Where in the World Is Carmen Sandiego? Detecting Person Locations via Social Media Discussions

Konstantina Lazaridou
Hasso-Plattner-Institut, Potsdam, Germany
konstantina.lazaridou@hpi.de

Toni Gruetze
Hasso-Plattner-Institut, Potsdam, Germany
toni.gruetze@hpi.de

Felix Naumann
Hasso-Plattner-Institut, Potsdam, Germany
felix.naumann@hpi.de

ABSTRACT

In today's social media, news often spread faster than in mainstream media, along with additional context and aspects about the current affairs. Consequently, users in social networks are up-to-date with the details of real-world events and the involved individuals. Examples include crime scenes and potential perpetrator descriptions, public gatherings with rumors about celebrities among the guests, rallies by prominent politicians, concerts by musicians, etc. We are interested in the problem of tracking persons mentioned in social media, namely detecting the locations of individuals by leveraging the online discussions about them.

Existing literature focuses on the well-known and more convenient problem of user location detection in social media, mainly as the location discovery of the user profiles and their messages. In contrast, we track individuals with text mining techniques, regardless whether they hold a social network account or not. We observe what the community shares about them and estimate their locations. Our approach consists of two steps: firstly, we introduce a noise filter that prunes irrelevant posts using a recursive partitioning technique. Secondly, we build a model that reasons over the set of messages about an individual and determines his/her locations. In our experiments, we successfully trace the last U.S. presidential candidates through millions of tweets published from November 2015 until January 2017. Our results outperform previously introduced techniques and various baselines.

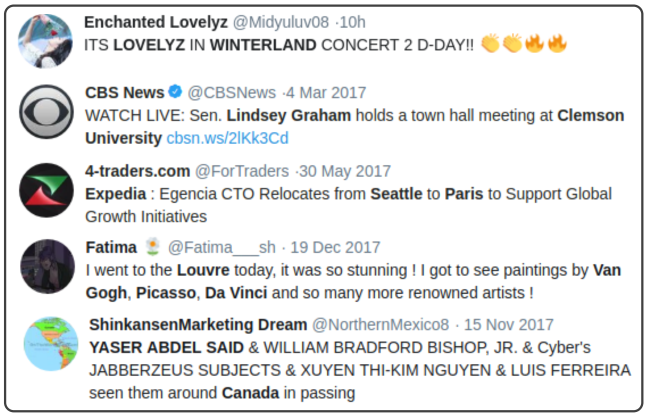
CCS CONCEPTS

• **Information systems** → **Web searching and information discovery**; **Social networks**; **Document filtering**; **Information extraction**; • **Computing methodologies** → **Supervised learning by classification**;

KEYWORDS

person tracking, event detection, text classification, social network analysis

ACM Reference Format:
Konstantina Lazaridou, Toni Gruetze, and Felix Naumann. 2018. Where in the World Is Carmen Sandiego? Detecting Person Locations via Social Media Discussions. In *WebSci '18: 10th ACM Conference on Web Science, May 27–30, 2018, Amsterdam, Netherlands.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3201064.3201068



Enchanted Lovelyz @Midyuluv08 ·10h
ITS **LOVELYZ** IN **WINTERLAND** CONCERT 2 D-DAY!! 👏👏🔥🔥

CBS News @CBSNews ·4 Mar 2017
WATCH LIVE: Sen. **Lindsey Graham** holds a town hall meeting at **Clemson University** cbsn.ws/2lKk3Cd

4-traders.com @ForTraders ·30 May 2017
Expedia : Egencia CTO Relocates from **Seattle** to **Paris** to Support Global Growth Initiatives

Fatima @Fatima___sh · 19 Dec 2017
I went to the **Louvre** today, it was so stunning ! I got to see paintings by **Van Gogh**, **Picasso**, **Da Vinci** and so many more renowned artists !

ShinkansenMarketing Dream @NorthernMexico8 · 15 Nov 2017
YASER ABDEL SAID & WILLIAM BRADFORD BISHOP, JR. & Cyber's JABBERZEUS SUBJECTS & XUYEN THI-KIM NGUYEN & LUIS FERREIRA seen them around **Canada** in passing

Figure 1: Tweets that indicate the locations of different entities: Lovelyz, Lindsey Graham, Expedia, Van Gogh, Picasso, Da Vinci and Yaser Abdel Said.

1 NEWS SPEED AND COVERAGE IN SOCIAL MEDIA

Millions of people publish their thoughts and experiences on various social networks and microblogs, such as Facebook, Twitter, etc. Users share real-time information via text messages, geo-located images, live videos etc. An example of the speed and brevity specifically of Twitter is the shooting outside the Texas Irving mall in 2011. The incident was reported by a very short tweet immediately after the shooting, in contrast to newspapers that reacted with a 3-hour delay [11]. Similarly, one is also likely to inform their peers about a natural disaster outbreak, even before the first news story is published [7]. Hence, Twitter can be seen as a fast and decentralized news media.

Furthermore, users keep their peers up-to-date, by retweeting, quoting and engaging in discussions about the current affairs. When considering that most of the posts on Twitter have no visibility restrictions, it is reasonable to claim that this platform "breaks down the communication barriers" [20]. According to Kwak et al., regardless the popularity of the original account, any random retweet spreads over the network almost instantly [9]. This means that every retweet is expected to reach 1,000 users on average, imposing its impact to the rest of the network. Thus, Twitter users can be extremely influential by sharing real-time ongoing news, including civil unrest, entertainment activities, earthquakes and floods, etc. This vast amount of information has attracted various Twitter analyses, particularly related to the problem of event [5] and location [17] detection in social media, with the latter being essential due to the very low amount of geo-tagged tweets.

In this work, we are interested in a location detection problem that leverages the up-to-dateness of social media (e.g. microblogs), that is: the task of "person tracking". Unlike related work on user location detection, we consider the individuals to be mentioned in discussions in the Twittersphere, rather than assuming that they hold a user profile. We prefer to rely on the wisdom of the crowd that discusses about a given person p, because we hypothesize that it brings many more tweets as evidence on p's locations than p might potentially share him/herself. We also do not assume that a location mentioned in a user post is identical to this user's current position. Thus, we allow users who discuss event locations asynchronously.

As shown in Figure 1, by detecting where a music band (**Lovelyz**) is or plans to be, a user can decide to join their concert and browse people's comments and anticipation about this specific event. Similarly for politicians (**Lindsey Graham**), we can leverage tweets discussing about them to discover the town hall meeting they hold. Additionally, target entities might also be companies that relocate (**Expedia**), or objects, such as famous art pieces that are moving to different countries over time (**Van Gogh, Picasso, Da Vinci**). Tracking vulnerability in software products over time could also be tackled by analyzing the mentions of exploit kits on the Web[1].

Moreover, an important use case covered by our approach is the ability to track people that are national risks, such as wanted criminals and warlords. An example of a well-known fugitive is **Yaser Abdel Said**, who is still missing and for whom FBI offers a high reward in exchange of valuable leads on his arrest. To demonstrate the benefits of a person tracking approach in social media, we performed a simple query in the Twitter Search API, namely, "Yaser Abel Said seen in". Only one tweet is returned by *NorthernMexico8* posted on November 2017 and as shown in Figure 1, it places him in Canada.

As we can observe, this basic test shows the challenge of analyzing a limited amount of valuable data, yet the potential of tackling the (person) entity tracking problem via social network discussions. Note that, even when the available data are more, i.e., individuals are very popular and draw a lot of attention in the media, there are still important challenges to face. Particularly, the high amount of spam and fake messages makes it crucial to filter the data, in order to detect correct person locations and avoid any misinformation or chatter, e.g. false positives and farces.

Hence, our goal is to harvest the wisdom of the crowd that can potentially provide us with ongoing events, but at the same time we must make sure to avoid noisy tweets. In addition, even though tracking people that do not want to be found is useful in the case of criminals, it might raise ethical concerns and have negative implications on the target individuals, such as when locating protesters and activists[2].

Unlike most prior work that mainly deals with location identification of social network users themselves [13], their home [16] or messages [24], we consider the tweets as a means to derive the physical position of mentioned individuals. We seek to answer the following question: Given a target person entity p, can we identify the locations of p over time only by observing what people say about p in social media? To address this problem, we analyze tweets that mention p to determine all p's physical locations and gain insights from a big tweet dataset spanning one year. Our approach uses millions of tweets relevant to the U.S. general elections in 2016 with the goal to track the presidential candidates and it considerably outperforms existing techniques and baselines. As demonstrated in Figure 2, our contributions include:

- A greedy two-phase algorithm that filters relevant tweets for person tracking
- A novel approach for predicting the locations of individuals by leveraging their third-person references in social media posts
- Evaluation results on tracing U.S. politicians' locations

The rest of the paper is organized as follows: Section 2 presents related work, Section 3 describes how we discover relevant tweets and Section 4 discusses how we determine the individuals' locations. Section 5 shows our results and Section 6 concludes this work with ideas for future work.

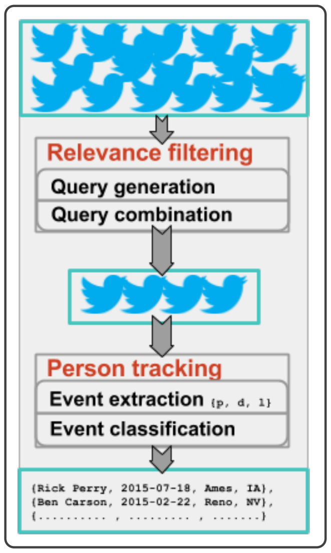


Figure 2: Our overall approach: Given a tweet stream, we apply a relevance filter that prunes the noise and supply the remaining tweets to a person tracker that outputs person locations.

2 RELATED WORK

An extensive body of literature focuses on novel information discovery in user-generated content, such as news pieces, trending topics, popular events, etc. Related research includes *TwitterStand* [20], a framework that provides a geographic overview of breaking news on Twitter and *TwitterMonitor* [14], which performs real-time trending topic tracking. Since we are interested in detecting localized events and their timestamps, which we consider as the places a target individual visits, we find our work relating better to the task of event detection, rather than trend detection. An event usually

[1]https://www.recordedfuture.com/tracking-exploit-kits/
[2]http://www.complex.com/life/2016/11/police-surveillance-activists-people-of-color

appears as a bursty occurrence of novel information in a certain time period [1] and a sudden increase of the occurrence of certain words [5]. The attention that events attract typically fades over time as other significant incidents arise, e.g., in our case, the target entity moving to another location.

Event detection has been studied extensively for various application areas, e.g., predicting earthquakes [18], real-time discovery of sports competitions [1] and detection of event-related information [12]. However, prior work is mainly motivated by the need to keep the users up-to-date in emergency situations and few works identify and analyze events independently of their type [5]. The majority focuses on the cases of incidents of public interest [25] , e.g., natural disasters, instances of civil unrest, or disease outbreaks.

In contrast, we do not address the problem of event detection aiming at public awareness, but we solve the task of person tracking in social media. Given an individual as a user query, we show that social media can help us create a timeline of his/her locations. Each event in the timeline is independent from the others regarding its kind and duration, and the frequency of these events depends entirely on the individual's profile. Hence, we are limiting our search to locations that these persons visit, yet at the same time we consider all possible types of events.

Another line of research that is closely connected to our work is the identification of locations in social media. Its emergence can be justified by the lack of geolocated user posts, especially on Twitter, since only 1% of the messages includes geotags [21]. Related works include *PETAR* [10], a time-aware point of interest (POI) extraction system and *TWILOC*, which determines the location of a tweet based on various content and network features [6]. Backstrom et al. study the relationship of social and spatial proximity and use the network properties to predict the location of users [2]. It is shown that social data, such as the location of a user's friends, can enhance prediction performance.

Unlike the above-mentioned works, we take into account tweets by various users that are published in a certain time frame, instead of performing a user-focused analysis [22]. Thus, we are not interested in geo-locating neither a tweet nor its user. Instead, we analyze the location and person mentions that are contained in tweets, in order to track the mentioned individuals.

Our goal is to gain insights about a discussed entity p and hence, we treat any potential tweet posted by p as all other tweets that share information about p in the third person. A representative example of a tweet we wish to discover is: "'History is made by the dreamers, not the doubters'. **Donald Trump** just now in **Des Moines**. #Politics @POTUS @realDonaldTrump @IvankaTrump @FLOTUS". This is an appropriate post for our task regardless the account that it originates from. By mining the textual content of such messages, we cope with the lack of geotags on Twitter, as well as with location inconsistencies. For instance, users might also share their thoughts about an event they attended earlier this day, which means that their current location is not identical with the event's location anymore. Thus, we choose to find locations in the content of the tweets instead, by applying a named entity linking approach [3].

The most relevant study to our current work is our previously introduced basic approach that discovers *located-in* patterns in the tweets [4] . We applied the Apriori algorithm on a very small set of tweets to discover frequent terms that can be used as queries for the Twitter API to retrieve relevant posts to politicians' locations. Consequently, it was naively assumed that each event that yields a minimum support of ten or more tweets in the result set corresponds to an actual event. Limited results were reported about the locations of Donald Trump, Hillary Clinton, Bernie Sanders, and Ted Cruz on the day prior to Super Tuesday.[3]

Drawing inspiration by these preliminary findings, we now perform a large-scale analysis on almost a billion of tweets and various events and individuals. We introduce a novel approach for noise detection in the context of person tracking, which is based on recursive partitioning and carefully generates higher quality queries than our previously proposed method. Instead of solely relying on the popularity of the mentioned events, we use a supervised constraint-based approach to detect which of the event locations are valid.

3 FINDING THE NEEDLE IN A HAYSTACK OF TWEETS

The first part of our person tracking approach is responsible for excluding noisy messages, which provide misleading information about the target entities and their associated events. We define an *event* as a triplet $e = (p, l, d)$, where an individual p appears in a specific location l on a particular date d. We model our noise detection task as an Information Retrieval task: given the tweets published in a certain time period, we wish to retrieve the ones that are relevant to person tracking. That is why we design a query for the Twitter API that will return suitable messages for our goal. Given the clean result set, we detail how we classify the discussed events into correct and incorrect in Section 4.

One can easily grasp that the terms {*rally*} or {*rally, today*} might be promising choices if one is searching for political campaigns in social media. However, given the almost infinite amount of words and hashtags that one can search with, choosing the right query is a cumbersome and complex task. The appropriate query terms depend on how users like to describe the locations of others, such as "live in", "don't miss the", or "just saw". Since the phenomenon of misinformation in media has risen in the past years [15], a naive query might return tweets that are fake or spam regarding the target entities.

Figure 3 depicts the number of tweets we found for four popular entities on four randomly selected dates: the U.S. politicians Donald Trump and Hillary Clinton, and the bands U2 and Red Hot Chili Peppers. The number of tweets that simply refer to an entity p is shown in *blue*, while the portion of them that contains a reference to an actual event location of p is depicted in *red*. The events are public speeches and concerts respectively. Although we depict a limited data sample[4] that is often biased by the medium's sampling process [19], we can already observe that irrelevant tweets are orders of magnitude more common than relevant ones are. Therefore, noise detection becomes an important, often domain dependent problem and a person tracking method is expected to distinguish which context provides correct person information and which not.

[3]https://en.wikipedia.org/wiki/SuperTuesday
[4]The Public Streaming API is limited to a maximum of 1% of the overall traffic on Twitter (i.e., around 5 million tweets per day).

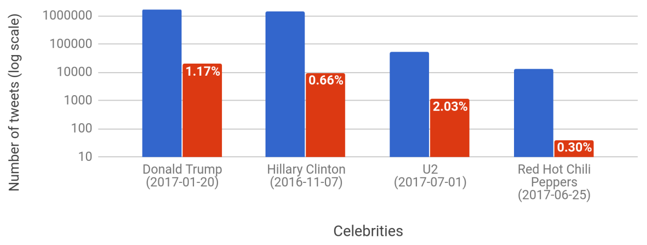


Figure 3: For four public figures and four respective dates, we depict the number of tweets referring to a) this entity (left) and b) the entity and a correct location (right).

3.1 Problem Statement

Given a set of entities we wish to detect, let us assume that the set of Twitter statuses mentioning at least one entity is denoted as T. Each tweet $t \in T$ represents a document that consists of a set of words, s.t., $t = \{w_1, w_2, ..., w_i\}$. All unique words in T form the existing vocabulary V. We aim to discover the tweets $T^+ \subset T$ that contain relevant information for our task. We refer to the rest of the tweets as $T^- \subset T$, where $T = T^+ \cup T^-$ and $T^+ \cap T^- = \emptyset$ hold.

A relevant tweet $t \in T^+$ is a message that refers only to correct event information, that is, contains an actual event triplet, $e = (p, l, d)$. For instance, during the U.S. election campaigns, the current U.S. president Donald Trump conducted a rally in Georgia on 29/2/2016. Thus, the tweet "LIVE Stream: **Donald Trump** Rally at Valdosta St. University in **Valdosta, GA**" belongs to T^+, whereas "It's Leap Day 2016. February has 29 days. And **Washington** is in an uproar. **Donald Trump** is trying to have the extra day deported" belongs to T^-. The second example is a tweet that refers to a false location of Donald Trump for that date and our approach makes it feasible to detect it, since it learns the context that individuals' locations are likely to be discussed on Twitter.

The Twitter API provides an interface for *Boolean queries*, where a query Q is a combination of terms $w \in V$ and boolean operators $\neg$, $\wedge$, and $\vee$. Our goal is to create a filtered tweet set T_Q, s.t., $T_Q \cap T^+$ is maximized and $T_Q \cap T^-$ is minimized. This optimization task can be reduced to the knapsack problem, which is known to be NP-hard. Given the fixed-size rucksack (queries allowed by the Twitter API), we aim to fill it with the most valuable items (most promising queries). Because the number of possible queries is exponential to $|V|$, it is not possible to enumerate them and select the best one. Therefore, it is not feasible to find an optimal solution in reasonable time.

To design a good query, we propose a greedy approach that is based on recursive partitioning. We generate Q in a disjunctive normal form. That is, Q is defined as an $\vee$-combination of queries, i.e. $Q = q_1 \vee q_2 \vee ... \vee q_i$, where each q_x is an conjunction of words or their negations, e.g., $q_x = w_1 \wedge \neg w_2 \wedge ... \wedge w_j$. For instance, we discovered that promising queries to trace politicians in the context of U.S. elections are "night $\wedge$ primary", "holds $\wedge$*in*" and "rally $\wedge\neg$*monday*". Our noise filtering algorithm consists of two phases: first, we discover promising conjuction queries q_x that maximize the positive examples in T_{q_x} and second, we combine candidate conjunctions in a query Q. The retrieved tweets are further examined by our event classifier in Section 4.

```
1:  function FIND_CANDIDATES($T, V, P$)
2:      $\Omega = \emptyset$
3:      for $p$ in $P$ do
4:          for $i$ in $1..\lfloor\sqrt{|V|}\rfloor$ do
5:              $V_i = fold(i, V \setminus p)$
6:              $\Omega = \Omega \cup \text{PARTITION}(T, V_i, \{p\})$
7:      return $\Omega$
8:  function PARTITION($T, V_i, q$)
9:      $\Omega = \{q\}$
10:     if $|q| < \theta_{len}$ then
11:         $w = \arg\max_{w \in V_i} \text{IG}(q, w)$
12:         if $\chi^2(T_q, w)$ then
13:             if $\frac{|T^+_{q\wedge w}|}{|T_{q\wedge w}|} > \frac{|T^+_{q\wedge\neg w}|}{|T_{q\wedge\neg w}|}$ and $|T^+_{q\wedge w}| \geq \theta_{supp}$ then
14:                 $\Omega = \Omega \cup \text{PARTITION}(q \wedge w, V_i \setminus w)$
15:             else if $|T^+_{q\wedge\neg w}| \geq \theta_{supp}$ then
16:                 $\Omega = \Omega \cup \text{PARTITION}(q \wedge \neg w, V_i \setminus w)$
17:     return $\Omega$
```

Figure 4: Candidate query discovery: recursive partitioning algorithm that creates queries that prune irrelevant tweets.

3.2 Candidate Query Discovery

Inspired by the principle of boosting in machine learning, we construct a variety of term-conjunctions that are built on independent data portions. Figure 4 demonstrates our approach for generating candidate queries, motivated by the principles of decision tree learners. Consider a set of pivot terms P (queries containing only one word) with a high coverage in T^+ (Line 3). Each seed term provides us with a high quality start, which propagates to the conjunctions that will be generated in the next recursive partitioning step (Line 6). For instance, let us assume the football player Luis Suarez and as pivot term the word *seen*. If *seen* is found in a high number of correct tweets (T^+) about Luis Suarez, e.g., "Just seen **#LuisSuarez** in Park Guell **#Barcelona**", this also increases the chances that the combination of *seen* and *in* would retrieve correct locations of the player.

Furthermore, for every pivot term, we split the vocabulary into $k = \sqrt{|V|}$ random and equally sized folds V_i (Line 4 and 5). In each iteration we expand the candidate query (that initially consists of p) with new terms from V_i. Note that every fold has the same size: $\forall i \in \{1, 2, ..., k\} : |V_i| \approx \sqrt{|V|}$, while $\bigcup_{V_i} = V$ and $\bigcap_{V_i} = \emptyset$ hold. The partitioning process (Lines 8- 17) works as follows: Assuming that the current q does not exceed the permitted length θ_{len} (Line 10), the algorithm expands it further. Although the length threshold is rarely hit, we adopt this constraint to prevent very long conjunctions that might lead to overfitting or conflict the restrictions of the Twitter API. Moreover, we perform a query expansion and select the term w (Line 11) that results in the highest information gain regarding the separation of the sets T^+ and T^-. We measure information gain as:

$$IG(q,w) = H(q) - \frac{\left|T_{q\wedge w}\right| * H(q \wedge w) + \left|T_{q\wedge \neg w}\right| * H(q \wedge \neg w)}{\left|T_q\right|}$$

where the *Shannon* entropy $H(\mathrm{q})$ is defined as:

$$H(q) = -\sum_r \left(\frac{\left|T_q^r\right|}{\left|T_q\right|}\right) \log\left(\frac{\left|T_q^r\right|}{\left|T_q\right|}\right), r \in \{-,+\}$$

and the set T_x refers to the tweets that x satisfies. The expansion based on the information gain is inspired by the greedy feature selection of the $C.45$ decision tree learner. It fits well to our task, because we leverage that the term conjunctions fulfill the monotonicity property.

Our overall goal is to distinguish between the vocabulary that users choose to discuss about actual events (of the target entities) and the vocabulary in any other topic that is irrelevant to our task. Thus, in order to capture and successfully avoid words that typically appear in incorrect context, we allow either w or $\neg w$ to expand q (Lines 13 to 16).

For the purpose of avoiding overly specific queries that overfit the data associated to the current fold, we stop expanding when the improvement of w (or $\neg w$) over q is not statistically significant (Line 12). To quantify the significance, we consider the null hypothesis that q's application will not affect the distribution of T^+ and T^-. We perform a χ^2 test to test the hypothesis and reject it if it cannot be supported with the typical significance level of at least $\alpha = 0.05$. We also prevent the query expansions $q \wedge w$ (or $q \wedge \neg w$) to be too specific, by ensuring that the new partition yields sufficient support over T^+, denoted as θ_{supp}.

3.3 Candidate Query Combination

Armed with a valuable set of promising queries Ω, we now combine them to generate our final query Q in a disjunctive normal form that provides us with fewer noisy tweets for person tracking. Given Ω and our document collection T, Figure 5 demonstrates our approach to greedily derive a good disjunction by maximizing the expected query quality *score*:

$$score(q,T) = \frac{\left|T_q^+\right|}{|T^+| + \left|T_q\right|}$$

It is evident that our *score* definition is proportional to the F_1 metric, given that T^+ is the set of relevant and T_q the set of retrieved documents. Therefore, COMBINE_CANDIDATES finds a local optimum for our problem.

Note that the number of possible combinations is exponential to the size of Ω and hence, enumerating all solutions is not feasible. If the maximum length of Q is reached or Q cannot be improved by adding further conjunctions $q \in \Omega$ (Line 6), the combination phase terminates. The monotonicity property of the disjunctions combined with the repeated improvement of the *score*, results in an extended query that covers a high number relevant tweets.

```
1: function COMBINE_CANDIDATES($\Omega, T$)
2:     $Q = \emptyset$
3:     repeat
4:         $Q' = Q$
5:         $Q = Q \vee \arg\max_{q' \in \Omega} score(Q \vee q', T)$
6:     until $score(Q,T) > score(Q',T)$ or $|Q| > \theta_{len}$
7:     return $Q$
```

Figure 5: Candidate query combination: combining different conjunctions to form a final disjunction query that minimizes the irrelevant and maximizes the relevant tweets.

4 CONSTRAINT-BASED PERSON TRACKING

Given the relevant data we discovered in Section 3, we can now address the question: How can one accurately extract people's locations by examining their references in social media posts? Inferring the places that individuals attend from social network discussions is a very challenging task. Realistic constraints should be taken into account, such as, any person cannot visit more than a reasonable number of locations per day, e.g., music artists usually schedule only one big gig per day, even during a tour. Additionally, many tweets are expected to talk about real-world events in contrast to incorrectly discovered events that won't dominate the online discussions. For instance, users share their experiences about various situations, ranging from popular global events (a concert by a famous band) to local community fairs that will most likely gain more attention in social than mainstream media.

We model this reasoning problem as a binary classification task and decide for each mentioned event on Twitter whether it is true or not. In order to ensure a good tracking performance, our constraint-based person tracking method leverages both the characteristics of the discussed events as well as the tweets themselves.

4.1 Event extraction

Each discussed event $e = (p, l, d)$ in our tweet set T is associated with a person p, a location l and a date d. It is denoted as $e \in E^T$, while the messages about e are denoted as T_e ($T_e \subset T$). To infer the date of e_i from a tweet t that discusses e_i, we use t's publication date, inspired by the up-to-dateness of microblogs as Twitter [9]. Thus, we leverage the daily reactions on e_i, by considering asynchronous discussions about it within the course of a day. We leave more flexible temporal tagging for our future work.

We allow that a person can visit the same location on different dates and can appear in multiple locations on the same date. To identify l and p, we use a named entity linking approach based on *CohEEL* [3] and apply it on the tweet text. Given a knowledge base, e.g. YAGO [23], CohEEL discovers potential mentions that are likely to be linked to a certain entity in the knowledge base. As a second step, the algorithm explores the entity graph derived from the knowledge base with a random walk approach and it extracts the final and coherent entity mentions.

We apply CohEEL with WIKIPEDIA and WIKIDATA (an open knowledge base) and extract from the tweets two different types of entities: persons (the target individuals) and locations (cities). We perform our analysis on a city level, that is, if an entity is found in n different city venues in n different tweets (various streets, buildings

Table 1: Twitter data extracted from November 2015 until January 2017.

All tweets	Tweets with entities	Correct tweets
903,239,572	29,208,457	321,530

etc.), we map the venues to the appropriate city name and consider each of them as a visit to this particular city. Taking into account that CohEEL can also be used for other kinds of target entities (e.g., companies and organizations) and locations in different granularities can be allowed (e.g., states, countries), our approach is easily adapted to other tracking use cases.

4.2 Event classification

After identifying the events mentioned in the tweets, we classify them using a number of features, inspired by the previously mentioned realistic constraints:

Popularity. The *popularity* or *prevalence* of an event on Twitter can be estimated based on the number of unique original tweets discussing about it (disregarding retweets). We refer to the popularity as $prev(e) = |T_e|$. This feature has already been proven as a good indicator for actual events in previous work [4]. For instance, on 31/8/2017 we found that the football player Cristiano Ronaldo was tweeted to be on a trip in the UK. There are more than 3 different tweets on that date all placing him in Manchester, as well as three others, about Tottenham, Longsight and Wolverhampton respectively. Thus, from a statistics point of view, Manchester seems more likely to be a true location.

Another interesting example is shown in Figure 6, which presents the city locations of the politician Jeb Bush during his South Carolina (SC) rally. The color indicates the number of tweets in a specific region. Despite the fact that many locations outside South Carolina are mentioned, the dominance of SC venues on Twitter gives a strong indication towards events in this particular region.

Distance. In the previously described example about Christiano Ronaldo, we observed that all tweets are published in a timeframe of only two hours, which raises the question of how far these three mentioned locations are from each other. Therefore, given all location mentions on a date, an event classifier should be able to understand how far an entity can travel within a certain time period.

We introduce the feature *distance*, i.e., the average pairwise distance of a certain location to the rest on a specific date. Similarly to the *popularity*, in a real-time experiment, this distance is updated as more locations are mentioned in newly published tweets. Moreover, each city is considered as a point on the earth and given its longitude and latitude, we calculate its Haversine distance from the other cities. Namely, given the locations of a person p on a date d, the associated tweets are denoted as $T_{p,d}$. The events found in $T_{p,d}$ are defined as follows:

$$E^T_{p,d} = \left\{ e_i \in E^T \mid p_i = p, d_i = d \right\}$$

and the distance feature of an event e is:

$$dist(e) = \frac{1}{\left|T_{p,d}\right|} \sum_{e_i \in E_{p,d}} |T_{e_i}| \cdot \textsc{Haversine}(geo(l), geo(l_i))$$

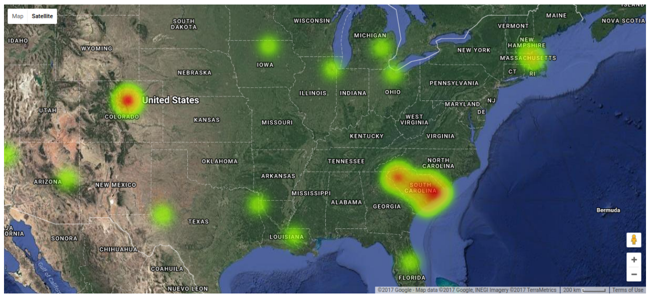


Figure 6: Heatmap of Jeb Bush's locations identified in tweets on 11/02/2016 during his South Carolina rally.

By using this feature, we aim to preserve the events that are held reasonably close to each other and eliminate locations that are very far from each other. We extract the geolocations of the cities from WIKIDATA: $geo(l) = \texttt{wd:P625}(l)$[5].

Population. We hypothesize that the size of a location, such as the *population* of an event's city, can be indicative of whether this event is true or not. We test this hypothesis by including the city's population as a feature of our event classifier and expect that the popularity of a target individual might be correlated to the size of the locations he or she visits. The city populations are retrieved from WIKIDATA with the query: $pop(e) = \texttt{wd:P1082}(l)$[6].

4.3 Datasets

We evaluate our approach for person tracking on a set of messages extracted via the Public Twitter API during a period of approximately one year.

Tweets. Our dataset consists of millions of messages published by more than 33 million users. The posts mention various individuals related to the last U.S. presidential election (2016). In order to ensure a high coverage on political discussions, we used 241 queries with politicians' names and usernames, as well as popular hashtags related to the election. Since the language found in the Twittersphere can be eccentric, the queries we posed contain not only the individuals' names and Twitter user accounts, but also potential aliases (such as *Hillary Rodham Clinton*, *Secretary of State*, *@HillaryClinton* and *#HRC*), extracted from WIKIPEDIA.

As shown in Table 1, the overall amount of tweets we gathered from the Twitter API is approximately one billion. This results in an amount of 2 million tweets per day. Among these data, there are 29 million posts that discuss our target entities (contain mentions to presidential candidates and locations discovered by *CohEEL* [3]). Furthermore, there are only 321,530 tweets revealing the actual locations of our target entities, which makes our task particularly challenging. The list of tweet ids for every discussed location and politician can be found in our homepage[7].

Events. To evaluate our approach for person tracking in social media, we collected a series of publicly available event records regarding the U.S. presidential candidates in 2016. Our ground truth is a set of events that presidential candidates hosted or participated

[5] https://www.wikidata.org/wiki/Property:P625
[6] https://www.wikidata.org/wiki/Property:P1082
[7] https://hpi.de/naumann/projects/web-science/social-media-analysis/politics-on-twitter.html

prior and after the general elections extracted from the 4President blog [8]. This website contains information about events related to the past four elections in the U.S.A. We automatically extract the reported events related to the presidential candidates of the last general election in 2016. Many entries on the website are usually a single event, e.g., the title of the entry page explicitly refers to an event triplet, `person-location-date` (e.g., Donald Trump, Youngstown, Ohio, 25/7/17).

In the cases where the title contains a broader location, i.e. a state, we apply *CohEEL* on the page's body text to determine the different cities within the state that a presidential candidate has visited. For the purpose of ensuring the validity of each event in our gold standard collection, any other blog entry whose title does not describe an event triplet, namely `politician-city-date`, is disregarded by our extractor. The resulting gold standard consists of almost three thousand events for various candidates, such as Ben Carson, Lincoln Chafee, Chris Christie, Hillary Clinton, Lindsey Graham, Rick Santorum, Jim Webb etc.

5 RESULTS

In this section, we evaluate our noise detector based on **Rec**ursive **Par**titioning (RecPar) and our **Co**nstraint-based **P**erson **T**racker (CoPT). RecPar leverages the wisdom of the crowd to discover relevant tweets and CoPT categorizes them into true or false person locations. First, we show the optimal setup of RecPar and compare it with our previously introduced approach that is based on **Freq**uent **Item**sets (FreqItem) [4]. Second, we demonstrate results on person tracking and compare CoPT with other approaches and baselines. In general, we conduct our experiments in consecutive monthly time intervals, namely we use the earliest months of our dataset to learn RecPar's query, afterwards we train CoPT, and in the last part of the dataset we test the performance of the overall approach.

5.1 Relevance filtering

RecPar consists of two consecutive phases: candidate query discovery and candidate query combination. The set of candidates, denoted as Ω, is generated by the first component and is also referred to as conjunctions or subqueries of the final query combination Q. In the current evaluation task, we show how RecPar behaves with different parameter settings. There are four parameters that we must consider in our approach:

- the maximum length of the final combined query Q (`maxQLen`)
- the maximum length of each subquery in Ω (`maxSubQLen`)
- the minimum support –number of tweets– that a subquery should exhibit to be included in Ω (θ_{supp})
- the minimum pivot support ($\theta^{p}_{\mathsf{supp}}$) that determines which terms will be the pivots

An example combination of the first two parameters could be a setting where `maxQLen` = 100 and `maxSubQLen` = 10. Herewith, RecPar would create a query Q with at most 10 subqueries, whose length will be 100/10=10 at maximum. For instance, a query combination that tracks art exhibitions of Picasso could be $(must \wedge see \wedge art \wedge exhibition \wedge Picasso) \vee (don't \wedge miss \wedge art \wedge work \wedge Picasso) \vee$

[8]http://blog.4president.org/

Table 2: Given different values of $\theta^{p}_{\mathsf{supp}}$ and θ_{supp}, the precision (PREC) and true negative rate (TNR) are shown, computed after the candidate query generation phase (Ω) and the candidate query combination phase (Q) respectively.

θ^{p}_{supp} (%)	θ_{supp} (%)	PREC		TNR	
		Ω	Q	Ω	Q
10	0.25	0.133	**0.523**	0.019	**0.928**
5	0.25	0.133	0.515	0.012	0.926
1	0.25	0.132	0.516	0.002	0.924
0.50	0.25	0.132	0.513	0.002	0.923
0.25	0.25	0.132	0.510	0.001	0.921
10	10	0.133	0.519	0.019	0.931
10	5	0.133	**0.530**	0.019	**0.937**
10	1	0.133	0.500	0.019	0.927
10	0.50	0.133	0.512	0.019	0.927
10	0.25	0.133	0.514	0.019	0.926

$(interesting \wedge exhibition \wedge inspired \wedge by \wedge Picasso)$. Both parameters are influenced by the restrictions of the Twitter API, yet affect RecPar's performance as well. In a real-time setting, our system would query the Twitter Streaming API with the target's name and meaningful keywords, and as the tweets arrive, it would categorize each mentioned event as true or false. Thus, we take into account that as of today, the Twitter API allows searches with at most 400 terms. This means that at least one of the query terms needs to be the name of the target person and the rest will be generated by our model.

Our intuition is that the more queries we allow our model to generate, the better the chances to capture more helpful tweets. In contrast, experimenting with different `maxQLen` values (i.e., 100, 200, 300 and 400) showed that this aspect influences our final event classification results only up to approximately 1%! We conclude that selecting promising and relevant queries is more essential than their number. Hence, in all our experiments `maxQLen` is set to its potential maximum, i.e. 395, leaving five terms to contain the person's name or alias (e.g., nickname) we aim to discover.

We also examined different values for `maxSubQLen` (between 2 and 20). Similarly to `maxQLen`, the results were not significantly affected for values higher than 5. Assigning a small number to `maxSubQLen` seems logical if we consider that tweets are limited to 140 characters, among which the name of the target person and a location have to appear. Therefore, we chose to set `maxSubQLen` to 5 for the rest of our experiments.

5.1.1 *Support thresholds.*

As shown earlier in Figure 3, the number of tweets mentioning real-world events is extremely low, i.e., below 2% of all tweets. Thus, we experiment with low values for θ^{p}_{supp} and θ_{supp} and define these two thresholds as a percentage of the correct tweets in our training set. We train RecPar's query with the first 3 months of our dataset (2015-11-01 – 2016-01-31) and use the next month (2016-02-01 – 2016-02-29) as a validation set to optimize the parameters. The results are shown in Table 2. The maximum depicted values for θ^{p}_{supp} and θ_{supp} are 3,686 tweets (i.e., 10%), given that there exist 36,868 positive examples (out of 2,011,085) in our training set. Note that we exclude the messages

Table 3: RecPar is compared to FreqItem in terms of precision (PREC), recall (REC), f1 score (F1) and accuracy (ACC).

Model	PREC	REC	F1	ACC
RecPar	0.48	0.47	0.47	0.83
FreqItem	0.25	0.60	0.35	0.64

that refer to multiple persons and locations as it is not clear how to assign one of the locations to one of the persons. Examining the word order in the text with the help of a syntax parser is a challenging problem and we leave this task for future work.

Initially, θ_{supp} is set constant and θ^{p}_{supp} is decreased, and then vice versa. By setting the pivot support higher than the overall support, we aim to be strict with our seed set so that limited ensemble models are created. The first conclusion we draw is that both thresholds affect RecPar's performance, but not drastically. For instance, in a strict setting where a pivot term has to appear in least 3,687 tweets (θ^{p}_{supp}=10%), the precision and the TNR are improved only by approximately 1% in comparison to the softest constraint (θ^{p}_{supp}=0.25%). Similarly for the θ_{supp}, its second highest value achieves the most successful result.

Another interesting finding is the crucial contribution of the candidate combination phase to RecPar's performance. It is evident that the naive usage of all subqueries would achieve poor precision results (first column under PREC). The reason behind this is that RecPar's first phase is recall-oriented and the candidates of this phase accomplish 95-99% True Positive Rate (TPR) and False Positive Rate (FPR). However, the combination phase improves the precision by a factor of 4 and the TNR by more than an order of magnitude. Additionally, our experiments show that the second phase diminishes the FPR and boosts the TNR significantly, leading to fewer noisy and irrelevant tweets in our dataset. To conclude, for the rest of our study, we use RecPar's best query combination, which is learned in 2015-11-01 – 2016-01-31 with $\theta^{p}_{supp} = 10\%$ and $\theta_{supp} = 5\%$.

5.1.2 Comparison between filtering approaches. This tweet-based experiment is an intermediate evaluation of our overall approach, before the evaluation of the event discovery. We measure how many of the remaining tweets after the filter are correct (i.e., refer to real events). We compare against the previously introduced approach for person tracking [4]. Similarly to RecPar, we apply FreqItem's query to every tweet t in the test set and if t satisfies it, then we classify t to the correct class. We expect FreqItem to perform poorer than RecPar, due to the fact that it is trained with a very small set of correct tweets and because it does not support negative predicates ($\neg w$).

Furthermore, the recursive nature of RecPar and the higher diversity of its query candidates, originating from independent data partitions in the generation phase, should lead to better queries. In contrast, in this work, we leverage millions more tweets and anticipate that the recursive nature of RecPar will dominate the naively constructed queries of FreqItem. The test set for both approaches is March 2016 (subsequent to RecPar's validation set).

As depicted in Table 3, RecPar prevails in terms of precision, f1-measure, and accuracy, since it generates more sophisticated and carefully designed queries, which guarantee that the result set will contain more relevant than irrelevant tweets. However, FreqItem achieves a higher recall, because it generates a very high amount of naive queries and hence many relevant (and irrelevant) tweets are covered by it.

Table 4: The results of our proposed solution (RecPar+CoPT), compared to alternative variations (RecPar+PoPT, CoPT, PoPT), the existing person tracking technique (FreqItem+Po), a naive baseline (Po) and the event detection algorithm MABED [5].

Approach	PREC	REC	F1
RecPar+CoPT	**0.68**	0.43	**0.53**
RecPar+PoPT	0.64	0.37	0.47
CoPT	0.32	0.24	0.28
PoPT	0.17	0.23	0.19
FreqItem+Po	0.15	0.67	0.25
Po	0.01	**0.87**	0.02
MABED	0.14	0.00	0.00

5.2 Person tracking

We now evaluate our constraint-based approach (CoPT) on the promising filtered tweets. We initially show the necessity of our realistic constraints (*population*, *popularity* and *distance*) by comparing our proposed solution RecPar+CoPT to RecPar+PoPT (**Po**pularity-based **P**erson **T**racking), which considers only the *popularity* of an event on Twitter. We use a Random Forest classifier for both approaches. Our goal is to see whether this obvious and simple constraint is adequate to retrieve the locations of the target individuals.

In addition, not only does RecPar enhance our overall efficiency in terms of time and memory consumption, but it also improves the tracking performance. Thus, in order to show the filter's necessity, we compare against CoPT and PoPT without filtering the tweets. As discussed earlier, our previous technique [4] applies FreqItem at first and then it assumes that each event that yields a *popularity* score higher than 10 corresponds to an actual event. We refer to this person tracking approach as FreqItem+Po (**Po**pularity) and we also compare simply against Po, as a naive baseline.

We train the above-mentioned models with events from April and May 2016 and test them monthly in a six-month period prior to the general elections in the US (from June till November). Various evaluation metrics are shown in Table 4, computed as an average of all test sets. RecPar+CoPT outperforms almost all techniques and competes closely to its variation, RecPar+PoPT, especially in terms of precision. That is, the *popularity* of a discussed event in social media is a very strong indicator about its validity, but not enough on its own. The importance of the RecPar phase is also evident, since CoPT and PoPT cannot outperform our overall proposed approach. Moreover, FreqItem+Po and Po achieve higher probability of detection (REC) than RecPar, due to their simplistic nature.

5.2.1 As time goes by. Multiple events related to our target persons happened prior to the US general elections[9], e.g., primaries/-caucuses in June, e-mail leakage in July and October, the Green National Convention in August, the first presidential debate in September, etc. In order to explore how the models work on each occasion, we show the monthly precision values in Table 5. We see that our person tracker outperforms all competitors, while having similar results to RecPar+PoPT for certain tests sets. For instance, in August 2016, the two techniques perform the same and in October 2016, RecPar+PoPT outstrips RecPar+CoPT.

As expected, the number of published tweets enhances significantly the performance of RecPar+PoPT, which can be observed in October 2016, the month with the highest amount of published tweets. However, our proposed combination RecPar+CoPT appears to be more consistent and robust, by always achieving a minimum precision of 60% and maintaining satisfying recall levels, as depicted in Table 4 as well.

5.2.2 Person tracking as event detection. One can argue that tracking the locations of mentioned entities in social media is a problem that can be tackled by an event detection algorithm. We hypothesize that existing literature on event discovery will not be as successful for our task, since the works are not focused on the involved individuals and thus, they will discover other events in our dataset that the target entities did not attend. To verify our intuition, we consider another competitor, namely MABED, a mention-anomaly-based event detection algorithm [5]. MABED leverages the creation frequency of dynamic mentions to discover events. Noise is avoided by allowing fine-tuned and dynamic events, which do not have to fit to a predefined time duration. This setting serves as a helpful noise "filter", given our highly imbalanced dataset.

An event is defined in MABED by a starting and ending date, a main keyword, and a set of related terms. We are looking for person and location mentions in these keywords by applying *CohEEL* and we use the event timeframe to create event triplets. As long as the detected event exists in our ground truth, we consider it a true positive.

In addition, the system is user-parametrizable and we tune it appropriately for our task. Namely, after experimenting with different parameter settings, we set the time window to 120 minutes to allow medium time precision and the number of words describing an event to 10. Increasing this number did not improve our results, because the longer the event summary is, the more are the chances that multiple politicians and locations are included in it and our evaluation setting does not allow such cases (as discussed in Section 5.1). The threshold for selecting relevant words is the default one (0.6). Since we perform monthly experiments and the most popular month in our dataset contains 400 events, we set k (the maximum number of returned events in MABED) to 400.

Unsurprisingly, we can see in Table 4 that MABED is not performing well, specifically it is unable to capture almost any event in our ground truth and it achieves similar precision to PoPT and FreqItem+Po. We observed that MABED can generally capture the political discussions and oftentimes, there exist mentions of presidential candidates and U.S. cities in the event descriptions. However, at

[9]https://en.wikipedia.org/wiki/United_States_presidential_election,_2016_timeline

Table 5: Monthly precision results in 2016 for each technique.

Approach	June	July	Aug.	Sept.	Oct.	Nov.
RecPar+CoPT	**0.62**	**0.66**	**0.66**	**0.67**	0.72	**0.80**
RecPar+PoPT	0.56	0.64	**0.66**	0.60	**0.80**	0.74
CoPT	0.14	0.26	0.45	0.34	0.31	0.47
PoPT	0.16	0.13	0.21	0.17	0.19	0.17
FreqItem+Po	0.13	0.13	0.16	0.14	0.21	0.15
Po	0.01	0.01	0.01	0.01	0.01	0.01
MABED	0.10	0.00	0.06	0.14	0.20	0.33

least one item in the discovered event triplets (`person-location-date`) is usually incorrect and thus the triplet does not refer to an actual location that a person visited on a certain date. This confirms our hypothesis that event detection models are not designed for predicting the precise locations of people mentioned in social media. The results are also not as consistent as of other models, e.g., there are no true positives discovered by MABED in July 2016, as shown in Table 5.

6 CONCLUSIONS AND FUTURE WORK

In this work, we tackled the problem of person tracking via online discussions in social networks. It is shown that social media posts reveal more than the obvious and they make it feasible to discover which places the discussed individuals visit and when. Our proposed approach extracts facts from tweet text and it could be applied to any domain whose entities move over time. The problem we study has several applications, such as detecting singers' concerts, politicians' speeches, companies' relocations, sport teams' games etc., but also in emergency situations, one can identify mentions of missing persons or any kind of threat, such as, fugitives, criminals etc.

We introduced RecPar, a recursive partitioning algorithm, which carefully generates queries for the Twitter API that return relevant information to the target entities and their locations. An extensive experimental analysis was conducted to examine RecPar's behavior and optimize its input parameters. We also proposed a constraint-based person tracking approach (CoPT), which reasons over the filtered tweets and categorizes the mentioned events as true or false. Social media as well as location characteristics were used to classify the events. Our overall person tracking method (RecPar + CoPT) outperformed the previously introduced tracking technique [4], the event detection algorithm MABED [5] and multiple baselines.

The more tweets are used for tracing the target entities, the more correct events can be discovered. Namely, one can use more sophisticated methods for assigning a time to an event, i.e., temporal labeling of the tweets instead of considering the publication time of the message. In this way, more tweets would contribute to the detection of the events and our intuition says that the person tracking results can be further enhanced. We currently perform daily analysis, i.e., we use the tweets of a certain day d to discover the events happened on d. Thus, we allow users who discuss events asynchronously, but only within 24 hours. One can use temporal expressions [8], e.g., yesterday, 2night, tomorrow and also dates mentioned in the text. Given a more flexible temporal tagging approach, we can update our confidence not only about today's events, but also about other future and past events.

Moreover, in order to report the crowd's impression about an individual's event and also the anticipation for an upcoming event (e.g., how inspiring a TED talk was by an entrepreneur and how long-awaited the next event is), our system could additionally provide the average sentiment of the past and newly published tweets respectively. Another interesting improvement would be to consider various granularities of locations. For instance, considering fine-grained locations, such as towns and villages, can be useful when the individual of interest is a national risk and the accuracy of the reported information about him/her is essential. More abstract locations, such as on a country-level, might also be adequate for entertainment or business related activities, such as concerts and conferences.

REFERENCES

[1] Hamed Abdelhaq, Christian Sengstock, and Michael Gertz. 2013. EvenTweet: online localized event detection from twitter. *VLDB Endowment* 6 (2013), 1326–1329.

[2] Lars Backstrom, Eric Sun, and Cameron Marlow. 2010. Find me if you can:Improving Geographical Prediction with Social and Spatial Proximity. In *WWW*. North Carolina, USA, 61 – 70.

[3] Toni Gruetze, Gjergji Kasneci, Zhe Zuo, and Felix Naumann. 2016. CohEEL: Coherent and Efficient Named Entity Linking through Random Walks. *Web Semantics: Science, Services and Agents on the World Wide Web* 37 (2016), 75–89.

[4] Toni Gruetze, Ralf Krestel, Konstantina Lazaridou, and Felix Naumann. 2017. What Was Hillary Clinton Doing in Katy, Texas?. In *WWW*. Perth, Australia, 783–784.

[5] Adrien Guille and Cécile Favre. 2014. Mention-anomaly-based event detection and tracking in twitter. In *ASONAM*. Beijing, China, 375–382.

[6] Bahareh Rahmanzadeh Heravi and Ihab Salawdeh. 2015. Tweet Location Detection. *Computation Journalism Symposium* (2015).

[7] Mengdie Hu, Shixia Liu, Furu Wei, Yingcai Wu, John Stasko, and Kwan-Liu Ma. 2012. Breaking News on Twitter. In *SIGCHI*. Texas, USA, 2751–2754.

[8] Ali Hürriyetoglu, Nelleke Oostdijk, and Antal van den Bosch. 2014. Estimating Time to Event from Tweets Using Temporal Expressions. In *EACL*. Gothenburg, Sweden, 8–16.

[9] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a social network or a news media?. In *WWW*. North Carolina, USA, 591.

[10] Chenliang Li and Aixin Sun. 2014. Fine-grained location extraction from tweets with temporal awareness. In *SIGIR*. 43–52.

[11] Rui Li, Kin Hou Lei, Ravi Khadiwala, and Kevin Chen-Chuan Chang. 2012. Tedas: A twitter-based event detection and analysis system. In *ICDE*. VA, USA, 1273–1276.

[12] Debanjan Mahata, John R. Talburt, and Vivek Kumar Singh. 2015. From Chirps to Whistles: Discovering Event-specific Informative Content from Twitter. In *WebSci*. Oxford, United Kingdom, 1–110.

[13] Jalal Mahmud, Jeffrey Nichols, and Clemens Drews. 2014. Home Location Identification of Twitter Users. *ACM Transactions on Intelligent Systems and Technology* 5 (2014), 1–21.

[14] Michael Mathioudakis and Nick Koudas. 2010. TwitterMonitor: trend detection over the twitter stream. In *SIGMOD*. Indiana, USA, 1155–1158.

[15] Eni Mustafaraj and Panagiotis Takis Metaxas. 2017. The Fake News Spreading Plague: Was It Preventable?. In *WebSci*. Troy, New York, USA, 235–239.

[16] Adam Poulston, Mark Stevenson, and Kalina Bontcheva. 2017. Hyperlocal Home Location Identification of Twitter Profiles. In *HT*. Prague, Czech Republic, 45–54.

[17] Søren B. Ranneries, Mads E. Kalør, Sofie Aa. Nielsen, Lukas N. Dalgaard, Lasse D. Christensen, and Nattiya Kanhabua. 2016. Wisdom of the Local Crowd: Detecting Local Events Using Social Media Data. In *WebSci*. Hannover, Germany, 352–354.

[18] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users. In *WWW*. North Carolina, USA, 851.

[19] Justin Sampson, Fred Morstatter, Ross Maciejewski, and Huan Liu. 2015. Surpassing the Limit: Keyword Clustering to Improve Twitter Sample Coverage. In *HT*. Guzelyurt, Northern Cyprus, 237–245.

[20] Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. 2009. TwitterStand: news in tweets. In *SIGSPATIAL*. Seattle, Washington, 42.

[21] Axel Schulz, Aristotelis Hadjakos, Heiko Paulheim, Johannes Nachtwey, and Max Mühlhäuser. 2013. A Multi-Indicator Approach for Geolocalization of Tweets.. In *ICWSM*. Boston, USA, 573–582.

[22] Yangqiu Song, Zhengdong Lu, Cane Wing-ki Leung, and Qiang Yang. 2013. Collaborative Boosting for Activity Classification in Microblogs. In *SIGKDD*. Chicago, IL, USA, 482–490.

[23] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*. Alberta, Canada, 697–706.

[24] Dennis Thom, Harald Bosch, Robert Krueger, and Thomas Ertl. 2014. Using large scale aggregated knowledge for social media location discovery. In *HICSS*. HI, USA, 1464–1473.

[25] Shiguang Wang, Prasanna Giridhar, Hongwei Wang, Lance Kaplan, Tien Pham, Aylin Yener, and Tarek Abdelzaher. 2017. StoryLine: Unsupervised Geo-event Demultiplexing in Social Spaces without Location Information. In *IoTDI*. Orlando, Florida, 83–94.