Erschienen in: The Computer Journal ; 60 (2017), 3. - S. 329-346
https://dx.doi.org/10.1093/comjnl/bxw056

Survey and Experimental Analysis of Event Detection Techniques for Twitter

ANDREAS WEILER*, MICHAEL GROSSNIKLAUS AND MARC H. SCHOLL

Department of Computer and Information Science, University of Konstanz, P.O. Box 188, 78457 Konstanz, Germany
**Corresponding author: Andreas.Weiler@uni konstanz.de*

Twitter's popularity as a source of up-to-date news and information is constantly increasing. In response to this trend, numerous event detection techniques have been proposed to cope with the rate and volume of Twitter data streams. Although most of these works conduct some evaluation of the proposed technique, a comparative study is often omitted. In this paper, we present a survey and experimental analysis of state-of-the-art event detection techniques for Twitter data streams. In order to conduct this study, we define a series of measures to support the quantitative and qualitative comparison. We demonstrate the effectiveness of these measures by applying them to event detection techniques as well as to baseline approaches using real-world Twitter streaming data.

Keywords: performance evaluation; event detection; Twitter data streams

1. INTRODUCTION

Microblogging is a form of social media that enables users to broadcast short messages, links and audiovisual content to a network of *followers* as well as to their own public timeline. In the case of Twitter, one of the most popular microblogging services, these so-called *tweets* can contain up to 140 characters. Twitter's 316 million monthly active users produce a total of over 500 million tweets per day.[1] As a consequence, several proposals have been made to leverage Twitter as a source of up-to-date news and information, e.g. to respond to natural disasters [1], to track epidemics [2] or to follow political elections [3].

A number of techniques have been designed and developed to detect such events in the Twitter social media data stream. Typically, they adopt the definition of an *event* introduced by research on Topic Detection and Tracking (TDT), i.e. a real-world occurrence that takes place in a certain geographical location and over a certain time period [4]. The main focus of these event detection techniques lies in addressing the specific requirements introduced by Twitter data, such as the brevity of tweets together with the fact that they contain a substantial amount of spam, typos, slang, etc. Although most proposals provide some qualitative evidence to motivate the benefits of the technique, few perform a quantitative evaluation or compare their results to competing approaches.

We argue that this lack of comparative evaluation without an existing manually created ground truth is explained by the fact that measuring the quantitative and qualitative performance of event detection techniques for Twitter data is itself a challenging research question. Crafting a gold standard manually in order to use textbook precision and recall measures is painstakingly slow and does therefore not scale to the volumes of data generated by Twitter users. In order to address this requirement, we build on our previous work in this field [5] and, in this paper, propose several scalable measures that can be automatically applied to the results of current and future event detection techniques. The specific contributions of this paper are as follows:

(1) Extensive survey of existing evaluation methods for event detection techniques for Twitter (Section 2).
(2) Definition of new evaluation measures to automatically evaluate the run-time and task-based performance of event detection techniques (Section 3).

[1]https://about.twitter.com/company/ (15 April 2016).

(3) Realization of several state-of-the-art event detection techniques as query plans in a data stream management framework (Section 4).
(4) Detailed study using real-life Twitter data that demonstrates the ability of our measures to evaluate the different techniques (Section 5).

As our evaluation approach is platform-based and modular, it will also enable further systematic performance studies of future event detection techniques.

This article is an extended presentation of Weiler *et al.* [6]. In comparison to the conference version, it features two additional contributions. First, this article provides a much more detailed survey of the state of the art in evaluation methods for event detection techniques for Twitter data streams. Since we also analyze the most recent approaches that have not yet been discussed in other surveys, this article closes a gap to these surveys with respect to existing evaluation methods of event detection techniques. Second, the collection of existing event techniques available in our platform has been extended to include *enBlogue* (ENB) [7]. As a consequence, this article also uses this additional technique to analyze the proposed evaluation measures. Apart from these two new contributions, this article also studies the run-time and task-based performance of event detection techniques in more depth by applying newly defined measures. We additionally examine the memory requirements of each technique and also measure the run-time performance on a second hardware setting.

2. BACKGROUND

In recent years, several research works have been conducted in the area of event detection and tracking techniques for Twitter. Consequently, a number of surveys exist that document the current state of the art. For example, the survey presented by Nurwidyantoro and Winarko [8] summarizes techniques to detect disaster, traffic, outbreak and news events. The survey by Madani *et al.* [9] presents techniques that each address one of the four challenges of health epidemics identification, natural events detection, trending topics detection and sentiment analysis. A more general survey with a wide variety of research topics related to sense making in social media data is the survey presented by Bontcheva and Rout [10]. The work defines five key research questions user, network and behavior modeling as well as intelligent and semantic-based information access. The part about semantic-based information access also includes an overview about event detection techniques in social media data streams. They classify event detection methods into three categories: clustering-based, model-based and those based on signal processing. Furthermore, an overview about techniques for 'sub-events' detection is presented. Finally, the most extensive survey on event detection techniques is presented by Farzindar and Khreich [11] with a listing of different techniques categorized by their detection methods, tasks, event types, application domains and evaluation metrics. Based on these surveys and the contained works, we can observe that most existing techniques are evaluated using *ad hoc* measures together with manually labeled reference data sets. Furthermore, only very few of the surveyed techniques have been compared against competing approaches.

In the following, we first summarize the presented evaluation methods used in existing works on event detection techniques. Second, we present available corpora that could be used to evaluate event detection techniques.

2.1. Evaluation of event detection approaches

In 2014, the 'Social News on the Web' (SNOW) challenge [12] attempted to compare different event detection techniques. In order to evaluate the different results of the submitted solutions, the measures of precision and recall, readability, coherence/relevance, and diversity were used. However, instead of evaluating the different submissions (11 teams) automatically, a manual evaluation was conducted by a group of human evaluators. This choice made by the organizers of the SNOW challenge is one example that demonstrates that evaluating event detection techniques automatically is a very challenging and complex problem. As a consequence, diverse manual and semi-automatic proposals for evaluation methods exist in the literature. Table 1 lists existing works on event detection techniques for Twitter and summarizes what methods were used to evaluate them. In the *Survey* column of the table, we indicate whether a given technique is already contained in one of the aforementioned surveys.

In the Application Programming Interface (API) column, we list the different Twitter APIs that are used to collect the data for the evaluation. Most of the works are based on the Streaming API, but with different levels or restrictions. The Filter API (12 of 42) is the most popular choice. With this API, it is possible to obtain the data in a streaming fashion and to pre-define filter queries based on keywords or geographical locations. The Spritzer access level (7 of 42) provides a uniform random 1% stream of the public timeline and is freely available to everyone. In contrast, the Gardenhose level (6 of 42) provides elevated access to a 10% stream, but needs some special authorization. Additionally, two special streams are available. First, the User Stream allows a pre-defined set of users to be followed directly and continuously. Second, the Trends Stream contains currently trending topics either on a global level or filtered by a geographic region. Apart from these streaming APIs, the Search API (5 of 42) can be used to retrieve tweets that match a given query. Which of these APIs is used to evaluate event detection techniques also impacts the number of tweets that can be retrieved (*cf.* column *Tweets*). The sizes of these collections range from 0.6 million to around 1.2 billion tweets.

TABLE 1. Evaluation methods of event detection techniques for Twitter.

Year	Reference	Survey	Type	GT	Measures	Tweets	API
2009	Ritterman *et al.* [13]	[8]	Case study			48 m	Filter
2009	Sankaranarayanan *et al.* [14]	[11]	Case study				Gardenhose, User Stream
2009	Schühmacher and Koster [15]						
2010	Benhardus [16]	[9]	Stand alone, user study	TT	Prec, rec, F_1		Gardenhose, Trends
2010	Cataldi *et al.* [17]	[9]	Case study			3 m	Spritzer
2010	Culotta [2]	[9]	Stand alone	CDC	Correlation	0.6 m	Search
2010	Lee and Sumiya [18]	[11]	Stand alone	Manual	Prec, rec	22 m	Filter
2010	Mathioudakis and Koudas [19]	[9]	Case study				Gardenhose
2010	Petrović *et al.* [20]	[8, 10, 11]	Stand alone	Manual	Avg. prec	163.5 m	Spritzer
2010	Sakaki *et al.* [1]	[8 11]	Stand alone, case study	Manual	Prec, rec, F_1		Search
2011	Achrekar *et al.* [21]	[8]	Stand alone	CDC	Correlation	4.7 m	Search
2011	Becker *et al.* [22]	[10, 11]	Stand alone	Manual	F_1	2.6 m	
2011	Lee *et al.* [23]		Case study			27 m	Spritzer
2011	Marcus *et al.* [24]	[10]	Stand alone	Manual	Prec, rec		
2011	Popescu *et al.* [25]	[11]	Stand alone	Manual	Prec, rec, F_1, avg. prec, a.r.o.c.	6000	
2011	Weng and Lee [3]	[8, 11]	vs. LDA	Manual	Precision	4.3 m	Search
2012	Abel *et al.* [26]	[8]					
2012	Adam *et al.* [27]	[8]					
2012	Aggarwal and Subbian [28]		Case study, stand alone	Manual	Prec, rec	1.6 m	Search
2012	Alvanaki *et al.* [7]		User study, vs. TM		Prec, runtime, rel. accuracy		Gardenhose
2012	Cordeiro [29]	[11]	Case study			13.6 m	Spritzer
2012	Ishikawa *et al.* [30]	[8]	Case study				
2012	Li *et al.* [31]	[8]	Case study, vs. EDCoW	Manual	Prec, rec, DER	4.3 m	Spritzer
2012	Li *et al.* [32]		Case study			0.1 m	Filter
2012	Nishida *et al.* [33]					0.3 m	Filter
2012	Osborne *et al.* [34]	[8]	Stand alone	Wik	Latency	48 m	Spritzer
2012	Ritter *et al.* [35]		Stand alone	Manual	Prec, rec	100 m	Spritzer
2012	Terpstra *et al.* [36]	[8]	Case study			0.1 m	Filter
2013	Aiello *et al.* [37]		vs. LDA	Manual	Prec, rec	0.3 m, 0.7 m, 3.6 m	Filter
2013	Bahir and Peled [38]		Case study				Filter
2013	Martin *et al.* [39]		Stand alone	Manual	Prec, rec		Filter
2013	Parikh and Karlapalem [40]		User study	Manual	Prec, rec, rt	1.3 m	Filter
2013	Walther and Kaisser [41]		Stand alone	Manual	Prec, rec, F_1		
2013	Abdelhaq *et al.* [42]		Case study			0.07 m	Filter
2013	Weiler *et al.* [43]		Case study			2 m	Gardenhose

Continued

TABLE 1. Continued

Year	Reference	Survey	Type	GT	Measures	Tweets	API
2014	Corney *et al.* [44]		Case study				
2014	Guille and Favre [45]		vs. TS, ET	Manual	Prec, rec, DER	3.5 m	Filter
2014	Ifrim *et al.* [46]		Stand alone	Manual	Precision	1 m	Filter
2014	Weiler *et al.* [47]		Case study			2 m	Gardenhose
2014	Zhou and Chen [48]		vs. OLDA	Manual	MD, FA	1.16b	Filter
2015	Meladianos *et al.* [49]		Stand alone	Match facts	Prec, rec, F_1	7.5 m	
2015	Thapen *et al.* [50]		User study	Manual	Prec, rec, F_1	96 m	Filter

The *Survey* column lists the survey(s), in which the technique is included. The *Type* column denotes the evaluation type used in the work, e.g. case study, stand alone, user study or comparison (vs.) with another technique (LDA, Latent Dirichlet Allocation [51]; TM, TwitterMonitor [19]; EDCoW, Event Detection With Clustering of Wavelet based Signals [3]; TS, Streaming Trend Detection in Twitter [16]; ET, Events from Tweets [40]; OLDA, Online LDA [52]). The *GT* column indicates which ground truth was used for the evaluation (TT, Twitter Trending Topics; CDC, Statistics of the Centers for Disease Control and Prevention). The *Measures* column specifies the used evaluation measures (prec, precision; rec, recall; F_1, F_1 score; DER, duplicate event recognition; rt, runtime; MD, missed detections; FA, false alarms). The *Tweets* column lists the number of tweets used in the evaluation. The *API* column gives the Twitter API used to collect the tweets.

For example, Ritterman *et al.* [13] use 48 million tweets that they crawled using the Filter API for a pre-defined set of domain-specific keywords over a 3-month period from April to June 2009, whereas Sankaranarayanan *et al.* [14] follow a hand-picked crowd of Twitter users that are known to publish news by using the User Stream API.

The *Type* column lists the different evaluation methods. We can observe that most of the works (17 of 42) performed one or several case studies to show the effectiveness and usefulness of their technique. Note that works that are evaluated by demonstrations are also marked with ‘case study’.

For example, Mathioudakis and Koudas [19] simply refer to a website that gives access to their TwitterMonitor (TM) technique and do not state whether they conducted any other evaluations. Another large group of works (15 of 42) performs a stand-alone evaluation in order to rate the outcomes of their own technique only. In this case, the focus of the evaluation consists in tuning different parameters or improving the different steps of a single technique. Unfortunately, the results obtained with this type of evaluation are very hard to interpret in terms of comparing them to other techniques. Only 6 of the 42 surveyed works perform a comparative evaluation. For example, Weng and Lee [3] and Aiello *et al.* [37] compared their technique with Latent Dirichlet Allocation (LDA) [51], whereas Zhou and Chen [48] compared their technique to the online version of LDA (OLDA [52]). Alvanaki *et al.* [7] compared their technique with Mathioudakis and Koudas [19], while Li *et al.* [31] compared their technique to Weng and Lee [3]. Finally, Guille and Favre [45] compared their technique to both Benhardus [16] (TS) and Parikh and Karlapalem [40] (ET). We also note that Aiello *et al.* [37] performed further comparative evaluations for a total of five self-defined event and topic detection techniques. A small fraction of evaluations (4 of 42) is based on user studies, where the results are shown to human evaluators.

The Ground Truth (GT) column gives an overview of different types of ground truths that are used to evaluate the results of a technique. Most of the works (17 of 42) use a manually labeled set of events as ground truth. Some of them also check the results of the technique manually to distinguish between real or non-real events. For example, Walther and Kaisser [41] checked manually for 1000 clusters whether they belonged to a real-world event or not. Three-hundred and nineteen clusters were labeled as positives (describe a real-world event), while the remaining 681 were labeled as negatives (do not describe a real-world event). We note that domain-specific event detection techniques can often be evaluated using an existing ground truth. For example, the statistics of the Centers for Disease Control and Prevention (CDC) can be used as ground truth to evaluate techniques that detect diseases (2 of 42). Similarly, match reports can be used for sport events, such as football games (*cf.* Meladianos *et al.* [49]). Finally, one work uses Wikipedia (Wiki) and another one uses Twitter’s Trending Topics (TT) as ground truth.

The *Measures* column summarizes the different measures that were proposed to evaluate the different techniques. Most of the times, the precision and recall measures are used (15 of 42). Additionally, some works calculate the F_1 score, average precision or the area under the receiver-operating curve (a.r.o.c.). While measures to evaluate the task-based performance of a technique are quite common, only the two works of Alvanaki *et al.* [7] and Parikh and Karlapalem [40] apply a measure (rt) to evaluate the run-time performance of their technique. Apart from these well-known measures, some novel measures were defined. For example, Alvanaki *et al.* [7] measure *relative accuracy*, whereas both Li *et al.* [31] and Guille and Favre [45] study the duplicate event rate (DER) of their techniques. Finally, Zhou and Chen [48] introduce two new measures: missed detections (MD) and false alarms (FA).

Benhardus [16] compares the outcome of his event detection technique to the Twitter TT. Twitter extracts these topics using a proprietary algorithm, which they describe as an 'algorithm that identifies topics that are popular now, rather than topics that have been popular for a while or on a daily basis.'[2] Trending topics can be crawled continuously using the Trends API[3] of Twitter. In the case of Benhardus [16], trending topics were only collected for the USA. The evaluation consisted of two experiments. In the first experiment, the precision, recall and F_1 score were compared with respect to the Twitter TT. In the second experiment, recall and relevance scores were calculated using human volunteers that labeled a list of valid terms and relevant topics. The results of the first experiment reflect that the precision, recall and F_1 score are very low as the average around 0.2 0.3. However, with the introduction of the human factor in the second experiment, the average value of the F_1 score increased to ~0.6 0.7.

Cullota [2] presents a domain-specific event detection technique for detecting influenza epidemics. Using the Search API with specific keywords, 574 643 tweets were collected for the 10-week period from 12 February 2010 to 24 April 2010. For evaluation purposes, statistics from the CDC were used in order to check for correlation with the outcome of the proposed models. The evaluation comprised 10 different models, with the best model achieving a correlation of 0.78 to the CDC statistics by leveraging a document classifier to identify relevant messages.

Lee and Sumiya [18] present a work on Twitter-based geo-social event detection and evaluate their results against a set of manually labeled events that occurred in the context of town festivals in Japan. Using the Filter API to only obtain tweets originated in the circumference of Japan, they crawled ~22 million tweets from the beginning of June to the end of July 2010. The results of the evaluation indicate a high recall value of 0.87 (13 out of 15 events detected). In contrast, the precision value of 0.018 is very low (13 of 724 reported events matched). However, closer analysis of the reported events showed that while some were unexpected, they were still interesting events.

Petrović *et al.* [20] evaluate their work on streaming first story detection with application to Twitter using a *gold standard*, which consists of manually labeled events. The data set for the evaluation consisted of 163.5 million tweets, collected over a period of 6 months using the Streaming API with the 1% Spritzer access. They demonstrated that their system can detect major events with reasonable precision and that the amount of spam in the output can be reduced by taking their definition of entropy into account.

Sakaki *et al.* [1] present a system for real-time event detection that uses Twitter users as *social sensors*. Their system is specifically tailored to detect events in data sets that are highly pre-filtered for disasters (e.g. for earthquakes using the filter word 'earthquake'). They prepared 597 positive examples, which report earthquake occurrences as a reference set. While they used different features within their classification methods, the best score they achieved was 0.87 for recall, 0.66 for precision and 0.73 for the F_1 score. Additionally, they present two case studies, in which they demonstrate the effectiveness of the proposed approach for detecting and following an earthquake and a typhoon.

Becker *et al.* [22] propose a system for real-world event identification on Twitter. In order to evaluate their approach, they crawled 2.6 million tweets in February 2010. They used the first 2 weeks for training the system, whereas the second 2 weeks were used as testing data. Human annotators produced a set of labeled event clusters. To evaluate the performance of each of their defined classifiers, they use a macro-averaged F_1 score. They compared the results of their approach to two baseline methods: fastest and random. As a result, their own approach had a higher score than the two other ones.

TwitInfo, presented by Marcus *et al.* [24], is a tool for aggregating and visualizing microblogs for event exploration. Their evaluation uses manually labeled events from soccer games and automatically extracted earthquake occurrences from the US Geological Survey. For soccer game events, they scored 0.77 in both precision and recall (17 of 22 events found). For major earthquakes, the score was 0.14 (6 out of 44) for precision and 1.0 for recall (5 out of 5). Therefore, they concluded that their peak detection algorithm identifies 80 100% of manually labeled peaks.

Popescu *et al.* [25] evaluate their work on extracting events and event descriptions from Twitter against a manually classified gold standard of 5040 snapshots, which were classified as events (2249) or non-events (2791). Their technique, which is called EventBasic, scores 0.691 for precision, 0.632 for recall, 0.66 for the F_1 score, 0.751 for average precision and 0.791 for the a.r.o.c. Their extension of EventBasic, which is called EventAboutness, does not show any improvements in the results, as its scores are almost the same.

Weng and Lee [3] present the above-mentioned EDCoW technique and evaluate their work by using a highly restricted data set containing only tweets from the top 1000 Singapore-based Twitter users. Also, they used a very strong pre-filtering on unique words, resulting in only 8140 unique words being contained in the data set after cleaning. In their evaluation, EDCoW is applied to detecting events for each day in June 2010. However, they argue that 'it is not feasible to enumerate all the real-life events [that] happened in June 2010 in the dataset.' Since it is, therefore, difficult to measure EDCoW's recall, they chose to concentrate on precision. For the precision score, they calculated a value of 0.76 (16 of 21 detected events). Apart from this stand-alone evaluation, they

[2]https://support.twitter.com/articles/101125 (15 April 2016).

[3]https://dev.twitter.com/rest/reference/get/trends/place (15 April 2016).

also evaluated the task-based performance of EDCoW in comparison to LDA. However, rather than performing a quantitative evaluation, they qualitatively evaluate EDCoW by discussing the differences between its outputs and an example output of the LDA technique.

Aggarwal and Subbian [28] divide their evaluation into two parts. In the first part, they show a case study to evaluate the unsupervised model of their event detection technique. In the second part, they use a self-generated ground truth to evaluate precision and recall of the supervised model for two sample events (*Japan Nuclear Crisis* and *Uganda Protest*). For the first event, they obtain a value of 0.525 (precision) and 0.62 (recall), while the precision value is 1.0 and the recall value is around 0.6 for the second event. However, it is important to note that they work with highly pre-filtered data, which does not constitute a real-life evaluation of event detection techniques.

Alvanaki *et al.* [7] conduct an evaluation that is based on the decisions of human evaluators with respect to whether a reported result is an event or not. For this user study, they created a website that displayed the result of both their ENB and the existing TM technique to users [19]. Users were then able to check if in their opinion a result is an event or not, and mark it accordingly. Apart from measuring precision and relative accuracy for these techniques, they were also the first group of researchers to study run-time performance. They obtained the following results. In terms of precision, ENB clearly outperforms TM. On average, ENB detected 2.5 out of 20 events, whereas TM only detected 0.8. To evaluate run-time performance, they measured the correlation between the parameters of the two techniques and the increase/decrease of execution time. As a consequence, it is difficult to derive meaningful and general conclusions based on these measurements. The same is true for their relative accuracy measurements, which are done stand-alone for ENB and only demonstrate the interplay of different parameters.

Twevent, proposed by Li *et al.* [31], is a technique for segment-based event detection from tweets. The approach was evaluated by comparing it to EDCoW based on the same input data set as used by Weng and Lee [3]. However, rather than reproducing the results of EDCoW, they were simply taken from the paper by Weng and Lee. The results of their evaluation seem to indicate that Twevent outperforms the state-of-the-art technique EDCoW with respect to both precision and recall. With a precision score of 0.86, it improves over EDCoW by a value of 0.1. In terms of recall, Twevent finds 75 real events within a total of 101 detected events, whereas EDCoW finds 13 real events within a total of 21 detected events. In terms of the DER, Twevent achieves the lowest rate, even though the technique detects much more events than EDCoW (101 vs. 21 events). However, since the original paper on EDCoW [3] did not include an evaluation of the DER, it is profoundly unclear how Li *et al.* [31] calculated it, given that they did not use an implementation of EDCoW. Apart from these quantitative evaluations, they also present a case study that shows the usefulness of Twevent in real-life.

In their work, Osborne *et al.* [34] present a first study of latency for event detection based on different sources. They evaluate performance in terms of the average distance between each time-aligned Twitter first story and the corresponding nearest neighbor Wiki page title. They conclude that there is a delay between events breaking on Twitter and on Wiki with an advantage for Twitter.

Ritter *et al.* [35] present a method for open-domain event extraction in Twitter. In their study, they demonstrate that precision and recall are increased by their technique in contrast to a baseline technique. Furthermore, they present a sample of extracted future events on a calendar layout in order to show the quality of the obtained results.

In the context of the SocialSensor project, Aiello *et al.* [37] compare six topic detection methods (BNGram, LDA, FPM, SFPM, Graph-based and Doc-p) using three Twitter data sets related to major events, which differ in their time scale and topic churn rate. They define three scoring measures: topic recall, keyword precision and keyword recall. They observe that the BNgram method always achieves the best topic recall, while always preserving a relatively good keyword precision and recall. They also observe that standard topic detection techniques such as LDA perform reasonably well on very focused events, whereas their performance is much lower when considering more 'noisy' events.

A follow-up work, presented by Martin *et al.* [39], includes a similar evaluation. Additionally, they attempt to derive the best slot size for the BNgram technique as well as the best combination of clustering and topic ranking techniques. Therefore, this work does not really contribute to the problem of comparative evaluations of event detection techniques. However, they observed that the results are very different between the three data collections that they used. One difference is particularly striking: the topic recall is far higher for football (over 90%) than for politics (around 60 80%). Therefore, they concluded that results of an evaluation also depend on the events that happen in the studied time frames.

Parikh and Karlapalem [40] evaluate their system (ET) on two data sets: one is provided by VAST Challenge 2011, whereas the other is published by US-based users in January 2013. For evaluation purposes, they use the same definitions of precision and recall as Weng and Lee [3]. For the VAST data set, ET detected a total of 23 events, out of which two events were trivial and insignificant. Thus, the precision value is 0.91 and the recall is 21. For the second data set, ET obtained a total of 15 events, out of which only one event was not related to any real event, yielding a precision of 0.93 and a recall of 14. Note that in this case, recall is simply represented as the number of 'good' detected events. In order to quantify the performance of ET, they present an execution time of 157 seconds to detect events from a total of 1 023 077 tweets, which corresponds to a throughput of 6516 tweets/seconds.

MABED, proposed by Guille and Favre [45], is a mention-anomaly-based event detection technique for Twitter. They conducted experiments on both English and French Twitter data. In their evaluation, MABED is compared with both ET [40] and TS [16]. The results indicate that MABED leads to more accurate event detection and improved robustness in presence of noisy Twitter content. Additionally, MABED showed better performance than MABED with ignoring mentions. The authors also demonstrated that MABED outperforms ET and TS in all of their tests.

Zhou and Chen [48] present two novel measures in their evaluation: MD and FA. Furthermore, they use a very large data set of 1.16 billion tweets and present a comparative evaluation of three variants of their technique and OLDA. Their experimental results demonstrate the superiority of their proposed approach in terms of effectiveness and efficiency.

Finally, Meladianos *et al.* [49] present a very strict evaluation of their techniques, which is applied to events during soccer games. They create a ground truth for their evaluation by crawling live reports of games in a sports website. The experiments show that their technique clearly outperforms the baseline techniques on the sub-event detection task, and also produces good summaries. In addition, their algorithm managed to detect a majority of key sub-events during each match.

2.2. Available corpora for evaluation

In this work, we address the challenge of defining general evaluation measures that can be used to compare various event detection techniques. Complementary to our approach, other works focus on the creation of evaluation corpora for Twitter-related analysis techniques. In the following, we present a series of works, which provide labeled reference data sets.

For example, McCreadie *et al.* [53] created a set of ~16 million tweets for a 2-week period. Therefore, the proposed corpus contains an average of ~50,000 tweets per hour. Since no language filtering is performed, which can be estimated to retain ~30% of these tweets (*cf.* Fig. 2), we can assume that only ~4 800 000 tweets of the corpus are in English. Furthermore, their list of 49 reference topics for the 2-weeks period is very limited and no description is given how these topics were created. Finally, this corpus focuses on *ad hoc* retrieval tasks and is, therefore, not very well suited for large-scale evaluation of event detection approaches.

Becker *et al.* [22] created a Twitter corpus that consists of over 2 600 000 tweets posted during February 2010. Since they only used their own approach to detect and label the events, the corpus is strongly biased to their technique and not very well suited for general evaluation purposes. Furthermore, no list of reference events is provided and the data set is geographically restricted to tweets from users who are located in New York City.

Petrović *et al.* [54] presented a corpus of 50 million tweets, created from a manual analysis of the Twitter data stream from July to mid-September 2011. This analysis led to the definition of 27 events for the whole time frame. This very low number of labeled events makes it very difficult to compare different event detection methods, especially when the techniques used are very diverse.

Papadopoulos *et al.* [12] provide three corpora for the purpose of development, training and testing event detection techniques. The development data set consisted of 1 106 712 tweets that were previously collected during the 2012 US Presidential election [37]. The data set for training was formed by applying filtering rules to keywords and user names. The keywords were set to {'flood', 'floods', 'flooding'} and the user names were filtered for a list of 'newshounds'. Newshounds are Twitter users that tend to tweet about news or events. As a filter query a total of 5000 UK-focused newshounds are selected. For the testing data set, the same user filter is used and the keywords are replaced by the set {'Syria', 'terror', 'Ukraine', 'bitcoin'}. In addition, a ground truth of 59 topics from UK media stories for the collection of 1 041 062 tweets and a 24-hour period was generated.

McMinn *et al.* [55] propose a methodology for creating a corpus to evaluate event detection methods. They used two existing state-of-the art event detection approaches [28, 54] together with Wikipedia to create a set of candidate events together with a list of associated tweets. The final corpus covers 4 weeks with ~120 million tweets and more than 500 events. However, events are described in prose and can, therefore, not be easily and automatically compared to the results of the various event detection techniques.

It is important to note that all of these corpora only consist of lists of tweet identifiers, since the Twitter terms of use do not permit redistribution of the tweets themselves. In order to use a corpus, the corresponding tweets have to be crawled, which is time-consuming and error-prone as some tweets might not exist anymore. For example, the organizers of the 2014 SNOW challenge [12] could only crawl 1 106 712 of the original 3 630 816 tweets of the above-mentioned 2012 US Presidential Election data set [37]. In order to assess how useable these collections of tweet identifiers are, we attempted to download the corpus of McMinn *et al.* [55]. The standard restriction of crawling tweets with the Twitter API[4] is set to 180 queries per 15 minute window. With one query, it is possible to obtain a bulk of 100 tweets.[5] Therefore, it is possible to crawl 18 000 tweets per 15-minute window and it would take ~6666 windows with an estimated total response time of 100 000 minutes (~1666 hours or ~69 days) on a single machine to crawl all the contained tweets. As this waiting

[4]https://dev.twitter.com/rest/public/rate limiting (15 April 2016).

[5]https://dev.twitter.com/rest/reference/get/statuses/lookup (15 April 2016).

time is prohibitive in practice, we implemented an alternative crawler that only retrieves the content of the tweets based on their identifiers. Note that this crawler does not retrieve the metadata that is otherwise included when crawling tweets using the Twitter API. Even so, our crawler was only able to download ~740 000 tweets (out of 1 850 000 total checked), which were still available, in a time frame of 7 days.

3. MEASURES

In order to address the lack of a common evaluation method for event detection in Twitter data streams, we propose a number of measures that partially rely on external services as a ground truth. By following this approach, our work is in line with previous proposals, e.g. Google Similarity Distance [56] and Flickr Distance [57], which have shown that using an external service is a valid way to evaluate results of different tasks. Our goal is to define measures that can easily be used by other researchers and that do not deprecate over time as most reference corpora do. While all of our measures support relative comparisons, we do not claim that they can be used to draw absolute conclusions. A single event detection technique can, therefore, only be evaluated 'against itself', e.g. with respect to different parameter settings or to confirm that improvements to the technique yield better results. For a set of techniques, the measures can be used to rank them with respect to different criteria. In this article, we focus on the second application.

In order to be able to measure and compare the performance of different approaches, a common definition of *event* and *event detection technique* is required. We define an event detection technique to be a process of four steps: (i) preprocessing, (ii) event detection, (iii) event construction and (iv) event reporting. In the context of this article, we use the same processing for the first and the last step, in order to render the different techniques more comparable. This approach is valid, since the major differences between the techniques stem from the second and the third step, i.e. from how events are detected and constructed. For the purpose of the work presented in this article, an event is defined to be a set of five terms $e = \{t_1, t_2, t_3, t_4, t_5\}$ that is reported by one of the event detection techniques. Note, however, that the actual number of reported terms depends on step (iii) of the corresponding technique. As we will see in the next section, some of the techniques need to be slightly adapted in order to meet our common event definition.

3.1. Run-time performance measures

Throughput We measure the throughput as the number of tweets that an approach processes per second. This measure is important to judge the feasibility of a technique. Most event detection techniques can be configured based on numerous parameters that influence both the processing speed and result quality. In combination with other measures, the run-time performance measure can, therefore, also be used to study the trade-off between these two objectives.

Memory usage While the throughput evaluates the time needed to process a data set, this measure captures how much space is required by each technique. Since all techniques process a stream of data, i.e. processing never stops, we are interested to study how memory usage evolves over time and how it is bounded.

3.2. Task-based performance measures

Duplicate event detection rate This measure captures the percentage of duplicate events detected by an approach. The implementations of state-of-the-art event detection techniques used in this article avoid the reporting of duplicate events within their processing time-frame, e.g. a 1-hour window. Nevertheless, important or long-lasting events can reoccur across several time-frames and, therefore, expecting a 0% rate of duplicate events is not reasonable.

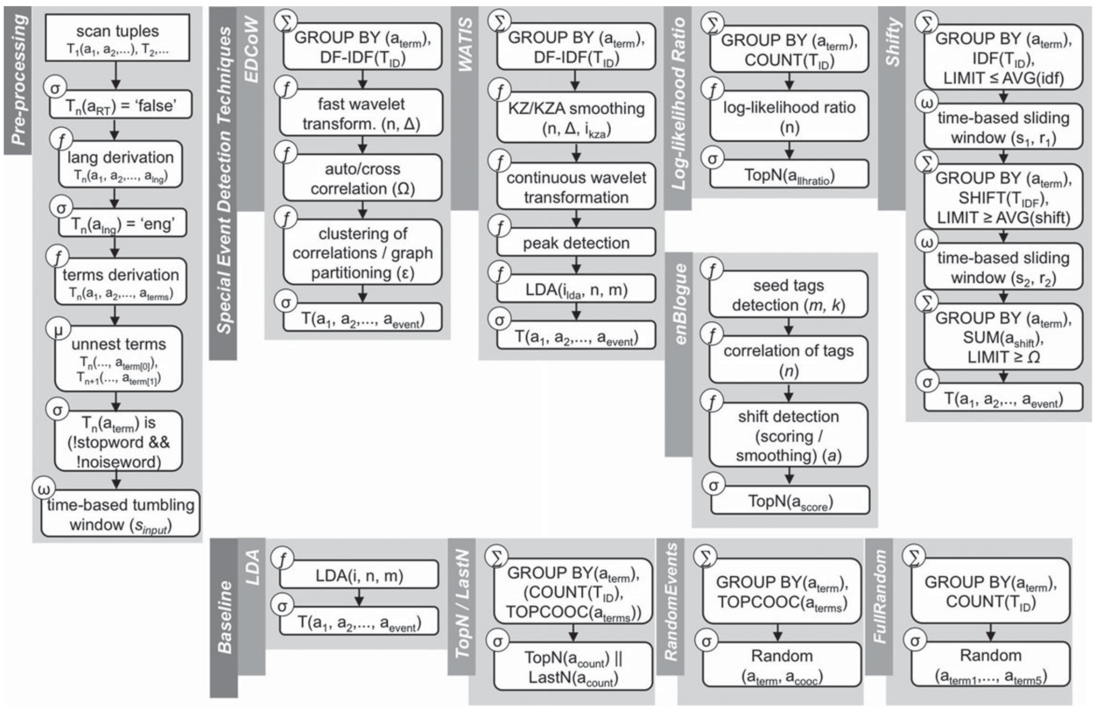
Precision Our precision measure is composed of two components. First, we query Google using the five event terms and a specific date range as search query input. Doing so, we are able to verify if the detected event has been described by an important article returned by Google for the corresponding time frame. As important articles, we define search results that are from one of the top 15 news websites such as CNN, CBSNews, USAToday, BBC and Reuters. For the second part of our precision measure, we query the archive of the New York Times[6] with the five event terms as well as the specific date range. Since the number of hits (h), which are in the range between 0 and 10 both for Google (h^{G}) or New York Times (h^{NYT}), is an indicator of how important a reported event is, we calculate the final precision score for all results (N) by weighting the single results as

$$\frac{1}{N}\sum_{i=0}^{N}\left(\frac{1}{2}h_i^{\mathrm{G}} + \frac{1}{2}h_i^{\mathrm{NYT}}\right).$$

Recall We propose to calculate recall by crawling the world news headlines on the Reuters website[7] for the days corresponding to the analysis. Each headline is represented as a list of terms T^{hl}. With this measure, we intend to reflect the percentage of detected events with respect to important news appearing on a real-world news archive. To weigh the single results, we check for each term in a news headline, which reported event, represented as a list of terms T^{e}, has the maximal similarity value (max _sim). Since we exclude matches on one term only, this similarity value can either be two,

[6]http://query.nytimes.com/search/sitesearch/ (15 April 2016).

[7]http://www.reuters.com/news/archive/worldNews?date 02112015/ (15 April 2016).



Pre-processing
scan tuples
T_1(a_1, a_2,...), T_2,...
T_n(a_RT) = 'false'
lang derivation
T_n(a_1, a_2,..., a_lng)
T_n(a_lng) = 'eng'
terms derivation
T_n(a_1, a_2,..., a_terms)
unnest terms
T_n(..., a_term[0]), T_n+1(..., a_term[1])
T_n(a_term) is (!stopword && !noiseword)
time-based tumbling window (s_input)
Special Event Detection Techniques
EDCoW
GROUP BY (a_term), DF-IDF(T_ID)
fast wavelet transform. (n, Δ)
auto/cross correlation (Ω)
clustering of correlations / graph partitioning (ε)
T(a_1, a_2,..., a_event)
WATIS
GROUP BY (a_term), DF-IDF(T_ID)
KZ/KZA smoothing (n, Δ, i_kza)
continuous wavelet transformation
peak detection
LDA(i_lda, n, m)
T(a_1, a_2,..., a_event)
Log-likelihood Ratio
GROUP BY (a_term), COUNT(T_ID)
log-likelihood ratio (n)
TopN(a_llhratio)
enBlogue
seed tags detection (m, k)
correlation of tags (n)
shift detection (scoring / smoothing) (a)
TopN(a_score)
Shifty
GROUP BY (a_term), IDF(T_ID), LIMIT ≤ AVG(idf)
time-based sliding window (s_1, r_1)
GROUP BY (a_term), SHIFT(T_IDF), LIMIT ≥ AVG(shift)
time-based sliding window (s_2, r_2)
GROUP BY (a_term), SUM(a_shift), LIMIT ≥ Ω
T(a_1, a_2,.., a_event)
Baseline
LDA
LDA(i, n, m)
T(a_1, a_2,..., a_event)
TopN / LastN
GROUP BY(a_term), (COUNT(T_ID), TOPCOOC(a_terms))
TopN(a_count) || LastN(a_count)
RandomEvents
GROUP BY(a_term), TOPCOOC(a_terms)
Random (a_term, a_cooc)
FullRandom
GROUP BY(a_term), COUNT(T_ID)
Random (a_term1,..., a_term5)

FIGURE 1. Niagarino query plans of the studied event detection techniques and baselines.

three, four or five terms. With this weighting, we calculate the final recall score for all headlines (N) as

$$\frac{1}{N}\sum_{i=0}^{N}\frac{1}{2}\max\_\mathrm{sim}(T_i^{\mathrm{hl}}, T^{\mathrm{e}}).$$

4. EVENT DETECTION APPROACHES

In order to realize streaming implementations of state-of-the-art event detection techniques for Twitter, we use Niagarino[8] [5], a data stream management system developed and maintained by our research group. The main purpose of Niagarino is to serve as an easy-to-use and extensible research platform for streaming applications such as the ones presented in the paper. Using its operator-based processing pipeline, our implementations are modular and can be easily configured. For example, we can configure the approaches to report the same number of events, which are represented as one main event term together with four associated event description terms. Using a common implementation, platform also has the advantage that run-time performance and memory usage results can be compared fairly.

For the evaluation of our measures, we take nine different approaches into account. Figure 1 shows the Niagarino-based implementations of these approaches as query plans. In Niagrarino, a query plan is represented as a directed acyclic graph $Q = (O, S)$, consisting of a set of query operators O and a set of streams S that connect these operators. Tuples enter the query plan at so-called source operators, which have no incoming streams. As tuples flow along streams through the query plan, they are processed by operators. Some of the operators that are currently supported by Niagarino are described in Table 2. Finally, query results are reported by sink operators, which have no outgoing streams. Additionally, the pre-processing pipeline, which is used by all approaches, is shown on the left. The pre-processing removes all non-English tweets and retweets. Then, it tokenizes and unnests the terms of the remaining tweets. It also discards terms that can be classified as stop-words or as noise (e.g. too short, invalid characters, etc.). Finally, a tumbling (non-overlapping) window of size s_{input} is continuously applied and its contents are forwarded to the subsequent operators.

At the bottom of Fig. 1, the query plans for LDA, TopN, LastN, RandomEvents (RE) and FullRandom (FR) are shown. Since these approaches are not specifically tailored to the task of event detection, we use them as baseline approaches in order to confirm that the proposed measures are discriminating.

LDA [51] uses the probabilities of terms in documents and groups those terms together that have the highest probability

[8]http://www.informatik.uni konstanz.de/grossniklaus/software/niagarino/ (15 April 2016).

TABLE 2. Niagarino query operators.

Symbol	Description
σ	*Selection*: used to select tuples from the stream
f	*User defined function*: applies a function to a one or more input tuples and emits one or more result tuples
μ	*Unnest*: splits a nested attribute value and emits a tuple for each nested value
ω	*Window*: segments the stream into time or tuple based tumbling (non overlapping) or sliding windows [59] of a pre defined size
Σ	*Aggregate*: applies a user defined or pre defined aggregation function, e.g. `SUM`, `AVG` and `IDF`, to a stream segment (window) and emits one result tuple

of belonging together. We realized LDA in Niagarino based on its user-defined function operator. Since LDA is normally used for topic modeling, we equate a topic to an event. The parameters that can be set for this approach include the number of topics, the number of terms per topic and the number of iterations of the probability modeling. As there are a lot of repeating terms in tweets and also per time window, we expect that this technique is not suitable for event detection and therefore classify it as a baseline method.

The other four-baseline techniques use a grouping operator followed by a selection operator. FR constructs 'events' by randomly selecting five terms from all distinct terms in a time window. RE selects the main event term in the same way as FR, but uses the four most co-occurring terms of the event term as the associated event description terms. Both of these approaches report N events per time window. The next two approaches, TopN and LastN, are based on the Inverse Document Frequency (IDF) [58] score of single terms among all distinct terms in the time window. While TopN selects the N most frequent terms, LastN selects the N terms with the lowest frequency. Both of them report the selected event terms together with the four most co-occurring terms. In addition to these baseline approaches, we implemented several techniques that have been proposed to detect events in Twitter data streams. We implemented all of these techniques to the best of our knowledge based on the information available in the original papers. The corresponding Niagarino query plans are shown at the top of Fig. 1.

The first technique, log-likelihood ratio (LLH), is a reimplementation of Weiler *et al.* [43], which is realized as a LLH user-defined function that is applied to the grouped set of terms of a time window. In contrast to the original technique that detected events for pre-defined geographical areas, we adjusted the approach to calculate the log-likelihood measure for the frequency of all distinct terms in the current time window against their frequency in the past time windows. Events are reported by selecting the top N terms with the highest LLH together with the corresponding top four most co-occurring terms. Since, these are the terms with the highest abnormal behavior in their current frequency with respect to their historical frequency, we define these terms to be events.

The second technique, Shifty, is a reimplementation of Weiler *et al.* [47]. In contrast to the original paper, which additionally considers bigrams, we now only use single terms in the analysis. The technique calculates a measure that is based on the shift of IDF values of single terms in pairs of successive sliding windows. First, the IDF value of each term in a single window is continuously computed and compared to the average IDF value of all terms within that window. Terms with an IDF value above the average are filtered out. The next step builds a window with size s_1 that slides with range r_1 in order to calculate the shift from one window to the next. In this step, the shift value is again checked against the average shift of all terms and only terms with a shift above the average are retained. In the last step, a new sliding window with size s_2 that slides with range r_2 is created. The total shift value is computed as the sum of all shift values of the sub-windows of this window. If this total shift value is greater than the pre-defined threshold Ω, the term is detected as event and reported together with its top four co-occurrence terms.

The third technique, WATIS, is an implementation of Cordeiro [29]. The algorithm partitions the stream into intervals of s seconds and builds Document Frequency-Inverse Document Frequency (DF-IDF) signals for each distinct term. Due to the noisy nature of the Twitter data stream, signals are then processed by applying an adaptive Kolmogorov-Zurbenko (KZA) [60] filter, a low-pass filter that smoothens the signal by calculating a moving average with i_{kza} iterations over N intervals. It then uses a continuous wavelet transformation to construct a time/frequency representation of the signal and two wavelet analyses, the tree map of the continuous wavelet extrema and the local maxima detection, to detect abrupt increases in the frequency of a term. In order to enrich events with more information, the above-mentioned LDA algorithm (with i_{LDA} iterations) is used to model one topic consisting of five terms After the LDA phase the event is reported.

The fourth technique, EDCoW, is an implementation of Weng and Lee [3]. The first step of the algorithm is to partition the stream into intervals of s seconds and to build DF-IDF signals for each distinct term in the interval. These signals are further analyzed using a discrete wavelet analysis that builds a second signal for the individual terms. Each data point of this second signal summarizes a sequence of values from the first signal with length Δ. The next step then filters out trivial terms by checking the corresponding signal autocorrelations against a threshold γ. The remaining terms are then clustered to form events with a modularity-based graph partitioning technique. Insignificant events are filtered out

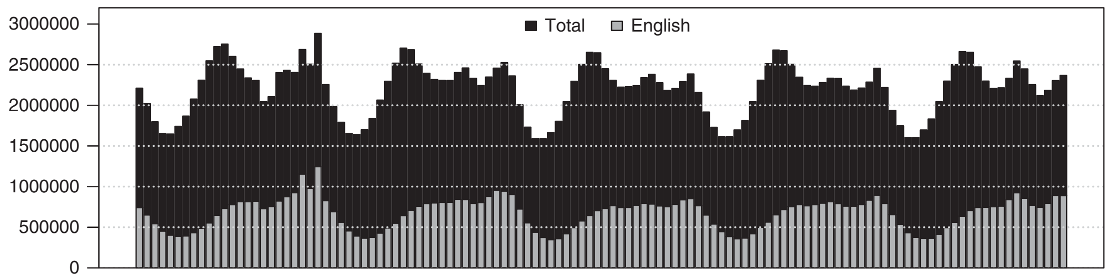


FIGURE 2. Average hourly total and English tweets for all three data sets.

using a threshold parameter ε. Since this approach detects events with a minimum of two terms, we introduced an additional enrichment step that adds the top co-occurring terms to obtain events with at least five terms. As the original paper fails to mention the type of wavelet that was used, we experimented with several types. The results reported in this paper are based on the *Discrete Meyer* wavelet.

Finally, the fifth technique, ENB, is an implementation of Alvanaki *et al.* [7]. This approach uses tumbling windows to compute statistics about tags and tag pairs.[9] An event consists of a pair of tags and at least one of the two tags needs to be a seed tag. These so-called seed tags are determined by calculating a popularity score. The tags with a popularity score within a pre-defined range of the top percentage terms (threshold k) are then chosen as seeds. Also a minimum of m tweets need to contain the tag. The correlation of two tags is calculated by a local and global impact factor, which is based on the corresponding sets of tweets that are currently present in the window. If two tags are strongly connected, they are considered to be related. A minimum of n correlations needs to exist. An event is considered as emergent if its behavior deviates from the expected. In order to detect this condition, the shifting behavior of correlations between terms is calculated by a scoring and smoothing function, which uses the fading parameter a to smooth out past values. Since ENB originally only considered pairs of tags as events, whereas we need a total of five tags for our evaluation, the three most co-occurring terms of both tags of the pair are added to complete the event. Also, the technique reports a pre-defined number of events, which are selected by ranking all events based on the calculated score of shift and reporting the top N events.

5. EVALUATION

In order to demonstrate that the measures proposed in this article are discriminating, we run experiments against three different real-world Twitter stream data sets (consisting of 5 days each) that we collected. The three data sets respectively contain the days of 1 6, 11 16 and 21 26 February 2015 (EST). By using the *Gardenhose* access level of the Twitter streaming API, we are able to obtain a randomly sampled 10% stream of all public tweets. Our collection contains an average of 2.2 million tweets per hour and almost 50 million tweets per day. We pre-filtered the data set for tweets with English language content by using a pre-existing Java library.[10] After this step, the data set contains an average of 660 000 tweets per hour and 16 million tweets per day. Figure 2 shows the distribution of the total and the English number of tweets per hour for each day as an average of all three data sets.

5.1. Experimental setup

The event detection techniques that we use for our evaluation have all been defined with slightly different use cases in mind. In order to fairly compare them, we defined a common task that all of the techniques can accomplish. As we are interested in (near) real-time event detection, we set the length of the time-window used for event reporting to 1 hour. This means that after each hour of processing the techniques need to report the results obtained so far. Note that within the time window of 1 hour no duplicate events are possible for any technique. As the number of events reported by the different techniques may vary significantly (depending on the parameter settings), we configured each technique to report about the same number of events per data set. In the simple case, where the number of detected events depends on a single parameter N, we set this parameter to 15 events per hour, which results in 1800 events per data set. In the more complex case, where the number of reported events depends on a number of parameters, we chose a setting that best approximates the number of 1800 events per data set. Note that in the former case the reported events are uniformly distributed over the data set, while in the latter case they are not. Some techniques report a few events with less than five terms,

[9]In their original paper, Alvanaki *et al.* [7] state that ENB uses sliding win dows. However, only the value for the size of the window is defined, while the value for the slide range is never given. Personal communication with one of the authors confirmed that indeed a tumbling (non overlapping) win dow is used.

[10]https://code.google.com/p/language detection/ (15 April 2016).

which are discarded. We compensated for this behavior by adjusting the parameters of such event detection techniques accordingly. Table 3 summarizes the parameter settings used. For all baseline techniques, we set the window size s to 1 hour. Note that all techniques report the first events after analyzing 1 hour of data. Since, for example, ENB needs to analyze a sequence of three windows before events can be reported, we set its window size s to 20 minutes. It is important to point out that these settings are purely required to obtain comparable output and might not correspond to the optimal settings for each technique. Also, it is unlikely that events are uniformly distributed over the hours of a day. Using these settings, we obtain an average of 1745 events for Shifty, 1520 for WATIS and 2020 for EDCoW per data set.

5.2. Results

In the following, we present the results of our evaluation. Note that we summarized the results of all three data sets as an average. We begin by discussing the results of the run-time performance measures, whereas the results of the task-based measures are discussed later in this section.

5.2.1. Run-time performance

5.2.1.1. Throughput. The throughput measure was evaluated on two different machine configurations. The first machine configuration (*M1*) consisted of Oracle Java 1.8.0 25 (64-bit) on server-grade hardware with 2 Intel Xeon E5345s processors at 2.33 GHz with 4 cores each and 24 GB of main memory. The second machine configuration (*M2*) consisted of Oracle Java 1.8.0 40 (64-bit) on server-grade hardware with 1 Intel Xeon E5 processor at 3.5 GHz with 6 cores and 64 GB of main memory. Regardless of the available physical memory, the *Xmx* flag of the Java Virtual Machine (JVM) was used in both configurations to limit the maximum memory to 24 GB. Note that ENB was only evaluated on *M2* as machine *M1* was not available anymore at the time of this evaluation. We note that the run-time performances of all other techniques scale by a similar factor of ∼2.5 to 2.9× when running them on machine *M2* instead of *M1*. Therefore, we can reason that the difference in the throughput between the techniques is always similar regardless of which hardware configuration is used.

TABLE 3. Parameters for Shifty, WATIS, EDCoW and ENB.

Technique	Parameters
Shifty	$s = 1$ min, $s_1 = 2$ min, $r_1 = 1$ min, $s_2 = 4$ min, $r_2 = 1$ min, $\Omega = 30$
WATIS	$s = 85$ sec, $N = 5$ intervals, $i_{\mathrm{kza}} = 5$, $i_{\mathrm{lda}} = 500$
EDCoW	$s = 10$ sec, $N = 32$ intervals, $\gamma = 1$, $\epsilon = 0.2$
ENB	$s = 20$ min, $m = 50$, $k = 20$, $n = 50$, $a = 0.8$, $N = 15$

Figure 3 shows the results of the throughput measure for all techniques, measured in terms of the average throughput (tweets per second), for all three data sets and both machine configurations *M1* and *M2*. Since the results obtained using configuration *M1* are already discussed in the original paper, we focus on the results by using the setting *M2*. The baseline techniques (with the exception of LDA) as well as the LLH and ENB techniques achieve the highest throughput with ∼35 000 tweets per second. ENB even achieves a slightly higher throughput, which can be explained by its use of minimum filters (thresholds m and n), which drastically reduce the amount of tag pairs that need to be considered for further analysis. The rate of our Shifty technique is lower at around 8000 tweets per second. However, it should be noted that Shifty is the only online technique that processes the input incrementally. Therefore, Shifty's performance does not depend on changes to the reporting schedule that we used (after each hour), which will affect the throughput of all other approaches. In contrast to WATIS, EDCoW scores very well. Since WATIS uses LDA at the end of processing to create the final events, this result is not surprising. As we see, applying LDA with 500 iterations is the slowest approach with around 4850 tweets per second. If we take into account the 50 million tweets per day (∼580 per second) of the 10% stream, we can observe that all techniques could process this stream in (near) real-time and are therefore feasible. However, if these techniques were applied to the full 100% stream (∼5800 tweets per second), LDA would not be feasible. Based on these observations, we conclude that our measure for throughput is discriminating and can be used to judge the feasibility of approaches.

5.2.1.2. Memory usage. In addition to execution time, the amount of memory required to process the stream is another important characteristic of a technique. In order to compare memory usage fairly, we measured the memory consumption at a fixed number of measurement points. We first recorded the total execution time needed by a technique to process all three data sets. Based on this total execution time, we then derived the

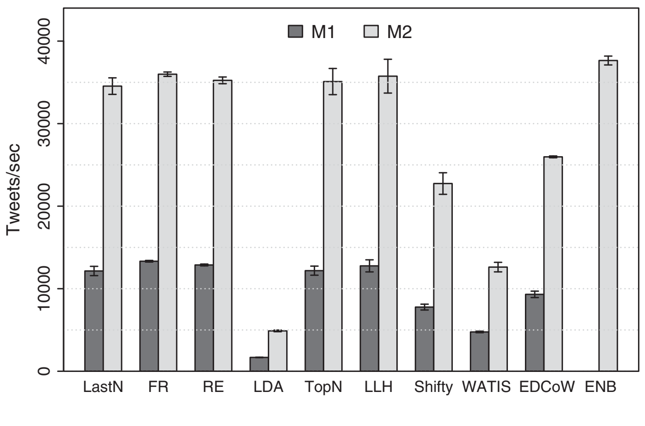


FIGURE 3. Run time performance.

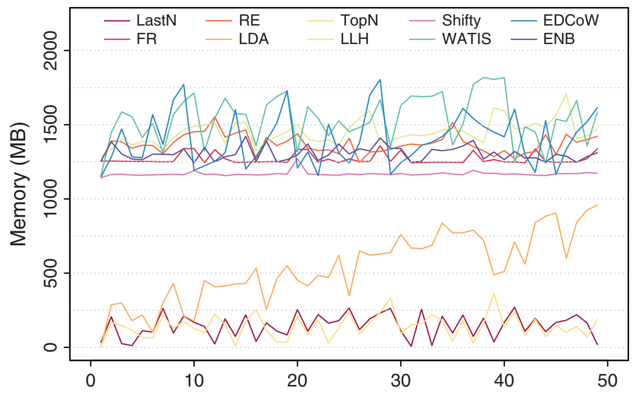


FIGURE 4. Memory usage.

intervals at which memory usage has to be measured in order to obtain a total of 50 measurements per execution. Before each measurement was taken, the garbage collector of the JVM was invoked to ensure better reproducibility of the results. All measurements reported in this article were recorded using the *M2* setting. Figure 4 plots the amount of used memory in megabytes at each of the 50 measurement points. As expected, the TopN and LastN techniques require the least memory. The memory usage of LDA is similarly low at the beginning of the computation, but steadily increases over time. Since our implementation relies on a third-party library to perform LDA, we are not able to explain this increasing memory usage. Both the FR and the RE technique use an almost constant amount of memory. Note that the high amount of consumed memory is due to the use of a non-optimized implementation. Since we mainly created these techniques to evaluate our relevance measures, these results are not unexpected at this point.

LLH, EDCoW and WATIS require substantially more memory with a usage that fluctuates between 1.2 and 1.6 GB. In contrast, ENB and Shifty require almost constant memory of ~1.3 and 1.15 GB, respectively. Again, this result is not unexpected as both of these techniques use windows that are smaller than 1 hour. Therefore, they are able to purge memory more often than the other techniques that we studied. Recall that ENB uses the parameters k, m and n to restrict the number of tags and tag sets in a window and thereby limits its memory consumption. The fact that Shifty's memory requirements are even lower than the ones of ENB is explained by the fact that its windows are shorter.

5.2.2. *Task-based performance*

In contrast to run-time performance, the remaining three measures assess the task-based performance, i.e. the quality of an event detection technique. To further evaluate our measures, we also include the results of applying them to the so-called TT of Twitter in the following discussion. We collected the top 15 trending topics and enriched them by querying the Twitter API for the most current tweets belonging to each topic. The enrichment process also tokenizes and cleans the obtained tweets, and summarizes the five most co-occurring terms to a final event. Hereby, we also get 1800 events per data set.

5.2.2.1. Duplicate event detection rate. We begin by presenting the results obtained from our duplicate event detection rate (DEDR) measure. For each technique, we calculate the percentage of events, which are classified as duplicates. As this classification is configurable, we present results obtained by requiring that one, two, three, four or all five event terms need to be equal (DEDR1, …, DEDR5). Figure 5 plots the average results of the DEDR for all data sets. We can observe that all techniques report a very high number of duplicates for DEDR1. Since the terms of FR and RE are randomly chosen, they generally report a lower number of duplicates. From the event detection techniques, the results for Shifty, WATIS and EDCoW closely resemble the results of applying our DEDR measure to *TT*, whereas all other approaches have significantly different profiles. ENB is the only non-baseline technique that differs from this profile. However, this can be explained by the fact that ENB also focuses on detecting topics, which remain in the news over time and uses pairs of tags as event terms. For example, the filters for the minimum number of tweets and correlations lead to more duplicates in the result set. However, we can conclude that, with the exception of ENB, the profile of duplicate detection is similar for all event detection techniques and differentiates them from the baselines.

5.2.2.2. Precsion, recall and F$_1$ *score.* For the evaluation of our precision and recall measures, we only use events that were not filtered out by DEDR3, i.e. all events with three or more common terms are removed from the result set and only the remaining non-duplicate events are further analyzed. Note that this results in an implicit inclusion of the DEDR measure in our precision and recall measures. Since we added ENB to our collection of studied event detection techniques for this article, we reran all precision tests[11] that we originally conducted for the conference version. Based on the comparison of the new results with the ones of the previous study, we can observe that they are almost the same. The highest deviation was a delta of 0.03 in the precision score of WATIS and TT, which does not affect the overall ranking of the approaches. Therefore, we argue that the additional results measured for ENB can be compared to the previous evaluation results of the other techniques. Figure 6 shows the average precision, recall and F_1 score over all three data sets for all techniques. Based on these measures, we observe that all of the dedicated

[11]Since we crawled and archived the headlines of the Reuters website for the corresponding time periods as a basis for our recall measure, it is not necessary to repeat these tests.

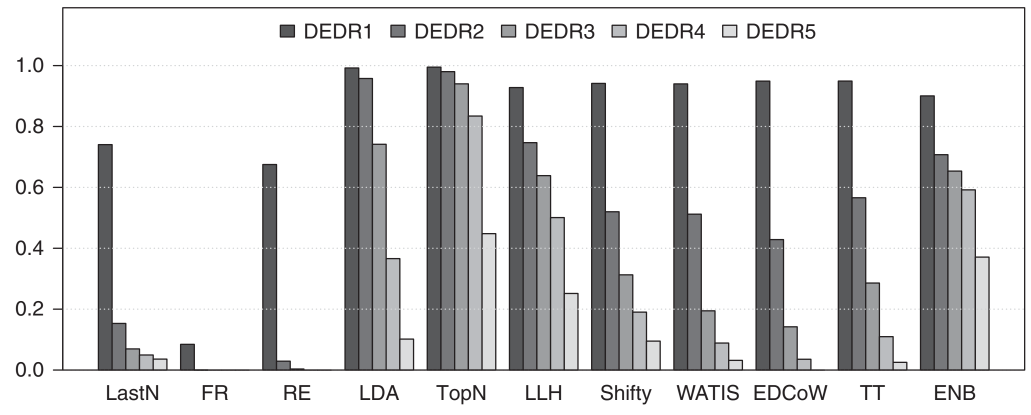


FIGURE 5. Average DEDR.

event detection techniques clearly outperform the baseline approaches. This finding confirms the validity of the precision and recall measure proposed in this article. We conclude our evaluation by discussing the results shown in Fig. 6 in more detail. We note that the scores are generally very low. However, since we are only interested in relative comparisons, this is not a problem.

Among the baseline approaches, both LDA and RE score competitive to dedicated event detection techniques with respect to specific measures. The precision of LDA is higher than the one of LLH and Shifty, RE scores well in terms of recall. In both cases, this result can be explained with the way these approaches work. Also, it demonstrates the importance of studying both precision and recall, which we support with our F_1 score. The best approaches according to our measures are the advanced techniques WATIS, EDCoW and ENB, which are also among the most cited event detection techniques. Since EDCoW produces the most events of all techniques, its parameters could also be adjusted to increase its precision score. Also, the basic enrichment process that we implemented for EDCoW could be improved. For example, WATIS uses LDA for the same purpose and scores very well in terms of recall. Our own techniques, LLH and Shifty, do not perform as well as the two advanced techniques. However, we note that Shifty is the only online event reporting technique and therefore only uses very short-time intervals (of 4 minutes in this case) instead of a full hour to classify terms as events. Additionally, we do not use bigrams in this article as opposed to the original Shifty algorithm. LLH was originally designed to use both the spatial and the time dimension to detect unusual rates of terms in pre-defined geographical areas over time. In this article, we only use the time dimension, which has weakened the performance of the approach. Finally, our measures assign high precision and recall scores to the Twitter TT, which further confirms their practical relevance. However, in contrast to the results presented in this paper, TT are based on the full 100% stream of Twitter.

6. DISCUSSION AND LIMITATIONS

Having presented the results of our evaluation in the previous section, we critically discuss the scope of our approach and some of its limitations in this section.

6.1. Evaluation parameters

In our evaluation, we have pre-defined several parameters, e.g. the number of events, the number of terms an event consists of and the size of the time windows. The settings of these parameters were experimentally determined by running a series of extended pre-tests. For example, we also evaluated both the precision and the recall measure using an event format that only included three terms. During this pre-test, we observed that there is a high probability that the three terms are very similar, which can lead to a large number of false positives. Due to this observation and due to the fact that existing techniques, e.g. Cordeiro [29], use five terms, we chose to use this event format.

Another challenge is to configure the number of events that are reported per time window. As the results of most techniques depend on a number of parameters, determining a set of settings that yields consistent and comparable results is non-trivial and very time-consuming. For our experimental setup, the common denominator of 1800 events per data set, i.e. ~15 events per hours, was empirically determined by iteratively adjusting the parameters of all techniques over several rounds of testing. One of the most important parameter that needs to be adjusted in accordance with this parameter setting is the window size. The window sizes used in the evaluations described in the original papers vary widely: ENB [7] reports ~1 or 2 hours, EDCoW [3] reports ~1 month, WATIS [29] reports about a week. Since all of these techniques are motivated by the promise of detecting events in (near) real-time, we started experimenting with very small windows and enlarged them gradually. By doing so, we empirically

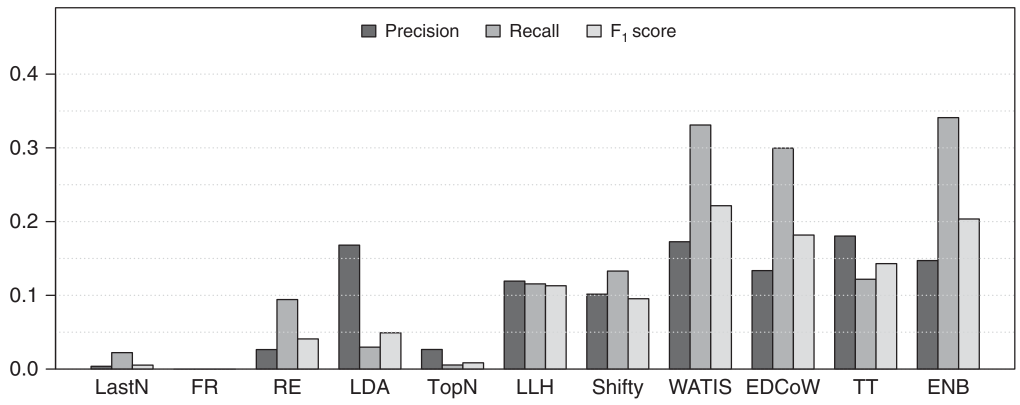


FIGURE 6. Average precision, recall and F_1 score of all techniques.

identified 1-hour windows as a workable tradeoff for all studied techniques.

6.2. Selection of the ground truth

Selecting an appropriate ground truth is another difficult problem. Since most existing works on event detection techniques for social media data stream characterize events as a newsworthy incident, we have chosen to use well-known news archives and websites as a ground truth for precision and recall. For precision, we calculate a weighted value for each detected event using a Google and a New York Times search query. Of course, this approach introduces bias towards a specific type of events. In order to evaluate techniques that detect events of a different nature, the services that serve as the ground truth would need to be replaced by an archive that better matches the event type. For example, the collection of events gathered by the GDELT Project[12] could be a possible option in a case, where events that are otherwise noteworthy are to be detected. In terms of recall, we crawled the Reuters news archive, but in previous work [5] we have shown that the Bloomberg news archive can also be used as an appropriate ground truth.

6.3. Stability of the measures

Our goal is to provide measures that are stable w.r.t. the ranking of the different techniques, while we tolerate fluctuations in their absolute scores over time. Since our precision measure relies on external services that cannot be crawled and archived, it is impossible to formally guarantee this property. However, in order to empirically study the stability of our precision measure, we reran all measurements and compared them to the results obtained over 1 year ago for the conference version of this article. Based on this comparison, we could conclude that the results obtained from Google and the New York Times search have changed very little. Furthermore, these small changes do not affect the ranking obtained from our measures. Even though this result is encouraging, it is only a current observation and does not enable predictions on how the indices of the search engines used will evolve in the future. In contrast to our precision measure, we can crawl the ground truth for our recall measure and, therefore, this measure is intrinsically stable.

[12]http://gdeltproject.org (15 April 2016).

6.4. Rate limits

One major drawback of our precision measure is the request limitation that are typically imposed by the external sources that we use to compute a score. For example, Google blocks access for an undisclosed amount of time if more than ~150 requests are issued at a time from a single IP address. Similarly, querying the archive of the New York Times is limited to 10 requests per second and no more than 10 000 requests per day from one IP address. While these limits put a clear restriction on the scalability of our precision measure, the permitted number of requests is large enough to evaluate the results of any of the event detection techniques as configured in this work in real-time.

7. CONCLUSIONS AND FUTURE WORK

In this article, we have addressed the lack of quantitative and comparative evaluation of event detection techniques by proposing a number of measures, both for run-time and task-based performance. In contrast to previous evaluation methods, all our measures can be automatically applied to evaluate large result sets without the requirement of an existing gold standard. In order to demonstrate the validity of our proposed measures, we have studied them based on several baseline approaches and state-of-the-art event detection techniques. We have shown that our measures are able to discriminate between different techniques and support relative comparisons.

As immediate future work, we plan to take advantage of our platform-based approach to extend our evaluations and study further techniques. By reviewing the survey of related work, we found several candidates for this venture. On the one hand, techniques such as TM [19] and Twevent [31] are interesting because the techniques that they use are closely related to our own techniques. On the other hand, also clustering and hashing techniques such as ET [40] or the work of Petrović *et al*. [20] would be interesting to compare. Since the source code of most of these works is not provided by their authors, it is a challenging task to correctly implement these techniques. Notable exceptions to this lack of reproducibility are SocialSensor [37] and MABED [45], which are both freely available as source code.

At the same time, the currently implemented techniques should be improved to process data continuously. Furthermore, the influence of the pre-processing on run-time and task-based performance could be studied. In our platform-based approach, we can easily remove existing operators (e.g. retweet filtering) and replace them with new operators (e.g. part-of-speech tagging or named-entity recognition). Finally, a deeper evaluation of how the different parameters of a technique influence the trade-off between run-time and task-based performance could give rise to adaptive event detection techniques. Also it would be interesting to include a crowd-based measure to evaluate how humans would rate the results of the different techniques in terms of precision and contrast to the automatic measures.

ACKNOWLEDGEMENTS

We would also like to thank our students Christina Papavasileiou, Harry Schilling and Wai-Lok Cheung for their contributions to the implementations of the WATIS, EDCoW and ENB event detection techniques.

FUNDING

The Deutsche Forschungsgemeinschaft (DFG), Grant No. GR 4497/4: 'Adaptive and Scalable Event Detection Techniques for Twitter Data Streams.'

REFERENCES

[1] Sakaki, T., Okazaki, M. and Matsuo, Y. (2010) Earthquake shakes Twitter users: real time event detection by social sensors. In *Proc. Intl. Conf. World Wide Web (WWW)*, pp. 851 860.

[2] Culotta, A. (2010) Towards detecting influenza epidemics by analyzing Twitter messages. In *Proc. Workshop on Social Media Analytics (SOMA)*, pp. 115 122.

[3] Weng, J. and Lee, B. S. (2011) Event detection in Twitter. In: *Proc. Intl. Conf. Weblogs and Social Media (ICWSM)*, pp. 401 408.

[4] Allan, J. (2002) *Topic Detection and Tracking: Event Based Information Organization*, Kluwer Academic Publishers.

[5] Weiler, A., Grossniklaus, M. and Scholl, M.H. (2015) Run time and task based performance of event detection techniques for twitter. In *Proc. Intl. Conf. Advanced Information Systems Engineering (CAiSE)*, pp. 35 49.

[6] Weiler, A., Grossniklaus, M. and Scholl, M.H. (2015) Evaluation measures for event detection techniques on Twitter data streams. In *Proc. British Intl. Conf. Databases (BICOD)*, pp. 108 119.

[7] Alvanaki, F., Michel, S., Ramamritham, K. and Weikum, G. (2012) See what's enBlogue: real time emergent topic identifi cation in social media. In *Proc. Intl. Conf. Extending Database Technology (EDBT)*, pp. 336 347.

[8] Nurwidyantoro, A. and Winarko, E. (2013) Event detection in social media: a survey. In *Proc. Intl. Conf. ICT for Smart Society (ICISS)*, pp. 1 5.

[9] Madani, A., Boussaid, O. and Zegour, D. E. (2014) What's happening: a survey of tweets event detection. *Proc. Intl. Conf. Communications, Computation, Networks and Technologies (INNOV)*, pp. 16 22.

[10] Bontcheva, K. and Rout, D. (2014) Making sense of social media streams through semantics: a survey. *Semantic Web*, **5**, 373 403.

[11] Farzindar, A. and Khreich, W. (2015) A survey of techniques for event detection in Twitter. *Comput. Intell.*, **31**, 132 164.

[12] Papadopoulos, S., Corney, D. and Aiello, L.M. (2014) SNOW 2014 data challenge: assessing the performance of news topic detection methods in social media. In *Proc. Workshop on Social News on the Web (SNOW) in conjunction with Intl. Conf. Companion on World Wide Web (WWW)*, pp. 1 8.

[13] Ritterman, J., Osborne, M. and Klein, E. (2009) Using pre diction markets and twitter to predict a swine flu pandemic. In *Proc. Intl. Workshop on Mining Social Media (MSM)*, pp. 1 7.

[14] Sankaranarayanan, J., Samet, H., Teitler, B.E., Lieberman, M. D. and Sperling, J. (2009) TwitterStand: news in Tweets. In *Proc. Intl. Conf. Advances in Geographic Information Systems (SIGSPATIAL)*, pp. 42 51.

[15] Schühmacher, J. and Koster, C. (2009) Signalling events in text streams. User Centric Media*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, **40**, 335 339.

[16] Benhardus, J. (2010) Streaming trend detection in Twitter. *National Science Foundation REU for Artificial Intelligence, Natural Language Processing and Information Retrieval, University of Colorado, Final report*, pp. 1 7.

[17] Cataldi, M., Caro, L.D. and Schifanella, C. (2010) Emerging topic detection on Twitter based on temporal and social terms evaluation. In *Proc. Workshop on Multimedia Data Mining*

(MDMKDD) in conjunction with Intl. Conf. on Knowledge Discovery and Data Mining (SIGKDD), pp. 4:1 4:10.

[18] Lee, R. and Sumiya, K. (2010) Measuring geographical regu larities of crowd behaviors for Twitter based geo social event detection. In *Proc. Workshop on Location Based Social Networks (LBSN) in conjunction with Intl. Conf. on Advances in Geographic Information Systems (SIGSPATIAL)*, pp. 1 10.

[19] Mathioudakis, M. and Koudas, N. (2010) TwitterMonitor: trend detection over the Twitter Stream. In *Proc. Intl. Conf. Management of Data (SIGMOD)*, pp. 1155 1158.

[20] Petrović, S., Osborne, M. and Lavrenko, V. (2010) Streaming first story detection with application to Twitter. In *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics (HLT)*, pp. 181 189.

[21] Achrekar, H., Gandhe, A., Lazarus, R., Yu, S. H. and Liu, B. (2011) Predicting flu trends using Twitter data. In *Proc. IEEE Conf. Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 702 707.

[22] Becker, H., Naaman, M. and Gravano, L. (2011) Beyond trend ing topics: real world event identification on Twitter. In *Proc. Intl. Conf. Weblogs and Social Media (ICWSM)*, pp. 438 441.

[23] Lee, C. H., Yang, H. C., Chien, T. F. and Wen, W. S. (2011) A novel approach for event detection by mining spatio temporal information on microblogs. In *Proc. Intl. Conf. Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 254 259.

[24] Marcus, A., Bernstein, M.S., Badar, O., Karger, D.R., Madden, S. and Miller, R.C. (2011) TwitInfo: aggregating and visualiz ing microblogs for event exploration. In *Proc. Intl. Conf. Human Factors in Computing Systems (SIGCHI)*, pp. 227 236.

[25] Popescu, A. M., Pennacchiotti, M. and Paranjpe, D.A. (2011) Extracting events and event descriptions from Twitter. In *Proc. Intl. Conf. Companion on World Wide Web (WWW)*, pp. 105 106.

[26] Abel, F., Hauff, C., Houben, G., Stronkman, R. and Tao, K. (2012) Twitcident: fighting fire with information from social web streams. In *Proc. Intl. Conf. Companion on World Wide Web (WWW)*, pp. 305 308.

[27] Adam, N., Eledath, J., Mehrotra, S. and Venkatasubramanian, N. (2012) Social Media Alert and Response to Threats to Citizens (SMART C). In *Proc. Intl. Conf. Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pp. 181 189.

[28] Aggarwal, C.C. and Subbian, K. (2012) Event detection in social streams. In *Proc. SIAM Intl. Conf. Data Mining (SDM)*, pp. 624 635.

[29] Cordeiro, M. (2012) Twitter event detection: combining wave let analysis and topic inference summarization. In *Proc. Doctoral Symposium on Informatics Engineering (DSIE)*.

[30] Ishikawa, S., Arakawa, Y., Tagashira, S. and Fukuda, A. (2012) Hot topic detection in local areas using Twitter and Wikipedia. In *Proc. Workshop on Complex Sciences in the Engineering of Computing Systems (CSECS) in conjunction with Intl. Conf. on Architecture of Computing Systems (ARCS)*, pp. 1 5.

[31] Li, C., Sun, A. and Datta, A. (2012) Twevent: segment based event detection from Tweets. *Proc. Intl. Conf. Information and Knowledge Management (CIKM)*, pp. 155 164.

[32] Li, R., Lei, K.H., Khadiwala, R. and Chang, K.C. C. (2012) TEDAS: a Twitter based event detection and analysis system. *Proc. Intl. Conf. Data Engineering (ICDE)*, pp. 1273 1276.

[33] Nishida, K., Hoshide, T. and Fujimura, K. (2012) Improving Tweet stream classification by detecting changes in word prob ability. In *Proc. Intl. Conf. on Research and Development in Information Retrieval (SIGIR)*, pp. 971 980.

[34] Osborne, M., Petrović, S., McCreadie, R., Macdonald, C. and Ounis, I. (2012) Bieber no more: first story detection using Twitter and Wikipedia. In *Proc. Workshop on Time aware Information Access (TAIA) in conjunction with Intl. Conf. on Research and Development in Information Retrieval (SIGIR)*.

[35] Ritter, A., Mausam, Etzioni, O. and Clark, S. (2012) Open domain event extraction from Twitter. *Proc. Intl. Conf. Knowledge Discovery and Data Mining (SIGKDD)*, pp. 1104 1112.

[36] Terpstra, T., de Vries, A., Stronkman, R. and Paradies, G. (2012) Towards a realtime Twitter analysis during crises for operational crisis management. In *Proc. Intl. Conf. Information Systems for Crisis Response and Management (ISCRAM)*, pp. 1 9.

[37] Aiello, L.M., Petkos, G., Martin, C., Corney, D., Papadopoulos, S., Skraba, R., Göker, A. and Kompatsiaris, I. (2013) Sensing trending topics in Twitter. *IEEE Trans. Multimedia*, **15**, 1268 1282.

[38] Bahir, E. and Peled, A. (2013) Identifying and tracking major events using geo social networks. *Soc. Sci. Comput. Rev.*, **31**, 458 470.

[39] Martin, C., Corney, D., Göker, A. and Macfarlane, A. (2013) Mining newsworthy topics from social media. In *Proc. BCS SGAI Workshop on Social Media Analysis in conjunction with Intl. Conf. of the British Computer Society's Specialist Group on Artificial Intelligence (SGAI)*, pp. 35 46.

[40] Parikh, R. and Karlapalem, K. (2013) ET: events from Tweets. In *Proc. Intl. Conf. Companion on World Wide Web (WWW)*, pp. 613 620.

[41] Walther, M. and Kaisser, M. (2013) Geo spatial event detection in the Twitter Stream. In Serdyukov, P., Braslavski, P., Kuznetsov, S. O., Kamps, J., Rger, S., Agichtein, E., Segalovich, I. and Yilmaz, E. Advances in Information Retrieval, *Lecture Notes in Computer Science*, **Vol. 7814**, 356 367. Springer Verlag.

[42] Abdelhaq, H., Sengstock, C. and Gertz, M. (2013) EvenTweet: online localized event detection from Twitter. *Proc. VLDB Endow.*, **6**, 1326 1329.

[43] Weiler, A., Scholl, M.H., Wanner, F. and Rohrdantz, C. (2013) Event identification for local areas using social media stream ing data. In *Proc. SIGMOD Workshop on Databases and Social Networks (DBSocial)*, pp. 1 6.

[44] Corney, D., Martin, C. and Göker, A. (2014) Spot the ball: detecting sports events on Twitter. In de Rijke, M., Kenter, T., de Vries, A. P., Zhai, C., de Jong, F., Radinsky, K. and Hofmann, K. Advances in Information Retrieval, *Lecture Notes in Computer Science*, **Vol. 8416**, pp. 449 454. Springer International Publishing.

[45] Guille, A. and Favre, C. (2014) Mention anomaly based event detection and tracking in Twitter. In *Proc. Intl. Conf. Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 375 382.

[46] Ifrim, G., Shi, B. and Brigadir, I. (2014) Event detection in Twitter using aggressive filtering and hierarchical Tweet clus tering. In *Proc. Workshop on Social News on the Web (SNOW) in conjunction with Intl. Conf. Companion on World Wide Web (WWW)*, pp. 33 40.

[47] Weiler, A., Grossniklaus, M. and Scholl, M.H. (2014) Event identification and tracking in social media streaming data. In *Proc. EDBT Workshop on Multimodal Social Data Management (MSDM)*, pp. 282 287.

[48] Zhou, X. and Chen, L. (2014) Event detection over Twitter Social Media Streams. *VLDB J.*, **23**, 381 400.

[49] Meladianos, P., Nikolentzos, G., Rousseau, F., Stavrakas, Y. and Vazirgiannis, M. (2015) Degeneracy based real time sub event detection in Twitter Stream. In *Proc. Intl. Conf. Weblogs and Social Media (ICWSM)*, pp. 248 257.

[50] Thapen, N. A., Simmie, D. S. and Hankin, C. (2015) The early bird catches the term: combining Twitter and News Data for event detection and situational awareness. *CoRR*, **abs/ 1504.02335**.

[51] Blei, D. M., Ng, A. Y. and Jordan, M. I. (2003) Latent Dirichlet allocation. *J. Mach. Learn. Res.*, **3**, 993 1022.

[52] Hoffman, M. D., Blei, D. M., and Bach, F. (2010) Online learning for latent Dirichlet allocation. In *Proc. of Annual Conf. Neural Information Processing Systems (NIPS).*

[53] McCreadie, R., Soboroff, I., Lin, J., Macdonald, C., Ounis, I. and McCullough, D. (2012) On building a reusable Twitter Corpus. In *Proc. Intl. Conf. Research and Development in Information Retrieval (SIGIR)*, pp. 1113 1114.

[54] Petrović, S., Osborne, M. and Lavrenko, V. (2012) Using para phrases for improving first story detection in news and Twitter. In *Proc. Conf. North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pp. 338 346.

[55] McMinn, A. J., Moshfeghi, Y. and Jose, J. M. (2013) Building a large scale corpus for evaluating event detection on Twitter. In *Proc. Intl. Conf. Information and Knowledge Management (CIKM)*, pp. 409 418.

[56] Cilibrasi, R.L. and Vitanyi, P.M.B. (2007) The Google similar ity distance. *IEEE Trans. Knowl. Data Eng*, **19**, 370 383.

[57] Wu, L., Hua, X. S., Yu, N., Ma, W. Y. and Li, S. (2008) Flickr distance. In *Proc. ACM Intl. Conf. Multimedia (MM)*, New York, NY, USA, pp. 31 40. ACM.

[58] Spärck Jones, K. (1988) *A Statistical Interpretation of Term Specificity and its Application in Retrieval*, Taylor Graham Publishing.

[59] Li, J., Maier, D., Tufte, K., Papadimos, V. and Tucker, P. A. (2005) No pane, no gain: efficient evaluation of sliding window aggregates over data streams. *SIGMOD Rec.*, **34**, 39 44.

[60] Yang, W. and Zurbenko, I. G. (2010) Kolmogorov Zurbenko filters. *Wiley Interdiscip. Rev. Computat. Stat.*, **2**, 340 351.