User-driven geolocated event detection in social media

Anes Bendimerad, Marc Plantevit, Céline Robardet, Sihem Amer-Yahia

User-driven geolocated event detection in social media

Anes Bendimerad, Marc Plantevit, Céline Robardet, Sihem Amer-Yahia

Abstract—Event detection is one of the most important research topics in social media analysis. Despite this interest, few researchers have addressed the problem of identifying geolocated events in an unsupervised way, and none includes user interests during the process. In this paper, we tackle the problem of local event detection from social media data. We present a method to automatically identify events by evaluating the burstiness of hashtags in a geographical area and a time interval, and at the same time integrating user feedback. We devise two algorithms to discover user-driven events. The first one relies on an exact enumeration process, while the other directly samples the space of events. In our empirical study, we provide evidence that geolocated events cannot be detected by non location-aware methods. We also show that our methods (i) outperform by a factor of two to several orders of magnitude state-of-the-art methods designed to discover geolocated events, (ii) are more robust to noise, (iii) and produce high quality events with respect to user interests.

Index Terms—Event detection, social media analysis, pattern mining.

1 Introduction

Social microblogging (Twitter, Weibo, Instagram, etc.) gives people the ability to interact at a worldwide scale by broadcasting their interests, feelings, reactions to their daily life and surrounding events. As such, they are an incredibly rich mean to know the pulse of the world, or of a specific neighborhood, in real time. Analyzing the abundant user-generated content can provide high valued information. Social media data have been analysed for several purposes, e.g. to understand the concerns of a population [1], study disease dynamics [2], or predict real-world outcomes [3].

Event detection has long been a research topic and has received much attention in the data mining community over the last decade [4–6]. Whereas real-life events are considered as phenomena that unfold over space and time, from a data mining point of view they are conventionally regarded as a set of terms (e.g. hashtags, user mentions, words) whose frequency bursts [7]. However, a bursting term is not necessarily related to an event, and real-life events can be blurred by pointless bursting terms. Several terms can also depict the same event, and thus detection methods must address problems related to synonymy. Furthermore, some methods are defined in supervised settings which require huge effort to collect labeled examples [6], [8–12]. In this paper, we address these pitfalls by considering geolocated events. Focusing on terms that burst for a geographical area meets the natural and intuitive notion of event. Extracting such events remains challenging because geolocated events depict small scale phenomena that are covered by much fewer terms than global events (i.e., large scale events). For instance, in a very large city like New York City, dozens of events can take place simultaneously and it can be tricky to find those that stimulate users' curiosity.

This paper responds to these concerns by (1) proposing efficient algorithms to detect geolocated events and (2) integrating user interests into the extraction process through interactions with the system. This is the first attempt to the discovery of user-driven events. In our approach, posts (e.g., tweets) are modeled as a labeled graph whose vertices encode geographical areas and edges depict neighborhood relationships. Time series of term occurrences are associated to vertices. In this unsupervised framework, a geolocated event is considered as a set of terms whose number of occurrences is large enough in a connected subgraph and a time interval. Events are then extracted in a time window and the user has the possibility to tag those he/she likes. This interactive process allows to extract events of interest to users. User preferences are then integrated in the quality measure used to define and rank events. This measure conveys user interest and promotes events that involve topics or geographical areas the user is interested in.

Fig. 1 describes our approach. From a social network, we extract time-stamped geolocated posts published during a given period. This set of posts, B (e.g., tweets), is used to generate a graph G_H of term co-occurrences. A second graph G_V depicts adjacency relations between geographic areas from which posts were emitted. Those graphs are used to guide the event discovery process making it possible to solve term synonymy problems as well as the spatial variability (size and location). Moreover, these graphs are the support of an interactive process with the user and are used to rank the identified events according to her preferences. By liking or not events, the user selects some events that are then used by the algorithm to derive locations and topics of preference. These preferences are then used to give a greater importance to events related to preferred geographical areas

- *A. Bendimerad and C. Robardet are with University of Lyon, INSA Lyon, CNRS UMR 5205.*
- *M. Plantevit is with University of Lyon, University Lyon 1, CNRS UMR 5205*
- *S. Amer-Yahia is with University of Grenoble Alpes, CNRS.*

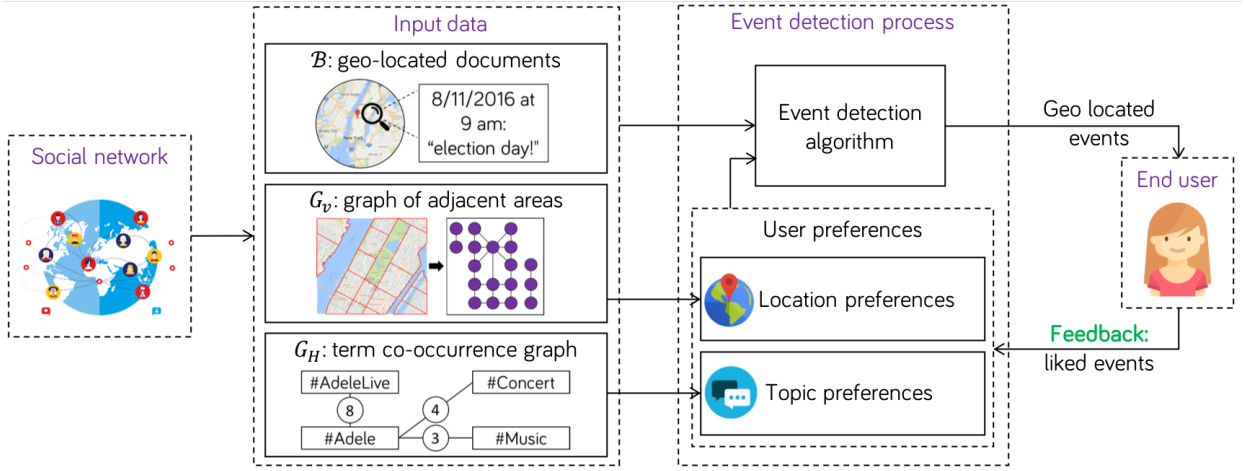


Fig. 1. Overview of the user-driven event detection system.

or terms of interest. If there is no interaction with the user, only data-driven events are detected.

The main contributions of this paper are the following:

- *Problem formulation.* We define, in a unified view, the problem of user-driven and data-driven geolocated event discovery. We propose an event interestingness measure that is guided by user interests.
- *Algorithms and analysis.* We propose two algorithms to discover geolocated events in social media. SIGLER-Cov[1] is based on the generate-and-test paradigm. It efficiently exploits upper and lower bounds that make it usable on large-scale microblogging data such as Twitter. SIGLER-Samp directly computes a sample of the geolocated events and retrieves interesting geolocated events in a very short time.
- *Evaluation.* We report a thorough empirical study that provides both quantitative and qualitative performances of our approach. First, we establish that data driven geolocated events cannot be discovered by non location-aware approaches. Second, we compare our method with MED [5] and GeoBurst [13], the state-of-the-art data driven geolocated event mining algorithms. We demonstrate that SIGLER (1) is several orders of magnitude faster than MED and three times faster than GeoBurst, and (2) is more robust to noise compared to both methods. Finally, the ability of our method to extract interesting user-driven events is experimentally validated. To that end, we ran experiments using a crowdsourcing platform on several cities (e.g., New York, Los Angeles, London) with different settings (e.g., unpaired and paired samples).
- Source code and datasets are publicly available: https://tinyurl.com/yywy2fqc.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 presents a unified framework for data-driven and user-driven geolocated event detection. Section 4 introduces the user-driven approach. Section 5 presents our event detection algorithms: SIGLER-Cov, an efficient event detection algorithm with coverage guarantee, and SIGLER-Samp that directly computes a sampling of the output space of user-diven geolocated events. Section 6 provides experimental results. Section 7 concludes and provides future directions.

1. SIGLER stands for SubjectIve GeoLocated Event discoveRy.

2 Related Work

This paper focuses on space-time event detection from social media with user interaction during the detection process. Detecting events in social media has raised great interest in the last decade. Several papers propose to identify targeted events using supervised [6], [9–12] or semi-supervised approaches [14], [15]. Such methods require a certain amount of labeled data to detect domain specific events, as opposed to the unsupervised problem considered in this paper.

In the seminal unsupervised approach [16], events are detected using signal processing methods on word occurrence time series. As several bursty terms can be associated to one real event (synonyms or correlated words), clustering-based approaches [17–19] are used to group terms with similar temporal patterns. In our experimental study, we show that these approaches fail to detect geo-spatial events. Other methods [4], [20] model the data as a heterogeneous interaction graph with content-related features associated to the nodes. Non-parametric statistics or clustering techniques are then applied on it to extract subgraphs. However, the subgraphs of strong interaction describe hardly local events that are generally not related to direct interactions within the graph. In fact, an event can be characterized with a sparse interaction network corresponding to tweets posted by people who mainly do not know each other. It is important to note that as non location-aware event detection methods fail to detect geolocated events, post-processing such events in order to promote events according to the user interest is not a solution.

Few approaches integrate spatial information during the event detection process. Among them, Triovecevent [8] discovers local events based on multimodal embedding. However, this method is supervised and requires a trained classifier to judge whether a detected cluster of tweets is related to an event or not. Several demo papers [21–24] present simple systems designed to identify groups of

messages that have been posted close in time and space, and assimilate them to events based on the co-occurrence of their terms. A more sophisticated Multiscale Event Detection (MED) method [5] computes the similarity between tweets based on the time series of the occurrences of their common terms considered at different temporal and spatial scales. The most appropriate scale is used to derive a similarity graph on top of which events are detected by a graph-based clustering process. Another state-of-the-art approach is GeoBurst [13]. It is based on an authority measure that captures the geo-topic correlations among tweets. These correlations make it possible to group the tweets using some elaborate clustering techniques. In [13] the authors show that GeoBurst provides better results than two other local event detection approaches: EVENTWEET [25] and WAVELET [26]. EVENTWEET is based on a clustering of spatially and temporally bursting terms. WAVELET uses wavelet analysis to extract events from geo-tagged Flickr photos. Thus, MED and GeoBurst are the main competitive approaches we consider in Section 6. We show the superiority of our approach towards these two competitors in terms of computation time and noise robustness.

Local event detection can be seen as a local modeling problem. Indeed, local events are related to specific parts of the data space. Discovering such peculiarities is the aim of local pattern mining, especially subgroup discovery [27] and exceptional model mining [28]. Our approach is rooted in the local pattern mining framework.

The problem of taking user interest in pattern mining was early identified in [29] and has seen a renewed interest in the last decade. User-driven and interactive pattern mining can be divided into three groups: (1) The approaches based on a prior model. Among them, the prominent work of de Bie [30] defines a general and statistically-founded framework of exploratory data analysis. Information theory is used to formalize the subjective interest as the divergence to prior belief; (2) The approaches that integrate user feedback interactively during the mining task [31–33]; (3) The methods which aim to learn an explicit model of user interests [34] and rank patterns in post-processing [35]. There is no approach in the literature that detects user-driven events. The one proposed in this paper integrates user feedback during the mining task by building a model of user interest on the fly.

Symbols	Definitions
$\mathcal{B} = \{b_1, b_2, ...\}$	The set of posts. Each post $b \in \mathcal{B}$ is described by (1) its terms $b.terms$, (2) the time $b.time$ at which it was sent, (3) and $b.loc$ the GPS coordinates of b.
$H = \{h_1, h_2, ...\}$	The set of terms that occur in posts $\mathcal{B}$.
$T = [\![t_1, t_m]\!]$	The ordered set of m timestamps of the posts.
$G_V = (V, E)$	A graph where the vertices of V are geographical areas and the edges of E connect adjacent areas.
$G_H = (H, H \times H)$	A weighted term co-occurrence graph. The weight $W(h_i, h_j)$ is the number of posts in which h_i and h_j co-occur.
$f(h, v, t)$	The number of occurrences of a term $h \in H$, for a vertex $v \in V$ and a time $t \in T$.
score(h, v, t)	The burstiness of the term $h \in H$ in the space time (v, t).
$\mathcal{H}(v, t)$	$\mathcal{H}(v, t) = \{h \in H \mid \text{score}(h, v, t) > 0\}$. The set of terms that burst in the space time (v, t). For the sake of simplicity, we generalize $\mathcal{H}(S, I) = \bigcap_{v \in S} \bigcap_{t \in I} \mathcal{H}(v, t)$ for $S \in V$ and $I \sqsubseteq T$.
$P = (S, I, K)$	A geolocated pattern defined with a set of vertices $S \subseteq V$, a time interval $I \sqsubseteq T$, and a set of terms $K \subseteq H$. We use $P = (S, I)$ instead of $P = (S, I, \mathcal{H}(S, I))$.
$\mathcal{M}, \mathcal{M}_u$	$\mathcal{M}(P)$ is the data-driven quality measure of the pattern P, and $\mathcal{M}_u(P)$ is the user-driven quality measure of P.
δ	A threshold on the minimum quality $\mathcal{M}_u(P)$ required to consider P as an event.
$\mathcal{R} = \{P_1, P_2, ...\}$	A set of patterns.
cov(A, B)	A function that measures how much the set B covers the set A: $cov(A, B) = \frac{\lvert A \cap B \rvert}{\lvert A \rvert}$.
cover(P_1, P_2)	A function that measures how much the pattern P_2 covers the pattern P_1.
`minCov`	A minimum threshold on the cover value to consider that a pattern covers enough another one.
$Q_u(h, v)$	The user interest on the term h and the vertex v. It is the average between $Q_{u,H}(h)$, the user interest on h, and $Q_{u,H}(v)$ the user interest on v. These functions take their values in $[1, \texttt{maxPref}]$, with $\texttt{maxPref} > 1$.

TABLE 1
Notations.

3 A UNIFIED FRAMEWORK FOR DATA-DRIVEN AND USER-DRIVEN GEOLOCATED EVENTS

In this section, we introduce the problem of data and user driven geolocated event discovery in a unified view. Table 1 summarizes the definitions and notations used along the paper. From microblogging social media, we consider a set of posts B, a batch[2], where each element b is described by: (1) the set of terms that appear in b ($b.terms$), (2) the time at which b was sent ($b.time$), (3) the GPS coordinates of the post of b ($b.loc$). From such a batch, we collect the following data: (1) H, the set of terms $H = \bigcup_{b \in B} b.terms$; (2) $T = [\![t_1, t_m]\!]$, the time interval made of m timestamps[3], and (3) $V = \{v_1, ..., v_n\}$, the districts obtained after discretizing the geographical space in n areas. For example, the geographical area defined by the square $[\min_{b \in B} b.loc.x, \max_{b \in B} b.loc.x] \times [\min_{b \in B} b.loc.y, \max_{b \in B} b.loc.y]$ can be divided along a grid and each square is associated to an element of V. Let area : $\mathbb{R}^2 \to V$ be the function that maps GPS coordinates (x, y) to V. V is hereafter considered as the vertices of a graph $G_V = (V, E)$ whose edges E connect vertices corresponding to adjacent areas.

Our approach is based on the detection of strong variations in term occurrences for a vertex and a timestamp. To this end, we consider the number of occurrences of a term

2. The posts emitted consecutively in a given period of time.

3. In our experiments, a timestamp corresponds to an interval of 3 hours.

$h \in H$, for a vertex $v \in V$ and a time $t \in T$ as function f:

$$f(h,v,t) = |\{b \in B \mid (h \in b.terms) \text{ and } (\text{area}(b.loc) = v) \text{ and } (b.time = t)\}|.$$

As generally admitted [13], [36], a geolocated event reflects that a large number of messages deal with the same subject during the same time and place. Each topic is associated with the set of its representative terms. For instance, in Fig. 12, the first detected event in New York is the *New York Comic Con*, associated with the terms (#nycc, #nycc2016, etc.). These terms burst in a period of three days, and around the location of Jacob K. Javits Convention Center where the convention took place at that time. Thus, a term is likely to correspond to an event in a space-time zone if its number of occurrences is significantly greater than what is observed in the other times for the same space zone. To that end, we compute the mean $\mu(h,v)$ and standard deviation $\sigma(h,v)$ of f over all the timestamps $[\![t_1, t_m]\!]$. If the difference between $f(h,v,t)$ and its average value is greater than θ times the standard deviation, then the number of occurrences of h is said to be significant for v and t. $\theta \geq 0$ is a parameter that controls the sensibility of the method, and whose default value is set to 1. Then, to reduce the impact of very frequent terms, the significance of each term is weighted by the normalized inverse document frequency factor [16]: $\text{idf}(h) = 1 - \frac{\log\left(f(h,V,T)\right)}{\log\left(f(H,V,T)\right)}$. Hence, the score function is:

$$\text{score}(h,v,t) = \Big(f(h,v,t) - \big(\mu(h,v) + \theta\sigma(h,v)\big)\Big) \times \text{idf}(h),$$

with $\sigma(h,v) = \sqrt{\frac{1}{m-1}\sum_{t\in T}\left(f(h,v,t) - \mu(h,v)\right)^2}$ and $\mu(h,v) = \frac{\sum_{t\in T} f(h,v,t)}{m}$. A term h is bursting for (v,t) if its score value is positive. The set of bursting terms for a space-time zone (v,t) is thus defined by:

$$\mathcal{H}(v,t) = \{h \in H \mid \text{score}(h,v,t) > 0\}.$$

For the sake of simplicity, given $S \in V$ and $I \sqsubseteq T$, we generalize $\mathcal{H}(S,I) = \bigcap_{v\in S}\bigcap_{t\in I}\mathcal{H}(v,t)$.

We consider a geolocated pattern P as a tuple (S,I,K) where $S \subseteq V$, $I \sqsubseteq T$, and $K \subseteq H$. We aim to identify patterns $P = (S,I,K)$ corresponding to events described by a location S, a time interval I and a set of terms K. The ability for P to represent an event is evaluated by the measure $\mathcal{M}(P)$:

$$\mathcal{M}(P) = \sum_{h\in K}\sum_{v\in S}\sum_{t\in I}\text{score}(h,v,t).$$

The larger $\mathcal{M}(P)$, the higher than expected the frequency of the terms in (S,I). This means that when $\mathcal{M}(P)$ is large, the pattern $P = (S,I,K)$ is likely to be an event. In our problem formulation, we are only interested in patterns $P = (S,I,\mathcal{H}(S,I))$ where $\mathcal{H}(S,I)$ are terms that burst in all the space-time (S,I). This makes it possible to filter out a large part of noisy terms. In fact, it is not likely that a noisy term bursts simultaneously in all the space-times of (S,I). Consequently, in what follows, we are only considering patterns $P = (S,I,\mathcal{H}(S,I))$ and denote them $P = (S,I)$ (since K is uniquely determined by S and I).

In practice, the interest of an event strongly depends on the end-user. Indeed, a user may be much more interested in events related to some subjects (e.g., sport, music) or may prefer events happening in some specific locations (e.g., near her residence). To take user interests into account, we propose to integrate the proper interests of the user through an interactive process. To this end, we define the quality measure $\mathcal{M}_u(P)$ that includes the user's preferences as:

$$\mathcal{M}_u(P) = \sum_{h\in\mathcal{H}(S,I)}\sum_{v\in S}\sum_{t\in I}\text{score}(h,v,t) \times Q_u(h,v),$$

where Q_u increases with the interest of u on h and v. Thus, if P contains some terms h or vertices v that are of interest according to user's feedback, then $\mathcal{M}_u(P)$ increases. When no user feedback is available for a pair (h,v), $Q_u(h,v)$ is equal to 1. We thoroughly discuss in Section 4 how the function Q_u is defined.

A user-driven geolocated event must be both spatially compact and have a significant value on its quality measure:

Definition 1 (User-driven geolocated event). Given a set of posts B, a set of timestamps T, a graph $G_V = (V,E)$, and a threshold $\delta > 0$, a pattern $P = (S,I)$, with $S \subseteq V$ and $I \sqsubseteq T$ (an interval of T), is a user-driven geolocated event iff (1) $G_V[S]$ is connected and (2) $\mathcal{M}_u(P) \geq \delta$.

If the user does not provide any feedback, the measure $\mathcal{M}_u$ is equal to $\mathcal{M}$. In such particular case, we speak of *data-driven* geolocated events.

Different geolocated events may overlap in terms, geographical area, and timestamps they share, depicting the same real-life event. This has two main disadvantages: (1) the size of the result set may be uselessly very large and redundant, and (2) the method performance may degrade due to the size of the output. Therefore, instead of finding the complete set of events, our goal is to return a concise summary that covers sufficiently all the events. To this end, we use the coverage measure cov. This function, defined for two sets or intervals A and B, measures how much the set B covers the set A: $\text{cov}(A,B) = \frac{|A\cap B|}{|A|}$. This coverage measure has been used in several problems in order to avoid the redundancy in the results [37], [38]. To measure how much a pattern $P_2 = (S_2,I_2)$ covers a pattern $P_1 = (S_1,I_1)$, we impose that P_2 sufficiently covers P_1 in all the dimensions. The function cover is thus:

$$\text{cover}(P_1,P_2) = \min\{\text{cov}(S_1,S_2), \text{cov}(I_1,I_2), \text{cov}(\mathcal{H}(S_1,I_1), \mathcal{H}(S_2,I_2))\}.$$

To get a set of patterns that cover all user-driven geolocated events while eliminating some redundancy, we propose to only retrieve a coverage guaranteed event summary.

Definition 2 (Coverage guaranteed event summary). Given a threshold $\texttt{minCov} \in [0,1]$, a coverage guaranteed event summary $\mathcal{R}_1$ of the set of all geolocated events $\mathcal{R}$ fulfills the following property:

$$\forall P \in \mathcal{R},\ \exists P' \in \mathcal{R}_1,\ \text{such that cover}(P,P') \geq \texttt{minCov}.$$

The last drawback that may appear is that two concomitant events in time and space are merged by our approach. To ensure that an event-pattern is semantically coherent – that is its related terms correspond to the same real life event

– a post-processing of the event patterns is performed. In Section 5.3, we explain how we apply a Louvain clustering [39] on terms to achieve this goal. The similarity between terms is computed based on their co-occurrences, since event-related terms tend to co-occur in the posts.

We propose two approaches to detect geolocated events: `SIGLER-Cov` that computes a coverage guaranteed event summary $\mathcal{R}_1$, and `SIGLER-Samp` that samples the pattern space according to the value of $\mathcal{M}_u$. These approaches are formally defined in Section 5. The following section details the computation of the user-driver weight used in the quality measure $\mathcal{M}_u(P)$.

4 Integration of User Feedback into Quality Measure

Let us now consider how to incorporate user interest into the geolocated event discovery process. Once a set of events has been identified, a user u appraises the detected events and indicates if she likes it. In doing so, she constructs a partial order on the patterns that is used to favor the discovery of the forthcoming geolocated events toward those of interest for u. User interest is expressed by $Q_u(h, v) = \frac{Q_{u,H}(G_H,h)+Q_{u,V}(G_V,v)}{2}$, the average of two interest measures on terms and vertices:

1) $Q_{u,H} : (G_H, h)$ takes as parameters a weighted graph G_H on terms and a term h. It assigns a value in $[1, \texttt{maxPref}]$, with $\texttt{maxPref} > 1$ a user-defined parameter, based on (1) the neighborhood of h in G_H and (2) a partial order on terms of H derived from the user event ranking. The closer h is to other liked terms in G_H, or the more recently it has been liked, the more $Q_{u,H}(G_H, h)$ is close to `maxPref`. Otherwise, it tends to 1.
2) $Q_{u,V} : (G_V, v)$ takes as parameters the graph G_V and a vertex $v \in V$. It takes its value in $[1, \texttt{maxPref}]$ and the closer v is to other liked vertices in G_V, or the more recently it has been liked, the more $Q_{u,V}(G_V, v)$ is close to `maxPref`.

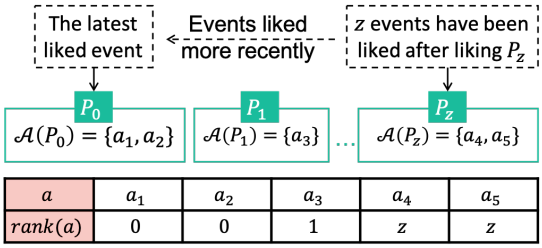
$\texttt{maxPref} > 1$ represents the maximum value that Q_u can reach. The choice of `maxPref` depends on how much we want to take into account the user preferences into the process. In the following, we use the value $\texttt{maxPref} = 3$, empirically chosen as the one that maximizes the number of liked events on NYC dataset (see the description of the experiment in section 6.3).

Both functions $Q_{u,H}(G_H, h)$ and $Q_{u,V}(G_V, v)$ are constructed above a weighted graph $G_X = (X, Y, W)$ with X a set of vertices, Y a set of edges and W the function that associates a weight to the edges ($W : Y \to \mathbb{N}$).

- $Q_{u,H}$ is evaluated on G_H, the term co-occurrence graph, such that $X = H$, $Y = H \times H$ and $\forall h_i, h_j \in H$, $W(h_i, h_j)$ equals the number of posts in which h_i and h_j appear simultaneously:

  $$W(h_i, h_j) = |\{b \in B \mid h_i \in b.terms \text{ and } h_j \in b.terms\}|.$$

  Related terms may co-occur in several posts and thus will tend to be connected by a shorter path whose sum of weights is high in G_H.



a	a_1	a_2	a_3	a_4	a_5
$rank(a)$	0	0	1	z	z

Fig. 2. Examples of liked events $\{P_1, P_2, ..., P_z\}$, and the values of $rank$ for terms associated to these events.

- $Q_{u,V}$ is evaluated on the graph G_V, with $X = V$, $Y = E$ and $W : E \to 1$.

In a similar way to PageRank score [40], we valuate the vertices of the graph such that (1) a high score associated to a vertex is transferred to its neighbors, with whom it is connected with a high weight, and (2) this effect decreases all the more as the neighboring vertex is far from the source vertex in the graph. However, unlike PageRank score, which is based on a random walk, we use a concentric model from the vertex to be valued. The vertex score is only influenced by the weights of its neighbors and not by their degrees:

$Q_X(G_X, x) = \alpha \sum_{(x,x')\in Y} \frac{W(x,x')}{\deg(x)} \times Q_X(G_X, x') + (1 - \alpha)\frac{1}{|X|}$, with $\deg(x) = \sum_{(x,x')\in Y} W(x, x')$. The first term, $\sum_{(x,x')\in Y} \frac{W(x,x')}{\deg(x)} \times Q(G_X, x')$, is simply the weighted average of qualities $Q_X(G_X, x')$ of the neighbor vertices. The second term, $\frac{1}{|X|}$, is a constant corresponding to the probability to directly reach a vertex. $\alpha \in]0, 1[$, whose default value is 0.7, is a balancing parameter between the two terms.

We propose to integrate the user preferences in the second term by replacing the constant value with the function $\mathcal{B}_u(x)$, a weight that amplifies the importance of the vertices involved in recently liked events:

$$Q_{X_u}(G_X, x) = \alpha \sum_{(x,x')\in Y} \frac{W(x, x')}{\deg(x)} Q_{X_u}(G_X, x') + (1 - \alpha)\mathcal{B}_u(x)$$

$\mathcal{B}_u(x)$ depends on the direct relation between x and the liked events. Let $\text{rank}(x)$ be the number of events that have been liked since the last event that (1) contained x and (2) was liked by the user. Fig. 2 shows the values of $rank$ for terms that appear in an ordered set of liked events $\{P_0, ..., P_z\}$. P_0 is the latest liked event, thus $rank(h_1) = 0$. The user liked P_z, but she liked z events after that, so $rank(h_5) = z$. Particularly, if a term or a vertex x does not belong to any liked event, then $rank(x) = +\infty$. We define $\mathcal{B}_u(x)$ as $\mathcal{B}_u(x) = 1 + \frac{\texttt{maxPref}-1}{1+log_2(1+\text{rank}(x))}$. We can observe that $\mathcal{B}_u(x) \in [1, \texttt{maxPref}]$, and if $rank(x) = +\infty$, then $\mathcal{B}_u(x) = 1$. It increases if x is related to a recently liked event (if $rank(x) = 0$, $\mathcal{B}_u(x) = \texttt{maxPref}$). The $\log_2$ function is used to smooth the effect of past liked events.

$Q_{X_u}(G_X, x)$ can be rewritten as the matrix equation $A.Q_{X_u} = B$ with (1) $a_{ij} = 1$ if $i = j$, and $a_{ij} = -\alpha\frac{W(x_i,x_j)}{\deg(x_i)}$ otherwise; (2) $b_i = (1 - \alpha)\mathcal{B}_u(x_i)$. This equation can be solved thanks to the Jaccobi method, as the convergence condition below is satisfied.

Proposition 1. The matrix A is strictly diagonally dominant.

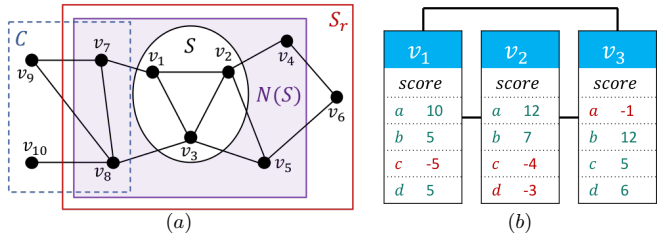


Fig. 3. A toy example that illustrates (a) the subgraph enumeration, and (b) the scores of terms in some vertices.

Proof 1. As $\sum_{i \neq j} |a_{ij}| = \alpha \sum_{i \neq j} \frac{W(x_i,x_j)}{\deg(x_i)} = \alpha < 1$ and $|a_{ii}| = 1$, we have $|a_{ii}| > \sum_{i \neq j} |a_{ij}|$. □

5 Computing Geolocated Events

We first present SIGLER-Cov, an algorithm that computes a set of patterns $\mathcal{R}_1$ whose quality $\mathcal{M}_u$ exceeds the threshold δ. In addition, $\mathcal{R}_1$ is a coverage guaranteed event summary (see Definitions 1 and 2). Then, we present SIGLER-Samp, a sampling event detection approach. Finally, we explain the post-processing step as introduced before.

5.1 Event Detection with Coverage Guarantee

We propose to extract user-driven geolocated events using a generate and test approach. It first enumerates a time interval I, and then explores the connected subgraphs S corresponding to areas where posts were sent during I. The quality measure $\mathcal{M}_u(S, I)$ evaluates whether the terms of the posts sent from $G_V[S]$ during I are characteristic of this time frame. As the number of such generated events is very large, especially the number of possible subgraphs, we propose several pruning techniques that makes the extraction feasible on real-life generated sets of posts.

Algorithm 1 enumerates all the intervals $[t_i, t_j]$ included in $T = [\![t_1, t_m]\!]$ thanks to the two loops in lines 2 and 4. For each interval I, it explores the connected subgraphs of $G_V[C]$ – the graph induced by the vertices for which there exists at least a term that bursts during the interval I – calling the function SGEnumerate presented in Algorithm 2. This backtracking algorithm uses two sets of vertices: $S \subseteq V$, the current enumerated subgraph induced by S, and $C \subseteq V \setminus S$ the vertices that are still to be explored. Initially, $S = \emptyset$ and C contains all the vertices for which there exists a bursting term in I. Algorithm 2 enumerates vertices from $C \cap N(S)$, with $N(S)$ the neighbors of vertices of S: $N(S) = \{v \in V \setminus S \mid \exists u \in S : (u, v) \in E\}$. Once the candidate set C is empty, P is tested to only be retained if it satisfies definitions 1 and 2 (lines 11 to 15). Fig. 3 (a) shows an example of sets S, $N(S)$ and C. In the next step of this example, SGEnumerate will choose v_7 or v_8 (in the intersection of C and $N(S)$) and add it to S. We can notice that some vertices do not belong to C (e.g., v_4) because they have already been enumerated and removed from C.

We use three pruning mechanisms without which the algorithm does not scale. The first one – the anti-monotonicity of $\mathcal{H}(P)$ – is used line 6 of Algorithm 1 to prune vertices given a time interval. This property is defined up to the intuitive partial order $\subseteq$: $P_1 \subseteq P_2$ if and only if $S_1 \subseteq S_2$ and $I_1 \subseteq I_2$.

Algorithm 1: SIGLER-Cov(δ, minCov, $\mathcal{R}_1$)

Input: δ the quality threshold, minCov the coverage threshold
Output: $\mathcal{R}_1$ a set of coverage guaranteed geolocated events

1 $\mathcal{R}_1 \leftarrow \emptyset$
2 **for** $i \leftarrow 1$ *to* m **do**
3 $C \leftarrow V$
4 **for** $j \leftarrow i$ *to* m **do**
5 $I \leftarrow [t_i, t_j]$
6 $C \leftarrow \{v \in C \mid \mathcal{H}(\{v\}, I) \neq \emptyset\}$
7 SGEnumerate(I, $\emptyset$, C,$\mathcal{R}_1$, δ, minCov)

Algorithm 2: SGEnumerate(I, S, C, $\mathcal{R}_1$, δ,minCov)

Input: I a time interval, S the current explored subgraph, C the candidate sets, δ the quality threshold, minCov the coverage threshold
Output: $\mathcal{R}_1$ the set of events

1 $P \leftarrow (S, I)$
2 **if** $C \cap N(S) \neq \emptyset$ **then**
3 **if** $UB(S \cup C, I, \mathcal{H}(P)) \geq \delta$ **then**
4 **for** $P_r \in \mathcal{R}_1$ **do**
5 **if** $LB(P, C, P_r) \geq minCov$ **then**
6 **return**
7 Choose a vertex $v \in C \cap N(S)$
8 SGEnumerate(I, $S \cup \{v\}$, $C \setminus \{v\}$, δ, minCov)
9 SGEnumerate(I, S, $C \setminus \{v\}$, δ, minCov)
10 **else**
11 **if** $\mathcal{M}_u(P) \geq \delta$ **then**
12 **for** $P_r \in \mathcal{R}_1$ **do**
13 **if** *cover*$(P, P_r) \geq minCov$ **then**
14 **return**
15 $\mathcal{R}_1 \leftarrow \mathcal{R}_1 \cup \{P\}$

Proposition 2 (anti-monotony of $\mathcal{H}(P)$). Considering two patterns $P_1 = (S_1, I_1)$ and $P_2 = (S_2, I_2)$, if $P_1 \subseteq P_2$ then $\mathcal{H}(P_2) \subseteq \mathcal{H}(P_1)$.

Proof 2. If $h \in \mathcal{H}(P_2)$, then $\forall v \in S_2, \forall t \in I_2$, score$(h, v, t) > 0$. As $S_1 \subseteq S_2$ and $I_1 \subseteq I_2$, thus, $\forall v \in S_1, \forall t \in I_1$, score$(h, v, t) > 0$, and thus $h \in \mathcal{H}(P_1)$. □

In Fig. 3 (b), the bursting terms of $S' = \{v_1\}$ are: $\mathcal{H}(S', t) = \{a, b, d\}$. If S' is expanded with a vertex x, the new set $\mathcal{H}(S' \cup \{x\}, t)$ is necessarily included in the previous one. For example, $\mathcal{H}(\{v_1, v_2\}, t) = \{a, b\}$.

The second pruning mechanism is based on the computation of an upper-bound of $\mathcal{M}_u(P)$ (used line 3 in Algorithm 2).

Definition 3 (Upper-bound on $\mathcal{M}_u$). Let $S \subseteq V$, $I \sqsubseteq T$, and $J \subseteq H$. UB is defined as:

$$UB(S, I, J) = \sum_{v \in S} \sum_{t \in I} \sum_{h \in J} \max\left(\text{score}(h, v, t) \times Q_u(h, v), 0\right)$$

We denote by $\Gamma(I, S, C)$ the set of all patterns that can be reached when expanding S by adding vertices from C: $\Gamma(I, S, C) = \{(S', I) \mid S' \subseteq S \cup C \text{ and } S \subseteq S'\}$, that is to say, the patterns that are generated from SGEnumerate(I, $S, C, \mathcal{R}_1, \delta,$ minCov). The following property states that $UB(S \cup C, I, \mathcal{H}(S, I))$ upper bounds $\mathcal{M}_u$ for all subsequent patterns:

Proposition 3. Let $S \subseteq V$, $I \sqsubseteq T$, and $C \subseteq V \setminus S$. For all patterns $P' = (S', I) \in \Gamma(I, S, C)$, $UB(S \cup C, I, \mathcal{H}(S, I)) \geq \mathcal{M}_u(S', I)$.

Proof 3. Since $S' \subseteq S \cup C$ and $\mathcal{H}(P') \subseteq \mathcal{H}(P)$, we have

$$
\begin{aligned}
&UB(S \cup C, I, \mathcal{H}(P)) = \\
&\sum_{v \in S'} \sum_{t \in I} \sum_{h \in \mathcal{H}(P')} \max\big(\text{score}(h, v, t) \times Q_u(h, v), 0\big) \\
&+ \sum_{v \in S \cup C} \sum_{t \in I} \sum_{h \in \mathcal{H}(P) \setminus \mathcal{H}(P')} \max\big(\text{score}(h, v, t) \times Q_u(h, v), 0\big) \\
&+ \sum_{v \in S \cup C \setminus S'} \sum_{t \in I} \sum_{h \in \mathcal{H}(P')} \max\big(\text{score}(h, v, t) \times Q_u(h, v), 0\big) \\
&\geq \sum_{v \in S'} \sum_{t \in I} \sum_{h \in \mathcal{H}(P')} \max\big(\text{score}(h, v, t) \times Q_u(h, v), 0\big) \\
&\geq \sum_{v \in S'} \sum_{t \in I} \sum_{h \in \mathcal{H}(P')} \text{score}(h, v, t) \times Q_u(h, v) = \mathcal{M}_u(P') \quad \square
\end{aligned}
$$

The last pruning technique is built on the coverage measure. As stated in Definition 2, there may exist several coverage guarantee summaries of geolocated events. Whereas it might be interesting to have a summary of smallest cardinality, the problem of finding the set of minimal size is NP hard. A practical approach consists in constructing the summary during the enumeration. We also use the coverage measure to prune large parts of the subgraph search space thanks to a lower bound LB (lines 4 to 6 in Algorithm 2) that will be defined next. The intuition behind is to prune a search space $\Gamma(I, S, C)$ if it is covered by an already found pattern P_r.

Proposition 4. Given a geolocated event pattern $P_r = (S_r, I_r)$ and a pattern $P' = (S', I)$ in $\Gamma(I, S, C)$, we have $\text{cov}(S', S_r) \geq \text{cov}(S \cup (C \setminus S_r), S_r)$.

Proof 4. As $S' = S \cup C'$ s.t $C' \subseteq C$, we have: $cov(S', S_r) = \frac{|S' \cap S_r|}{|S'|} = \frac{|S \cap S_r| + |C' \cap S_r|}{|S| + |C' \setminus S_r| + |C' \cap S_r|} \geq \frac{|S \cap S_r| + |C' \cap S_r|}{|S| + |C \setminus S_r| + |C' \cap S_r|}$. We define $g(x) = \frac{|S \cap S_r| + x}{|S| + |C \setminus S_r| + x}$, with $x = |C' \cap S_r|$. Knowing that the derivative $g'(x) = \frac{|S| + |C \setminus S_r| - |S \cap S_r|}{(|S| + |C \setminus S_r| + x)^2} \geq 0$, $g(x)$ takes its lower value when x is minimal, that is when $x = 0$. Thus $g(x) \geq \frac{|S \cap S_r|}{|S| + |C \setminus S_r|} = cov(S \cup (C \setminus S_r), S_r)$ and we conclude that $cov(S', S_r) \geq cov(S \cup (C \setminus S_r), S_r)$. $\square$

For instance, in Fig. 3, S_r covers S and a part of C. $\text{cov}(S \cup \{v_9, v_{10}\}, S_r) = \frac{3}{5}$ is a lower bound of $\text{cov}(S', S_r)$ for all $(S', I) \in \Gamma(I, S, C)$. In fact, $S \cup \{v_9, v_{10}\}$ contains only the vertices of C that are not in S_r, which minimizes the coverage of S_r.

Definition 4. Let $\mathcal{H}(P) \cap \mathcal{H}(Pr) = \{h_1, \ldots, h_q\}$ – the set of terms of P covered by those of P_r – be ordered by $h_i \leq_{UB} h_j$ iff $UB(S \cup C, I, \{h_i\}) \geq UB(S \cup C, I, \{h_j\})$. We consider the minimal set $\{h_1, \ldots, h_{q^\star}\}$ of terms that can be added to $\mathcal{H}(P) \setminus \mathcal{H}(P_r)$ while still satisfying the upper-bound:

$$q^\star = \underset{r \in \{1 \ldots q\}}{\text{argmin}}\, UB(S \cup C, I, \mathcal{H}(P) \setminus \mathcal{H}(P_r) \cup \{h_1, \cdots, h_r\}) \geq \delta$$

and define $H^\star$ as $\mathcal{H}(P) \setminus \mathcal{H}(P_r) \cup \{h_1, \cdots h_{q^\star}\}$. In other words, $\mathcal{H}^\star \subseteq \mathcal{H}(P)$ is the set of terms that overlaps the least with $\mathcal{H}(P_r)$ while verifying the condition $UB(S \cup C, I, \mathcal{H}^\star) \geq \delta$

Proposition 5. For each pattern $P' = (S', I) \in \Gamma(I, S, C)$ such that $\mathcal{M}_u(P') \geq \delta$, we have $\text{cov}(\mathcal{H}(P'), \mathcal{H}(P_r)) \geq \text{cov}(\mathcal{H}^\star, \mathcal{H}(P_r))$.

Proof 5. We know that $|\mathcal{H}(P') \cap \mathcal{H}(P_r)| \geq q^\star$, otherwise $UB(S', I, \mathcal{H}(P')) < \delta$ and $\mathcal{M}_u(P') < \delta$. We can rewrite $cov(\mathcal{H}(P'), \mathcal{H}(P_r)) = \frac{|\mathcal{H}(P') \cap \mathcal{H}(P_r)|}{|\mathcal{H}(P') \setminus \mathcal{H}(P_r)| + |\mathcal{H}(P') \cap \mathcal{H}(P_r)|} = \frac{q^\star + x}{|\mathcal{H}(P') \setminus \mathcal{H}(P_r)| + q^\star + x}$ with $x \geq 0$. Let's denote $g(x) = \frac{q^\star + x}{|\mathcal{H}(P) \setminus \mathcal{H}(P_r)| + q^\star + x}$. We have $cov(\mathcal{H}(P'), \mathcal{H}(P_r)) \geq g(x)$ as $\mathcal{H}(P') \subseteq \mathcal{H}(P)$. Knowing that the derivative $g'(x) = \frac{|\mathcal{H}(P) \setminus \mathcal{H}(P_r)|}{(|\mathcal{H}(P) \setminus \mathcal{H}(P_r)| + q^\star + x)^2} \geq 0$, then $g(x)$ takes its lower value when $x = 0$. Thus, $g(x) \geq \frac{q^\star}{|\mathcal{H}(P) \setminus \mathcal{H}(P_r)| + q^\star} = cov(\mathcal{H}^\star, \mathcal{H}(P_r))$ and we conclude the proof. $\square$

Definition 5. We define the function LB as

$$
\begin{aligned}
LB(P, C, P_r) \quad = \quad & \min\{\text{cov}(S \cup (C \setminus S_r), S_r), \text{cov}(I, I_r), \\
& \text{cov}(\mathcal{H}^\star, \mathcal{H}(P_r))\}
\end{aligned}
$$

Proposition 6. For each pattern $P' = (S', I) \in \Gamma(I, S, C)$ such that $\mathcal{M}_u(P') \geq \delta$, we have $\text{cover}(P', P_r) \geq LB(P, C, P_r)$.

Proof 6. It is sufficient to prove that : $cov(S', S_r) \geq cov(S \cup (C \setminus S_r), S_r)$ and $cov(\mathcal{H}(P'), \mathcal{H}(P_r)) \geq cov(\mathcal{H}^\star, \mathcal{H}(P_r))$ and $cov(I, I_r) \geq cov(I, I_r)$ The first two equations are respectively guaranteed with Propositions 4 and 5, and the third one is trivial. $\square$

The exploration of the search space $\Gamma(I, S, C)$ is stopped if there exists $P_r \in \mathcal{R}_1$ such that $LB(P, C, P_r) \geq minCov$, because all the subsequent patterns are covered by it.

5.2 Pattern Sampling Based Event Detection

In this section, we explore another approach to discover geolocated events: Pattern sampling [32]. Given a time budget, the proposed algorithm `SIGLER-Samp` mines events using a random exploration of the search space that favors events with high $\mathcal{M}_u$ values. Such an approach enables instant mining which is required in interactive pattern mining.

The pattern sampling process we consider is based on a random walk on a graph whose vertices are patterns $P = (S, I)$ and edges (transitions) are chosen following a probability measure that overweights high quality patterns. The random walk starts from a singleton pattern $P = (S, I)$ where $S = \{v\}$ and $I = [t, t]$. Next, P is expanded by adding randomly drawn vertices from $N(S)$, the neighborhood of S, or by adding a timestamp to I. The random walk is basically composed of two main steps described in Algorithm 3:

1) $\mathcal{M}_u(P)$ is computed for each pattern $P = (S, I)$, where $S = \{v\}$, $v \in V$ and $I = [t, t]$, $t \in T$. The probability of drawing a singleton pattern P is defined as $\mathcal{P}(P) = \frac{\mathcal{M}_u(P)}{\sum_{v' \in V, t' \in T} \mathcal{M}_u(\{v'\}, [t', t'])}$
2) From a current pattern $P = (S, I)$ with $I = [t_i, t_j]$, the next step consists to draw a pattern P' from the set:

$$
\begin{aligned}
\text{Next}(P) \quad = \quad & \{(S, I)\} \cup \{(S, [t_{i-1}, t_j])\} \cup \{(S, [t_i, t_{j+1}])\} \\
\cup \quad & \{(S', I) \mid S' = S \cup \{v\},\ v \in N(S)\}
\end{aligned}
$$

This is done based on $\mathcal{P}(P'|P)$, the probability to reach the pattern $P' \in \text{Next}(P)$ from P: $\mathcal{P}(P'|P) = \frac{\mathcal{M}_u(P')}{\sum_{P_2 \in \text{Next}_{(P)}} \mathcal{M}_u(P_2)}$.
This distribution of probabilities rewards transitions toward patterns P' with large $\mathcal{M}_u(P')$ values. After drawing P', if $P' \neq P$, we continue the expansion by repeating Step 2 using the new pattern P'. If $P' = P$, the pattern P is returned to the user and the sampling is repeated from Step 1 until the whole consumption of the time budget.

The two main steps are repeated (lines 8 to 21) until $P' = P$. At each iteration, the pattern $P = (S, I)$ is extended by adding a vertex to S or a timestamp to I. Since S and I are respectively bounded by V and T, this loop necessarily stops after at most $|V| + |I|$ iterations. All patterns with nonzero $\mathcal{M}_u$ value have a non zero probability to be generated.

Proposition 7. For each pattern $P = (S, I)$, if $\mathcal{M}_u(P) > 0$ then $\mathcal{P}(P \in \mathcal{R}_2) > 0$

Proof 7. Let us prove it by induction on $n = |S| + |I|$.

- For $n = 2$, P is such that $|S| = 1$ and $|I| = 1$ (in other cases, $\mathcal{M}_u(P) = 0$). P can be drawn in the first step, and if it is chosen from Next(P) in step 2, it is added to $\mathcal{R}_2$. Thus, $\mathcal{P}(P \in \mathcal{R}_2) \geq \frac{\mathcal{M}_u(P)}{\sum_{v' \in V, t' \in T} \mathcal{M}_u(\{v'\},[t',t'])} \times \frac{\mathcal{M}_u(P)}{\sum_{P_2 \in \text{Next}_{(P)}} \mathcal{M}_u(P_2)} > 0$.
- Let us suppose that the proposition is true for n. Let $P = (S, I)$ be a pattern such that $|S| + |I| = n + 1$ and $\mathcal{M}_u(P) > 0$. Let $P' = (S', I') \subseteq P$ be a pattern containing one less vertex or one less timestamp than P. This means that $|S'|+|I'| = n$, by the recursion hypothesis we have $\mathcal{P}(P' \in \mathcal{R}_2) > 0$. Thus, the probability $\mathcal{P}(P')$ to reach P' is not null. Since P can be reached from P' during the random walk, then: $\mathcal{P}(P \in \mathcal{R}_2) \geq \mathcal{P}(P') \times \frac{\mathcal{M}_u(P)}{\sum_{P_2 \in \text{Next}_{(P')}} \mathcal{M}_u(P_2)} \times \frac{\mathcal{M}_u(P)}{\sum_{P_2 \in \text{Next}_{(P)}} \mathcal{M}_u(P_2)}$
$\mathcal{P}(P \in \mathcal{R}_2) > 0$ □

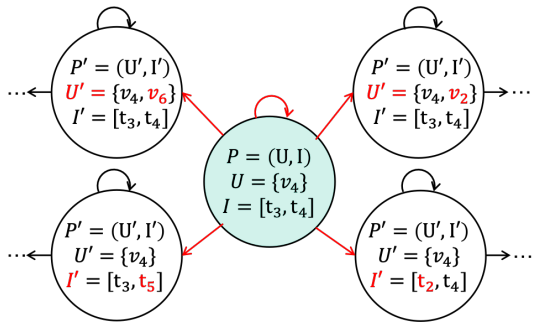


Fig. 4. A step of SIGLER-Samp: P' are the patterns that can be generated from the current P when considering the graph of Fig. 3.

Fig. 4 gives an example of an iteration in SIGLER-Samp. The current generated pattern is $P = (\{v_4\}, [t_3, t_4])$. In the next iteration, one of the neighbors of P, or P, is randomly chosen. The neighbors are generated by either adding $\{v_2\}$ or $\{v_6\}$, vertices connected to v_4 in the graph of Fig. 3, or increasing the time interval.

Algorithm 3: SIGLER-Samp

Input: time_budget
Output: $\mathcal{R}_2$ a set of sampled patterns

1 **for** $v \in V, t \in T$ **do**
2 compute $\mathcal{M}_u(\{v\}, [t, t])$
3 **while** *current_time* $<$ time_budget **do**
4 // Step 1: draw a singleton pattern P
5 draw $P = (\{v\}, [t, t]) \sim \frac{\mathcal{M}_u(P)}{\sum_{v' \in V, t' \in T} \mathcal{M}_u(\{v'\},[t',t'])}$
6 // Step 2: expansion of P
7 $P' \leftarrow P$
8 **repeat**
9 $P \leftarrow P'$
10 // Compute the set Next(P) for $P = (S, [t_i, t_j])$
11 Next$(P) \leftarrow \{P\}$
12 **for** $v \in N(S)$ **do**
13 Next$(P) \leftarrow$ Next$(P) \cup \{(S \cup \{v\}, [t_i, t_j])\}$
14 **if** $i > 1$ **then**
15 Next$(P) \leftarrow$ Next$(P) \cup \{(S, [t_{i-1}, t_j])\}$
16 **if** $j < m$ **then**
17 Next$(P) \leftarrow$ Next$(P) \cup \{(S, [t_i, t_{j+1}])\}$
18 **for** $P' \in$ *Next*(P) **do**
19 compute $\mathcal{M}_u(P')$
20 draw $P' \sim \frac{\mathcal{M}_u(P')}{\sum_{P_2 \in \text{Next}_{(P)}} \mathcal{M}_u(P_2)}$
21 **until** $P' = P$;
22 $\mathcal{R}_2 \leftarrow \mathcal{R}_2 \cup P$

5.3 Discussion

We discuss here some potential issues that may appear, and the related post-processing to fix them. By applying one of the aforementioned algorithms, we find the set of patterns $\mathcal{R}_1$ or $\mathcal{R}_2$, let's denote it $\mathcal{R}_*$. In some particular cases, two real life events happen at the same time and the same location. However, these two events will be merged in the same pattern $P \in \mathcal{R}_*$. It is important to separate them before displaying the result to the end user. Therefore, for each pattern $P = (S, I, \mathcal{H}(S, I)) \in \mathcal{R}_*$, we apply a community detection algorithm on the terms $\mathcal{H}(S, I)$ in order to partition them into groups $K_1, ..., K_d$ where (1) $\forall i \in [\![1, d]\!] : K_i \subseteq \mathcal{H}(S, I)$ (2) $\cup_i K_i = \mathcal{H}(S, I)$ (3) each K_i corresponds to a single real life event. We use Louvain community detection algorithm [39], and we express its result by the function $Louvain_1(P) = \{K_1, ..., K_d\}$. The similarity measure sim_1 used in this clustering is defined for two terms $h_1, h_2 \in \mathcal{H}(S, I)$ w.r.t the pattern P:

$$sim_1(h_1, h_2, P) = \begin{cases} 1, \text{if } |\{b \in B \mid (\{h_1, h_2\} \subseteq b.terms) \text{ and} \\ \quad (\text{area}(b.loc) \in S) \text{ and } (b.time \in I)\}| \geq 1 \\ 0, \text{otherwise} \end{cases}$$

In other words, $sim_1(h_1, h_2, P) = 1$ if h_1 and h_2 co-occur at least once in the space S and the time interval I. Thus, each cluster $K_i \in Louvain_1(P)$ would be a set of terms that co-occur in this space-time. Each cluster K_i gives a pattern (S, I, K_i) with the same space-time than the current P. After applying this clustering to each $P \in \mathcal{R}_*$, we have the post-processed result $R' = \cup_{(S,I,\mathcal{H}(S,I)) \in \mathcal{R}_*} \{(S, I, K) \mid K \in Louvain_1(S, I, \mathcal{H}(S, I))\}$

In order to deal with the redundancy issue, we have defined SIGLER-Cov that computes a summary $\mathcal{R}_1$. However, it is not necessarily the optimal summary, that is to say the summary of minimal size whose events partially cover

each geolocated events that does not belong to the summary. Indeed, the computation of an optimal summary is NP hard. Thus, some redundancy can still remain in the result, we post-process the set R' to fix this issue. We apply Louvain algorithm on patterns $P \in R'$ to merge the ones with similar location, time interval, and terms. We define a similarity measure sim_2 for two patterns $P = (S, I, K)$ and $P' = (S', I', K')$: $sim_2(P, P') = \frac{|S \cap S'|}{|S \cup S'|} \times \frac{|I \cap I'|}{|I \cup I'|} \times \frac{|K \cap K'|}{|K \cup K'|}$. The result will be the communities: $\mathcal{C}_1, ..., \mathcal{C}_l$ where each community $\mathcal{C}_i \subseteq R'$ is a set of similar patterns. From each community $\mathcal{C}_i$ we reconstitute a pattern $P_{\mathcal{C}_i} = \cup_{(S,I,K) \in \mathcal{C}_i}(S, I, K)$. The final result set of pattern is: $R'' = \{P_{\mathcal{C}_1}, ..., P_{\mathcal{C}_l}\}$.

6 Experiments

In this section, we report our experimental results. We start by describing the real-world datasets we use, as well as the questions we aim to answer. Then, we provide a thorough comparison with the state-of-the-art algorithms and we report a performance study. Eventually, we evaluate the ability of our approach to take user interests into account through different testbeds[4].

SIGLER is implemented in C++ and the experiments were executed on a machine equipped with i7 CPU @ 2.5GHz, and 16GB main memory, running macOS Sierra version 10.12.2. For reproducibility purposes, the source code and the data are available[5].

6.1 Experimental Setting

Experiments are performed on 3 real-world datasets that contain the tweets obtained by querying three different cities on Twitter: New York (NYC), Los Angeles (LA) and London. For each city, we collected geolocated public tweets and removed those produced by bots (i.e, accounts that produce more than 100 tweets in a period of 10 days). We only retained tweets containing hashtags or user mentions. The main characteristics of these datasets are given in Table 2.

dataset	starting date	ending date	# tweets	# distinct terms
New York	2016/10/08	2017/01/07	652,244	332,618
Los Angeles	2017/05/17	2017/07/27	353,541	224,769
London	2017/05/17	J 2017/07/27	270,648	177,166

TABLE 2
Description of the real-world datasets.

This empirical study aims to answer the following questions: Are SIGLER-Cov and SIGLER-Samp more effective and efficient than their competitors? Do they scale well according to the size of the dataset and the different parameters? Does SIGLER-Samp capture all the events? Is the approach able to make use of user feedback to discover user-relevant events?

In the first bunch of experiments, we compare our approach with two local event detection approaches: (1) Multi-scale Event Detection algorithm (MED) [5], a state-of-the-art algorithm which aims to identify geolocated events based on a wavelet analysis of time series of terms, (2) GeoBurst [13], an online local event detection method, that first detects geo-topic clusters using a random walk on a keyword co-occurrence graph, and then ranks all the clusters with a weighted combination of spatial and temporal burstiness. We also considered INSIGHT [18] and EDCoW [19], two non location-aware event detection methods. INSIGHT is one of the best methods to detect events in tweets, as it won a recent challenge [41].

These experiments show that (1) non location-aware approaches are not appropriate to detect geolocated events, and (2) MED and GeoBurst algorithms are less robust to noise than our approaches, and encounter scalability issue. Finally, we demonstrate the ability of our methods to extract interesting user-driven events via experiments performed on a crowdsourcing platform.

6.2 Effectiveness

We first study the ability of our approach to detect user-driven geolocated events. Using the method described in [5], we generate artificial datasets for which the ground-truth geolocated events are known. Twenty events, denoted hereafter R_0, are artificially created. Each of them lasts between 2 and 8 timestamps, involves from 2 to 16 vertices and is defined by 10 unique terms. The datasets contain 1024 vertices, 32 timestamps and 1200 unique terms. Posts are artificially produced and sent at different timestamps and spatial locations. Each post contains between 9 to 13 terms. These posts are either related to embedded events, or are randomly drawn: (1) Event-related posts are uniformly distributed over the vertices and times stamps of its associated event and contain 5 of the 10 event-related terms as well as between 4 to 8 other terms; (2) Non-event related posts are uniformly distributed over the other timestamps and vertices. The terms they contain are drawn using a Zipf law probability distribution [42] among the 1000 non-event related terms. 10 to 50 event-related posts are randomly drawn and the number of non-event related posts is controlled by the $noise\_rate$ parameter.

Let $R = \{e_1, \ldots, e_k\}$ be the set of discovered events. The quality of R is assessed based on the following adapted $Fscore$ measure:

$$Fscore(R, R_0) = 2 \times \frac{Precision(R, R_0) \times Recall(R, R_0)}{Precision(R, R_0) + Recall(R, R_0)}$$

with $Precision(R, R_0) = \frac{\sum_{e \in R} max_{e' \in R_0} \text{COV}(e, e')}{|R|}$ and $Recall(R, R_0) = \frac{\sum_{e \in R_0} max_{e' \in R} \text{COV}(e, e')}{|R_0|}$.

Fig. 5 presents the $Fscore$ values achieved by the different approaches when noise rate is varying. From this figure we can draw the following conclusions:

1) *Non location-aware event detection approaches cannot detect geolocated events*, as INSIGHT and EDCoW $Fscore$ is always lower than 0.2.
2) SIGLER, MED and GeoBurst perform well with low noise rate. However, *both SIGLER-Cov and SIGLER-Samp are more robust to noise than MED and GeoBurst* . The $Fscore$ of our approaches remains high even when the noise rate increases, whereas MED and

4. We report experiments performed on a crowd-sourcing platform with real-users in this paper. Additional experiments with virtual users are reported in supplementary material.

5. https://tinyurl.com/yywy2fqc

GeoBurst $Fscore$ decreases almost linearly. More precisely, MED and GeoBurst keep a good recall but the precision decreases when the noise rate increases.

3) Furthermore, the $Fscore$ of SIGLER-Samp is obviously bounded by the $Fscore$ obtained by SIGLER-Cov which adopts a more exhaustive exploration of the search space. Nevertheless, the bigger the time budget, the better the $Fscore$.

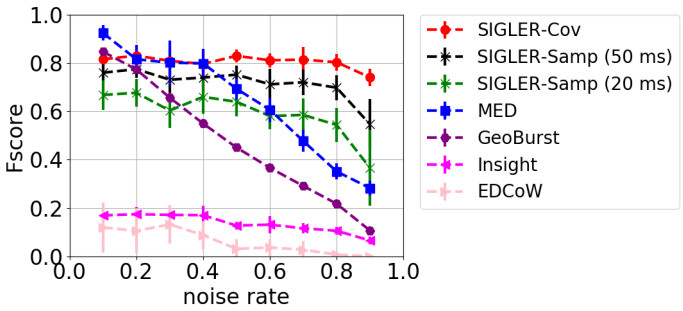


Fig. 5. Average and error bars of $Fscore$ values obtained over 10 generated datasets for a given noise rate value ($\delta = 5$ and minCov $= 0.8$). Non location-aware event detection approaches are not able to detect geolocated events. MED fails in presence of noise. Runtime of SIGLER-Cov increases from 10ms to 2.5s when the noise rate increases.

We also study the impact of parameters of our approach on the value of $Fscore$. To this aim, we variate minCov and δ and we show the value of $Fscore$ in Fig 6. The highest $FScore$ is achieved with values of δ between 1 and 2, this leads to the best trade-off between precision and recall. Increasing δ allows to increase the precision, but decreases the recall. Although the value of $Fscore$ slightly increases when minCov is higher, this parameter does not significantly impact the quality of the result.

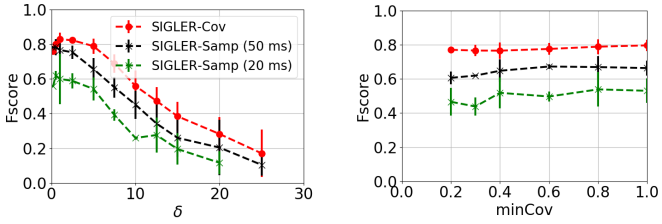


Fig. 6. Average and error bars of $Fscore$ obtained according to δ and minCov. Default values are: $noiseRate = 0.8$, minCov $= 0.8$, $\delta = 5$.

Ideally, one would prefer to perform the effectiveness study based on real world dataset. However, we do not have the ground truth of such data. This makes it very hard to achieve an objective comparison on them. Nevertheless, in supplementary materials, we display and compare the top 10 events returned by SIGLER-Cov, MED and GeoBurst in the New York dataset. We show that there are several events that are identified by both SIGLER-Cov and MED. We also show that GeoBurst provides events that are much smaller than the ones of SIGLER-Cov and MED. We also discuss the quality of these top 10 events. MED and GeoBurst tend to detect false positive events.

6.3 Efficiency

To evaluate the scalability of the algorithms, we consider New York tweets, our largest dataset. Fig. 7 reports the runtime and the number of events obtained by SIGLER[6], MED, and GeoBurst when dataset parameters are varying. However, MED is only reported for at most 10,000 tweets because of its scalability issues (in the figures at left).

MED, GeoBurst, and SIGLER-Cov discover a comparable number of events but MED performances raise scalability issues. Indeed, SIGLER-Cov outperforms MED with several orders of magnitude for all the configurations, especially when the number of tweets increases. Even if MED uses some indexing techniques, its computational complexity is quadratic in the number of tweets, and MED fails to handle large datasets. Although MED is able to process one day of tweets in our experiments, it is without considering the fact that the Twitter API gives access to less than 1% of the posted tweets. MED scale limitation is therefore a real issue on these data.

In Fig. 7 - right, we report the runtime and the number of discovered events with higher numbers of tweets (we consider the whole dataset). We observe that the execution time of SIGLER-Cov increases with the number of tweets, but there is no order of magnitude change (non-logarithmic scale). Although GeoBurst runtime also increases linearly, it is considerably higher than the one of SIGLER-Cov.

We also study the impact of number of vertices and time granularity on our methods. We show their behavior in Fig. 8 when these two dimensions are varying. The execution time of SIGLER-Cov increases when the number of vertices and time granularity increase. The execution time of SIGLER-Samp is controlled by a parameter and we can observe that its number of results tends to the one of SIGLER-Cov when the time budget increases.

To go further on evaluating the discovered events, Fig. 9 investigates the ability of SIGLER-Samp to capture similar events as SIGLER-Cov, that is to say it verifies that the computed event sample well covers all the events obtained with the exhaustive approach. To this end, we compare the events provided by SIGLER-Samp (denoted R_1) with the events discovered by SIGLER-Cov (denoted R_2) as follows. Using the Jaccard similarity measure of two patterns $P1 = (S_1, I_1)$ and $P_2 = (S_2, I_2)$, $J(P_1, P_2) = \frac{1}{3} \times \left(\frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} + \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|} + \frac{|H_{P1} \cap H_{P2}|}{|H_{P1} \cup H_{P2}|}\right)$, the similarity of R_1 and R_2 is defined by $Sim(R_1, R_2) = \frac{\sum_{P_1 \in R_1} \max_{P_2 \in R_2} J(P_1, P_2)}{|R_1|+|R_2|} + \frac{\sum_{P_2 \in R_2} \max_{P_1 \in R_1} J(P_1, P_2)}{|R_1|+|R_2|}$. We executed SIGLER-Samp with different time budgets and post-processed the result R_2 by removing redundant patterns (using minCov $= 0.8$) and low quality ones (using $\delta = 40$). Fig. 9 reports the Sim values with respect to SIGLER-Samp time budget for 300K tweets (left) and the whole dataset (right). The runtimes of SIGLER-Cov for these two cases are respectively 95s and 173s. With a time budget fixed to 11% of the execution time of SIGLER-Cov (around 9s and 19s), SIGLER-Samp retrieves most of the high-quality patterns ($sim > 0.9$).

6. The runtime for SIGLER corresponds to the complete process including the post-processing step described in Section 5.3. This explains the slight variation of the runtime of SIGLER-Samp for different configurations.

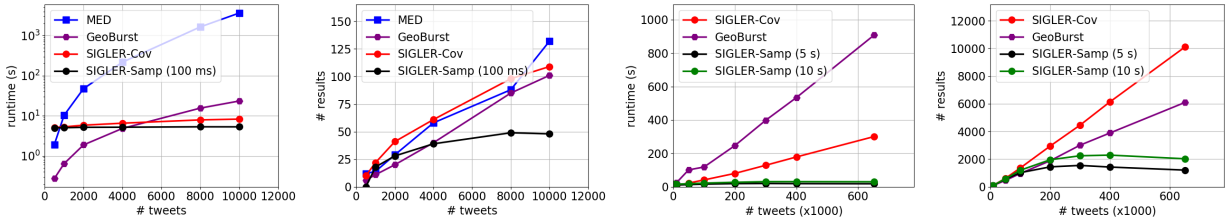


Fig. 7. Runtime and number of events of SIGLER−Cov, SIGLER−Samp, MED and GeoBurst when varying the number of tweets (default values of SIGLERs: 2000 vertices, $\Delta t = 3h$, $\delta = 10$ and minCov $= 0.8$).

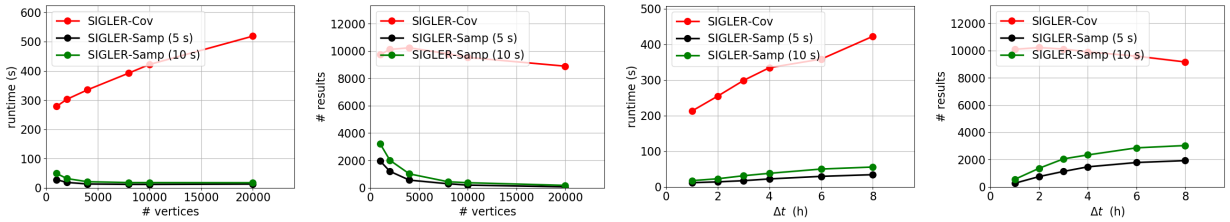


Fig. 8. Number of events by SIGLER−Cov and SIGLER−Samp and runtime of SIGLER−Cov according to the number of vertices, and the time granularity when considering the whole dataset (default values: 652k tweets, 2000 vertices, $\Delta t = 3h$, $\delta = 10$ and minCov $= 0.8$).

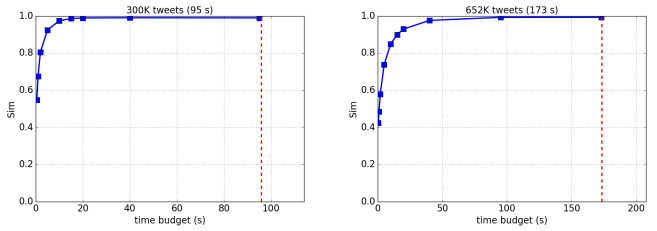


Fig. 9. Similarity of the results of SIGLER−Cov with the ones of SIGLER−Samp with respect to SIGLER−Samp time budget.

Finally, Fig. 10 evaluates the impact of the two main parameters, minCov and δ, on the results. minCov is a very intuitive parameter that eliminates a pattern if it is covered at least by minCov% of a pattern belonging to the solution. When minCov=1, only non maximal patterns are removed, and the more minCov decreases, the more disjoint the patterns. From Fig. 10 we can observe that this parameter has also a major impact on the execution time. Indeed, this parameter is involved in the computation of the upper-bounds and when large, it drastically reduces the execution time. In our experiments, we set this parameter to 0.8 to remove highly redundant events while allowing some intersections.

The quality of an event is evaluated by the measure $\mathcal{M}$, also used to rank the patterns when presented to the user (see next subsections). The function of the parameter δ is to cut the tail of the pattern distribution in order to only keep those of high quality. So the larger δ, the smaller the number of patterns and the faster the execution.

For the following experiments, we fixed the value of δ according to the number of events that we wanted to present to the user. Thus, we fixed δ value so as to have around 800 events for NYC, which contains 3 months of tweets, and around 400 events for LA and London that contain 70 days of tweets. This led us to set $\delta = 40$ for NYC, and $\delta = 15$ for London and LA dataset.

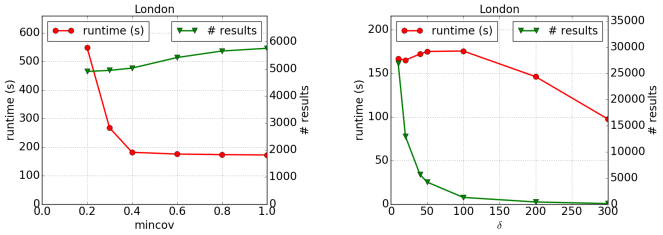


Fig. 10. Runtime and number of discovered events by SIGLER−Cov according to minCov and δ (default value minCov=0.8, $\delta = 40$).

6.4 User-driven discovery of geo-located events

To evaluate the ability of SIGLER to take benefit of user feedback, we performed interactive event detection process with real users on real datasets. We used a crowdsourcing platform – Figure Eight[7]– to hire people living in the country where the data is located. Indeed, our user feedback requires some expertise about the city and its events. To this end, we developed a graphical application[8] and deployed it on Figure Eight. For each user, the process consists in several iterations. At each of them, a batch containing the tweets emitted for 6 consecutive days is given as input to the algorithms (with a recovery of 3 days between 2 batches). Geolocated events are computed using either $\mathcal{M}$ (data-driven detection), or $\mathcal{M}_u$ (user-driven detection). Then, the user is asked to mark the events that she likes. Finally, the batch index is incremented and the process iterates. Since evaluating with the overall datasets can be very long and

7. https://www.figure-eight.com
8. 134.214.104.134:6001 and 134.214.104.134:6002.

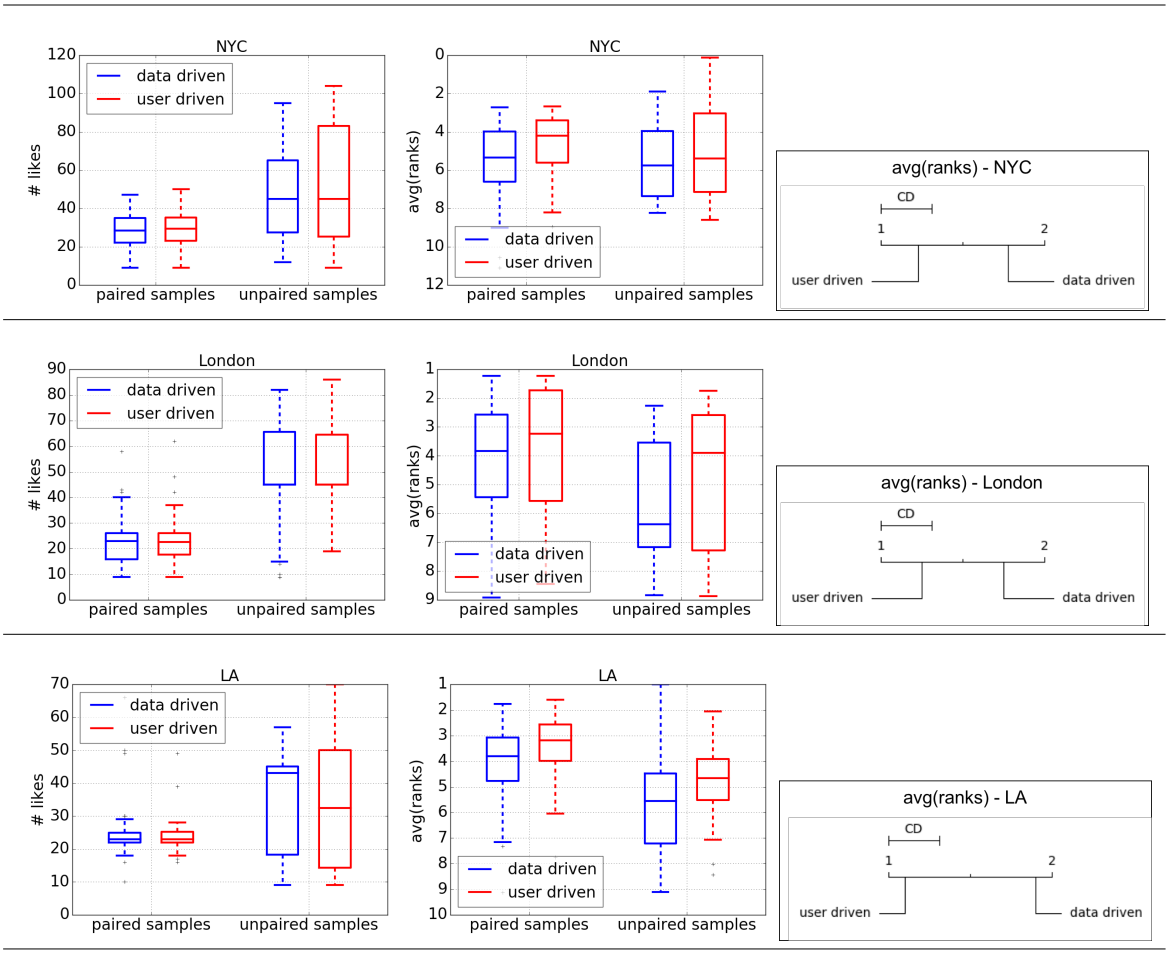


Fig. 11. Ability to take user feedback into account with real users: number of likes (left), average ranks of liked events (center), Nemenyi tests on average ranks (right) for paired samples.

exhausting for real users, we limited the number of batches to 10, that is approximately 33 days of data.

We used two different settings[9] to compare the data-driven and the user-driven approaches. In the *paired sample* (also referred to as comparative evaluation), each of the 40 participants evaluated both approaches in a blind way, i.e. the two lists of events were randomly displayed to the user. In the *unpaired sample* (also known as independent evaluation), 60 participants evaluated either the data or the user-driven approach while not being aware of the type of method used.

Fig. 11 reports the results of this crowdsourcing-based evaluation. For the paired sample, the number of likes is greater or equal in the user-driven setting than in the data-driven one, while results are less obvious for the unpaired sample. The purpose of paired samples is to get better statistics by controlling for the effects of other "unwanted" variables. And so, as our sample sizes are quite small, results obtained on paired samples are probably the most reliable. In addition, the test of Wilcoxon [43] is applicable on the *paired sample*, while it is not on the *unpaired* ones. However, even for paired samples, the Wilcoxon test does not allow to reject the null hypothesis "the number of likes are similar", and the difference is not considered as significant. Considering the average ranks of liked events, we can observe that they are always better in the user-driven configuration than in the data-driven one. The difference in the values is considered as significant by both Wilcoxon and Nemenyi tests [43]. The Nemenyi test value is shown in Fig. 11: when the rank difference on the graduated line is greater than the CD value, the rank difference is considered not due to chance.

Thus, this experience with real users leads to nuance the claim that the user-driven approach makes it possible to identify more interesting events than the data-driven one. However, it confirms that the identified events are of much better quality for the user-driven setting than for the data-driven one. Similar experiments with simulated users are reported in supplemental material.

6.5 Illustrative results

Finally, we show some examples of events detected by our approach. Fig. 12 reports the top 3 events detected in New York, London, and Los Angeles datasets. Each event is described by the locations, time interval and top 8 most frequent terms of its related tweets.

The first event in New York is the Comic Con[10] , which

9. The evaluation frameworks are available at: (1) for the paired sample: 134.214.104.134:6002, (2) for the umpaired sample: 134.214.104.134:6001

10. https://goo.gl/BR7kgp

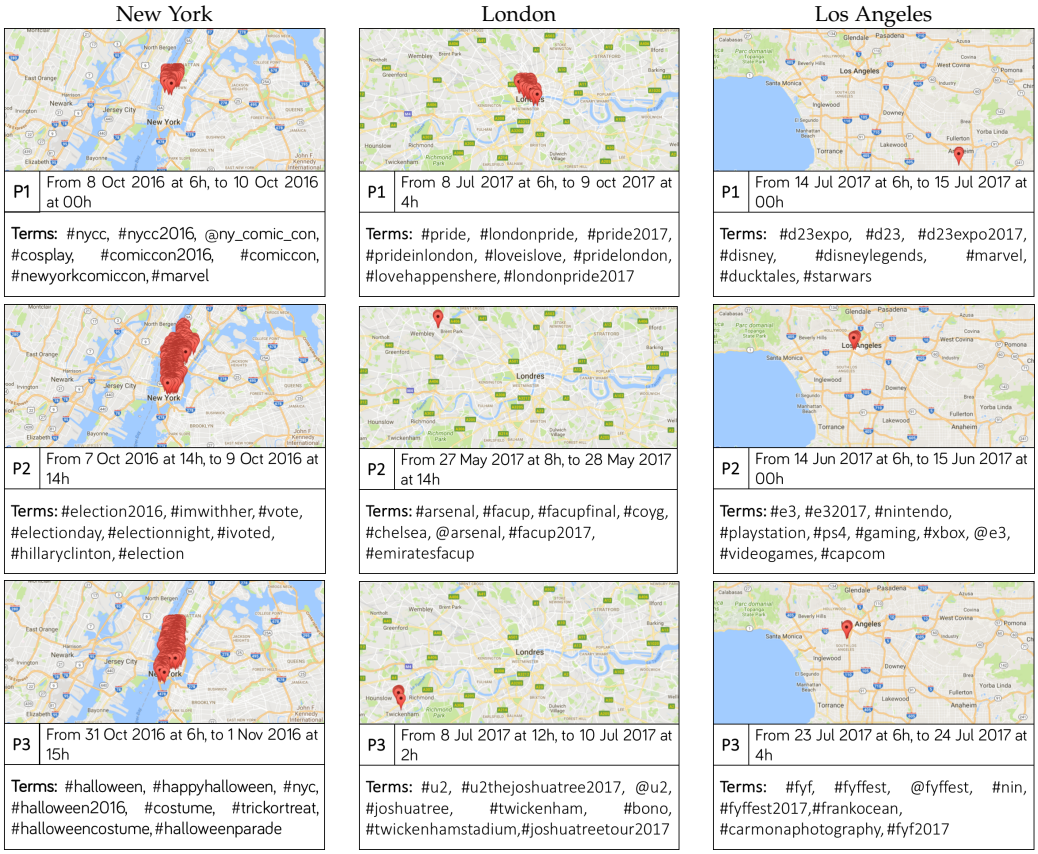


Fig. 12. Top 3 events detected in New York (left), London (center), and Los Angeles (right).

is an annual convention mainly dedicated to comics. It was organized in Jacob K. Javits Convention Center, the location associated to the related tweets. The two other events correspond to USA presidential elections, and the celebration of Halloween. In London, the first event is a Pride Parade organized in July 8th, 2017. The remaining are two geolocated events corresponding to a soccer event and a concert of U2. In Los Angeles, the top 3 events are respectively: the Official Disney Fan Club, E3[11] (a video game related event), and the FYF Fest[12] (a music festival).

7 Conclusion

In this paper, we introduced the novel problem of user-driven geolocated event discovery in social media. We handled the discovery of data-driven and user-driven event detection in a unified view. We designed two different algorithms to efficiently and effectively discover events based on geolocation and user feedback. Experiments demonstrate that, in the data-driven setting, our approach outperforms state of the art methods by a factor of two to several orders of magnitude. Furthermore, our approach is more robust to noise. We also provide evidence that non location-aware event detection approaches fail to discover geolocated events. Thus, user feedback cannot be considered by post-processing the events obtained by existing approaches. We also showed the ability of our method to directly discover geolocated events that are of interest for the user based on her feedback with crowdsourcing-based experiments. We believe that this work opens new directions for future research. For example, the event detection can be enhanced by thoroughly taking into account prior knowledge to detect really unexpected events and better propagate user feedback to the semantic neighborhood of liked events. Another interesting direction is to learn an explicit model of user interests and provide active learning based heuristics to foster the interactive process.

11. https://goo.gl/pio4dS
12. https://goo.gl/k1yWmK

Acknowledgment

This work was supported by Group Image Mining (GIM) which joins researchers of THALES Group and LIRIS Lab, and by the ACADEMICS grant of the IDEXLYON, project of the Université of Lyon, PIA operated by ANR-16-IDEX-0005. The authors would like to warmly thank the reviewers for their valuable remarks, and Adnene Belfodil and Aimene Belfodil for interesting discussions.

References

[1] H. Xiao, P. Rozenshtein, and A. Gionis, "Discovering topically and temporally coherent events in interaction networks," in *ECMLP-KDD*, 2016.
[2] V. Carchiolo, A. Longheu, and M. Malgeri, "Using twitter data and sentiment analysis to study diseases dynamics," in *ITBAM 2015*, 2015, pp. 16–24.
[3] S. Asur and B.-A. Huberman, "Predicting the future with social media," in *WI-IAT*. IEEE Computer Society, 2010, pp. 492–499.

[4] F. Chen and D. B. Neill, "Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs," in *KDD '14*. New York, NY, USA: ACM, 2014, pp. 1166–1175.

[5] X. Dong, D. Mavroeidis, F. Calabrese, and P. Frossard, "Multiscale event detection in social media," *Dami*, vol. 29, no. 5, pp. 1374–1405, 2015.

[6] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang, "Tedas: A twitter-based event detection and analysis system," in *ICDE'12*, 2012, pp. 1273–1276.

[7] J.-M. Kleinberg, "Bursty and hierarchical structure in streams," in *KDD*, 2002, pp. 91–101.

[8] C. Zhang *et al.*, "Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams," in *ACM SIGKDD*, 2017, pp. 595–604.

[9] M. Akbari, X. Hu, L. Nie, and T.-S. Chua, "From tweets to wellness: Wellness event detection from twitter streams." in *AAAI*. AAAI Press, 2016, pp. 87–93.

[10] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on twitter," in *ICWSM*, 2011.

[11] Y. Wei, L. Singh, B. Gallagher, and D. Buttler, "Overlapping target event and story line detection of online newspaper articles," in *DSAA 2016*, 2016, pp. 222–232.

[12] T. Sakaki, M. Okazaki, and Y. Matsuo, "Earthquake shakes twitter users: real-time event detection by social sensors," in *WWW*, 2010, pp. 851–860.

[13] C. Zhang *et al.*, "Geoburst: Real-time local event detection in geo-tagged tweet streams," in *ACM SIGIR*, 2016, pp. 513–522.

[14] T. Hua *et al.*, "STED: semi-supervised targeted-interest event detectionin in twitter," in *ACM SIGKDD*, 2013, pp. 1466–1469.

[15] ——, "Automatic targeted-domain spatiotemporal event detection in twitter," *GeoInformatica*, vol. 20, no. 4, pp. 765–795, 2016.

[16] Q. He, K. Chang, and E.-P. Lim, "Analyzing feature trajectories for event detection," in *SIGIR*. ACM, 2007, pp. 207–214.

[17] L. M. Aiello *et al.*, "Sensing trending topics in twitter," *Trans. Multi.*, vol. 15, no. 6, pp. 1268–1282, 2013.

[18] G. Ifrim, B. Shi, and I. Brigadir, "Event detection in twitter using aggressive filtering and hierarchical tweet clustering," in *snow@WWW*, 2014, pp. 33–40.

[19] J. Weng and B.-S. Lee, "Event detection in twitter," in *ICWSM*, 2011.

[20] C.-C. Aggarwal and K. Subbian, "Event detection in social streams," in *SIAM DM*, 2012, pp. 624–635.

[21] P. Houdyer *et al.*, "Gazouille: Detecting and illustrating local events from geolocalized social media streams," in *ECML PKDD 2015*, 2015, pp. 276–280.

[22] M. Walther and M. Kaisser, "Geo-spatial event detection in the twitter stream," in *European Conference on Advances in IR*, 2013, pp. 356–367.

[23] K. Watanabe, M. Ochi, M. Okabe, and R. Onai, "Jasmine: A real-time local-event detection system based on geolocation information propagated to microblogs," in *CIKM*, 2011, pp. 2541–2544.

[24] D. Garcia-Gasulla *et al.*, "Social network data analysis for event detection," in *ECAI*, 2014, pp. 1009–1010.

[25] H. Abdelhaq, C. Sengstock, and M. Gertz, "Eventweet: Online localized event detection from twitter," *PVLDB*, vol. 6, no. 12, pp. 1326–1329, 2013.

[26] L. Chen and A. Roy, "Event detection from flickr data through wavelet-based spatial analysis," in *CIKM*, 2009, pp. 523–532.

[27] M. Atzmueller, "Subgroup discovery," *Data Mining and Knowledge Discovery*, vol. 5, no. 1, pp. 35–49, 2015.

[28] W. Duivesteijn, A. J. Feelders, and A. Knobbe, "Exceptional model mining," *Data Mining and Knowledge Discovery*, vol. 30, no. 1, pp. 47–98, 2016.

[29] A. Silberschatz and A. Tuzhilin, "On subjective measures of interestingness in knowledge discovery," in *KDD*, 1995, pp. 275–281.

[30] T. de Bie, "An information theoretic framework for data mining," in *KDD*, 2011, pp. 564–572.

[31] V. Dzyuba and M. van Leeuwen, "Interactive discovery of interesting subgroup sets," in *IDA*, 2013, pp. 150–161.

[32] M. Bhuiyan and M. Al Hasan, "Interactive knowledge discovery from hidden data through sampling of frequent patterns," *Statistical Analysis and Data Mining*, vol. 9, no. 4, pp. 205–229, 2016.

[33] H. Wu *et al.*, "Interactive discovery of coordinated relationship chains with maximum entropy models," *TKDD*, vol. 12, no. 1, pp. 7:1–7:34, 2018.

[34] D. Xin, X. Shen, Q. Mei, and J. Han, "Discovering interesting patterns through user's interactive feedback," in *KDD*, 2006, pp. 773–778.

[35] S. Rüping, "Ranking interesting subgroups," in *ICML*, 2009, pp. 913–920.

[36] H. Abdelhaq, C. Sengstock, and M. Gertz, "Eventweet: Online localized event detection from twitter," *Proc. VLDB Endow.*, vol. 6, no. 12, pp. 1326–1329, Aug. 2013.

[37] J. Wang, J. Cheng, and A. W. Fu, "Redundancy-aware maximal cliques," in *KDD 2013*, 2013, pp. 122–130.

[38] A. A. Bendimerad, M. Plantevit, and C. Robardet, "Unsupervised exceptional attributed sub-graph mining in urban data," in *ICDM 2016*, 2016, pp. 21–30.

[39] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

[40] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," in *WWW*, 1998, pp. 161–172.

[41] S. Papadopoulos, D. Corney, and L. Maria Aiello, Eds., *SNOW Data Challenge*, vol. 1150, 2014.

[42] J. A. Pérez-Melián, J. A. Conejero, and C. F. Ramirez, "Zipf's and benford's laws in twitter hashtags," in *EACL*, 2017, pp. 84–93.

[43] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

Anes Bendimerad is a PhD student in Computer Science at INSA Lyon. He is a member of the DM2L group in the LIRIS Lab. His research is mainly focused on pattern mining and attributed graph mining.

Marc Plantevit received the PhD degree in computer science from the University of Montpellier 2 in 2008. He is currently an associate professor at University Claude Bernard Lyon 1 and member the DM2L group in the LIRIS lab. His research is mainly concerned with pattern mining, especially in sequence and graph data.

Céline Robardet is a full professor at INSA Lyon and member of the LIRIS Lab. She leads the DM2L team. She is particularly interested in clustering analysis, pattern extraction under constraints and complex dynamic network analysis.

Sihem Amer-Yahia Sihem Amer-Yahia is a Research Director at LIG in Grenoble where she leads the SLIDE team. Her interests are at the intersection of large-scale data management and user data exploration. Before joining CNRS, she was Principal Scientist at QCRI, Senior Scientist at Yahoo! Research and Member of Technical Staff at at&t Labs. Sihem served on the SIGMOD Executive Board, the VLDB Endowment, and the EDBT Board. She is the Editor-in-Chief of the VLDB Journal for Europe and Africa and has been on the editorial boards of TODS and the Information Systems Journal. She is chairing VLDB 2018 and WWW Tutorials 2018. Sihem received her Ph.D. in CS from Paris-Orsay and INRIA in 1999, and her Diplôme d'Ingénieur from INI, Algeria.

User-driven geolocated event detection in social media
–
Supplementary material

Anes Bendimerad, Marc Plantevit, Céline Robardet, Sihem Amer-Yahia

1 User-driven discovery based on simulated users

In this section, we provide supplementary experiments of the user-driven approach. We simulated virtual users who, depending on their "own interest", tend to prefer events of a given type. We evaluate their "satisfaction" according to the number of events that the system presents to them that they are inclined to like. Thus, virtual users having topics and/or location preferences are simulated and the experiment consists in studying the number of liked events when considering or not user feedback during the event discovery process.

To that end, we first extracted events in the three real datasets[1] and retain those spanning at least over 2 timestamps (the set $\mathcal{E}$). Then, we manually annotated the events. The tags used for the annotation are Topics $=$ {Business/Economics, Politics, Science/Technology, Art/Culture, Celebration, Music, Sport, Accident/Disaster}. Each event P can be annotated with several topics and the function $\text{Tag}(P,\tau) \rightarrow [0,1]$ expresses the importance of the topic for the event (with $\sum_{\tau \in \text{Topics}} \text{Tag}(P,\tau) = 1$). Some detected events did not match to any category and the obtained distributions are presented in Table 1.

dataset	#	Art/Culture	Music	Celebration	Sport	Politics	Business
NYC	800	97	89	212	87	88	1
LA	489	157	70	18	30	0	7
London	353	120	32	8	53	3	36

TABLE 1
Distribution of events according to the topics.

A virtual user u prefers a specific topic, or location, or both of them. The function $\mathcal{P}_u(P)$ captures the preferences of the user u for the event P. It is defined according to 3 cases:

- If τ is the preferred topic of u, $\mathcal{P}_u(P) = \text{Tag}(P,\tau)$
- If ℓ is the preferred location of u, we consider that u is interested in events at a distance from ℓ at most equal to L and $\mathcal{P}_u(P) = \max(\frac{L-\text{dist}(\ell,P)}{L}, 0)$. Based on the surface area of the cities, we choose $L = 5km$ for New York, and $L = 10km$ for Los Angeles and London.
- If u has both topic and location interests, $\mathcal{P}_u(P) = \frac{\text{Tag}(P,\tau)+\max(\frac{L-\text{dist}_{(\ell,P)}}{L},0)}{2}$

Topics with fewer than 20 events were discarded. For the preferred locations, we consider several well-known places for each city[2]. Finally, to be able to automatically annotating computed events on these datasets, we used the Tag function to annotate hashtags and locations (for $x = h, v$, $\text{Tag}'(x,\tau) = \sum_{P \in \mathcal{E}\ s.\ t.\ x \in P} \text{Tag}(P,\tau)$) and then use them to automatically annotate events ($\text{Tag}^\star(P,\tau) = \sum_{x \in P} \text{Tag}'(x,\tau)$).

To evaluate how `SIGLER-Cov` and `SIGLER-Samp` effectively discover user-driven events, we simulate the same interactive process that we did with the real users in Section 6.4, but with virtual users this time.

Fig. 1 presents results of these experiments when simulating between 19 and 29 virtual users depending on the number of considered locations and topics on each dataset. For each dataset, we show boxplots of the average number of likes using `SIGLER-Cov` and `SIGLER-Samp` in the data and user-driven settings. We can observe that (1) the average number of likes in the user-driven setting is always greater than the one in data-driven configuration. This difference is considered as significant by the Wilcoxon and the Nemeny post-hoc [Dem06] tests (the later is shown on Fig. 1.(2) results obtained by `SIGLER-Samp` are below those obtained by `SIGLER-Cov`, and the difference is significant according

- *A. Bendimerad and C. Robardet are with University of Lyon, INSA Lyon, CNRS UMR 5205.*
- *M. Plantevit is with University of Lyon, University Lyon 1, CNRS UMR 5205*
- *S. Amer-Yahia is with University of Grenoble Alpes, CNRS.*

1. To obtain around 800 events on NYC, we used $\delta = 40$, and to obtain around 400 events on LA and London, we used $\delta = 15$. On all datasets we set $\texttt{minCov} = 0.8$.

2. **NYC**: Barclays Center, Javits Center, Madison Square Garden and Metlife Stadium; **LA**: City Hall, DisneyLand, Museum of Art and Rose Bowl Stadium; **London**: City of London, Royal Albert Hall, Soho Theatre and Wembley Stadium.

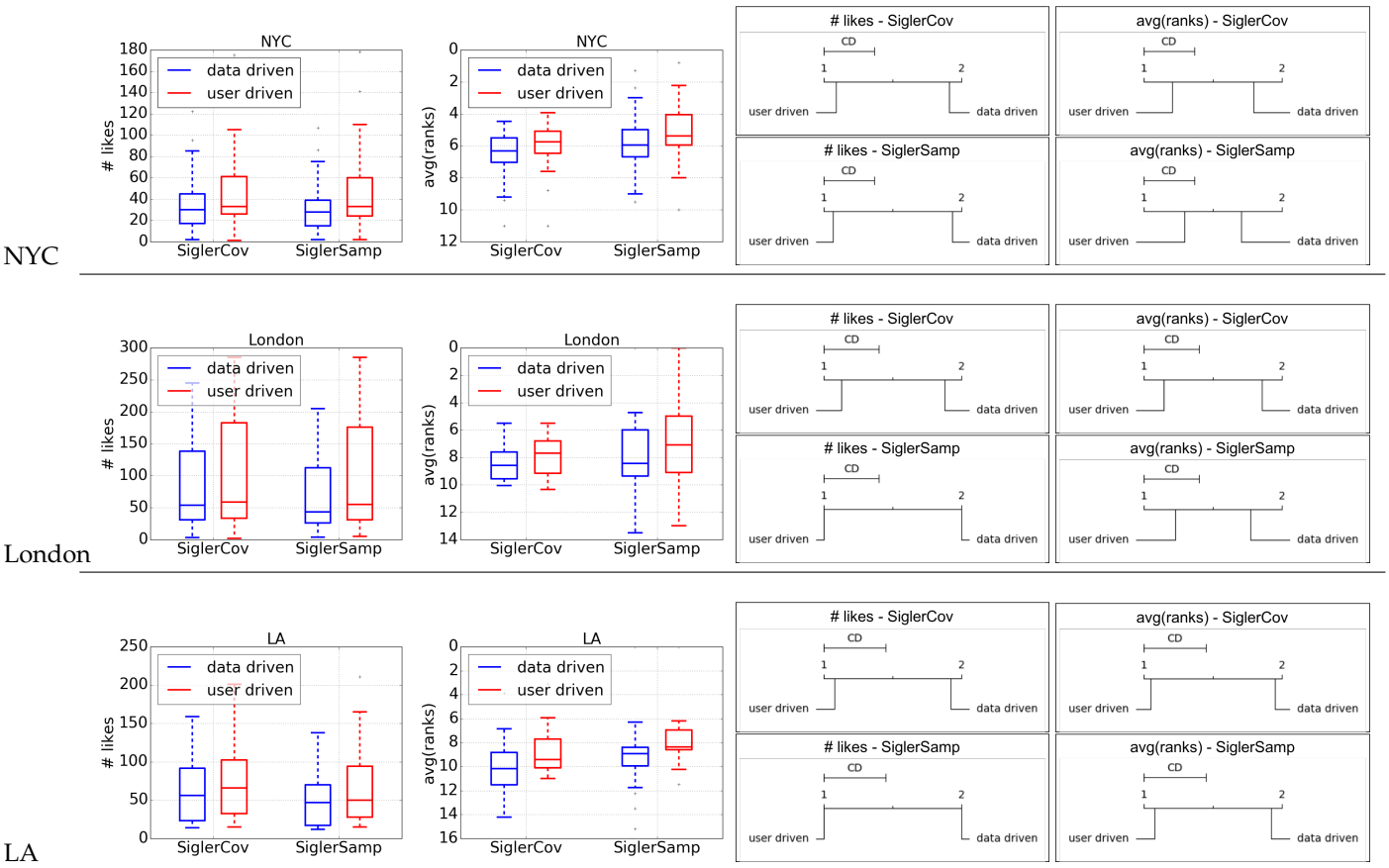


Fig. 1. Virtual user-driven geo-spatial event detection with SIGLER-Cov and SIGLER-Samp. Number of likes (1st column) and average ranks of events liked by both data and user-driven settings (2nd column), Nemenyi tests on number of likes and average ranks (3rd and 4th columns).

to the Wilcoxon test, except for the NYC dataset, where the number of likes is considered to be similar. Indeed, SIGLER-Cov is more exhaustive and finds more events than SIGLER-Samp, which explains this point.

Notice that in this simulation between 300 and 800 events are presented to the virtual-users who, on the contrary to human users, have the ability to evaluate all of them. As already mentioned above, the order in which the events are presented to the users is essential. In order to evaluate this point, Fig. 1 presents the average ranks of events liked by both data and user-driven settings. Clearly the setting for which the liked events are ranked first is advantageous in real situations. We can observe that the average rank is always lower in the user-driven setting than in the data-driven one. This difference is considered significant by the Wilcoxon test on all datasets. This is also confirmed, in a more visual way, by the Nemenyi tests [Dem06] displayed on the figure. When comparing the average ranks obtained by SIGLER-Cov and SIGLER-Samp, it appears that the later obtains significantly better ranks (smaller) for liked events that the former. Thus SIGLER-Samp identifies fewer events, but that are of high quality for users.

This simulation with virtual users allows us to conclude that the user-driven setting makes it possible to identify more relevant events than the data-driven one, whether in terms of quantity and quality. Besides, SIGLER-Samp identifies fewer geolocated events than SIGLER-Cov, but they are of better quality.

2 Comparative Study in Real World Dataset

Based on synthetic data, we have studied in Section 6.2 the ability of our approach to detect local events, and we compared it with other state-of-the-art methods. Ideally, one would prefer to use real world datasets to perform such evaluation. However, we do not have the ground truth of the studied real world data. This makes it very hard to achieve an objective comparison on them. Nevertheless, we show in Table 2 the top 10 events returned by SIGLER-Cov, MED, and GeoBurst, on the first 10k tweets of NYC dataset, and we make a discussion about them. The number of tweets is limited to only 10k, in order to be able to compare with MED which has scalability issues.

We can notice that there are some similar results of SIGLER-Cov with those of MED and GeoBurst. In fact, SIGLER-Cov and MED have both returned the New York Comic Con[3] (1, 4 and 8 in SIGLER-Cov, 3 and 4 in MED), Beyoncé Concert[4] (2 in SIGLER-Cov, 5 and 8 in MED), and Taylor Mac concert[5] (5 in SIGLER-Cov and 10 in MED) .

3. https://goo.gl/BR7kgp
4. https://goo.gl/FrZEBu
5. https://goo.gl/9pM6z3

#	SiglerCov		MED		GeoBurst	
	time	top 5 terms	time	top 5 terms	time	top 5 terms
1	0h to 23h	**nycc, nycc2016, cosplay, ny_comic_con, newyorkcomiccon.**	0h to 23h	nyc, newyork, love, manhattan, saturday.	15h to 23h	**nilerodgers, foldfest, bettemidler, foresthillsstadium, walkeratconcert.**
2	0h to 21h	**formationworldtour, beyonce, formationtour, beyhive, theformationworldtour.**	0h to 23h	newyork, job, hiring, newyorkcity, photo.	15h to 21h	**raniahatoum, thelondonnyc, tripleb, blackbridalbliss, bridalgown.**
3	6h to 18h	**rnrbrooklyn, runrocknroll, halfmarathon, brooklynwerunhard.**	0h to 23h	**nycc2016, cosplay, nycc, newyorkcomiccon, wonderwoman.**	18h to 23h	**intercoiffure, wella, icamoments, nerolisalonspa, wellapro.**
4	12h to 21h	**smashingnycc, nigelthornberry, nigel, smashing, wildthornberries.**	0h to 23h	**nycc, ny_comic_con, cosplay, comiccon, marvel.**	9h to 15h	ridetheferry.
5	12h to 21h	**24decadehistoryofpopularmusic, sawtaylormac, 24decades, marskado, afraidoffun.**	0h to 23h	**formationworldtour, beyonce, beyhive, metlifestadium, beyonce.**	18h to 23h	**sturgillsimpson, kingsbklyn, asailorsguidetoearth.**
6	15h to 23h	**greenday, websterhall, revrad, saturdaynight, 90s.**	0h to 23h	**brooklyn, bushwick, williamsburg, sigurros, music.**	0h to 12h	**elitefridays, cityscapesny, imsobx, reposting, cityscapesnyc.**
7	18h to 21h	**dosgualas, livvinyl, monies, freeze, megaman.**	0h to 23h	repost, montanoy27, regram, alofokemusicnet, parkslopemoms_.	18h to 23h	**descendents.**
8	12h to 18h	**ronswwadventure, 75thanniversary.**	0h to 23h	**formationtour, beyonce, theformationworldtour, nj, kendricklamar.**	0h to 15h	**50cent, dozadrumdealer, mynameisjuan, industrykilla, narcotechs.**
9	15h to 21h	**foresthillsstadium, nilerodgers, fold, bettemidler, foresthills.**	3h to 23h	foodporn, food, foodie, yummy, eeeeeats.	0h to 12h	**deadrabbitnyc.**
10	18h to 21h	**knicks, nets, nyknicks, brooklynnets, preseason.**	0h to 23h	**24decadehistoryofpopularmusic, sawtaylormac, 24decades, proofoflifenumber, marskado.**	0h to 23h	**doomocracy, pedroreyes, doomacracy, creativetime.**

TABLE 2
Top 10 events returned by SIGLER−Cov, MED, and GeoBurst, in NYC dataset, for the first 10k tweets (the day of 8 Oct. 2016). True positive events are market in bold while false negative events are not.

Both SIGLER−Cov and GeoBurst identified the FOLD Festival of Nile Rodgers[6] (9 in SIGLER−Cov and 1 in GeoBurst). However, the rest of top results of GeoBurst are different from the ones of other approaches. GeoBurst seems to give more importance to small events. In fact, each of the top 10 events of GeoBurst contain at most 10 tweets, while the number of tweets in top results of SIGLER−Cov (resp. MED) varies between 43 and 3k (resp. between 42 and 1280).

We believe that the results 1, 2, 7, and 9 of MED, and the result 4 of GeoBurst are not relevant. Indeed, they are defined with terms that do not correspond to any real life event (e.g., nyc, job, hiring, foodporn, etc.). Concretely, The terms ”nyc, newyork, love, manhattan, job, hiring, newyorkcity, photo, repost” are very frequent in New York dataset. Each of them appear at least 30 times in 90% of the days. The term ”saturday” is frequently used in Saturday (more than 30 times in 80% of cases). The terms ”food, foodie, yummy, eeeeeats” also appear in a large number of posts where people want to share their feeling about some food experience. Each of them is used at least 7 times in 50% of the days. The term ”ridetheferry” is used by people who pass by the NY Waterway Ferry. It occurs between 1 and 6 times in 22 different days.

REFERENCES

[Dem06] Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

6. https://goo.gl/1htnhk