EviDense: A Graph-Based Method for Finding Unique High-Impact Events with Succinct Keyword-Based Descriptions

Oana Balalau
Max Planck Institute for Informatics
obalalau@mpi-inf.mpg.de

Carlos Castillo
Universitat Pompeu Fabra
chato@acm.org

Mauro Sozio
LTCI, Télécom ParisTech University
sozio@telecom-paristech.fr

Abstract

Despite the significant efforts made by the research community in recent years, automatically acquiring valuable information about high impact-events from social media remains challenging. We present EviDense, a graph-based approach for finding high-impact events (such as disaster events) in social media. Our evaluation shows that our method outperforms state-of-the-art approaches for the same problem, in terms of having higher precision, lower number of duplicates, while providing a keyword-based description that is succinct and informative.

Introduction

Social media have been playing increasingly a major role during crises and disasters. For example, the American Red Cross (ARC) pointed out the effectiveness of social media and mobile apps in handling emergency situations such as those generated by a disaster event.[1].

Unfortunately, despite the significant efforts made by the research community in recent years, automatically acquiring valuable information about high impact-events from social media remains challenging. This is due to fact that social content is often noisy, inconsistent and ambiguous. As a result, making sense of a large collection of tweets, for example, is non-trivial even in the case when the collection of tweets is static and it does not evolve over time.

One of the challenges we address in our work is how to provide a succinct keyword-based description of high-impact events containing the most relevant information about the events, such as what happened, where, and when. According to a survey by the US Congressional Service, the administrative cost for monitoring multiple social media sources, which typically produce large amounts of noisy data, is significant (Lindsay 2011). Therefore, in order to alleviate the burden of analyzing social content, a succinct and informative description of the events is needed.

Our approach consists of the following steps: i) filtering of the tweets by retaining only those containing at least one term in a given lexicon. ii) finding locations whose number of occurrences in tweets deviates significantly on a given time window from their average frequency; iii) a graph mining approach for selecting the relevant keywords in the description, based on a novel definition of a clique and a quasi-clique in the weighted case. Our definition of a weighted clique, which is a generalization of a clique in an unweighted graph, ensures an event is described in a succinct manner by relevant keywords. We call our approach EviDense, as our graph mining approach is based on finding "dense" regions in the graph representing the co-occurrence of keywords in the tweets.

We evaluate this algorithm against state-of-the-art approaches for the same or similar problems on a collection of tweets covering the period between November 2015 and February 2016, by means of a crowdsourcing platform. Our experimental evaluation shows that our approach outperforms the baselines both in terms of precision (at k) and number of duplicates, while the keyword-based description provided by our algorithm is succinct and informative.

Given these results, we consider EviDense could represent a valuable tool for analyzing social content, but also other datasets such as news articles from mainstream media, as well as for studying how they compare. It could also be used to boost the performance of other approaches. For example, collections of tweets labeled as related to disasters by our approach could be used to create training sets for automatic classifiers.

Related Work. In our survey of the related work and in our experimental evaluation, we focus on those approaches that allow a large-scale analysis over a long period of time. In particular, we focus on unsupervised approaches that enforce or allow to enforce some kind of constraint on the output size. We summarize the relevant related work as follows. (Weng and Lee 2011) developed an event detection algorithm (EDCoW) based on clustering of the wavelet-based signal of words. The approach developed in (Guille and Favre 2014) aims at finding words with similar temporal patterns in a given time window. EDCoW has been shown to perform better than several other event detection techniques in a recent study (Weiler, Andreas, Grossniklaus, Michael and Scholl 2015). (Angel et al. 2014) developed an algorithm for maintaining overlapping dense subgraphs with limited size in a dynamic graph. Another graph mining tech-

[1] http://www.redcross.org/news/press-release/More-Americans-Using-Mobile-Apps-in-Emergencies

nique for real-time event discovery is presented in (Agarwal, Ramamritham, and Bhide 2012). Finding dense subgraphs of an input graph has received increasing attention in recent years, in the graph mining community: (Balalau et al. 2015), (Danisch, Balalau, and Sozio 2018), (Danisch, Chan, and Sozio 2017), (Epasto, Lattanzi, and Sozio 2015).

Algorithm

Our algorithm consists of the following main steps: 1) collection of tweets containing keywords related to disaster events by means of the Twitter API; 2) recognition and tagging of mentions of locations in the tweets; 3) finding bursts of mentions of locations; 4) mentions of locations are finally complemented with related keywords so as to provide more informative results. Each of these steps is described in the following paragraphs.

Collection of Tweets and Preprocessing. Tweets are filtered using disaster keywords (Olteanu et al. 2014) and for the recognition and tagging of locations, we use an entity recognition tagger that was trained on Twitter data (Ritter et al. 2011). We remove stopwords, URLs and special characters such as emoticons.

Finding Bursts of Locations. When an event such as a disaster event occurs, we observe a burst of activity on Twitter with terms pertinent to the event increasing suddenly their frequency in tweets. In our approach (where tweets contain keywords related to disaster events), a burst in the number of mentions of a location gives us a first signal that a disaster event is unfolding at that location. Previous works (Pan and Mitra 2011) have used geolocation of posts and not mentions of locations. We argue that using mentions of locations makes our algorithm more versatile, allowing it to analyze datasets coming from different sources, such as news articles which don't have a geolocation. Burstiness of words in streams of data is a well-studied topic (Zhu and Shasha 2003; Lappas et al. 2009) and in our approach, we use a simple technique, similar to other event detection methods (Guille and Favre 2014).

For each location, we compute a set of intervals in which the deviation between the frequency of the location and its expected frequency is always above a threshold. Our intuition is that all tweets (dealing with the same location) posted during such an interval are discussing the same event. We refer to such intervals as *interesting intervals*. The expected frequency of a location is computed assuming that location frequencies can be approximated by the binomial distribution. Then, all the (location, interesting interval) pairs are ranked according to how much the frequency of a location deviates from its expected frequency and we retain the top k pairs.

Finding Quasi-Cliques. In order to complement the set of locations with additional information about the corresponding event, we employ a graph mining approach. In particular, for each location and each interesting interval for that location, we wish to find a set of terms which induce a dense region in the co-occurrence graph during that time interval. Given an interesting interval $\mathcal{I}$ and a collection of tweets, we define a weighted undirected graph $G_{\mathcal{I}} = (V_{\mathcal{I}}, E_{\mathcal{I}})$, where $V_{\mathcal{I}}$ consists of the set of terms in the collection of tweets, while there is an edge between two nodes if the corresponding terms co-occur in at least one tweet posted within $\mathcal{I}$. A weight function $c : E \to \mathbb{R}^+$ represents the number of co-occurrences of terms in tweets posted within $\mathcal{I}$.

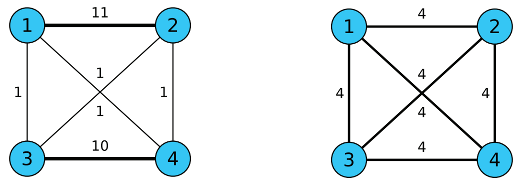

Figure 1: The left subgraph has a larger weight than the right one, but the nodes in the right subgraph are better connected.

Several definitions of weighted cliques have been provided in the literature, such as a subgraph of maximum total weight where any two nodes are connected (Östergård 1999), as well as a subgraph with maximum total weight and number of nodes no larger than a threshold provided in input (Alidaee et al. 2007). Observe that both definitions would favor the graph on the left in Figure 1, which exhibit weak connections between the set of nodes $\{1, 2\}$ and $\{3,4\}$. As a result, those two different parts of the graph might actually refer to two different events. It is more likely that the nodes of the graph on the right in Figure 1 refer to the same event, as the edges of the graph have the same weight.

This motivates the following novel definitions of cliques and quasi-cliques. We say that a graph H is a *weighted clique* if all pairs of nodes in H are connected by an edge with the same weight. Given a parameter $\gamma > 0$, we then define a *weighted quasi-clique* as follows.

Definition 1 (Weighted Quasi-Clique) *Given an undirected weighted graph* $H = (V(H), E(H), w)$*,* $0 < \gamma \leq 1$*, we say that* H *is a* weighted γ-quasi-clique *if the following holds:*

$$\sum_{e \in E(H)} w(e) \geq \gamma \cdot w_{max}(H) \binom{|V(H)|}{2},$$

where $w_{max}(H) = \max_{e \in E(H)}(w(e))$*.*

We define the function $q_G : V \to (0, 1]$ (q for short) to be the function which associates to every set $S \subseteq V$ a rational number γ such that the subgraph induced by S in G is a γ-quasi-clique and γ is the largest value for which this holds.

Finding quasi-cliques is an NP-hard problem, therefore we resort to the following heuristic for finding quasi-cliques containing a node v and at most s nodes. The algorithm starts with v and adds the edge with maximum weight containing v. At any given step, let S be set of current nodes. If $|S| = s$, the algorithm stops. Otherwise, it adds a node x in the neighborhood of S maximizing $q(S \cup \{x\})$ provided that by adding x the resulting subgraph is still a γ-quasi-clique.

Experimental Evaluation

Corpora. We collect tweets posted over a period of 4 months between November 2015 and February 2016. We use the Twitter Streaming API while filtering the tweets so that they contain at least one term related to disasters (Olteanu et al. 2014) and they are written in English. We obtain approx 3M tweets in total, which we divide into four datasets (one per month).

Related work. We compare against MABED (Guille and Favre 2014) and EDCoW (Weng and Lee 2011). All approaches are evaluated on the same collection of tweets. There has been significant disagreement among the crowd workers when interpreting the results of (Angel et al. 2014). Therefore, we omit such an approach from our study, deferring a more careful evaluation to future work.

Parameter settings. In our approach, we find the top 20 events and we set the size of the quasi-clique to be at most 10 and the γ parameter at least 0.5. We run MABED using the implementation provided by the authors and the setting specified in the original paper (Guille and Favre 2014), that is $p = 10, \theta = 0.7$ and $\sigma = 0.5$. For the EDCoW algorithm we use the implementation of (Weiler, Andreas, Grossniklaus, Michael and Scholl 2015) and we set the parameters as follows: the size of first level of intervals is $s = 100s$, while we take $\Delta = 32$, setting a size of $3200s$ for the second-level intervals and, as in (Weiler, Andreas, Grossniklaus, Michael and Scholl 2015), we set $\gamma = 1$. As EDCoW does not enforce any constraint on the size of the output, we order the results according to ϵ (as defined in the original paper), which measures the relevance of the results and retain only the top k results.

Metrics. We evaluate the precision at k, denoted with $P@k$ (or precision for short), which is defined as the number of true events in the top k results, divided by k. In addition to the precision, we compute the fraction of duplicate events among all the events retrieved, i.e. the *DeRate* (Li, Sun, and Datta 2012). From these two metrics, we can infer a third one, which measures the fraction of unique events (i.e. duplicates do not contribute) in the top k results. We denote such a metric with $U@k$. We observe that if an algorithm performs best in terms of $U@k$, it performs best also in terms of recall. Therefore, we do not report recall in our experimental evaluation.

Crowdsourcing Settings. In order to ensure a fair comparison, we use a crowdsourcing service, GetHybrid[2]. For each result produced by any of the approaches, we ask 5 workers to determine whether it was a disaster event, that is an event considered to be a disaster by the US Government or the International Disaster Database.[3] We added to the description of a result two relevant tweets in order to facilitate the labeling task. In order to evaluate a result, a worker would select one of the following answers to the question of what type of event do the keywords and tweets describe:

(A) A natural disaster (earthquake, landslide, volcano, extreme temperature, hurricane, large and dense fog, large storm, flood, tsunami, drought, wildfire, epidemic, large accident involving animals, asteroid impact),

(B) A technological disaster (chemical spill, building collapse, explosion, fire, gas leak, large poisoning, nuclear, radiation, cyberattack),

(C) A human-induced disaster (war, terror attack),

(D) A large transport accident (air, road, rail, water),

(E) Not a natural, technological, or human-induced disaster, or large transport accident.

Answers A-D correspond to a disaster. We label a result as a disaster if 3 out of 5 workers confirmed. In order to compute the *DeRate*, we considered duplicate events to be subevents or consequences of an event. Two events are considered to be duplicates if 3 out of 5 workers confirmed.

Estimating *DeRate* and U@k. The task of estimating the average number of duplicate events is non-trivial, given the large number of results. Asking the workers in GetHybrid to estimate the number of duplicate events in a list containing approximately 20 or more results is time-consuming and most probably would result in a non-accurate evaluation. Therefore, we draw a random sample from the set of all possible event pairs. Each worker is then asked to determine whether a given pair of events in the sample contains duplicate events or not. The resulting fraction of duplicate event pairs in the sample is used to infer a 95% confidence interval on the fraction of duplicate event pairs on the whole dataset, using the Wilson score method (Newcombe 1998). From the fraction of duplicate pairs we can estimate the number of unique events, $U@k$.

Comparison. In Table 1 we present the average $P@20$ for all the approaches over the time period Nov. 2015 - Feb. 2016. We observe that EviDense outperforms the other two approaches.

Method	Average $P@20$
EDCoW	0.325
MABED	0.537
EviDense	0.737

Table 1: Average precision over Nov. 2015 - Feb. 2016.

We evaluate the number of duplicate events in the results. First, we measure the fraction of duplicate event pairs in our sample. This is shown in Table 2. Observe, that EviDense produces less duplicate event pairs. In particular, our results are better with a 95% confidence.

Method	Fraction in sample	95% Confidence Interval
EDCoW	0.120	0.049 to 0.250
MABED	0.220	0.145 to 0.316
EviDense	0.020	0.003 to 0.077

Table 2: Fraction of duplicate event pairs.

From the results shown in Table 2, we obtain a 95% confidence interval on the fraction of duplicate events, i.e. the *DERate*. We obtain a 95% confidence interval of $[0.000, 0.846]$ for EDCoW, $[0.372, 0.906]$ for MABED, and $[0.000, 0.474]$ for EviDense. From the latter result, it is difficult to determine which algorithm performs best in

[2] https://www.gethybrid.io

[3] https://www.ready.gov/be-informed, http://www.emdat.be/classification

terms of *DERate*. Moreover, observe that approaches with higher precision might be penalized by the *DERate*, in that, they tend to have a larger number of duplicates. For example, an approach which retrieves exactly one event has a *DERate* of zero. Therefore, we also consider the $U@k$ metric, that is, the fraction of unique events in the top k results. The results are shown in Table 3. We can see that even with a pessimistic estimate, EviDense outperforms the other approaches in terms of $U@k$, while the fraction of unique events in the top-20 results can be up to 73.7%.

Method	U@20
EDCoW	0.050 to 0.325
MABED	0.050 to 0.337
EviDense	0.387 to 0.737

Table 3: Average ratio of unique events Nov. 2015 - Feb. 2016.

The output of our algorithm is shown in Table 4. We can see that the description of each of the events is succinct and informative. In particular, one can easily retrieve the location of the event (in bold), its time-frame and what happened.

Time (UTC)	Event keywords
Dec 03 02:20, Dec 07 08:50	**San Bernardino**, dead, female, #sanbernardino, killed, male, police, shooting
Dec 27 01:04, Dec 28 03:37	**Dallas**, Rowlett, tornado
Dec 07 03:24, Dec 10 21:11	**Chennai**, damaged, floods, fresh, issue, lost, passport, psks, sushmaswaraj
Dec 04 09:12, Dec 04 23:23	**Cairo**, attack, firebomb, killed, nightclub, people, restaurant
Dec 28 20:15, Dec 29 09:21	**Cleveland**, 12-year-old, Tamir Rice, charged, death, grand, indict, jury, police

Table 4: Top 5 events discovered in December 2015 by EviDense. The event is centered on the location given in bold.

Conclusions

We presented EviDense, a graph-based approach for finding high-impact events in social media. We address the challenge of providing a succinct and informative description of the events retrieved with our approach while focusing on disaster events.

EviDense could represent a valuable tool for analyzing social content and could also be used to boost the performance of other approaches. For example, collections of tweets labeled as related to disasters by our approach could be used to create training sets for automatic classifiers.

Acknowledgments

Part of this work was done while O. Balalau was a student at Télécom ParisTech University. C. Castillo is partially funded by La Caixa project LCF/PR/PR16/11110009. M. Sozio is partially funded by the French National Agency (ANR) under project FIELDS (ANR-15-CE23-0006).

Data and code sharing. The ids of the tweets we use in our evaluation are publicly available, together with our code.[4]

References

Agarwal, M. K.; Ramamritham, K.; and Bhide, M. 2012. Real Time Discovery of Dense Clusters in Highly Dynamic Graphs: Identifying Real World Events in Highly Dynamic Environments. *VLDB* 5(10):980–991.

Alidaee, B.; Glover, F.; Kochenberger, G.; and Wang, H. 2007. Solving the maximum edge weight clique problem via unconstrained quadratic programming. *European Journal of Operational Research* 181(2):592 – 597.

Angel, A.; Koudas, N.; Sarkas, N.; Srivastava, D.; Svendsen, M.; and Tirthapura, S. 2014. Dense subgraph maintenance under streaming edge weight updates for real-time story identification. *VLDB Journal* 175–199.

Balalau, O. D.; Bonchi, F.; Chan, T. H.; Gullo, F.; and Sozio, M. 2015. Finding subgraphs with maximum total density and limited overlap. In *WSDM*, 379–388.

Danisch, M.; Balalau, O.; and Sozio, M. 2018. Listing k-cliques in sparse real-world graphs. In *WWW*.

Danisch, M.; Chan, T. H.; and Sozio, M. 2017. Large scale density-friendly graph decomposition via convex programming. In *WWW*, 233–242.

Epasto, A.; Lattanzi, S.; and Sozio, M. 2015. Efficient densest subgraph computation in evolving graphs. In *WWW*, 300–310.

Guille, A., and Favre, C. 2014. Mention-anomaly-based Event Detection and Tracking in Twitter. *ASONAM* 375–382.

Lappas, T.; Arai, B.; Platakis, M.; Kotsakos, D.; and Gunopulos, D. 2009. On Burstiness-Aware Search for Document Sequences. 477–485.

Li, C.; Sun, A.; and Datta, a. 2012. Twevent: Segment-based Event Detection from Tweets. *CIKM* 155–164.

Lindsay, B. R. 2011. Social media and disasters: Current uses, future options, and policy considerations.

Newcombe, R. G. 1998. Two-sided confidence intervals for the single proportion: comparison of seven methods. *Statistics in medicine* 17(8):857–872.

Olteanu, A.; Castillo, C.; Diaz, F.; and Vieweg, S. 2014. CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises. In *ICWSM*.

Östergård, P. R. J. 1999. A New Algorithm for the Maximum-Weight Clique Problem. *Electronic Notes in Discrete Mathematics* 3:153–156.

Pan, C.-C., and Mitra, P. 2011. Event detection with spatial latent dirichlet allocation. In *JCDL*, 349–358.

Ritter, A.; Clark, S.; Mausam; and Etzioni, O. 2011. Named entity recognition in tweets: An experimental study. In *EMNLP*, 1524–1534.

Weiler, Andreas, Grossniklaus, Michael and Scholl, M. 2015. Evaluation Measures for Event Detection Techniques on Twitter Data Streams. *Bicod* 1–157.

Weng, J., and Lee, B.-S. 2011. Event Detection in Twitter. In *ICWSM*.

Zhu, Y., and Shasha, D. 2003. Efficient Elastic Burst Detection in Data Streams. In *KDD*, 336–345.

[4] https://github.com/nyxpho/evidense