Erschienen in: Proceedings of the 5th Ph.D. workshop on Information and knowledge - PIKM '12. - New York, New York, USA : ACM Press, 2012. - S. 87-90. - ISBN 978-1-4503-1719-1
https://dx.doi.org/10.1145/2389686.2389704

Towards an Advanced System for Real-Time Event Detection in High-Volume Data Streams

Andreas Weiler, Svetlana Mansmann, Marc H. Scholl
Database and Information Systems Group, University of Konstanz
Box D 188, 78457 Konstanz, Germany
{firstname.lastname}@uni-konstanz.de

ABSTRACT

This paper presents an advanced system for real-time event detection in high-volume data streams. Our main goal is to provide a system, which can handle high-volume data streams and is able to detect events in real-time. Additionally, we perform further steps, such as classifying and ranking events with retrospective analysis. To solve this task we take advantage of a high-performance database system for semi-structured data and extend it with the functionality of continuous querying. The combination of executing queries on the incoming data stream and fast queries on the historical datasets is used as a powerful tool for developing an event detection and information system. Furthermore, we define several event features for improving event classification and for discovering parallelisms, relations, duration, and coherences of events.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Storage and Retrieval

Keywords

event detection, information extraction, stream processing

1. INTRODUCTION AND MOTIVATION

Recent years have witnessed a continuous development of social media services on the web. Unprecedented success and active usage of these services result in massive amounts of user-generated data produced within a very short time. The continuous emergence of new services, such as social media platforms and technologies for generating and receiving streamed data, imposes new challenges on the way such data volumes are processed and analyzed in real-time or near real-time. Popular examples are microblogging services, such as Twitter. Initially introduced in 2006 as a simple platform for exchanging short messages ("tweets") on the Internet, Twitter rapidly gained worldwide popularity and has evolved into an extremely influential channel of broadcasting news and the means of real-time information exchange. Twitter has revolutionized the ways of interacting and exchanging information on the Internet and opened new ways for knowledge acquisition from social interaction streamed in real-time. As of 2012 the Twitter platform attracts over 140 million active users generating over 340 millions of tweets daily[1]. A crowd of users broadcast a variety of personal matters, most current news, or spam and advertisements into the World Wide Web. The resulting data flow contains a large volume of live data with all types of information from all over the world. Considering the speed, the volume and the noisiness of the streaming data, it becomes a crucial task to discriminate between useful and useless information in the incoming data stream in appropriate time. This brings up new challenges for data processing and analysis. An important prerequisite for achieving the fastest possible response times is efficient processing of the streaming data. The fast propagation of news and current events over microblogging services makes it very interesting to detect such events automatically and in real-time. We have to tackle several issues to provide an efficient and effective live event detection system.
On the one hand, a system is required to keep the data moving through the on-line algorithms to get the most valuable results in the shortest time frame possible. On the other hand, there is a need for the functionality of a database system for accessing the historical data already stored for further steps of the analysis and for comparing freshly detected events with past occurrences. For example a recent earthquake in Indonesia (see chapter 5) is detected based on mostly non-English terms. However by using the historical data belonging to the event we are able to enrich the event by English terms such as "earthquake", "indonesia", and "warning". As a result, there is an urge to combine a passive and inflexible database system with a real-time and active one and deploy the advantages from both into a powerful combination.

2. SYSTEM ARCHITECTURE

We adopt a native XML database system BaseX[2] as an underlying architecture for our proposed system. The core component combines the storage layer and the streaming processor into one system. The event detection system as well as the visualizations can be assembled in a distributed fashion by using several connections to the streaming data or

[1]http://blog.twitter.com/2012/03/twitter-turns-six.html
[2]http://www.basex.org

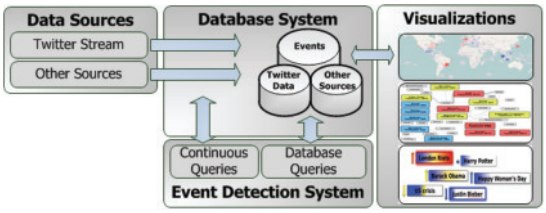

Figure 1: Architecture overview of the proposed system.

the database server. Hereby we ensure that all components are decoupled and can be extended with further components. One of the challenging tasks is the combination of continuous querying applied to the stream and queries against data already stored in the database. This combination leads to two challenges we have to tackle: first, we encounter an ongoing load of queries for the event detection and, second, we have to deal with uncertain peaks of query load for querying the historical data. Due to storage and performance constraints it is impossible to store the entire incoming data stream in the database for unlimited time. Therefore, we rather focus on the last few hours of recent data for extracting additional information for the events.

Another challenge is to combine and integrate multiple heterogeneous data sources into a homogeneous system. With the usage of XML we tackle this challenge and provide an adaptable system for all kinds of sources. For example, Twitter data is streamed in JavaScript Object Notation format and can be easily and quickly converted into XML. With this solution the system automatically adapts to possible future format changes of the streamed Twitter data.

Figure 1 shows our proposed architecture, which supports direct and continuous querying applied to on-line stream as well as execution of queries against data already stored in the database. As for query language, the database system supports the full functionality of XQuery[3]. The incoming data stream runs through the continuous query system for detecting events on-the-fly. For example, it is possible to filter the incoming items by defining a query for returning only tweets with geographic information included. Simultaneously to the stream processing, the data items will be persistently stored into the database to be available for future information acquisition. Additionally, we store already detected event patterns into the database to provide support for matching newly detected events with past events, detecting relationships between events, and offering an information system about all past events.

3. TWITTER

Twitter has fundamentally changed the way messages are formulated, published, and distributed in the World Wide Web. Prominent characteristics of Twitter, such as the limitation of the message length to just 140 characters and the freedom for users to create messages in a fast and easy way without rules, regulations or inspections applied, brought a new era of communication into being.

A further important property of the Twitter data is that each tweet object in the data stream contains the tweet message itself and a very large amount of meta-data for the tweet (e.g. count of retweets, geographic location) and the user's profile (e.g. count of followers). Through this multidimensionality with over 60 data fields we can derive such dimensions as the time of the tweet, its geographic location or the source, the latter describing the way the tweet is created. For example, the Twitter message can be sent from a mobile device, by just clicking a Twitter button on any website or by entering the message directly via the Twitters' web interface. This information can be really helpful for separating between a personally created tweet and a pure forwarding of a web article. Other kind of meta-data specific for Twitter is the count of retweets and the mention of hashtags or users in the tweets. By using the count of retweets we can provide an indication about the range of users who already received the respective tweet. This additional knowledge can be very useful for detecting and characterizing different events.

[3]http://www.w3.org/TR/xquery/

4. EVENT DETECTION

Event Detection is a classical problem that has been under discussion for several years in many research areas. A lot of research tackles the detection of anomalies [4], which can be an indication of an event. Our research is mostly related to anomaly detection in text data streams and our analysis also mainly concentrates on the content of short messages. Hereby we can discover abnormally high rates of terms in the textual input. However, we also want to combine the analysis of the textual part of the tweet with acquired knowledge about events and anomalies from different dimensions provided by the meta-data. For example, we can apply the event detection analysis on the geographical dimension of tweets to determine abnormally large amount of tweets in specific regions or cities, which could be an indication of a local or global happening.

We also want to classify events in order to distinguish between certain types of events. Using this acquired knowledge we can set up hierarchies and relations between the recognized and classified events. To achieve all these goals in real-time we need to develop algorithms to make them applicable for on-line event detection and combine them with retrospective analysis for further knowledge gains [12].

4.1 Phases of Event Detection

For an easier separation of single steps in the event detection framework, we defined three phases that have to be implemented using either the on-line stream or the historical dataset. The following summarizes the phases of the event detection system:

1. **Detect and Identify:** The first phase is concerned with detecting unusual behavior of defined measures in the incoming data stream. We have to distinguish between useful and useless items in a fast-moving, never ending, noisy, and high-volume data stream, which imposes the necessity to develop new or adapt existing algorithms for on-line execution and immediate delivery of the results.
2. **Specify and Classify:** The second phase is specification and classification of the event. Here we identify various features of an event and assign the latter to a defined or newly generated event class. For example, the classification can be supported by meta-data inspection or details from information sources such as DBPedia[4] or WordNet[5]. Additionally we enrich the events with fre-

[4]http://dbpedia.org/
[5]http://wordnet.princeton.edu/

quent terms extracted out of the historical data, which occurred in conjunction with the event term. Further information gain can be achieved by correlation with other sources (e.g., RSS feeds).

3. **Rank and Relate:** The next phase is event ranking and discovery of relations between and among certain events. We have to distinguish between important (e.g. highest amount of tweets or most retweeted) and less important events as well as whether and how they are related to each other. This knowledge opens up the possibility to create hierarchical structures of events and to model relationships between them.

4.2 Event Types and Features

By enumerating a multitude of possible characteristics of an event, we can detect related events, such as equal or similar events at different geographical locations or events that are follow-ups of other events. A further advantage is that we can find duplicate events or events caused by other events and build up a taxonomy of event types.
On the one hand, events can be described by a pre-defined set of characteristics and features. On the other hand, events are highly versatile and it should be possible to discover arrival of new events, i.e., such that have never occurred before, in order to describe them by introducing new features. This is a kind of unsteadiness we have to tackle within our research. The most common descriptive characteristics of events can be summarized as follows:

Type: Determines what other information is needed to properly characterize the given event. Events can be of various types, e.g., social, natural disaster, or criminal activity.

Geographic: Describes where the event took place and what locations are involved in the event. This way, we can distinguish between global and local events or between centralized and widely scattered ones, etc.

Temporal: Describes the event's time frame along with the starting and the ending point. By using the time frame we can differentiate between short-term, medium-term or long-term events [5]. With the knowledge from the event history we can additionally distinguish between one-time happenings or reoccurring events.

Social: Contains data about the actors or the concerned persons of an event. Hereby we can classify individual events (e.g., death of a famous artist) or mass events (e.g., political riots).

Sentiment: Reflects the emotions of the event, giving us an opportunity to distinguish between positive and negative events. We also have to deal with mixed emotional characteristics, such as political elections or sport events.

5. MOTIVATING EXAMPLE

The data collection for our work was obtained by storing the incoming items of the public Twitter stream directly into a database, with new database instances created on an hourly basis. We are connected to the Twitter endpoint using the Streaming API[6] with the so-called "gardenhose" level, which contains 10% of the public stream provided by Twitter. As a result, we obtain a data stream with up to 2 million tweets per hour with the average of 20.000 tweets per minute. We can conclude that around 10% of the incoming tweets have geographic information, which is either set automatically by the mobile device, manually by the author of the tweet or both of them, available. Our training collection currently covers the last few months of the Twitter stream. In the following example we show event detection concentrating on the textual analysis of the content of the tweets. We analyze abnormal peaks in the IDF value of terms and assume that the resulting terms are "event terms". In this case we simulate a live stream with the Twitter data of the hour from an earthquake happening on the Indonesian island of Sumatra at 08:38:38 AM UTC on Wednesday, April 11th 2012. Since we only need the text of the tweets, we run a continuous filter query for the text field on the stream and discard all other unused data fields.
Since pure tokenization of texts results in a lot of terms, but the main subject of a statement is normally present in nouns, the tokens are tagged with their part of speech by using especially tailored for Twitter Part-Of-Speech Tagger [6] and then limited to nouns and proper nouns for further treatment. All other tokens are currently discarded. Figure 2 shows the IDF Trend of the resulting possibly event terms. For the purpose of checking whether a term is an event term or not we calculate the IDF Trend of the current occurrence of a term by using the following rules:

1. Let x_n (n is the actual occurrence) be the number of tweets run through the system since the last occurrence of the term. If result res ($res = log(x_n) + log(x_{n-1}) - log(x_{n-2})$) is less than the threshold 4, increase number of *hits* by one, else decrease number of *hits* by one.

2. If number of *hits* is greater than 20, set the term as possible event term.

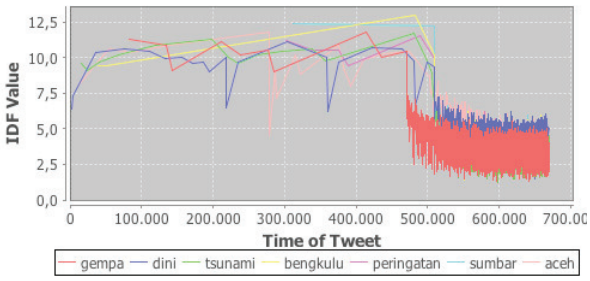

Figure 2: IDF Trend of resulting event terms.

By using the above rules we can detect seven candidates for event terms in the hour of the earthquake event already mentioned before. Most of them are related to the event and can be detected in the last quarter of the hour, which corresponds to the time when the event takes place. Future work and further analysis is needed to validate the possible event terms and to mark them as definite or unreliable. Additionally we have to enrich the identified event terms with the information from the historical data and investigate their correlation and relationships. First exemplary results show that we can enrich the event by detecting frequent terms, such as "earthquake", "indonesia", and "warning", in the appropriate tweets containing one of the event terms.

6. RELATED WORK

The work related to our research can be summarized by

[6]https://dev.twitter.com/docs/streaming-api

topics such as stream processing systems and knowledge discovery and event detection from data streams and especially from Twitter. Stream processing was introduced into the world of relational databases by systems such as Aurora [1], Borealis [2], and STREAM [3] which provide a SQL interface for accessing and processing the streaming data. In contrast to these systems we concentrate on extending an XML database for accessing and processing semi-structured data in an on-line fashion.
Mathioudakis and Koudas [8] presented a system called "Twitter Monitor", which detects trends from Twitter streams in real-time by identifying and grouping bursty keywords in the tweets. The system further actively involves the users to order the trends and provide their own descriptions for them. Contrary to this system we propose to automatically process the phases of our proposed event analysis, such as the functionality for specifying, classifying and correlating events. Sankaranarayanan et al. [10] proposed a tool called "TwitterStand", which detects breaking news from Twitter by observing some identified special users ("Seeders"), who are mainly responsible for the occurrence of news stories. An event detection system dedicated to earthquakes was presented by Sakaki et al. [9]. In 2011 Weng et al. [11] used wavelet analysis on the frequency-based raw signals of the words from tweet for detecting events. They used a keyword-filtered dataset to show their practical usage for identifying events during the Singapore General Election in 2011. In the same year Marcus et al. [7] demonstrated an application called "TwitInfo", which identifies and labels event peaks for given search queries related to the event. In contrast to our proposed idea which uses an unfiltered data stream, all of the above mentions systems are somehow restricted.

7. CONCLUSIONS AND FUTURE WORK

In this paper we proposed an advanced system for event detection on high-volume data streams. We demonstrated the ability to detect events in a large collection of tweets by identifying abnormal peaks via inverse document frequency analysis without predefined keywords. These results provide evidence that the Twitter data stream is rich in information and events can be identified. Since the data contains further interesting dimensions such as the geographic information, we also extend the event detection algorithm to take such dimensions into account.
The main challenges are now to extend and devolve our existing approaches to work in an on-line and real-time fashion on the data streams by developing and enhancing algorithms and to implement further extensions and improvements of the underlying system. For example, the database system has to be equipped with a functionality to execute parallel writing transactions to store the incoming streaming data and reading transactions for acquiring knowledge for further event analysis. One step in the future work is to analyze large amounts of historical tweets to identify daily or weekly patterns and to include these in the analysis. Hereby we get the knowledge about default progression of terms at particular times. Another step is to add further data sources to our system and correlate these data sources with the detected events from the Twitter stream. Hereby we can obtain further information about the detected events and that can also help to decide if an event is "real" or not. Additionally we also want to detect events occurring in the incoming stream of other sources, such as RSS feeds, which do not appear in the Twitter stream.
We contribute a scalable and high-performance system that provides both real-time and retrospective analysis combined in a core component. The database system plays a powerful role as a continuous active and a constantly available passive query processor. A further contribution is the integration and consolidation of many heterogeneous data sources for obtaining the largest possible information gain through our system. We make use of various independent sources and combine them into a single information base for real-time event detection and enrichment.

8. REFERENCES

[1] D. Abadi et al. Aurora: A Data Stream Management System. In *In ACM SIGMOD Conference*, 2003.

[2] D. Abadi et al. The Design of the Borealis Stream Processing Engine. In *Second Biennial Conference on Innovative Data Systems Research (CIDR 2005)*, 2005.

[3] A. Arasu et al. STREAM: The Stanford Data Stream Management System. Technical Report 2004-20, Stanford InfoLab, 2004.

[4] V. Chandola et al. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.

[5] M. Cheong and V. Lee. Integrating web-based intelligence retrieval and decision-making from the twitter trends knowledge base. In *Proceedings of the 2nd ACM workshop on Social web search and mining*, SWSM '09, pages 1–8. ACM, 2009.

[6] K. Gimpel et al. Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments. In *ACL (Short Papers)*, pages 42–47, 2011.

[7] A. Marcus et al. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 227–236. ACM, 2011.

[8] M. Mathioudakis and N. Koudas. TwitterMonitor: trend detection over the twitter stream. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10, pages 1155–1158. ACM, 2010.

[9] T. Sakaki et al. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860. ACM, 2010.

[10] J. Sankaranarayanan et al. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 42–51. ACM, 2009.

[11] J. Weng et al. Event Detection in Twitter. Technical report, HP Labs, 2011.

[12] Y. Yang et al. A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 28–36. ACM, 1998.