Towards Reproducible Research of Event Detection Techniques for Twitter

Andreas Weiler
Institute of Applied Information Technology
Zurich University of Applied Sciences
Winterthur, Switzerland
andreas.weiler@zhaw.ch

Harry Schilling, Lukas Kircher, Michael Grossniklaus
Department of Computer and Information Science
University of Konstanz
Konstanz, Germany
firstname.lastname@uni-konstanz.de

***Abstract*—A major challenge in many research areas is reproducibility of implementations, experiments, or evaluations. New data sources and research directions complicate the reproducibility even more. For example, Twitter continues to gain popularity as a source of up-to-date news and information. As a result, numerous event detection techniques have been proposed to cope with the steadily increasing rate and volume of social media data streams. Although some of these works provide their implementation or conduct an evaluation of the proposed technique, it is almost impossible to reproduce their experiments. The main drawback is that Twitter prohibits the release of crawled datasets that are used by researchers in their experiments. In this work, we present a survey of the vast landscape of implementations, experiments, and evaluations presented by the different research works. Furthermore, we propose a reproducibility toolkit including Twistor (*Twi*tter *S*tream Simula*tor*), which can be used to simulate an artificial Twitter data stream (including events) as input for the experiments or evaluations of event detection techniques. We further present the experimental application of the reproducibility toolkit to state-of-the-art event detection techniques.**

***Index Terms*—reproducibility, evaluation, event detection, twitter stream processing**

I. Introduction

Reproducibility of results is gaining importance in all research areas. For example, the SIGMOD[1] conference honors publications that fulfill replication criteria with the "ACM Results Replicated" label and the most-reproducible paper award. Furthermore, the data, scripts, and code of the paper can be hosted on the ACM servers to get the "ACM Artifacts Available" label. This movement demonstrates the importance of availability, replicability, and reproducibility of research results. The fast and easy access to Twitter data streams and the possibility to analyze these streams in real-time have fostered many research efforts specialized on social media data streams. In this area, the task of event detection is one of the major topics which led to countless approaches in this direction. In general, all event detection approaches have in common that they attempt to detect patterns that differ from the normal behavior of the data stream. However, almost all of them also have in common that very little attention is being paid to reproducibility. Cheung [7] extracts a number of characteristics from 50 Twitter event detection publications in order to estimate a reproducibility score. While the presented experiment shows no significant correlation between estimated reproducibility scores and actual reproduction efforts, it still reveals that almost none of the reviewed works put effort into reproducibility. For example, the source code of most works is not provided by their authors and, therefore, it is a challenging task to correctly implement these techniques. Notable exceptions to this poor reproducibility are *SocialSensor* [2] and *MABED* [11], which are both freely available as source code. However, even if the source code is available, the diversity of implementations makes it very difficult to recreate the original environment and to exactly reproduce the research results obtained in the original work. In this context, Weiler *et al.* [32] show that minor modifications in the different phases or parameters of event detection techniques can strongly impact the stability of their results. Another challenge is that experiments and evaluations are done in very different ways. For example, the used datasets are very diverse with regard to type, size, time frame, granularity, and data source. In this work, we, therefore, propose a solution to share and distribute a common data stream and ground truth for evaluation purposes. To address this problem, we introduce Twistor (*Twi*tter *S*tream Simula*tor*), a simulator for the Twitter data stream based on a statistical analysis of historical Twitter data. It is possible to simulate the default background noise and to inject predefined events, which are also defined based on a statistical analysis of historical events that appeared in the original Twitter data stream. It is also possible to scale the simulated data stream to different resolutions and therefore simulate the hundred percent public Twitter stream (*Firehose*), which is otherwise extremely costly to obtain. Twistor is publicly shared and all researchers in the area of event detection for Twitter data streams can apply their work to a common data stream and ground truth without painstakingly collecting any data. The two main contributions of this paper are as follows. First, we highlight the main issues of reproducibility in existing research works on event detection techniques for Twitter (*cf.* Section II). We also give an overview of existing research on creating or simulating Twitter corpora. Second, we present our proposed reproducibility toolkit including Twistor, which can be used to simulate an artificial Twitter stream (including events) as input for the evaluation of event detection techniques (*cf.* Sec-

[1] http://db-reproducibility.seas.harvard.edu/ (Feb 10, 2019)

tion III). In addition, we present experiments to automatically evaluate state-of-the art event detection techniques applied to the Twistor data stream. Finally, we draw conclusions about the presented proposal and indicate aspects for future work.

II. Issues of Reproducibilty

Over the last years, Twitter gained significant importance for researchers, especially in the context of event detection techniques applied to social media data streams. The great variety of event detection techniques for Twitter is also reflected in a couple of recently presented surveys [6], [9], [18], [23], [36]. At the same time and with the same variety, methods to evaluate the results of event detection techniques for Twitter appeared. For example, the "Social News on the Web" (*SNOW*) challenge [24] attempted to compare the results of different event detection techniques. However, instead of evaluating the different submissions of the teams automatically or even semi-automatically, a manual evaluation was conducted by a group of human evaluators. This choice is just one example that demonstrates the challenging and complex problem of evaluating event detection techniques automatically. As a consequence, proposals [32], [34], [35] for semi-automatic evaluation have been presented. In the following, we present the two major issues of reproducibility of event detection techniques: (i) the difficulty of reproducing the implementations of the techniques itself and (ii) the challenge to reproduce the evaluation or experiments of previous research works. We evaluated a total of 48 research works with regard to these issues.

A. Implementation Issues

A major goal of reproducible research is that successive researchers are able to reproduce previous works in order to build on them. Unfortunately, most current research works do not provide sufficient implementation details, source code, or even pseudo code to do so. We can derive that only 9 of the 48 research works provide the source code of their implementation to the public. The programming language of the source code is divided into Java (5), Python (3), and R (1). However, most of the works also depend on further components such as databases or libraries for which they fail to provide exact product or version information. At least, further 10 of the left 38 research works, provide pseudo code in the paper to enable partly reimplementation of the technique. The remaining publications offer a description of the algorithm, but lack all other information vital to support reproducibility.

B. Evaluation Issues

By investigating the different evaluation methods, we figured out that only 12 of the 48 research works perform a comparative evaluation. The largest number of competitors (three) is considered by Hua *et al.* [12] by comparing to [16], [27], [38]. Two competitors are each considered by Guille and Favre [11], Unankard *et al.* [29], Doulamis *et al.* [8], Xie *et al.* [40], and Zhang *et al.* [41]. One competitor is considered by further six research works [2], [3], [15], [38], [39], [42]. This analysis also shows that Weng and Lee [38] is the most commonly used comparison technique for the evaluations. We can observe that most works (17 of 48) performed one or several case studies to show the effectiveness and usefulness of their technique. Another large group of works (23 of 48) perform a stand-alone evaluation in order to rate the outcomes of their own technique only. In this case, the focus of the evaluation consists of 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 w.r.t. other techniques. A small fraction of evaluations (4 of 48) is based on user studies, where the results are evaluated by human evaluators. Most of the works are based on the Streaming API, but with different levels or restrictions. The Filter API (16 of 48) is the most popular choice. With this API, it is possible to obtain the data in a streaming fashion and to predefine filter queries based on keywords or geographical locations. The Spritzer access level (7 of 48) provides a uniform random 1% stream of the public timeline and is freely available to everyone. In contrast, the Gardenhose level (5 of 48) provides elevated access to a 10% stream, but needs special authorization. Note that Twitter does no longer provide Gardenhose access since the end of 2014. Apart from the streaming APIs, the Search API (9 of 48) 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. The sizes of these collections range from 0.6 million to around 1.2 billion tweets. With regard to the diversity in types of ground truths that are used to evaluate the results of a techniques, we figured out that most works (31 of 48) use a manually labeled set of events. Some of them also check the results of the technique manually to distinguish between real and non-real events. For example, Walther and Kaisser [30] manually checked for 1,000 clusters whether they belonged to a real-world event or not. 319 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, statistics of the Centers for Disease Control and Prevention can be used as ground truth to evaluate techniques that detect diseases (2 of 48). Similarly, match reports can be used for sport events, such as football games (*cf.* Meladianos *et al.* [21]). Finally, one work uses Wikipedia and another one uses Twitter's Trending Topics as ground truth. We also figured out that the works are very diverse in the measures that were proposed to evaluate the different techniques. Most of the times, precision and recall are used (18 of 48). Additionally, some works calculate the F_1 score, average precision, or the area under the receiver-operating curve. While such measures that evaluate the task-based performance of a technique are quite common, only six works ([3], [13], [22], [25], [39], [41]) apply a measure to evaluate the run-time performance. Apart from these well-known measures, some novel measures were defined. For example, Alvanaki *et al.* [3] measure *relative*

accuracy, whereas both Li *et al.* [15] and Guille and Favre [11] study the duplicate event rate of their techniques. Wurzer *et al.* [39] present the normalized topic-weighted minimum cost, which is a combination of miss and false alarm probabilities.

C. Evaluation Corpora

The second main issue in reproducible research on event detection techniques for Twitter is the lack of a common corpora of data, which can be used in experiments and evaluations. A couple of works focus on supplying evaluation corpora for Twitter-related analysis techniques. On the one hand, there are tools to create special tailored Twitter corpora, *e.g.*, TWORPUS [4] and TweetCaT [17]. However, both are simply interfaces to crawl the Twitter API for tweets that are defined by their identifier or other characteristics. On the other hand, there are research works which offer predefined tweet collections. Since the publication of the tweets themselves is prohibited by Twitter, these corpora are given as lists of tweet identifiers. McCreadie *et al.* [19] created a set of approximately 16 million tweet identifiers for a two-week period. Therefore, the proposed corpus contains an average of about 50,000 tweets per hour. Since no language filtering is performed, which can be estimated to retain approximately 30% of these tweets, we can assume that only about 4,800,000 tweets of the corpus are in English. Furthermore, their list of 49 reference topics for the two-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.* [5] created a Twitter corpus that consists of over 2,600,000 tweet identifiers 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 dataset is geographically restricted to tweets from users who are located in NYC. Petrović *et al.* [26] presented a corpus of 50 million tweet identifiers, created from a manual analysis of the Twitter data stream from July to 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 difficult to compare different event detection methods as they typically produce far larger numbers of events during the same period of time. McMinn *et al.* [20] propose a methodology for creating a corpus in order to evaluate event detection methods. They use two existing state-of-the-art event detection approaches [1], [26] together with Wikipedia to create a set of candidate events together with a list of associated tweets. The final corpus covers four weeks with about 120 million tweet identifiers and more than 500 events. However, events are described in prose and can, therefore, not be easily compared automatically to the results of various event detection techniques. Again, it is important to note that all of these corpora only consist of lists with tweet identifiers. In order to use these corpora for evaluation purposes, the actual tweets have to be crawled from Twitter, which is a time-consuming and error-prone process as tweets can get deleted over time. In order to study the implications of this process on reproducibility, we attempted to download the corpus of McMinn *et al.* [20]. The standard restriction of crawling tweets with the Twitter API is set to 180 calls per 15 minute window. With one call, it is possible to obtain a bulk of 100 tweets. Therefore, it would be possible to crawl 18,000 tweets per 15 minute window and it would take about 6,666 windows with an estimated total response time of 100,000 minutes ($\sim$1,666 hours or $\sim$69 days) on a single machine to crawl all tweets of the corpus. As this waiting time is prohibitive in practice, we implemented an alternative crawler that retrieves the content of tweets only using their identifiers. Even so, our crawler was only able to retrieve about 740,000 still available tweets out of a total of 1,850,000. More evidence of this phenomenon is shown by Lee *et al.* [14] by trying to recreate a Twitter corpora with originally 10,822 tweets, they could only reacquire 7,100 tweets (65.6%).

III. Experiments

In the previous section, we argued that it is currently almost impossible to reproduce existing evaluations of Twitter event detection techniques. In this section, we, therefore, present an approach—consisting of the design and implementation of a reproducibility toolkit[2] —to improve the reproducibility of experiments and evaluations in this setting. The survey presented in the previous section clearly outlines that there are three major issues propelled by the great diversity of research on Twitter event detection techniques. First, the input data used in the different research works is not common. Researchers are unable to publish their datasets for use in future research efforts. To solve this problem, we designed a simulator for the Twitter data stream specifically for the task of event detection. In this domain, the artificially created data stream can be used to detect events with techniques that build on statistic distributions on terms in the stream. Second, the implementations of the different techniques are not standardized. For this, we implemented two event detection techniques [33], [37] in a single data stream management system, which also can be extended with further techniques. Third, the results of the different event detection techniques are not evaluated in a common way. We solved this issue by designing an evaluation module with an integrated ground truth (dependent on the content of the artificial Twistor stream), that can be used for evaluation purposes with three evaluation measures. The three measures consist of the F_1 score for quality, the *throughput* for performance, and the *latency* as a usability measurement.

A. Setup

1) Twistor [28]: simulates the Twitter stream and provides the ability to embed predefined events into the data. With Twistor, it is possible to create artificial Twitter streams with very similar statistical properties as the original public Twitter stream. Twistor itself consists of two components. The first

[2]https://github.com/AWe/ecir2019 (Feb 10, 2019)

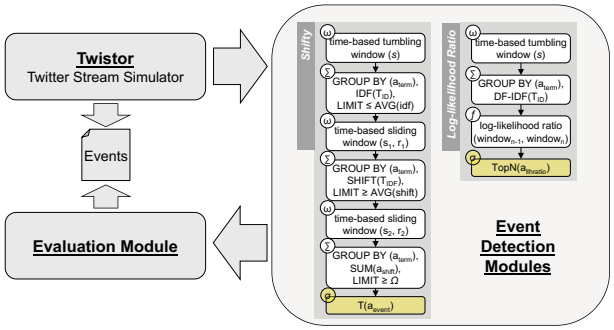

Fig. 1: Architecture of the reproducibility toolkit and query plans of the techniques implemented in the data stream system.

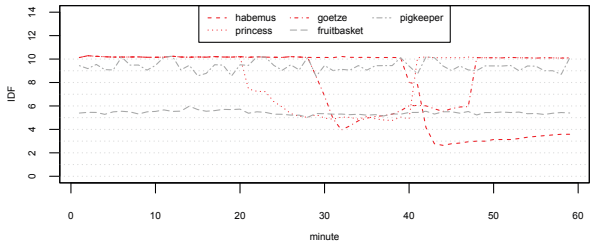

Fig. 2: IDF time-series of events (“habemus”, “princess”, “goetze”) and non-events (“fruitbasket”, “pigkeeper”).

component is the general simulation of the Twitter stream and the second one is the integration of events into the simulated stream. As a consequence, Twistor can be used as standardized input for evaluating different event detection techniques in a consistent way. To create a precise representation of the original Twitter stream the simulated Twitter stream is based on original Twitter data (collected with the Gardenhose access which represents 10% of the original Twitter stream). Over a time period of 24 hours the statistical term distribution in the original Twitter data is captured in a resolution of one-minute windows. The term distribution is stored for each window as base information. This base information needs to be created only once. As we need to make sure that no original information, with regard to contents, of the Twitter stream is contained in the artificial stream, we choose new terms with a random selection from the *Leipzig Corpora Collection* [10], which consists of terms in all languages. Since the distribution of the terms is quantified on the base information, the simulated stream can also be scaled up to 100% (still based on the 10% Gardenhose access) and so the Firehose (100% of the original Twitter stream) access level can be simulated. The integration of events into the simulated Twitter stream is also realized on the basis of original Twitter data. We provide the statistical representation of 10 predefined events. However, further events can easily be added by describing them with parameters or by providing their statistical properties. An event itself is represented by at least two words. The IDF values of the words representing the event are captured with a sample rate of one second at the time the event happened and stored as event description. To embed an event into the simulated Twitter stream, the IDF values of the terms representing the event are mapped into it. To map the IDF values of an event term into the artificial Twitter stream, the number of tweets per second which should contain the event term is computed and the event term is then inserted into this number of tweets. Figure 2 presents the evolution of event terms (“goetze”, “princess”, “habemus”) and non-event terms (“fruitbasket”, “pigkeeper”) during one hour of simulated Twitter data. We can see that the event terms have a drastic drop in their values, while the non-event terms have an almost constant rate. We can also see that the term “fruitbasket” is a very common term and the term “pigkeeper” is rarely used. However, note that these are the random selected terms from the *Leipzig Corpora Collection* and not the terms from the original stream.

2) Event Detection Modules: consist of the two event detection techniques—Shifty [33] and Log-Likelihood Ratio (LLH) [37]—that are realized as query plans (*cf.* Figure 1) in a data stream management system. Shifty is a technique that detects “abnormal” shifts in the IDF frequency of terms with a sliding window model. In contrast to that LLH detects events by calculating the difference of the log-likelihood ratio of terms in subsequent windows. Since the simulated data stream of Twistor is already cleaned-up, we removed all preprocessing steps of the event detection techniques. In previous works, events are defined as a collection of an event term with the corresponding most co-occurring terms as context. However, as the content of simulated data stream is not reflecting real terms as co-occurrences of terms in the tweets, we modified some of the operators in the query plans. We modified the output operators (*cf.* yellow marked operators in Figure 1) of the two techniques to only report the event terms without any added co-occurrence terms to them. The output operators create a result file which contains tab-separated rows with the attributes *id*, *date*, and *event_term*.

3) Evaluation Module: The evaluation module analyzes the reported events from the event detection modules against the ground truth, *i.e.*, the events introduced by Twistor into the data stream. For this, the Twistor module shares the information about the injected events as a description file with the evaluation module. In the description of an event, the properties of an event as the terms for the F_1 score measurement and the starting time for the latency measurement are included. To compare the task-based performance of the event detection modules, the evaluation module calculates values for precision, recall, and F_1 score. Additionally, it can be used to compare the run-time performance by tracking the throughput (tweets per second) of the event detection modules. As a further important measure, we evaluate the latency of the techniques. Especially for event detection it is very important that the techniques report the detected events as soon and close as possible to the real occurrence of the event.

B. Results

To evaluate the presented reproducibility toolkit, we applied the two aforementioned event detection techniques as well as two baseline techniques (TopN and RandomEvents) to an artificial Twistor data stream and automatically evaluated the results with the evaluation module. All experiments were conducted on virtual machines with two Intel single core processors at 2 GHz with 32 GB of main memory, running Oracle Java 10.0.1 (64-bit). As basis for the evaluation, we created a 10% Twitter data stream with 10 embedded events (*e.g.*"Boston Marathon Bombing, 2013", "Papal Conclave, 2013" or "MH17 Airline Crash, 2014"), which need to be detected by the techniques. The creation of the artificial stream takes less than a minute and results in a stream with about 1.5 million tweets in total and an average of 25,000 tweets per minute. In our experiments, we modified different parameters of each technique. For the RandomEvents, TopN, and LLH technique, we adjusted the size of the input window to 5, 10, 15, or 20 minutes and varied the number of reported events from 6 to 20. For Shifty, we adjusted the threshold value, which indirectly controls the number of reported events, from 10 to 69. The modification of these parameters results in 61 term sets for each technique. We then removed result sets that did not correspond to the detection of an event. Consequently, we excluded all results of the RandomEvents approach from further study. To improve the comparison of the results, we normalized the scores between 0 and 1. The latency measure is hereby calculated using difference between the detection time and the actual start of the event as a percentage of the full duration of the stream (60 minutes). For the throughput measure we had a look at the evolution of tweets per seconds for major events in the Twitter history. For example, the airing of the animation movie *Castle in the Sky*, caused an average of about 25,000 tweets per second in the year 2013. Therefore, we calculate the throughput measure on the assumption that processing of 30,000 tweets per second would be a perfect processing performance and leads to the best score of 1.0.

Figure 3 presents the results of these experiments. For the F_1 score, we can derive that LLH provides the best score for all parameter settings. Shifty provides a slightly better F_1 score than TopN. However, the variance of the F_1 score is much higher than TopN. The reason for this is that Shifty only performs well for a certain range of threshold values. Also, LLH and TopN analyze larger time windows and therefore have more data available than the streaming event detection technique Shifty. In terms of latency, we can observe that Shifty only has a latency of a few seconds, due to its streaming implementation. In contrast, the other two techniques have a latency of a couple of minutes. By looking at the throughput score, we can derive that Shifty is the slowest approach. Due to its streaming implementation, Shifty processes many small windows over the data stream. Nevertheless, the throughput is still high enough to process all incoming tweets in real-time and report events with very low latency.

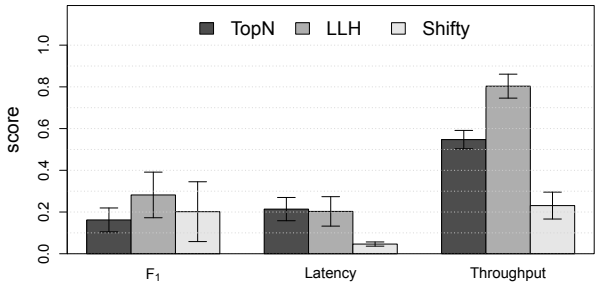

Fig. 3: F_1, Latency, and Throughput of TopN, Shifty, and LLH.

IV. Conclusions and Future Work

In this paper, we addressed the problem of reproducibility of event detection techniques by applying our techniques to a simulated artificial data stream. Our results show that it is possible to evaluate the techniques in respect to quality, latency, and performance. Since all of the techniques require a certain set of parameters, it is possible to tune the parameter setting by using our reproducibility toolkit. One drawback of the artificial data stream is that, because of the lack of metadata such as retweet information, location, or user information, only event detection techniques using statistical measures to detect the events can be applied to this stream. Therefore techniques that integrate metadata into the event detection (*e.g.*, [30], [31]) are currently unable to use the artificial stream created with Twistor. As future work, it is important to categorize the injected events according to importance levels as some events need to be detected (*e.g.*, "Goal at the final of the World Cup"), while others could be detected (*e.g.*, "Boston Marathon Bombing") or are almost impossible to detect (*e.g.*, "Killing of Boris Nemzow"). With this information, the different techniques can be evaluated in a more detailed manner. Since the ground truth only contains events that are known beforehand, it will be interesting to combine these evaluations with studies (*e.g.*, [34]–[36]) that are also allow for serendipitous events. From a technical point of view, it would be a great opportunity to have a framework in which researchers could add their own event detection modules and evaluate their approach against all others.

Acknowledgement

The research presented in this paper is funded in part by the Deutsche Forschungsgemeinschaft (DFG), Grant No. GR 4497/4: "Adaptive and Scalable Event Detection Techniques for Twitter Data Streams".

References

[1] C. C. Aggarwal and K. Subbian. Event Detection in Social Streams. In *Proc. SIAM Intl. Conf. on Data Mining (SDM)*, pages 624–635, 2012.

[2] L. M. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Göker, and I. Kompatsiaris. Sensing Trending Topics in Twitter. *IEEE Transactions on Multimedia*, 15(6):1268–1282, 2013.

[3] F. Alvanaki, S. Michel, K. Ramamritham, and G. Weikum. See What's enBlogue: Real-time Emergent Topic Identification in Social Media. In *Proc. Intl. Conf. on Extending Database Technology (EDBT)*, pages 336–347, 2012.

[4] A. Bazo, M. Burghardt, and C. Wolff. TWORPUS - An Easy-to-Use Tool for the Creation of Tailored Twitter Corpora. In *Proc. Intl. Conf. on Language Processing and Knowledge in the Web (GSCL)*, pages 23–34, 2013.

[5] H. Becker, M. Naaman, and L. Gravano. Beyond Trending Topics: Real-World Event Identification on Twitter. In *Proc. Intl. Conf on Weblogs and Social Media (ICWSM)*, pages 438–441, 2011.

[6] K. Bontcheva and D. Rout. Making Sense of Social Media Streams through Semantics: a Survey. *Semantic Web*, 5(5):373–403, 2014.

[7] W.-L. Cheung. Reproducibility of Event Detection Techniques on Social Media Data. Master's thesis, University of Konstanz, 2017.

[8] N. Doulamis, A. Doulamis, P. Kokkinos, and E. Varvarigos. Event Detection in Twitter Microblogging. *IEEE Transactions on Cybernetics*, 46(12):2810–2824, 2016.

[9] A. Farzindar and W. Khreich. A Survey of Techniques for Event Detection in Twitter. *Computational Intelligence*, 31(1):132–164, 2015.

[10] D. Goldhahn, T. Eckart, and U. Quasthoff. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *Proc. Intl. Conf. on Language Resources and Evaluation (LREC)*, pages 759–765.

[11] A. Guille and C. Favre. Mention-anomaly-based Event Detection and Tracking in Twitter. In *Proc. Intl. Conf. on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 375–382, 2014.

[12] T. Hua, F. Chen, L. Zhao, C.-T. Lu, and N. Ramakrishnan. Automatic targeted-domain spatiotemporal event detection in twitter. *GeoInformatica*, 20(4):765–795, 2016.

[13] S. Kaleel and A. Abhari. Cluster-discovery of Twitter messages for event detection and trending. *Journal of Computational Science*, 6:47–57, 2015.

[14] K. Lee, A. Qadir, S. A. Hasan, V. Datla, A. Prakash, J. Liu, and O. Farri. Adverse Drug Event Detection in Tweets with Semi-Supervised Convolutional Neural Networks. In *Proc. Intl. Conf. on World Wide Web (WWW)*, pages 705–714, 2017.

[15] C. Li, A. Sun, and A. Datta. Twevent: Segment-based Event Detection from Tweets. In *Proc. Intl. Conf. on Information and Knowledge Management (CIKM)*, pages 155–164, 2012.

[16] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang. TEDAS: A Twitter-based Event Detection and Analysis System. In *Proc. Intl. Conf. on Data Engineering (ICDE)*, pages 1273–1276, 2012.

[17] N. Ljubešić, D. Fišer, and T. Erjavec. TweetCaT: a Tool for Building Twitter Corpora of Smaller Languages. In *Proc. Intl. Conf. on Language Resources and Evaluation (LREC)*, pages 2279–2283, 2014.

[18] A. Madani, O. Boussaid, and D. E. Zegour. What's Happening : A Survey of Tweets Event Detection. In *Proc. Intl. Conf. on Communications, Computation, Networks and Technologies (INNOV)*, pages 16–22, 2014.

[19] R. McCreadie, I. Soboroff, J. Lin, C. Macdonald, I. Ounis, and D. McCullough. On Building a Reusable Twitter Corpus. In *Proc. Intl. Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 1113–1114, 2012.

[20] A. J. McMinn, Y. Moshfeghi, and J. M. Jose. Building a Large-scale Corpus for Evaluating Event Detection on Twitter. In *Proc. Intl. Conf. on Information and Knowledge Management (CIKM)*, pages 409–418, 2013.

[21] P. Meladianos, G. Nikolentzos, F. Rousseau, Y. Stavrakas, and M. Vazirgiannis. Degeneracy-Based Real-Time Sub-Event Detection in Twitter Stream. In *Proc. Intl. Conf. on Weblogs and Social Media (ICWSM)*, pages 248–257, 2015.

[22] D. T. Nguyen and J. E. Jung. Real-time event detection for online behavioral analysis of big social data. *Future Generation Computer Systems*, 66:137–145, 2017.

[23] A. Nurwidyantoro and E. Winarko. Event Detection in Social Media: a Survey. In *Proc. Intl. Conf. on ICT for Smart Society (ICISS)*, pages 1–5, 2013.

[24] S. Papadopoulos, D. Corney, and L. M. Aiello. 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)*, pages 1–8, 2014.

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

[26] S. Petrović, M. Osborne, and V. Lavrenko. Using Paraphrases for Improving First Story Detection in News and Twitter. In *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pages 338–346, 2012.

[27] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. In *Proc. Intl. Conf. Companion on World Wide Web (WWW)*, pages 851–860, 2010.

[28] H. Schilling. Twistor - Simulation des Twitterstroms für Evaluationszwecke. In *46. Jahrestagung der Gesellschaft für Informatik, Informatik*, pages 2133–2143, 2016.

[29] S. Unankard, X. Li, and M. Sharaf. Emerging event detection in social networks with location sensitivity. *World Wide Web*, 18(5):1393–1417, 2015.

[30] M. Walther and M. Kaisser. Geo-spatial Event Detection in the Twitter Stream. In P. Serdyukov, P. Braslavski, S. O. Kuznetsov, J. Kamps, S. Rüger, E. Agichtein, I. Segalovich, and E. Yilmaz, editors, *Advances in Information Retrieval*, volume 7814 of *Lecture Notes in Computer Science*, pages 356–367. Springer-Verlag, 2013.

[31] H. Wei, J. Sankaranarayanan, and H. Samet. Finding and Tracking Local Twitter Users for News Detection. In *Proc. Intl. Conf. on Advances in Geographic Information Systems (SIGSPATIAL)*, pages 64:1–64:4, 2017.

[32] A. Weiler, J. Beel, B. Gipp, and M. Grossniklaus. Stability Evaluation of Event Detection Techniques for Twitter. In *Proc. Advances in Intelligent Data Analysis (IDA)*, pages 368–380, 2016.

[33] A. Weiler, M. Grossniklaus, and M. H. Scholl. Event Identification and Tracking in Social Media Streaming Data. In *Proc. Workshop on Multimodal Social Data Management (MSDM) in conjunction with Intl. Conf. on Extending Database Technology (EDBT)*, pages 282–287, 2014.

[34] A. Weiler, M. Grossniklaus, and M. H. Scholl. Evaluation Measures for Event Detection Techniques on Twitter Data Streams. In *Proc. British Intl. Conf. on Databases (BICOD)*, pages 108–119, 2015.

[35] A. Weiler, M. Grossniklaus, and M. H. Scholl. Run-time and Task-based Performance of Event Detection Techniques for Twitter. In *Proc. Intl. Conf. on Advanced Information Systems Engineering (CAiSE)*, pages 35–49, 2015.

[36] A. Weiler, M. Grossniklaus, and M. H. Scholl. Survey and Experimental Analysis of Event Detection Techniques for Twitter. *Oxford Computer Journal*, 60:329–346, 2017.

[37] A. Weiler, M. H. Scholl, F. Wanner, and C. Rohrdantz. Event Identification for Local Areas Using Social Media Streaming Data. In *Proc. Workshop on Databases and Social Networks (DBSocial) in conjunction with Intl. Conf. on Management of Data (SIGMOD)*, pages 1–6, 2013.

[38] J. Weng and B.-S. Lee. Event Detection in Twitter. In *Proc. Intl. Conf on Weblogs and Social Media (ICWSM)*, pages 401–408, 2011.

[39] D. Wurzer, V. Lavrenko, and M. Osborne. Twitter-scale New Event Detection via K-term Hashing. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[40] W. Xie, F. Zhu, J. Jiang, E.-P. Lim, and K. Wang. TopicSketch: Real-Time Bursty Topic Detection from Twitter. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2216–2229, 2016.

[41] C. Zhang, G. Zhou, Q. Yuan, H. Zhuang, Y. Zheng, L. Kaplan, S. Wang, and J. Han. GeoBurst: Real-Time Local Event Detection in Geo-Tagged Tweet Streams. In *Proc. Intl. Conf. on Research and Development in Information Retrieval (SIGIR)*, pages 513–522, 2016.

[42] X. Zhou and L. Chen. Event Detection over Twitter Social Media Streams. *The VLDB Journal*, 23(3):381–400, 2014.