SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM

TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Volume 5 No. 6
ISSN 2054-7390

Twitter Sports: Real Time Detection of Key Events from Sports Tweets

[1]Jeyakumar Kannan, [2]AR. Mohamed Shanavas, [3]Sridhar Swaminathan
[1,2]Department of Computer Science, Jamal Mohamed College, Tiruchirappalli, India;
[3]Department of Computer Science and Engineering, Bennett University, Greater Noida, India;
meetjey@gmail.com; arms3375@gmail.com; sridhar.swaminathan@bennett.edu

ABSTRACT

Twitter users play a role of human sensors and update information about real-life events by posting their tweets about them. Event detection in Twitter is the process of detecting an event which is an occurrence causing change in the volume of tweets that discuss the associated topic at a specific time and a location by Twitter users. Twitter has been extensively used to detect major social and physical events such as *earthquakes*, *celebrity deaths*, *presidential elections*, *traffic jam* and others. Real time event detection in Twitter is detecting real-life events from live tweets instantly as soon as the event has occurred. Real time event detection from Cricket sports using Twitter media is an interesting, yet a complex problem. Because, event detection algorithm needs live tweets streamed at real-time about the game and should detect events such as *boundary* and *sixer*, at near real-time within few seconds from their occurrences. In this paper, a novel real-time event detection approach is proposed for the Cricket sports domain. The proposed approach first computes the post rate of an adaptive window, which is the ratio between the volumes of tweets in the second half window and the volume of tweets in the first half. An event has occurred if the post rate is above the pre-defined threshold, otherwise the algorithm selects the next big window in an adaptive manner. The predefined threshold helps to filter out the small spikes in the streaming tweets volume. Once an event is detected in a time window along the tweet stream, the event represented inside the window is recognized using the event lexicon representing different events of a cricket game. The proposed real-time event detection algorithm is extensively evaluated on 2017 IPL T20 Cricket sports dataset using ROC and AUC evaluation measures. The experimental results on the performance of the proposed approach show that the adaptive sliding window detects sports events with over 80% true positives and around 15% false positive rates.

Keywords: Social media; Microblogs; Twitter; Event detection; Sports events; Adaptive sliding windows.

1 Introduction

Modern day internet, web and mobile technologies have now enabled vast majority of people around the world to communicate with each other through social media services such as Facebook, Twitter etc. Online social media services have changed the way communication happened between people, groups, and communities [1]. These social media has become a platform for many of its users to express their ideas, opinions and share information to the rest of the world. Microblogging is one such form of social media broadcasting medium where users share small digital content such as short texts, links, images, or

DOI: 10.14738/tmlai.56.3729
Publication Date: 2nd Oct, 2017
URL: http://dx.doi.org/10.14738/tmlai.56.3729

videos [2]. Microblogging has become famous and highly used by numerous people, organizations and researchers from different fields around the entire world, despite it is a totally new medium of communication when comparing to the traditional social media. It allows users to share information and respond to different opinions quickly. Despite the constraints on size of the information being shared in the microblogging, it has gained wide spread usage due to its features such as easier portability, unrestricted content, quickness and ease of usage in communication.

Among the popular social media services in microblogging, Twitter is one of the most widely used and fast-growing microblogging social networking service which has more than 500 million users around the world [3]. Twitter allows its users to share a short text called *Tweet*, which is no longer than 140 characters, by using different communication services such as smartphones, web interfaces and social media apps. Twitter differs from other social networks by being a micro-blogging service that limits the size of messages. This feature allows Twitter users to publish short messages, in a faster and summarized way, makes it the preferred tool for the quick dissemination of information over the web. The content of a tweet highly varies based on individual user's interests and behaviors [4]. These tweets contain wide range of information such as advice, opinions, moods, concerns, facts, rumors, world news, and general information, report of important events [5]. In the context of online social networking, social media users can be regarded as sensors reporting important information.

People, community and organizations can be well informed of live happenings around the world from the dynamic source of information in Tweets. Twitter users are also interested in receiving different tips, opinions, live updates on news from the other Twitter users [4]. Corporations use Twitter to make announcements of products, services, events, and news media companies use Twitter to publish near real-time information about breaking news. Several organizations have started utilizing Twitter as a platform for advertisements, product, and service recommendations. They also started to exploit reports from sentiment analysis to build and maintain reputations of their products by responding to the complaints from customers and dynamically improving their decisions [6]. Twitter has become a quick communication medium for obtaining and sharing of viral news [7], election results prediction [8], and crime prediction [9]. Invaluable information can be gained by the continuous monitoring and analysis of rich user-generated content. For example, it is easy to understand the top trending topics in sports such as cricket by analyzing the most frequent terms from tweets. Figure 1 to 4 depict the top 25 terms and bigrams of two of the games we crawled during IPL T20 season 2017. From these figures, it is evident that Twitter users generally discuss their favorite players and teams throughout the game, besides Cricket events. This information would not have been obtained from any other traditional media outlets.

Events can be generally defined as real-world occurrences that unfold over space and time [10]. Research on social media analytics has shown that important real-life events can be detected from the information provided by the human sensors where the credibility of the information being shared by the users were always high most of the time. Automatic event detection from microblogging social media becomes necessary, due to a large volume of data, redundancy in information reporting events and presence of noises or false information about the events. Recent research on event detection has shown that Twitter can provide insights into a detection of major social and physical events such as earthquakes, celebrity deaths, and presidential elections [11]. Event detection from Twitter is considered difficult since the detection algorithm should cope with challenging factors such as limited text length, structural and grammatical errors, rumors and misinformation. In Twitter event detection, the underlying assumption

that some related words would show an increase in the usage when an event is happening is not a viable method. In comparison to traditional event detection from news wire, Twitter stream include a much higher volume of data flooded by high amounts of meaningless messages.

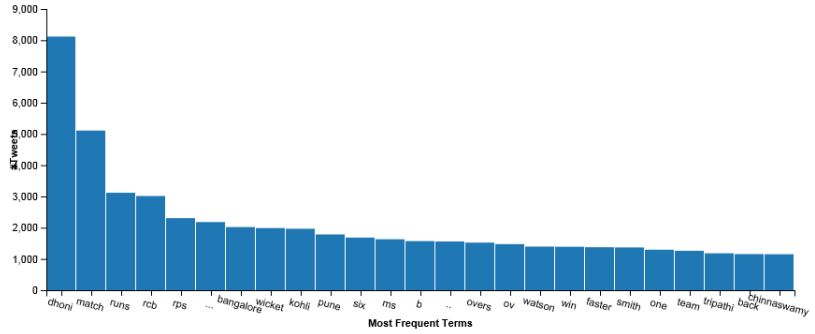


Figure.1. Most frequent terms for RCBvRPS on 16 April 2017

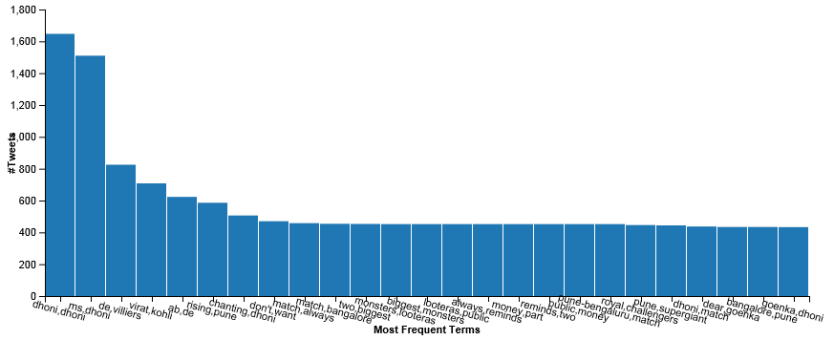


Figure 2. Most frequent bigrams for RCBvRPS on 16 April 2017

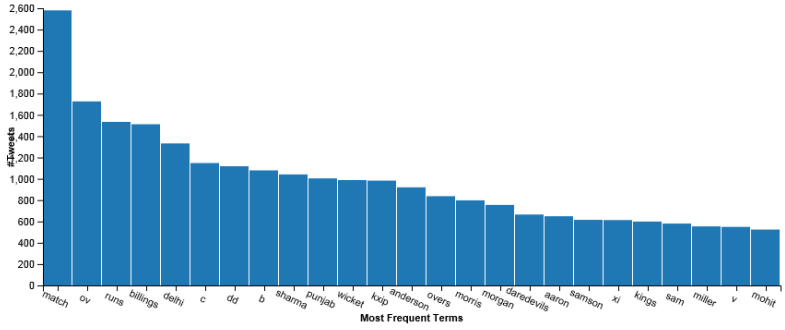


Figure 3. Most frequent terms for DDvKXIP on 15 Apr 2017

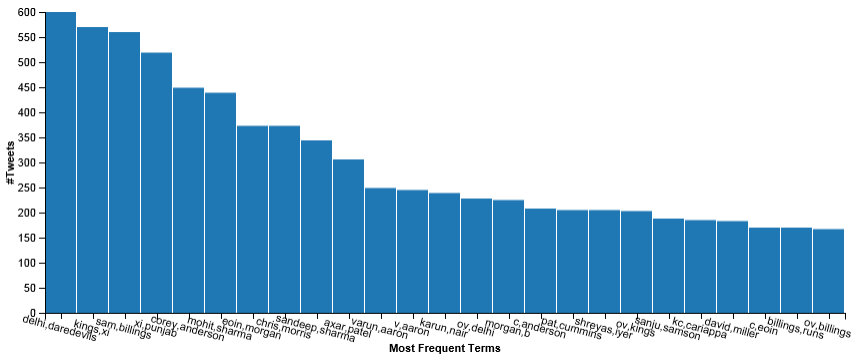


Figure 4. Most frequent bigrams for DDvKXIP on 15 Apr 2017

In this digital era, news about an event instantly reaches the other side of the world within seconds as an event happens. Millions of people around the world have started using Twitter for reporting significant events in real-time. Real-time detection and recognition of events from Twitter is yet another challenging study in recent times. There are lot of advantages in detecting events from real-life using real-time event detection in Twitter in situations such as catastrophic events, important deaths, political events, elections and campaigns. Real-time event detection is considered highly challenging due to various factors such as gathering and processing data at real-time for societal and business applications.

Although event detection has been a well-studied problem for over a decade, limited number of research work have been carried out in sports domain. Even in domain of sports, most of research work focused on NFL soccer games and detected events from offline datasets. Recently, few researchers have considered detecting key events from NFL games at real-time. Nevertheless, there are no studies that investigated *Cricket* sports as a domain for the detection of events from live tweets at real-time.

To address this demand, this paper proposes a novel event detection approach, *TwitterSports*, based on two complementary ideas namely event lexicon and adaptive sliding windows. The event lexicon is a dictionary of terms representing each event and populated with cricket terminologies gathered from the popular website for sports. The event detection and recognition algorithm recognizes events by examining the post rate of tweets when a game is ongoing. It helps to reduce the computational load significantly because detection can be achieved without analyzing the content of tweets, utilizing the event lexicon. The major contributions of this paper are summarized as follows:

1. Unlike previous approaches which used offline datasets and training data for event detection, we present a novel approach which detects events at real-time based on event lexicon and adaptive sliding windows. Our event detection algorithm recognizes all key events on the fly and does not require training data. This is particularly important as some applications such as *terrorism* and *presidential elections*, do not have any training data. Also, it does not require any natural language preprocessing steps as the event lexicon can handle all possible name variations for a possible key event and greatly reduces the computation overhead for the real-time detector.
2. For sports event detection and reporting, many studies have been made in NFL soccer sports domain. Similar to soccer sports, cricket has been one of the popular sports and attract a lot of viewers during the game. Since all viewers post tweets about the happenings of a game, a widely agreed event lexicon for Cricket sports will help researchers for detecting events. Therefore, this

paper proposes an event lexicon that has not been reported before in the literature. The event lexicon represents 37 key events for Cricket sports.

Our work is unique in a way that we study Twitter as the real-time output of the human sensors to infer the physical world. We investigate how Twitter users, as human sensors, report key events of sports games such as cricket at real-time. To the best of our knowledge, ours is a first step towards proposing an event lexicon for cricket terminologies and our algorithm is the first of its kind that detects and recognizes cricket events at real-time utilizing live tweets, when a game is ongoing.

The rest of the paper is organized as follows. In section 2, we survey the related work on Twitter event detection approaches. In section 3, we introduce Twitter APIs and our data collection method. We describe the adaptive window based real-time event detection method in section 4 and examine the performance of the proposed approach in section 5. Finally, we conclude the paper and describe our future work in section 6.

2 Related Work

A wide number of research contributions can be found in the recent literature on Twitter event detection. Generally Twitter event detection approaches can be classified into different types such as environment, social and sports events detection based on the domain in which events are detected. Based on the class of solutions, they can be categorized into term-interestingness based, incremental clustering based, topic modeling based and frequency based approaches [12]. Among them frequency based event detection is considered as a simple yet effective class of approach where no training data is needed for the detection process. For more detailed study on Twitter event detection, the readers are recommended to refer to some recent surveys [2, 12]. However in this section, recent research belong to different domains and solution classes will be discussed.

Some of the earlier work involved detection of social and physical events from Twitter. Earthquakes detection using Twitter was explored by Sakaki et al [13] and Qu et al [14]. Environment related events such as grassfire and floods in microblogs were studied by Vieweg et al [15]. In TwitterStand [16], news topics were discovered from the Twitter data where news stories among them were found using clustering the related tweets. A major drawback of the aforementioned approaches is that they could detect the event only several minutes after the actual event happened.

In a similar work by Hannon et al [17], tweet post rate is exploited to generate World Cup game highlights in a form of video. However, the approach generates the highlights in offline mode where the specific game events were not recognized. Some previous work in the sports domain have not applied event detection at real-time. Chakrabarti and Punera [18] trained Hidden Markov Models on training data of tweets of previous events to describe the events with the assumption that a game event is recognized already.

Detection and recognition of events in the sports domains has been vastly studied by the Computer Vision community earlier. Visual features has been analyzed to summarize the important events in the videos of soccer games [19]. Due to a low correlation between the visual features and the sports events, detection of events using visual features is not a highly reliable strategy. Some work have exploited the textual information associated with the videos. Rui et al [20] exploited speech detection to detect important events in baseball game videos. In another work by Zhang and Chang [21], closed captions were utilized

to detect and summarize key events in the baseball videos. Petridis et al [22] and Xu et al [23] utilized textual features such as MPEG-7 and webcast text to detect events in sports. The practicality of utilizing the aforementioned textual features highly depend upon their availability. It can be seen that these textual features do not always come in handy to support detection of events at real-time. However, Twitter is one such source which is both instant and easily available for processing as long as the game events are witnessed by huge audience.

System developed by Mathioudakis and Koudas [24] detects events based on high-rate and unusual appearances of keywords. Another event identification system named EDCoW by Weng and Lee [25] identifies events by exploiting wavelet analysis on frequencies of words by measuring new features of words. Further, words with low signal auto-correlations are filtered out, where modularity-based graph partitioning is applied for clustering the remaining words. To improve the scalability, each new document is compared against the previous document using locality-sensitive hashing [26, 27]. Twitter stream is first clustered where a classifier is trained on Twitter data, annotated using temporal, social, topical and Twitter-specific distinguishable features, in an event detection system proposed by Becker et al. [28].

Emerging topics in Twitter are detected [29] by locating strongly connected components of a directed graph containing emerging topic terms which are identified by comparing frequency of current terms with previous terms for a given period of time. An event visualization and summarization system, TwitInfo, developed by Marcus et al. [30] detects events by analyzing temporal peaks. The event detection results are presented to the users using a timeline-based visualization display where the temporal peaks are highlighted in the timeline. In an event detection system by Valkanas and Gunopoulos [31], users are clustered based on their geographical locations and the emotional states of the group of users are monitored where events are detected if there is a sudden change in a group's emotional state.

Controversial events involving celebrities are identified by analyzing the public discussions in the Twitter in response to those events by Popescu and Pennacchiotti [32]. Factor graph model is used to analyze individual Twitter messages and to cluster them to detect concert events [33]. In his work, clusters are automatically formed according to the type of events and a canonical value is generated for each property of event. Geo-social event detection system developed by Lee and Sumiya [34] identifies local festivals by modeling and monitoring different behaviors of the crowd in Twitter. Their approach detects events by analyzing the geographical regularity found from the usual behavior patterns by using geo-tags.

Sakaki et al [35] exploited tweets to detect specific types of events such as earthquakes and typhoons. They formulated event detection as a classification problem and trained an SVM on a manually labeled Twitter data set that comprises of a sample of positive and negative tweets. By combining simple rules and strategies for query building, Becker et al [36] successfully augmented information about planned events in Twitter data where events are identified by applying precise query strategies that are derived from description of events. High-quality, useful and relevant Tweets related to an event are extracted using centrality-based approach by Becker et al. [37].

Generative language modeling approach [38] based on microblog's quality indicators and query expansion is used to retrieve microblog messages. Weerkamp and de Rijke [39] proposed quality indicators such as emoticons, length of tweet post, angry expressions, capitalization of words, and hyperlinks along with other characteristics such as recency of message, number of shares of messages and the number of

followers of the user. N-grams based event modelling approach called ETree [40] groups large volume of tweets into relevant information blocks using content analysis approaches.

In summary, some research work have investigated the network properties of Twitter such as *geo-locations*, some user properties such as *influence* and *emotion*, while others used Twitter users as human sensors for detecting physical and social events such as *earth quakes*, *festivals* and *presidential elections*. However, all these research work performed their processing offline. Notably, few research work have been carried out in the domain of sports, but these studies focused on NFL soccer games. No other previous approaches have considered a Cricket domain and solved the problem of real-time event detection for Cricket sports.

In this paper, we explore how good human sensors are for the real-time detection of events in *Cricket* sports. We demonstrate the feasibility of using Twitter for real-time event detection for Cricket sports which has frequent and rapid key events[1] (aka, *key moments*) like *boundary* and *sixer*. Most importantly, our Twitter-based approach can be readily extended to recognize social and physical events beyond Cricket sports as long as these events are witnessed by a large number of Twitter users. Therefore, the insights gained from this study will help other novel applications, such as *reporting traffic jam* and *festivals*, to use human sensors for the event detection at real-time.

3 Twitter API for *TwitterSports*

Twitter has become an ideal platform for people to publish spontaneously, as it limits its length to just 140 characters. As a result, it has the shortest delay in delivering user comments to citizens, compared to other social media platforms such as *blogs*. During international Cricket games, Twitter receives a huge volume of tweets from cricket fans and audience of the live game, tweeting about game moments that they find exciting or notable. *TwitterSports* leverages this activity, associating particular streams of tweets with game moments or sub-events (e.g. Boundary, Sixer, Catch) to perform robust event detection and event recognition in real-time.

Twitter supports three types of APIs namely *REST API, Search API* and *Streaming API*. The Representational State Transfer (REST) API allows application developers to access tweets stored in the main database that contains all tweets. The *Search API* will search only those tweets that were posted only in the past 7 days and return 100 tweets for a given query. The *Streaming API* returns tweets in real-time based on the filter predicates such as *follow* a user, *track* a keyword and a *location*.

The *Streaming API* is the most suitable type for our event detection task from live tweet streams. The advantages are as follows. It returns up to date tweets; there are no rate limits; we can filter tweets based on keywords. For example, we have used the keyword, *RCBvRPS* to stream all tweets at real-time. The keyword *RCBvRPS* denotes an IPL T20 game between the teams *Royal Challengers Bangalore* and *Rising Pune Supergiant.* Similarly, we have used another keyword, *DDvKXIP* to crawl all tweets of the game between the teams *Delhi Daredevil* (DD) and *Kings XI Punjab* (KXIP) at real-time. Our *TwitterSports* runs

[1] We use the terms *key moment* and *key event* interchangeably

continuously collecting tweets without any break during the entire game time, detects events from tweets at real-time and also archives all gathered tweets in JSON format for later offline analysis.

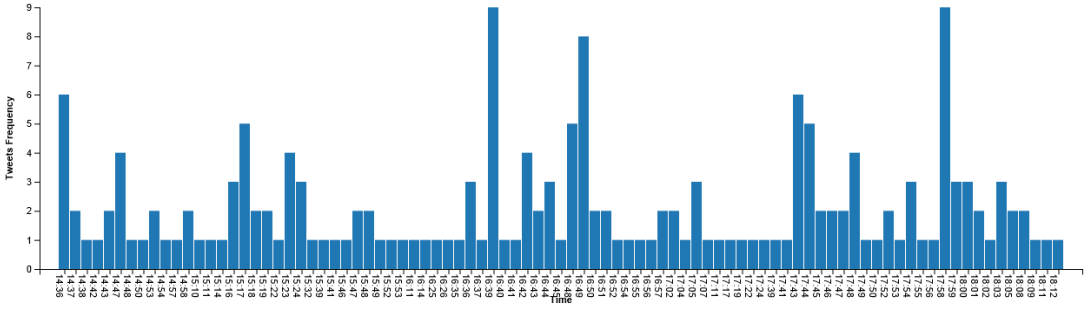


Figure 5. Tweet frequency of *Boundary* in RCBvRPS

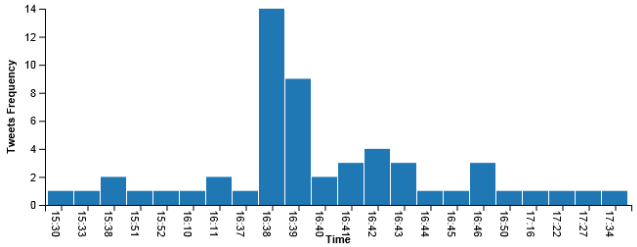


Figure 6. Tweet frequency of *Wide* in RCBvRPS

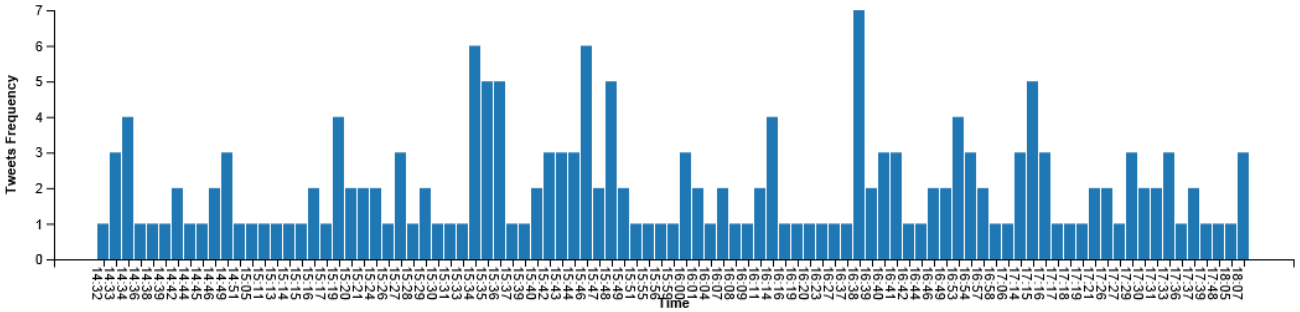


Figure 7. Tweet frequency of *Boundary* in DDvKXIP

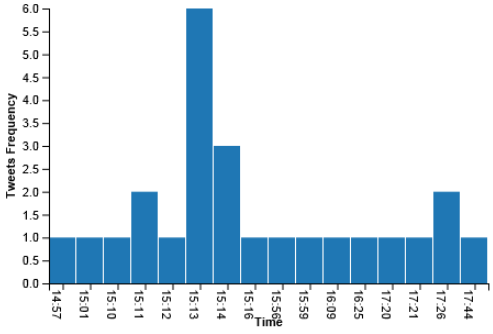


Figure 8. Tweet frequency of *Wide* in DDvKXIP

As a sports enthusiast, one can easily understand the interesting aspects of a game, by analyzing the collected tweets offline as well. We can also plot the timing of key events (aka, moments) such as *boundary, catch, wide* and so on by plotting the frequency of a term such as *catch* against time. Figure 5

to 8 depict the frequency of tweets about the events, *Boundary* and *Wide*, during the entire match time. The peaks in the graph indicate that an event might have happened in a crucial time of the game. Note that for an event, not all users will tweet at the same time, some may post their tweets little late, thereby some of the events are just noise as well.

4 Proposed Approach

In this section, we first describe our proposed approach for event detection and then present techniques to improve its performance in accuracy and responsiveness. We will describe our solution in the context of 2017 IPL T20 Cricket games and detect key events that happened during the Cricket games. The architecture of the proposed event detection framework is depicted in figure 9. Our two stage event detection and recognition solution for sports event, such as IPL T20 and ICC Champions is based on two complementing concepts namely, sliding windows and event lexicons.

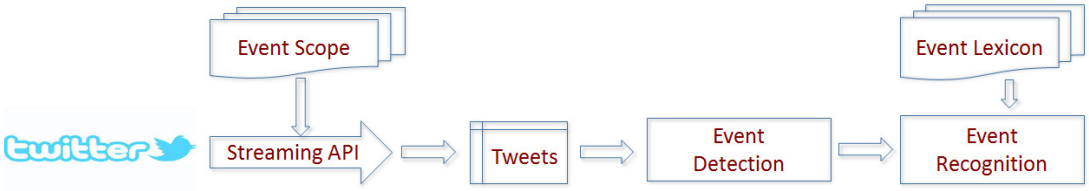


Figure 9. System architecture of *TwitterSports*

Our proposed event detection approach has the following critical steps.

1. Live Tweets collection: Tweets with the event scope are collected from the internet using Streaming Twitter API.
2. Removing noise from live tweets: Noisy tweets are removed from the stream of live tweets for further processing.
3. Adaptive sliding window based event detection: Sliding windows concept is applied to calculate the volume of tweets in a window. If tweet rate of the window is above a threshold, the window is marked as an event.
4. Lexicon-based event recognition: If the window contains enough tweets, it calculates the aggregated votes for the events in the tweets that occur in the middle of a window. Finally, using a lexicon defined for the events, called as *event lexicon*. An event with the maximum vote will be declared the event.

Figure 10 shows the algorithm for the proposed approach.

4.1 Live tweet collection

The rationale of event detection is that an interesting event may immediately trigger its audience and participants to talk about the happenings through social media platforms such as Twitter and Facebook in real-time. The spectators would talk about the event that may have either happened in physical world or reported in mass media such as newspapers, blogs and others. Note that *real-time* means that events need to be discovered as early as possible after they start unraveling in the online social networking service stream. Such information about emerging events can be immensely valuable if it is discovered in near real-time and made available timely to those people who are interested.

Input: Tweets of past 60 seconds
Output: Event name and its tweets
1: create event *lexicon* for pre-determined event types
2: **repeat**
3: *tweets* ← filtered live tweets of past 60 seconds
4: **for** each *tweet* in *tweets* **do**
5: *tweet_frequency* ← number of tweets in each *second*
6: *tweets_per_time* ← tweets in each *second*
7: **end for**
8: *window* = [6,10,20,30,60]
9: **if** *window* does not contain enough tweets **then**
10: select next window
11: **end if**
12: **if** *post rate of window* > pre-defined *threshold* **then**
13: get all tweets that appear in the middle of the window from *tweets_per_time*
14: select an event that has maximum votes using *lexicon*
15: display event name and its tweets using *lexicon*
16: **end if**
17: **until** connection closed

Figure 10. Proposed algorithm for key events detection

Generally, many physical world events, such as product announcements, celebrity deaths and natural disasters like earth quakes, attract a lot of viewers to witness the event. Therefore a sizeable increase in tweets about the physical world event on Tweet will occur, even if a small fraction of viewers talk about the physical event on Twitter. Further, it is a normal human tendency to share the current updates about the physical world event to others. So, people can thus be regarded as sensors who can be leveraged to get updates in real-time. With the help of Streaming API from Twitter, we will be able to collect live tweets continuously and to analyze them so as to detect all events as quickly as possible. The event detection method operates on the stream of live tweets based on the scope of events such as hashtags in Twitter.

4.2 Removing noise from live tweets

Noise elimination is an important preprocessing step for detection and recognition of key events from live tweets which are streamed at real-time. We first remove all tweets which contain up to 3 correct English words. Even though, Twitter specifies the language for those tweets is English, an underlying language used by the viewers is not always English. For example, many viewers of the game transliterate Hindi phrases in English. This step also takes care of short tweets that are unlikely giving us any additional information. Spam tweets are removed by using a dictionary of common words. Similarly, stop words are also removed from the raw tweets. Furthermore, we remove all tweets that contain URLs and pronouns. Obviously, this preprocessing step ensures that signals from tweets dominate noise, otherwise the performance will be very poor for event detection.

4.3 Adaptive sliding window based event detection

Our real-time streamer based on Twitter's Streaming API continuously streams tweets every second which are available in a pre-defined queue. In general, the live tweets that are gathered for the past 60 seconds fixed time window will be considered for a possible event. A fixed window approach usually computes the tweets volume of the first and second half of the window and subsequently calculates the post rate of the window as a ratio between the tweets volume of the second half of the window and the first half of the

window. We detect whether an event just happened by examining the volume of tweets. If the tweets volume is greater than the predefined threshold, the system concludes that some event might have occurred in a particular window time. This works based on a simple rationale that the percentage of change is the post rate obviously indicates the trend of an event related discussion.

Lexicon based event detection utilizing a fixed window of size 60 seconds would suffer from delay issues. Because, it is possible that an event would have occurred during the beginning of the window. Since our event detection will be a real-time detection system, longer delay will dampen the performance of the proposed system. In order to address the real-time challenge and to minimize the delay, we adopt an adaptive window approach. Here, the size of the window will have a significant impact on the tradeoff between the delay and accuracy of event detection. If the size of the window is short, the delay in event detection will also be small. However, the performance of the event detection may be poor, as there may not be several tweets posted during the window and thus post rate would be low.

In order to achieve a better trade-off, the window size should be selected adaptively based on two scenarios. First, viewers of the game have not posted tweets continuously for every second in the current window. That is, tweets were not available in each second of the window. Obviously, the size of the window should be increased. Second, the post rate of the current window is less than the predetermined threshold. Hence, the next window size should be selected automatically. We determine the value for the threshold by analyzing the tweets of the games using our offline dataset of IPL T20 2017 season. The performance of both fixed window and adaptive window approaches is discussed in section 5.3.2 where influence of different fixed window sizes are studied in section 5.3.3.

Our solution is based on an adaptive selection of a sliding window, as depicted in figure 11. The size of the sliding window can be a variable size of 6, 10, 20, 30, 60 seconds. The tweets in a window are sliced into two sub windows which contain tweets in the first half and second half of the window. For example, a window of 10 seconds includes two sub windows of 5 seconds each containing tweets. Our event detection system starts with the smallest window, 6 seconds. First it checks whether there are tweets spread in every second of the selected window. Otherwise, the window size will be incremented to have a size of 10 seconds. The basic assumption is that every key event will result into a continuous tweeting activity by the viewers throughout the window. It also avoids the computation of post rate of the window.

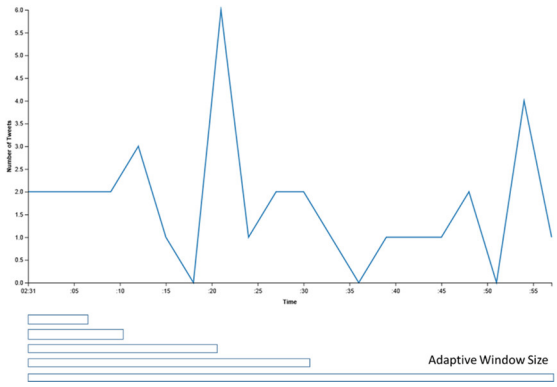


Figure 11. Tweets of one min at 14.31 hours from game DDvKXIP. The adaptive window moves from 6 seconds to 60 seconds.

Second, our event detection system checks whether the post rate of the window exceeds the threshold. If the post rate exceeds the threshold, the system proceeds to recognize the event, such as *sixer*, *catch* etc, otherwise the window size will increment. The post rate ratio is the volume of tweets in the second half window to the volume of tweets in the first half. The threshold value is set to 1.0 based on the analysis of tweets of games in our data set. The value of 1.0 indicates that the post rate in the second half window has to be at least 1.0 times of the post rate in the first half to proceed to recognize an event. The threshold denoting the average number of tweets helps to filter out the small spikes in the tweets. Because, the key events will result into huge spikes in the tweets frequency, which will be above the threshold. Once the events are detected in a time window along the tweet stream, the event represented inside the window is recognized using a lexicon based approach.

4.4 Lexicon-based event recognition

The event recognizer identifies the specific event in the detected event window based on the idea of maximum voting. For simplicity and consistency, we choose all tweets that are available in the middle of the window. We look for the occurrences of event related terms such as event names in each tweet and increase the vote for the occurrence of event related terms. The event recognizer selects the event whose has the maximum votes and declare the event as the winner.

We create a lexicon for 37 events in IPL T20 Cricket games. For example, *boundary*, *sixer*, *catch*, *bowled out*, *run-out*, *lbw* and *leg bye* are some of the key events in Cricket sports. A sample lexicon for the key event *boundary* would contain variants of the event name *boundary* such as *four, fours* and *4* in addition the term *boundary*. The way game viewers tweet the details of events are unique, the vocabulary of lexicons should be more descriptive in order to recognize the designated event. Further, the size of each tweet is limited to 140 characters and the event name is a highly preferred way to describe the event. Therefore, the domain specific lexicon for the key events should precisely describe the event names for the corresponding events. The vocabulary of our lexicon is populated with event terminologies collected from a website ESPNCricInfo[2]. Figure 12 shows a section of our event lexicon for Cricket sports.

```
BOUNDARY = ['boundary', 'four', 'fours', '4']
SIXER = ['sixer', 'six', '6']
ONE = ['one', '1']
CATCH = ['catch', 'c']
BOWLED_OUT = ['bowled out', 'bowled by', 'clean bowled', 'bowled off']
```

Figure 12. Lexicon for few events of *Cricket* sports

Our lexicon based event detection approach is simple in implementation, but at the same time enjoys a better performance for event detection from live tweets which are streamed in real-time. Unlike statistical learning approaches which require training data to build models for further prediction tasks, our lexicon based approach does not require them for event detection and recognition. Furthermore, there are real-time applications to whom training data are not available a priori. For instance, celebrity deaths and terrorist attacks have no training data available in advance. Nevertheless, the keywords related to such

[2] http://www.espncricinfo.com/ci/content/story/239756.html

events are predictable. Therefore, it is very practical to use lexicon based approaches for event recognition.

4.5 Preventing duplicate event alerts

Duplicate events have been an important issue for our event detection method, in which our algorithm reports duplicate events many times even after the specified event had occurred. The viewers of the game continue to discuss even after the event was just now over because this was a key event. For example, in a cricket match, the batsmen keep accumulating runs aggressively and the opponent is in need of a wicket desperately. Now, one of the batsmen gets out due to a catch and thus it is a key event for the audience, triggering a huge discussion. Therefore, our detection algorithm may repetitively recognize the event during the discussion. The primary reason for duplicate event alerts is associated with a shorter window size. Since small number of tweets are analyzed in the short window, the detection algorithm is unable to distinguish whether the short spike in tweets volume denotes a beginning of a new event or the continuation of the current event. Many users, in addition to viewers of a game, also forward the received tweet to other users, as retweets. Therefore, retweets can be considered as noise and retweeting is yet another important reason for the duplicate event alerts.

The solution to duplicate event alerts problem can be approached in two ways. First, we can ignore all retweets coming from the live streaming of tweets, thereby we can minimize the intense discussion over the current key event. Influence of the retweets in detecting events is experimentally studied in section 5.3.5. Second, we can assume that no event of the same kind can happen within 60 seconds. Based on this idea, if the detection algorithm reports the same event again within 60 seconds, we can ignore this event as a duplicate event alert. For example, in Cricket games, an over containing six balls should be delivered by a bowler to a batsman within 5 minutes. So, the process of bowling and batting should be finished within the time frame of 60 seconds. Therefore, an event of the same type cannot happen once again within 60 seconds.

As discussed above, our approach to event detection on live tweets first detects whether an event occurs and then recognizes the specific event type such as *boundary*, *sixer*, *catch* and others. Our method is computationally efficient as it detects key events without analyzing the content of tweets. It detects 37 types of pre-defined key events for Cricket games from live tweets posted by game viewers in real-time.

5 Experimental Results

This section presents the extensive evaluation of the proposed *TwitterSports* approach which we have implemented in Python. The following sections will present the dataset, evaluation criteria and parameter setup and evaluation results respectively.

5.1 Dataset

We collected tweets during game time with the Streaming API using IPL's official hashtags of every game, during 2017 IPL T20 season that was held during April and May 2017. We have successfully crawled tweets at real-time for 44 games. The total file size of the collected tweets is over 6GB. Out of 44 games, we have selected 3 games for detecting key events. Table 1 and 2 show the description of the 2 games. From the tweet statistics in terms of number of tweets and tweet rate, it can be assessed that the game on 16 April 2017, RCB vs RPS is an interesting and most anticipated match.

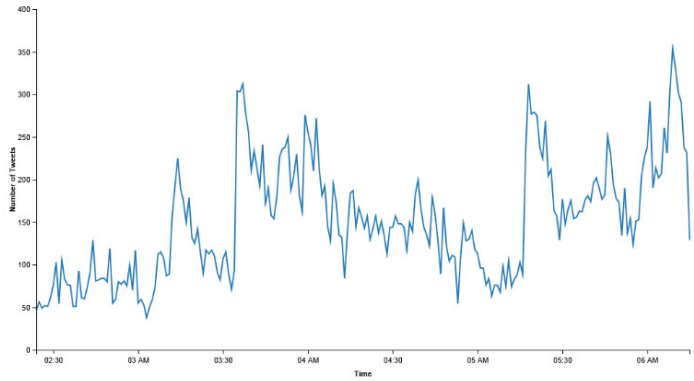
Table 1. Game statistics of RCBvRPS

RCBvRPS game	Total	Total min	Mean (re)tweets per min	Min (re)tweets per min	Max (re)tweets per min	Standard deviation
Tweets	34967	232	150.72	38	354	67.4779909455
Retweets	16162	232	69.664	13	176	34.4786150528

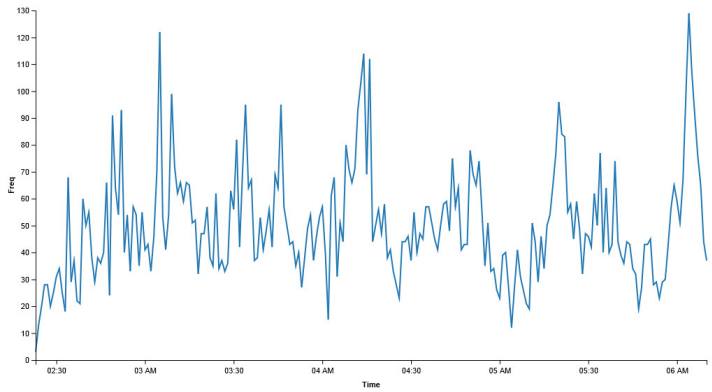
Table 2. Game statistics of DDvKXIP

DDvKXIP game	Total	Total min	Mean (re)tweets per min	Min (re)tweets per min	Max (re)tweets per min	Standard deviation
Tweets	11407	228	50.030	3	129	20.7684377065
Retweets	3473	228	15.234	1	80	9.48236586451

Figure 13 (a) and (b) depict the tweet post rate for games RCBvRPS and DDvKXIP correspondingly. Figures show that the volume of tweets posted during the end the game is high in both games. Both games contain several exciting moments throughout the entire game.



(a)



(b)

Figure 13. Post rate of tweets of games (a) RCBvRPS and (a) DDvKXIP

The ground truth of all events are gathered from the online IPL live commentary site (*http://www.iplt20.com/*). The validity of the event timings have been validated with other IPL commentary sites. Table 3 shows the description of ground truth events in these games.

Table 3. Summary of events in ground truth

Game	No. of ground truth events	No. of Boundaries	No. of Catches	No. of Sixers	Other events
RCBvRPS	81	24	6	9	42
DDvKXIP	89	34	10	8	37

5.2 Evaluation Criteria and Parameter Setup

The performance of the approaches are evaluated using ROC curves and its corresponding Area Under Curve (AUC). Detection result is compared against the ground truth events of a game. A detection is considered a *hit* if the detection event is reported within a particular time (such as 1, 5, 10, 15 minutes) from when the actual event happened. It will be considered a *miss* if the event is not detected within the window time. From this, True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) are computed. Then, True Positive Rate and False Positive Rate are computed as:

$$TPR = TP/(TP+FN) \quad (1)$$

$$FPR = FP/(FP+TN) \quad (2)$$

For a particular approach, different sets of results are obtained by adjusting the tweet rate threshold (0.2, 0.4, 0.5 and 1.0). Different sets of TPRs and FPRs are computed where the True Positive Rates are plotted against False Positive Rates which results in ROC. For completeness of the curve, the curve starts and ends in the coordinates (0,0) and (1,1) respectively. In addition, Area Under Curve of the ROC is calculated for each approach. The AUC of each ROC line depicts the degree of performance in terms of detection accuracy where AUC is high when true positive rate is high and false positive rate is low and AUC is low when true positive rate is low and false positive rate is high.

5.3 Results

We evaluate *TwitterSports* using IPL T20 games and show the accuracy of the event detection for major events such as *boundary*, *catch*, *sixer* and *boundary+catch*. The evaluation results prove that the adaptive window approach can detect key events faster than fixed window approach.

5.3.1 Performance on detecting events

Figure 14 shows the performance of the adaptive window approach in detecting different events such as *boundary*, *catch*, *sixer* and major events (*boundary+catch*) in RCBvRPS game. The evaluation is conducted for different evaluation window sizes such as 1, 5, 10 and 15 minutes.

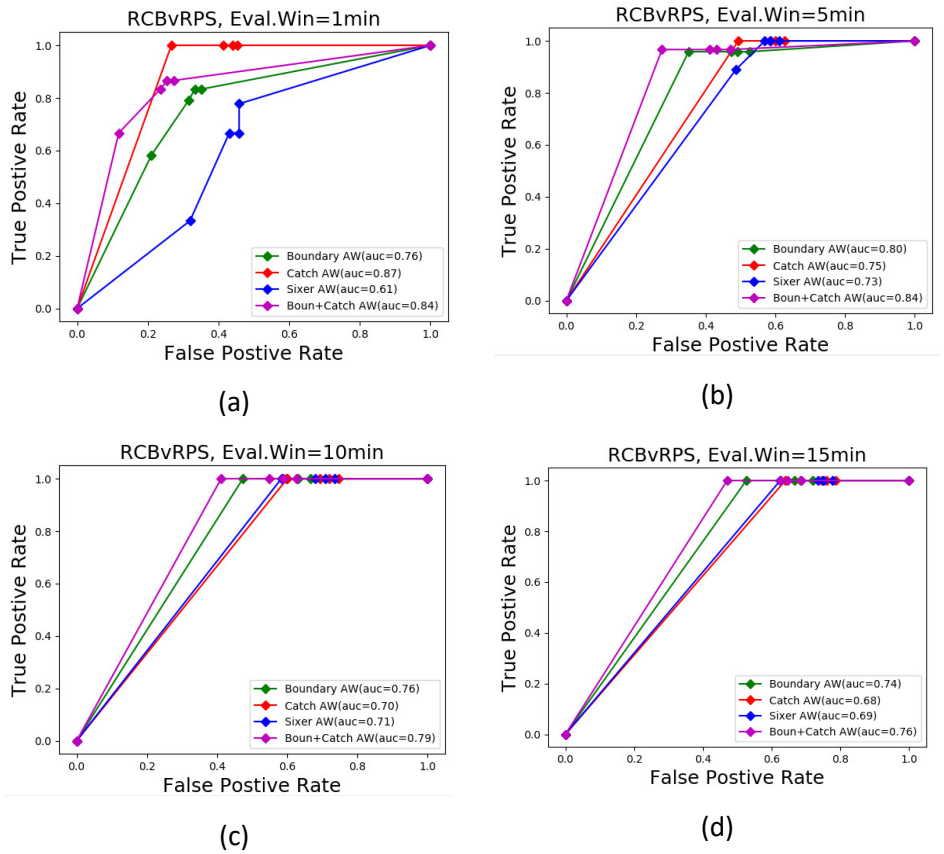


Figure 14. Detection performance for individual events

Results show that the adaptive window approach gives decent performance in detecting the game events. Boundary events are detected quicker than the *sixer* and *catch* events even within an evaluation window of 1 minute. Major events (*boundary+catch* combined) of a Cricket game are also detected well with the adaptive window based approach. Almost all game events are detected well with decent accuracy within an evaluation window of size 5 minutes where the performance for evaluation windows 10 and 15 minutes are highly similar. It shows that most of the events are detected and reported well even within a time of 5 minutes from the actual game event happened. Here performance in detecting each event can be directly assessed by the AUC of the ROC line.

5.3.2 Performance of fixed vs. adaptive windows

Figure 15 shows the performance of adaptive window and fixed window approaches in detecting major events for different evaluation windows. The results are shown for performance in both games. The evaluation for comparing fixed and adaptive window approaches is conducted for different evaluation window sizes such as 1, 5, 10 and 15 minutes in detecting major events of the games DDvKXIP (Figure15 (a), (b), (c) and (d)) and RCBvRPS (Figure15 (e), (f), (g) and (h)). For fixed window approach, the performance is also shown for different fixed window sizes such as 6, 10, 20 and 30.

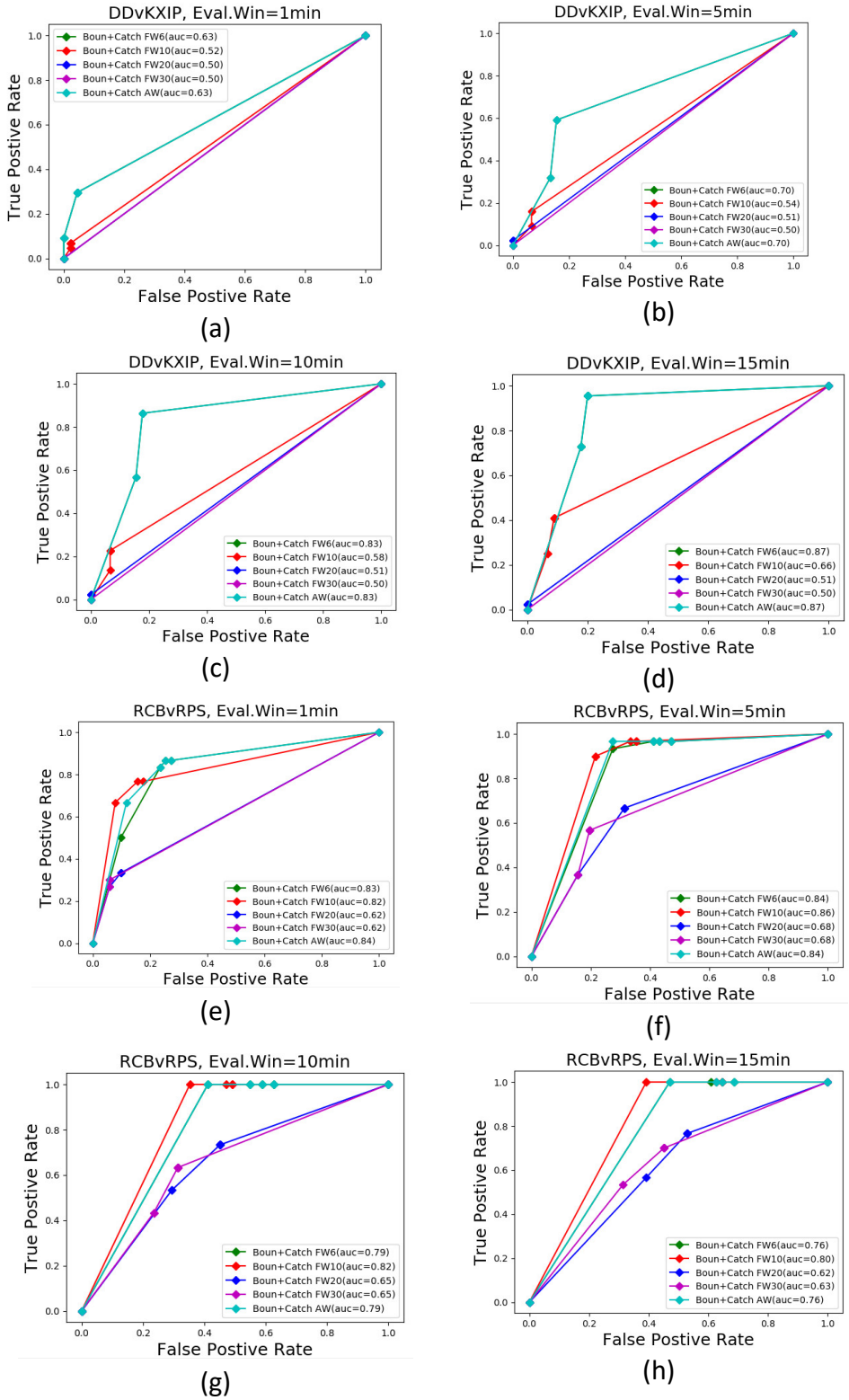


Figure 15. Detection performance of fixed and adaptive window approaches

In game DDvKXIP, both adaptive window approach and fixed window approach (window size 6) show similar performance under different evaluation windows. In RCBvRPS game, adaptive window approach outperforms fixed window approach in small evaluation window where fixed window approach (window

size 10) marginally outperforms adaptive window approach in other evaluation window sizes. From the results we can assess that there is no standard best performing window size for fixed window approach. Since the best performing window size for fixed window approach is changing for different games and cannot be known forehand, adaptive window approach is mostly preferred for any game.

5.3.3 Performance of fixed window for different windows sizes

Figure 16 shows the performance of fixed window approach for different window sizes such as 6, 10, 20 and 30 in detecting the *boundary*, *catch* and *sixer* events of a game. The evaluation is conducted for different evaluation window sizes for the game RCBvRPS.

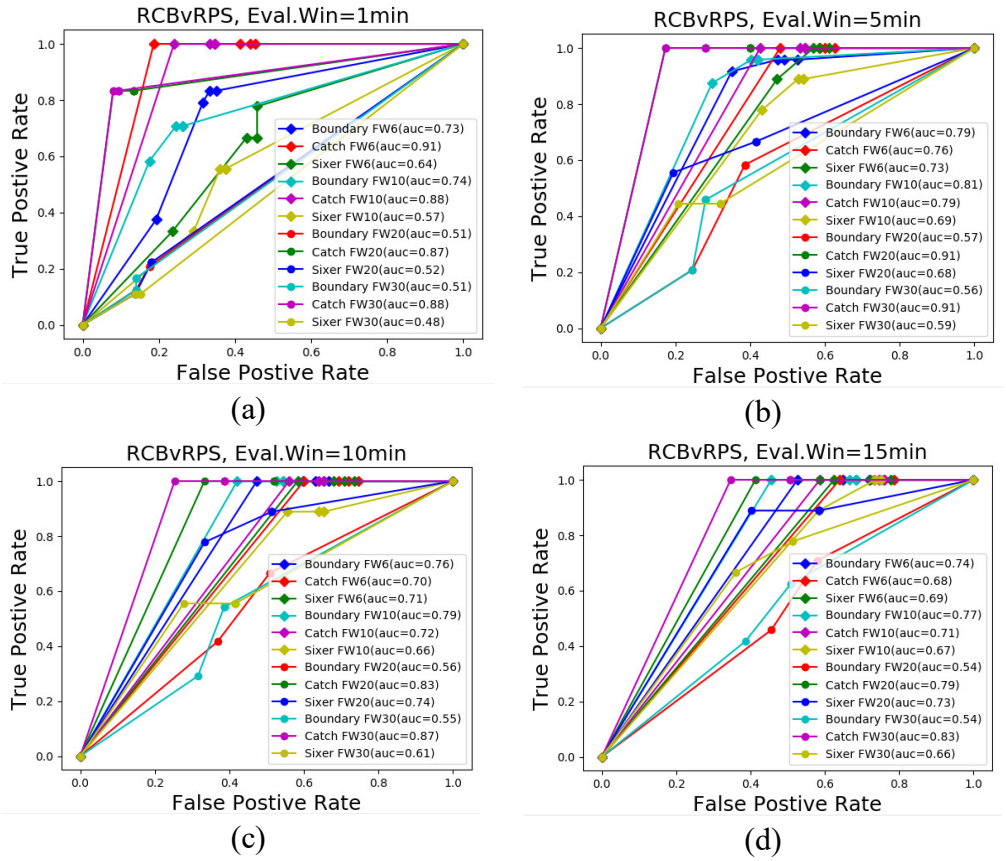


Figure 16. Detection performance of fixed widow approach with different window sizes

For a small evaluation window 1 minute, detection performance degrades as the size of the fixed window increases for almost all the events. However, for a bigger evaluation window 5 minute, detection accuracy improves for medium sized fixed windows such as 10 and 20 for detecting events such as *catch*. It can be concluded that for detecting events quickly in real-time, fixed windows with smaller sized windows can be preferred. However, for consistent performance, medium sized windows can be used in situations where bigger event detection time like 5 minutes is considered. As discussed in section 5.3.1, the performance is similar for evaluation window sizes 10 and 15 minutes.

5.3.4 Performance under different evaluation windows

Figure 17 depicts the detailed performance in detection of individual events under different evaluation window sizes such as 1, 5, 10 and 15 minutes. This evaluation is conducted for gauging the effect of

different evaluation window sizes by performing adaptive window approach in detecting events of the game RCBvRPS.

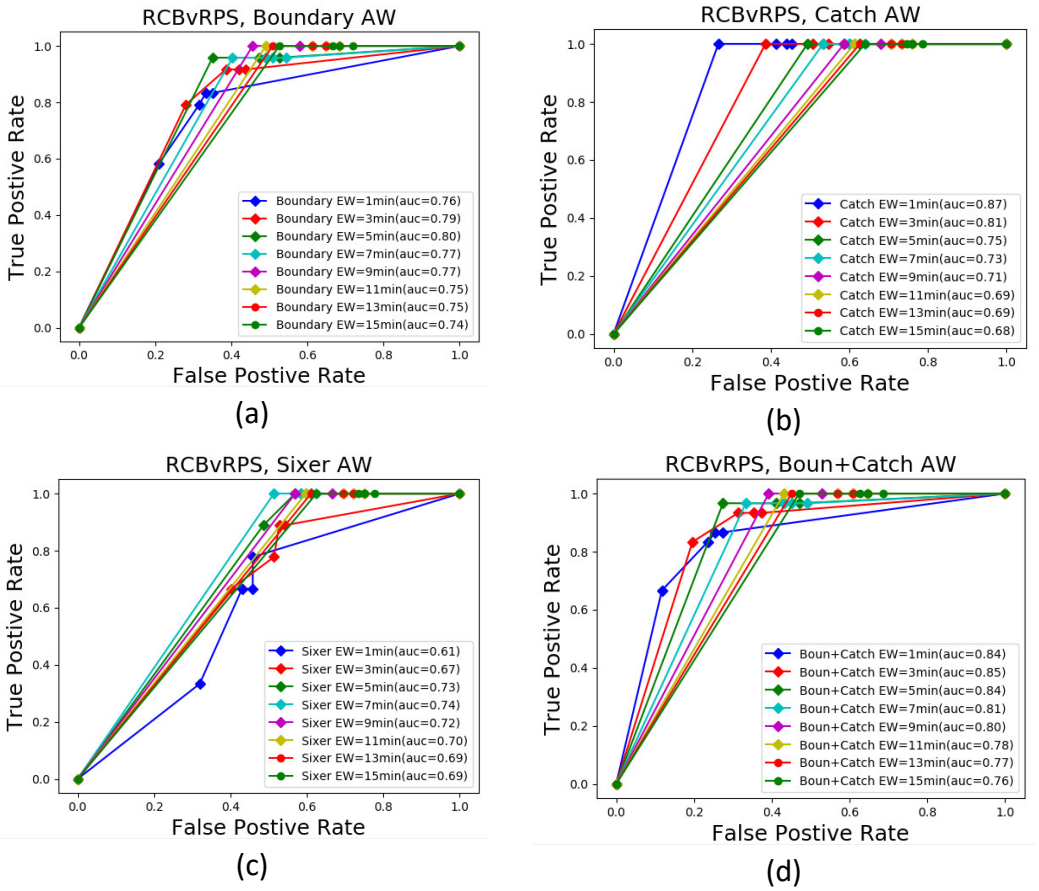


Figure 17. Detection performance for different evaluation windows

The results show that *catch* event is quickly detected within evaluation window size of 1 minute where the performance degrades as the evaluation window size increases. Events such as *boundary* and *sixer* are detected well within a time delay of 5 minutes where the accuracy reduces as the evaluation window size is increased further. Similarly major events (*boundary+catch* events) are detected well within smaller evaluation window sizes such as 1, 3 and 5 minutes where larger evaluation window sizes show lesser detection accuracy.

5.3.5 Performance of all tweets vs. no retweets

Since the processed tweets contain both tweets and their corresponding retweets, influence of retweets in detecting events is analyzed by evaluating the adaptive window approach with *all tweets* and with *no retweets*. The evaluation is conducted for adaptive window based approach in detecting *boundary*, *catch* and *sixer* events in the game RCBvRPS under different evaluation windows as depicted in figure 18.

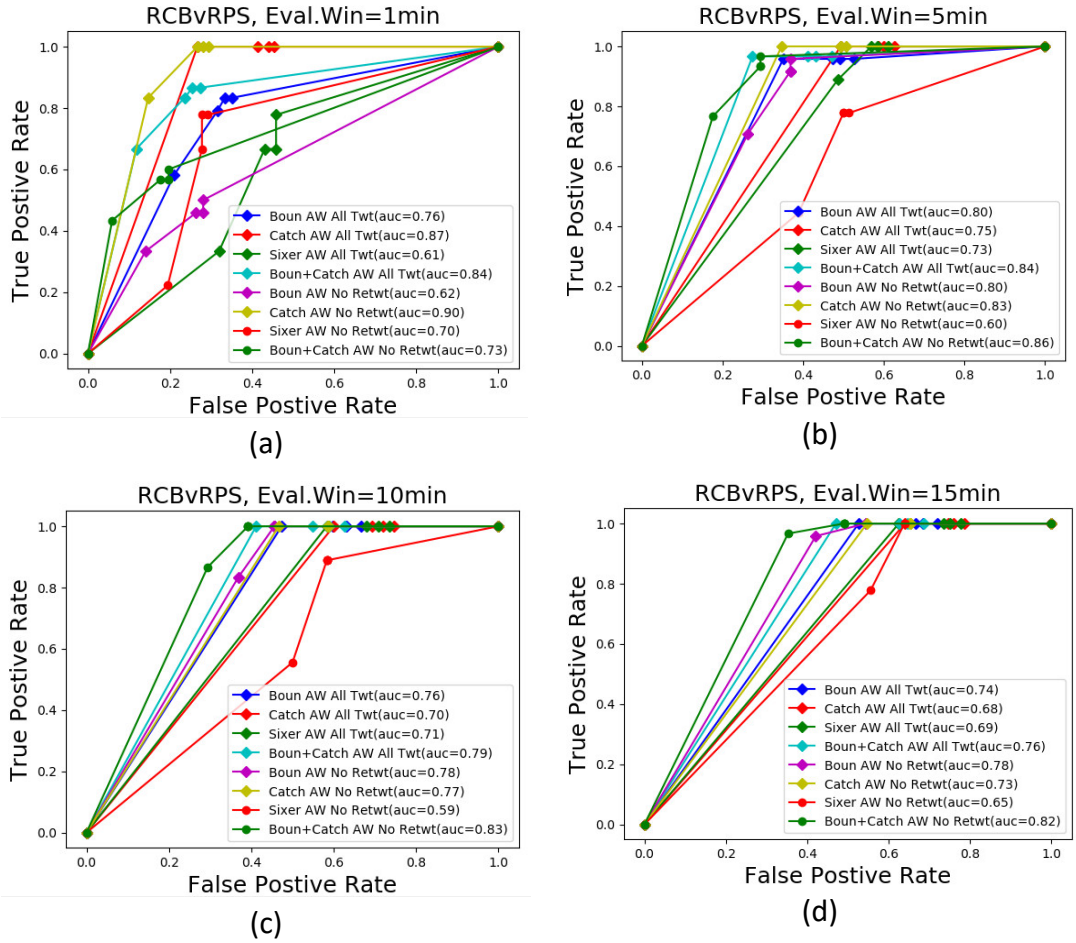


Figure 18. Detection performance for *all tweets* and *no retweets*

The results show that *catch* events are detected well within small evaluation windows (for ex. 1 and 3 minutes) when *no retweets* are considered for detection. The detection improves as the evaluation window size increases where tweets including the retweets help the event detection. However, *boundary* events are detected easily when *all tweets* are considered along with retweets. Similarly, as the evaluation window size increases the *sixer* events are detected well when all tweets are considered. Overall, it can be concluded that for quick detection of rare events such as catches, removal of retweets helps the detector. Also, inclusion of retweets highly influences detection accuracy if considerable time delay in detection is permissible i.e. bigger window sizes.

5.4 Limitations of *TwitterSports* and Twitter

Although our *TwitterSports* is a simple, yet powerful lexicon based event detection algorithm, performance will be directly impacted by the amount of tweets posted by the viewers. The algorithm needs enough tweets so that the post rate would be above the predefined threshold. This will enable our event detector to recognize a key event, otherwise the detection algorithm will perform badly. We can notice such an average performance of our *TwitterSports* in the game DDvKXIP for a key event *sixer*. The area under ROC value is just an average and below average value for the evaluation windows 1, 5, 10 and 15 minutes as shown in figure 19.

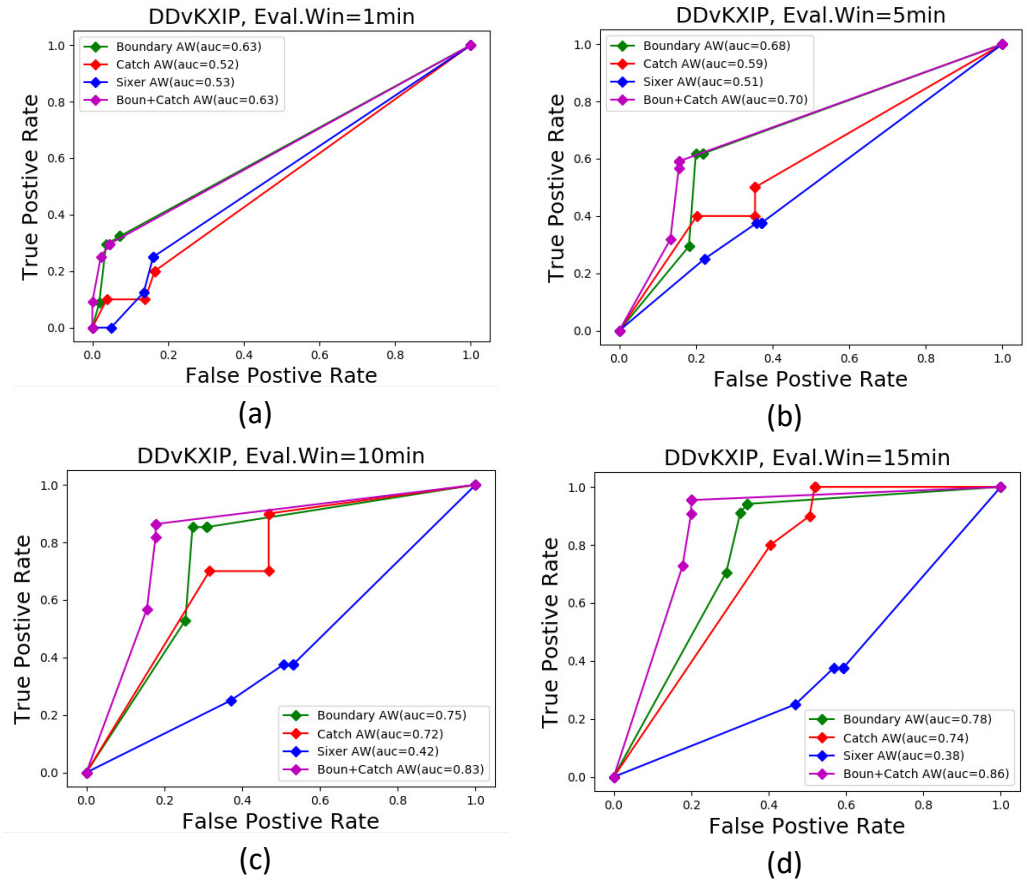


Figure 19. Detection performance of key events from DDvKXIP game

Performance of our Twitter event detection approach *TwitterSports*, also depends on many factors. One such factor is *Delay* or *Latency* in the flow of signals from the human sensors in the social media. Delay plays a vital role in determining the performance of a real-time event detection system. There are three types of delays encountered in the Twitter social media, namely human delay, Twitter delay, and processing delay [3]. Many applications require game events to be detected at real-time. So that, the information provided to society will be meaningful. Therefore, we need to analyze the delay of our *TwitterSports* approach in detecting key events.

Human delay is a period of time between a user observing an event such as *boundary* and typing and posting a tweet about this event. Human delay depends on how fast humans observe, react and publish tweet about the event to the social media, such as Twitter. In this case, delay is caused by the humans in various degrees. For example, delay in reporting the events depend on the user's interest in reporting the event, ability to type faster, type device (PC, laptop, or mobile) of used for tweeting, type of mobile (Nokia, Blackberry, iPhone, Samsung etc), speed of internet, location of the user (watching the match live in stadium or in TV in home).

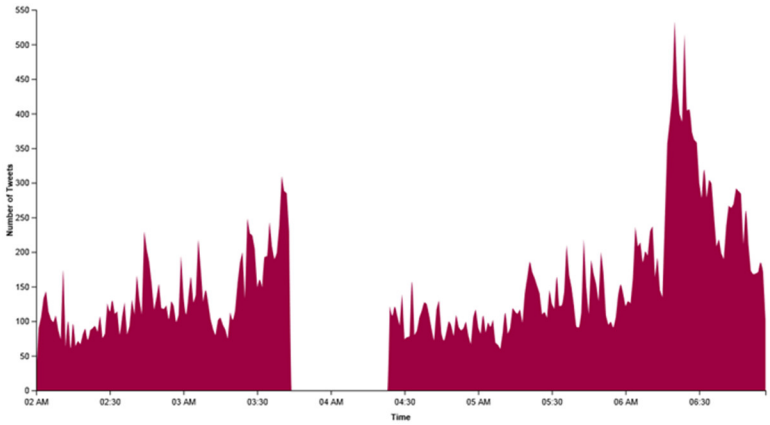


Figure 20. Delay in Twitter streaming of tweets for game RPSvMI held on 21 May 2017. There were no tweets streamed from 3.44 PM to 4.24 PM, which includes an innings break of 15mins.

Twitter also introduces a delay in providing tweets to the querying users. On the technical part of the data streaming, Twitter often faces heavy workload from the millions of users and its indexing mechanism leads to certain delay in delivering the relevant tweets to the crawler through its API. One way to find the delay is to compare the timestamp of the tweet and the timestamp when we acquire the tweet. There is no explicit mentioning about the degree of delay introduced by Twitter API. One such Twitter API issue can be noted in figure 20 where there was no streaming of Tweets for 40 minutes despite there were no internet or API connection problem for streaming. Due to the different Twitter indexing mechanisms, sometimes the choice of keywords for crawling Twitter also makes difference in retrieval speed of tweets. Therefore, the quality of search predicates defined by queries for crawling is another factor for delay.

In addition, minimal amount of delay is also introduced by our event detection engine. This is, due to the processing time involved in data collection and analysis of data. Due to the faster nature of the proposed approach, this delay is highly reduced during the processing. In case of extremely high tweet rate, this analysis delay can be reduced by using parallel processing.

6 Conclusion

In contrast to the existing event detection approaches for sports domain, *TwitterSports*, a novel real-time event detection approach is presented in this paper. *Twittersports* is based on two complementary ideas event lexicon and adaptive sliding windows. An event is declared detected when the post rate of the adaptive window exceeds the pre-defined threshold. The detected event is recognized utilizing the domain specific event lexicon for Cricket sports. The event lexicon can handle all possible name variations for a possible key event and greatly reduces the computation overhead by eliminating the need for a natural language preprocessing. Further, the predefined threshold efficiently reduces noise by ignoring small spikes in the volume of tweets.

Results of the extensive experiments have shown the efficacy of the proposed adaptive window based event detection. In the experiments, it was found that certain events such as boundary are detected easily and quickly than other events, which are found to be more appealing to the audience to react quickly in Twitter. Comparative evaluation between fixed and adaptive window based detection has revealed the advantages of using an adaptive window approach for robust and consistent performance. Influence of different fixed window sizes were also analyzed in the experimental evaluation. It was found that different fixed windows sizes perform differently in detecting each event in Cricket game. The time delay in

detecting the events were studied using evaluation with different evaluation window sizes. Influence of including retweets for detection of different events were analyzed. Challenges in real time scenario in terms of various delays such as human, streaming API and processing delays were also discussed.

There are many challenges left in the area of real time event detection. The proposed event detection algorithm detects only events that are documented in the event lexicon. It would be useful to detect interesting but unexpected events that are not included in the lexicon by performing NLP preprocessing steps.

REFERENCES

[1] Boyd, D. M and N. B. Ellison. *Social network sites: Definition, history, and scholarship*. Journal of Computer-Mediated Communication, 2007. 13(1): p. 210–230.

[2] Atefeh, F and Khreich, W. *A survey of techniques for event detection in twitter*. Computational Intelligence, 2015. 31(1): p. 132-164.

[3] Zhao, S., Zhong, L., Wickramasuriya, J., Vasudevan, V., LiKamWa, R and Rahmati, A. *Sportsense: Real-time detection of NFL game events from Twitter*. ArXiv preprint, 2012. arXiv:1205.3212.

[4] Zhao, D and M. B. Rosson. *How and why people Twitter: The role that micro-blogging plays in informal communication at work*. In Proc. ACM International Conference on Supporting Group Work, GROUP '09, ACM, New York, NY, 2009. p. 243–252.

[5] Hurlock, J and M. Wilson. *Searching Twitter: separating the tweet from the chaff*. In Proc. International AAAI Conference on Weblogs and Social Media, Barcelona, Spain, 2011.

[6] Jiang, L., M. Yu., M. Zhou., X. Liu and T. Zhao. *Target-dependent Twitter sentiment classification*. In Proc. 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies – vol. 1, HLT '11, ACL, Stroudsburg, PA, 2011. p. 151–160.

[7] Amer-yahia, S., S. Anjum, A. Ghenai, A. Siddique, S. Abbar, S. Madden, A. Marcus and M. El-haddad. *MAQSA: A system for social analytics on news.* In Proc. ACM SIGMOD International Conference on Management of Data, SIGMOD '12, ACM, New York, NY, 2012. p. 653–656.

[8] Tumasjan, A., T. O. Sprenger., P. G. Sandner and I. M. Welpe. *Predicting elections with Twitter: What 140 characters reveal about political sentiment.* In Proc. 4th International Conference on Weblogs and Social Media, ICWSM. The AAAI Press: Washington, DC, 2010.

[9] Wang, X., Gerber, M. S and D. E. Brown. *Automatic crime prediction using events extracted from Twitter posts*. In Proc. 5th International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction, SBP'12. Springer-Verlag: Berlin, Heidelberg, 2012. p. 231–238.

[10] Troncy, R., B. Malocha and A. T. S. Fialho. *Linking events with media*. In Proc. 6th International Conference on Semantic Systems, I-SEMANTICS '10, ACM, New York, NY, 2010. 42: p. 1–42:4.

[11] Zhao, S., Zhong, L., Wickramasuriya, J and Vasudevan, V. *Human as real-time sensors of social and physical*

events: A case study of twitter and sports games. ArXiv preprint, 2011. arXiv:1106.4300.

[12] Hasan, M, Orgun, M. A and Schwitter, R. *A survey on real-time event detection from the Twitter data stream.* Journal of Information Science, 2017. 0165551517698564.

[13] T. Sakaki, M. Okazaki and Y. Matsuo. *Earthquake shakes Twitter users: real-time event detection by social sensors*. In Proc. ACM WWW '10, 2010.

[14] Y. Qu, C. Huang, P. Zhang and J. Zhang. *Microblogging after a major disaster in China: a case study of the 2010 Yushu earthquake*. In Proc. ACM 2011 conference on Computer supported cooperative work, 2011.

[15] S. Vieweg, A. L. Hughes, K. Starbird and L. Palen. *Microblogging during two natural hazards events: what twitter may contribute to situational awareness*. In Proc. ACM CHI '10, 2010.

[16] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman and J. Sperling. *TwitterStand: news in tweets.* In Proc. ACM SIGSPATIAL, 2009.

[17] J. Hannon, K. McCarthy, J. Lynch and B. Smyth. *Personalized and automatic social summarization of events in video.* In Proc. ACM IUI, 2011.

[18] D. Chakrabarti and K. Punera. *Event Summarization using Tweets*. In Proc. AAAI ICWSM, 2011.

[19] Ekin, A. M. Tekalp and R. Mehrotra. *Automatic soccer video analysis and summarization*, Image Processing, IEEE Transactions on, 2003. 12: p. 796-807.

[20] Y. Rui, A. Gupta and A. Acero. *Automatically extracting highlights for TV Baseball programs*. In Proc. ACM Multimedia 2000.

[21] D. Zhang and S.-F. Chang. *Event detection in baseball video using superimposed caption recognition*. In Proc. ACM Multimedia, 2002.

[22] K. Petridis, S. Bloehdorn, C. Saathoff, N. Simou, S. Dasiopoulou, V. Tzouvaras, S. Handschuh, Y. Avrithis, Y. Kompatsiaris, and S. Staab. *Knowledge representation and semantic annotation of multimedia content*. *Vision*, Image and Signal Processing, IEE Proceedings, 2006. 153: p. 255-262.

[23] C. Xu, Y.-F. Zhang, G. Zhu, Y. Rui, H. Lu and Q. Huang. *Using Webcast Text for Semantic Event Detection in Broadcast Sports Video*, Multimedia, IEEE Transactions on, 2008. 10: p. 1342-1355

[24] Mathioudakis, M and Koudas, N. *TwitterMonitor: Trend Detection over the Twitter Stream*. In Proc. SIGMOD/ PODS, 2010. p. 1155–1158.

[25] Weng, J and Lee, B.-S. *Event Detection in Twitter.* In Proc. ICWSM, 2011. p. 401–408.

[26] Shane Fitzpatrick. *Improving new event detection in social streams*. 2014. Master Thesis.

[27] Petrovi´c, S., Osborne, M and Lavrenko, V. *Streaming First Story Detection with Application to Twitter*. In Proc. NAACL HLT, 2010. p. 181–189.

[28] Becker, H., Naaman, M and Gravano, L. *Beyond Trending Topics: Real-Wrold Event Identification on Twitter*. In Proc. ICWSM, 2011. 11: p. 438–441.

[29] Cataldi, M., Di Caro, L and Schifanella, C. *Emerging Topic Detection on Twitter Based on Temporal and Social Terms Evaluation*. In Proc. MDM/KDD, 2010. p. 4:1–10.

[30] Marcus, A., Bernstein, M. S., Badar, O., Karger, D. R., Madden, S and Miller, R. C. *TwitInfo: Aggregating and Visualizing Microblogs for Event Exploration.* In Proc. CHI, 2011. p. 227–236.

[31] Valkanas, G and Gunopulos, D. *How the Live Web Feels About Events*. In Proc. In Proc. 22nd ACM International Conference on Information and Knowledge Management CIKM, 2013. p. 639–648.

[32] Popescu, A. M and M. Pennacchiotti. *Detecting controversial events from Twitter*. In Proc. 19th ACM International Conference on Information and Knowledge Management, CIKM '10, ACM, New York, NY, 2010. p. 1873–1876.

[33] Benson, E., A. Haghighi and R. Barzilay. *Event discovery in social media feeds*. In Proc. 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, HLT '11, Association for Computational Linguistics, Stroudsburg, PA, 2011. p. 389–398.

[34] Lee, R and K. Sumiya. *Measuring geographical regularities of crowd behaviors for Twitter-based geo-social event detection.* In Proc. 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks, LBSN '10, ACM, New York, NY, 2010. p. 1–10.

[35] Sakaki, T., M. Okazaki and Y. Matsuo. *Earthquake shakes Twitter users: Real-time event detection by social sensors*. In Proc. 19th International Conference on World Wide Web, WWW '10, ACM, New York, NY, 2010. p. 851–860.

[36] Becker, H., F. Chen, D. Iter, M. Naaman and L. Gravano. *Automatic identification and presentation of Twitter content for planned events.* In Proc. International AAAI Conference on Weblogs and Social Media, Barcelona, Spain, 2011.

[37] Becker, H., M. Naaman and L. Gravano. *Selecting quality Twitter content for events*. In Proc. International AAAI Conference on Weblogs and Social Media, Barcelona, Spain, 2011b.

[38] Massoudi, K., M. Tsagkias, M. De Rijke and W. Weerkamp. *Incorporating query expansion and quality indicators in searching microblog posts*. In Proc. 33rd European Conference on Advances in Information Retrieval, ECIR'11. Springer-Verlag: Berlin, Heidelberg, 2011. p. 362–367.

[39] Weerkamp, W and M. De Rijke. *Credibility improves topical blog post retrieval.* In Proc. ACL, Columbus, OH, 2008. p. 923–931.

[40] Gu, H., X. Xie, Q. Lv, Y. Ruan and L. Shang. *ETree: Effective and efficient event modeling for real-time online social media*. In Proc. Web Intelligence and Intelligent Agent Technology, WI-IAT 2011, IEEE/WIC/ACM International Conference, 2011. 1: p. 300–307.