by the authors. Experiments were executed on a machine with Intel Core i7 CPU@3.40GHz, 16 GB RAM and 64-bit OS.

Reproducibility Datasets and code are available online[4].

[1]https://www.dev.twitter.com/rest/public
[2]https://en.wikipedia.org/wiki/Portal:Current_events
[3]https://networkx.github.io/
[4]https://www.di.uoa.gr/~antoniasar/deli/

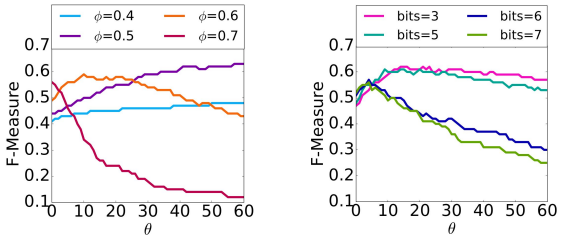

Fig. 3. Parameter Study. *Left:* DeLi, with varying θ and ϕ. *Right:* DeLi$^{\#}$, with varying θ and fingerprint size (#bits).

V. RESULTS AND DISCUSSION

A. Event Detection

The results of our comparative study appear in Table I. First, we observe that DeLi has the best overall performance. This underlines the significance of the heterogeneous network representation, which the other techniques do not consider. Nevertheless, our approach performs well even with alternative graph representations, e.g., plain content network (DeLi$_{Con}$) and term graph (DeLi$_{Ter}$). The K-Cores method identified many noisy instances as events (e.g., "happy birthday" messages). We also performed a t-test to evaluate the statistical significance of these results and confirmed ($p < 0.01$) that: i) DeLi is superior to the best performing baseline Act, and ii) DeLi performs better than its variants. Finally, DeLi$^{\#}$ shows a minor drop in F1-score, but it drastically reduces the runtime (see Section "Scalability Experiments").

Parameter Study Figure 3 shows the effect of system parameters on performance: On the left, DeLi's F1-score for varied values of θ (controls the size of CCs considered as events) and ϕ (cosine similarity threshold). Values close to 0.5 perform better due to the short text in tweets. Naturally, higher values lead to missing events since connected components are not formed. On the right, DeLi$^{\#}$'s F1-score against θ and #bits (the SimHash fingerprint size). A small number of bits seems to suffice. In both cases, large θ values decrease performance, due to more false negatives.

Scalability Experiments DeLi is also shown to be considerably faster than K-Cores in Table III. Figure 4 demonstrates (a) the efficiency of DeLi using SimHash when continuously increasing data at each time window (left), (b) a break down of the execution time required by each of DeLi$^{\#}$'s modules (middle), with similarity links being the most demanding part, and (c) the runtime for a longer time period (up to a year) and total number of tweets (right).

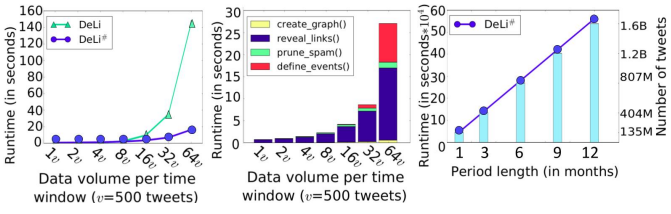

Fig. 4. Scalability Experiments.

TABLE II
SUB-EVENT TYPES WITH THE NUMBER OF APPEARANCES

Sub-Event Type	Ground Truth	DeLi 15-min	K-Cores 1-min	K-Cores 15-min
Total goals	86	**0.70**	0.33	0.05
Penalties	3	**0.33**	0.0	0.0
Own goals	6	**0.83**	0.17	0.0
Yellow cards	85	**0.17**	0.01	0.0
Red cards	1	**1.0**	**1.0**	0.0
Final Score	29	**0.72**	0.31	0.07
All Sub-Events	210	**0.49**	0.19	0.03

TABLE III
PRECISION / RECALL / F-SCORE / RUNTIME FOR SUB-EVENT DETECTION

Method	Precision	Recall	F-Score	Runtime
K-Cores (1-min)	0.14	0.19	0.16	-
K-Cores (15-min)	0.2	0.03	0.05	11091s
DeLi	**0.15**	**0.49**	**0.22**	**139s**

B. Sub-Event Detection

To evaluate sub-event detection, we followed the state-of-the-art line of thinking [22] and constructed a dataset with the sub-events of the 29 Premier League games in the same time period using SkySPORTS[5]: total goals, penalties, own goals, final score, yellow cards and red cards. We run K-Cores with our dataset and time window size set to 15 min and 1 min, as shown in Table III. We had human annotators manually annotate each output, and identify whether there is a sub-event description posted within a few minutes that contains all information or not.

Table II presents the ground truth sub-events, their frequency and the recall attained by each method, per type. DeLi is again better at this task. "Yellow cards" are the least detected type, since users rarely focus on it. Table III presents more performance results for each method on this task. The low precision is mainly due to false positives, i.e., posts that are uninformative or are improperly timed, e.g. "GOAL!". Low interest in a game can impact recall, as discussions lack momentum / popularity. K-Cores also suffers from the fact that it reports a single representative post per time window, despite different events taking place concurrently, which explains its extremely low recall. On the other hand, DeLi outputs summaries only when a sub-event is detected.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we define the problem of event delineation and propose DeLi, a method that detects events and sub-events as they occur by tracking connected components of user and content nodes. DeLi also summarizes those (sub-)events by selecting the most central content nodes. Future directions include learning topical thresholds l and L, evaluating alternatives for describing sub-events (e.g., influential users) and improving the precision for sub-events.

Acknowledgements This research has been financed by the European Union through the FP7 ERC IDEAS 308019 NGHCS project, the Horizon2020 688380 VaVeL project and a Google Faculty Research Award.

[5]https://www.skysports.com/premier-league

REFERENCES

[1] E. van der Goot, H. Tanev, and J. P. Linge, "Combining twitter and media reports on public health events in medisys," in *ACM WWW*, 2013.
[2] E. Aramaki, S. Maskawa, and M. Morita, "Twitter catches the flu: Detecting influenza epidemics using twitter," in *EMNLP*, 2011.
[3] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: Real-time event detection by social sensors," in *ACM WWW*, 2010.
[4] M. Walther and M. Kaisser, "Geo-spatial event detection in the twitter stream," in *ECIR*, 2013.
[5] A. Saravanou, G. Valkanas, D. Gunopulos, and G. L. Andrienko, "Twitter floods when it rains: A case study of the UK floods in early 2014," in *WWW*, 2015.
[6] A. Le, Y.-R. Lin, and K. Pelechrinis, "Information network mining: A case for emergency scenarios," in *ACM SIGKDD LESI*, 2014.
[7] H. Abdelhaq, C. Sengstock, and M. Gertz, "Eventweet: Online localized event detection from twitter," in *VLDB*, 2013.
[8] O. Ozdikis, P. Senkul, and H. Oguztuzun, "Semantic expansion of tweet contents for enhanced event detection in twitter," in *ASONAM*, 2012.
[9] G. Valkanas and D. Gunopulos, "How the live web feels about events," in *ACM CIKM*, 2013.
[10] ——, "Event detection from social media data," in *IEEE Data Eng. Bull.*, 2013.
[11] B. Perozzi and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods," in *CoRR*, 2016.
[12] P. Rozenshtein, A. Anagnostopoulos, A. Gionis, and N. Tatti, "Event detection in activity networks," in *ACM SIGKDD*, 2014.
[13] S. Rayana and L. Akoglu, "Less is more: Building selective anomaly ensembles with application to event detection in temporal graphs," in *SIAM SDM*, 2015.
[14] E. Desmier, M. Plantevit, C. Robardet, and J.-F. Boulicaut, "Trend mining in dynamic attributed graphs," in *ECML PKDD*, 2013.
[15] F. Chen and D. B. Neill, "Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs," in *ACM SIGKDD*, 2014.
[16] A. Saravanou, I. Katakis, G. Valkanas, V. Kalogeraki, and D. Gunopulos, "Revealing the hidden links in content networks: An application to event discovery," in *CIKM*, 2017.
[17] Q. Wu, S. Ma, and Y. Liu, "Sub-event discovery and retrieval during natural hazards on social media data," in *WWW*, 2016.
[18] A. Zubiaga, D. Spina, E. Amigó, and J. Gonzalo, "Towards real-time summarization of scheduled events from twitter streams," in *Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, 2012.
[19] S. Unankard, X. Li, M. Sharaf, J. Zhong, and X. Li, "Predicting elections from social networks based on sub-event detection and sentiment analysis," in *WISE*, 2014.
[20] D. Pohl, A. Bouchachia, and H. Hellwagner, "Automatic sub-event detection in emergency management using social media," in *WWW*, 2012.
[21] D. Abhik and D. Toshniwal, "Sub-event detection during natural hazards using features of social media data," in *ACM WWW*, 2013.
[22] P. Meladianos, G. Nikolentzos, F. Rousseau, Y. Stavrakas, and M. Vazirgiannis, "Degeneracy-based real-time sub-event detection in twitter stream," in *AAAI ICWSM*, 2015.
[23] H. Xiao, P. Rozenshtein, and A. Gionis, "Discovering topically- and temporally-coherent events in interaction networks," in *CoRR*, 2016.
[24] C. Xing, Y. Wang, J. Liu, Y. Huang, and W. Ma, "Hashtag-based sub-event discovery using mutually generative lda in twitter," in *AAAI*, 2016.
[25] G. S. Manku, A. Jain, and A. Das Sarma, "Detecting near-duplicates for web crawling," in *ACM WWW*, 2007.
[26] F. E. Grubbs, "Procedures for detecting outlying observations in samples," in *Technometrics*, 1969.
[27] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," in *ACM Comput. Surv.*, 2009.
[28] S. Rayana and L. Akoglu, "Less is more: Building selective anomaly ensembles," in *ACM Trans. Knowl. Discov. Data*, 2016.
[29] L. C. Freeman, "A set of measures of centrality based on betweenness," in *Sociometry*, 1977.