Building timelines of soccer matches from Twitter

Amosse Edouard
Université Côte d'Azur
Inria, CNRS, I3S, France
amosse.edouard@unice.fr

Elena Cabrio
Université Côte d'Azur
Inria, CNRS, I3S, France
elena.cabrio@unice.fr

Sara Tonelli
Fondazione Bruno Kessler
Trento, Italia
satonelli@fbk.eu

Nhan Le-Thanh
Université Côte d'Azur
Inria, CNRS, I3S, France
nhan.le-thanh@unice.fr

Abstract

This demo paper presents a system that builds a timeline with salient actions of a soccer game, based on the tweets posted by users. It combines information provided by external knowledge bases to enrich the content of tweets and applies graph theory to model relations between actions (e.g. goals, penalties) and participants of a game (e.g. players, teams). In the demo, a web application displays in nearly real-time the actions detected from tweets posted by users for a given match of Euro 2016. Our tools are freely available at https://bitbucket.org/eamosse/event_tracking.

1 Introduction

In the latest years, social media platforms have become a popular communication channel, thanks to their coverage and speed, which allow users to follow and comment events in real time. In particular, the need to monitor, categorize and organize information is very relevant during large sports events like the Olympic Games or FIFA World Cup, since several matches take place in a limited time span, sometimes in parallel. While summaries are usually made by journalists during the matches, user-generated content from microblogs can be used for the same purpose by building complete summaries of sports games from tweets. For example, Nichols et al. (2012) and Xu et al. (2013) present simple approaches based on the observation of peaks in the tweets' volume. Even though these works can effectively detect the most salient actions in games (e.g. goals), they fail to capture actions that do not generate high volumes of tweets (e.g. shoots). To address this issue, we propose an algorithm to build a timeline with salient actions in sports games based on the tweets posted by users (Edouard et al., 2017). We combine information provided by external knowledge bases to enrich the content of the tweets and apply graph theory to model relations between actions (e.g. goal, penalties) and participants of a game (e.g. players, teams).

In this paper, we present the system built on top of such approach, resulting in a web-based application that displays a fine-grained, real-time summary of sub-events occurring in a soccer game based on the content of tweets, combined with information from external knowledge bases. In the remainder of the paper, we first describe the server component for real-time tweet processing (Section 2) and then the client component, displaying timelines to final users (Section 3).

2 The server component

The server component pipeline is displayed in Figure 1. It consists of the following components: tweet collection and preprocessing, Named Entity Recognition (NER) and timeline generation.

2.1 Tweet collection

The server component is configured to extract tweets either from the Twitter stream or from a local database. In both cases, it requires the names of the opposing teams to be provided (e.g. England VS Wales). To retrieve tweets from Twitter, we provide as query parameters to the Twitter streaming API a set of keywords, that are generated from the names of the teams. Soccer fans tend to use their teams standard abbreviation (e.g. ENG or WAL) more often than full names of the teams

Proceedings of Recent Advances in Natural Language Processing, pages 208–213,
Varna, Bulgaria, Sep 4–6 2017.
https://doi.org/10.26615/978-954-452-049-6_029

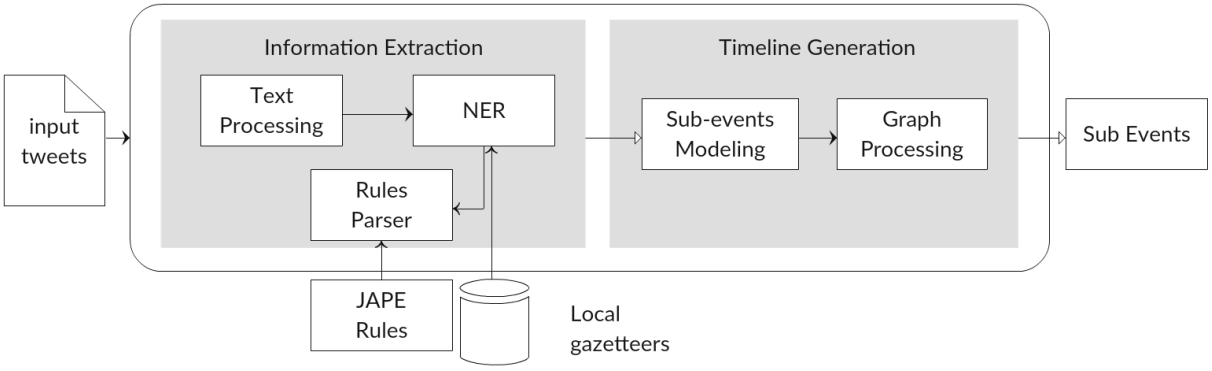

Figure 1: Server component pipeline in which data is flowing in the sense of the arrows. The input are tweets coming from Twitter stream or from a local database and the output is the sub-events detected from the input tweets.

(e.g. England, Wales). Based on this observation, we define heuristic rules to create keywords using different versions and combinations of the team names. For example eng, wal, engwal, waleng were used to query tweets related to the match between England and Wales. Furthermore, the keywords can be also enriched with other terms such as the competition (e.g. euro, euro2016) or soccer actions (e.g. goal).

2.2 Information Extraction Module

After retrieving the tweets, the pipeline extracts the participants and actions mentioned in the tweet, and sets relations between them. In the case of soccer, actions are defined by the Fédération Internationale de Football Association (FIFA). Table 1 shows the complete list of soccer actions detected in the current implementation of the system. Participants are players or teams who are involved in the actions. These information are extracted based on the following steps.

Tweet preprocessing. Input tweets are preprocessed in order to remove noise and redundant information similar to (Edouard et al., 2017). We clean the input tweets in order to remove repetitive characters (e.g. goooooaaaalll), emoticons or URL mentions. Manual inspection suggests that re-tweets are usually observed a few minutes after the original tweets have been sent. Thus, for the specific task of detecting real-time actions in soccer games, we make the assumption that re-tweets are not relevant to the actions reported in a current time window in our processing phase. Note that we do not perform stop words removal, since they are useful for determining relationships between actions and participants (see Listing 1).

```
Rule: involved_in
Priority: 20
(
    {Lookup.minorType == "foot_action"}
    {Token.category == "IN"}*
    {Lookup.minorType == "football\_player"}

): participate
```

Listing 1: JAPE rule to detect relationship between actions and participants in tweets exploiting preposition and subordinating conjunctions.

NER module. For detecting mentions of actions and participants in tweets, we rely on GATE (Cunningham et al., 2002), because it includes a highly flexible NER tool that allows the integration of custom gazetteers. Indeed, in order to detect *actions*, we update its gazetteer based on the Sports Markup Language (Council, 2017), a controlled vocabulary used to describe sports events. SportsML core schema provides concepts allowing the description of events for 11 major sports including Soccer, American football, Basketball or Tennis. For soccer games, we extract actions such as goals, substitutions, yellow/red cards and penalties. Furthermore, we enrich the list of actions with synonyms extracted from Wordnet (Fellbaum, 1998).

For *participants*, we use football-data API[1] that, given a soccer game in input, returns the name of the teams and the players in each team. We employ a set of hand-crafted rules to define different mentions of players' and teams' names in tweets. For instance "giroud", "oliviergiroud" or "olivier_giroud" are considered as mentions

[1] http://api.football-data.org

Event	Description	Participants involved
D1P, D2P	Beginning of the first or second period	Both opponent teams
F1P, F2P	End of the first or second period	Both opponent teams
TIR	Shoot, goal attempt, blocked...	1 player
BUT	Goal, score	1 player
CGT	Substitution, replacement	2 players
CJA	Yellow card	1 player
CRO	Red card	1 player

Table 1: Set of actions detected by the system.

to "Oliver Giroud", a striker of the French national team. Finally, we update the GATE's local gazetteer with actions and participants according to the game.

We initialize GATE with the custom gazetteers and use its in-built NER to identify mentions of actions and participants in the tweets. Whenever actions and participants occur in the same tweet, we extract them and connect them through a link in a graph. Furthermore, we set additional links using JAPE (Java Annotation Pattern Engine) rules, a GATE-specific format to define regular expressions needed for pattern matching. As an example, we report in Listing 1 the JAPE rule matching all tokens whose type is "foot_action" and "football_player", separated by any preposition or subordinating conjunction, all matching patterns are labeled as a "participate".

2.3 Timeline generation

The output of the NER component is a list of quadruples $\langle a, p, t, \omega \rangle$, where a is a soccer action, p the participants, t the timestamp of the tweet and ω is the weight whose value is either $= 2$ if the relation is extracted from JAPE rules or $= 1$, otherwise. Next, the extracted entities are modeled in a temporal event graph. Specifically, we split the game in fixed time windows (e.g. 2 minutes), and create an event graph that models the relationships between actions and participants, represented by nodes, for each time window. The edges connecting two nodes correspond to the retrieved links, whose weight is increased every time a relation is found in a tweet.

At this stage, the weighted relations between actions and participants are considered as sub-event candidates. In a further step, we select the events (i.e. relations between actions and participants) that are to be included in a timeline by retaining those that are above an adaptive threshold. The thresholds are tuned by taking into account both the type of the action and the popularity of the teams involved in the game using Kreyszig standard score formula (Kreyszig, 2007). The detailed description of the algorithm, and its evaluation is presented in (Edouard et al., 2017). The output of the server component is thus a timeline where the extracted actions and participants are connected and temporally ordered.

2.4 Technical implementation

We implement the server component in the Java programming language. As local database, we use MongoDB[2] to store tweets and Twitter4J[3] to collect tweets from the Twitter streaming API.[4] We use the JGraph library (Naveh et al., 2008) to generate and process the event-graph. The server component also includes a socket listener allowing bidirectional communication between the server and the client component.

3 The client component: the web demo

The goal of the client component is two-fold. On the one hand, it displays the timelines containing the salient actions in the match and the statistics related to the selected soccer game (e.g. the current score, the ball possession). On the other hand, it allows users to modify the empirical threshold for the actions included in football games (see details below).

The client component is built as a web application with HTML5, CSS3 and Java Script libraries such as angularJS.[5] Data between the client and the server are exchanged using web socket. The web interface is inspired by the website of

[2]https://www.mongodb.com/
[3]http://twitter4j.org/en/index.html
[4]https://dev.twitter.com/streaming/overview
[5]Angular JS https://angularjs.org/

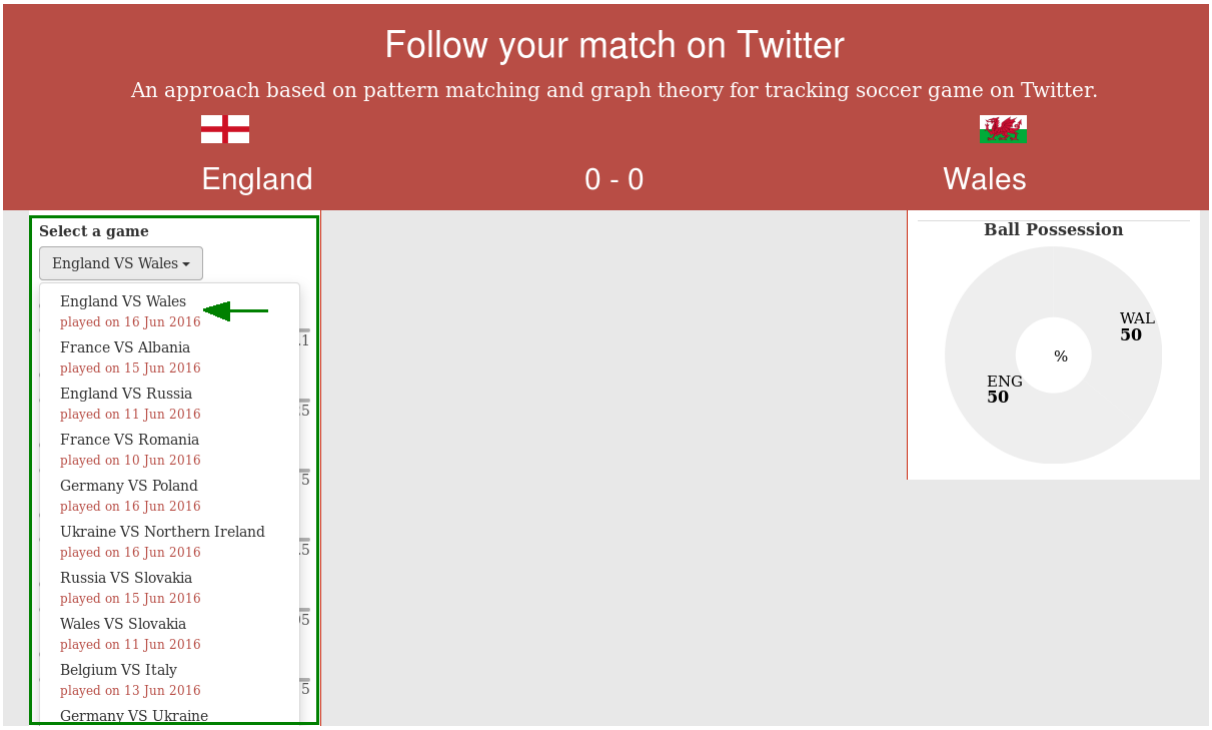

Figure 2: Screenshot of the web demo. To start, the user can select from a list the matches he wishes to track (for demo purposes, the list contains the matches from the first stage of the Euro 2016 championship.

The Guardian[6] describing soccer matches in real time. However, while the list of salient actions in The Guardian is manually compiled by journalists, our goal is to show that this can be automatized through our processing pipeline based solely on tweets.

End users can define different values from the client side: they can select a match and set the algorithm parameters, such as the frequency threshold, to retrieve the actions. Then, the server processes the tweets every two minutes (i.e the time-window) and notifies the clients when salient events are detected in the tweets.

In order to better describe the different demo components, we mark in the screenshot of the web platform (see Figure 3) its five main blocks: *i)* game selection, *ii)* threshold settings for actions selection, *iii)* ball possession, *iv)* salient actions, and *v)* updated score of the match.

Game Selection. For the purpose of the demo, the stream of tweets comes from a local database containing the tweets related to 24 games of the first stage of the Euro 2016 championship.[7] In the current implementation, the server can track one game at a time, which can be selected from the list at the top of the left panel (see Figure 2 for the game selection list, and partition 1, Figure 3). Example matches are Switzerland-Albania or England-Wales.

Action thresholds. As introduced before, the action threshold is the minimal confidence value used by our algorithm to include or discard actions reported in tweets. The action threshold are automatically adapted according to the popularity of the game (i.e the more a game is discussed in tweets, the higher the thresholds and conversely). The lower the confidence, the smaller the number of tweets reporting a certain action on which the algorithm relies in order to add such action to the timeline (i.e. a lower threshold generates a higher number of actions in the timeline, detected with a lower confidence score). Conversely, the higher the threshold, the more confident the algorithm is (but less actions appear in the timeline). For each action, the user can easily modify the thresholds using the range sliders. Thus, the end user can adapt the different empirical values for the thresholds as needed.

[6] https://goo.gl/MWc6dN
[7] https://github.com/HackaTAL/2016/tree/master/Tweets

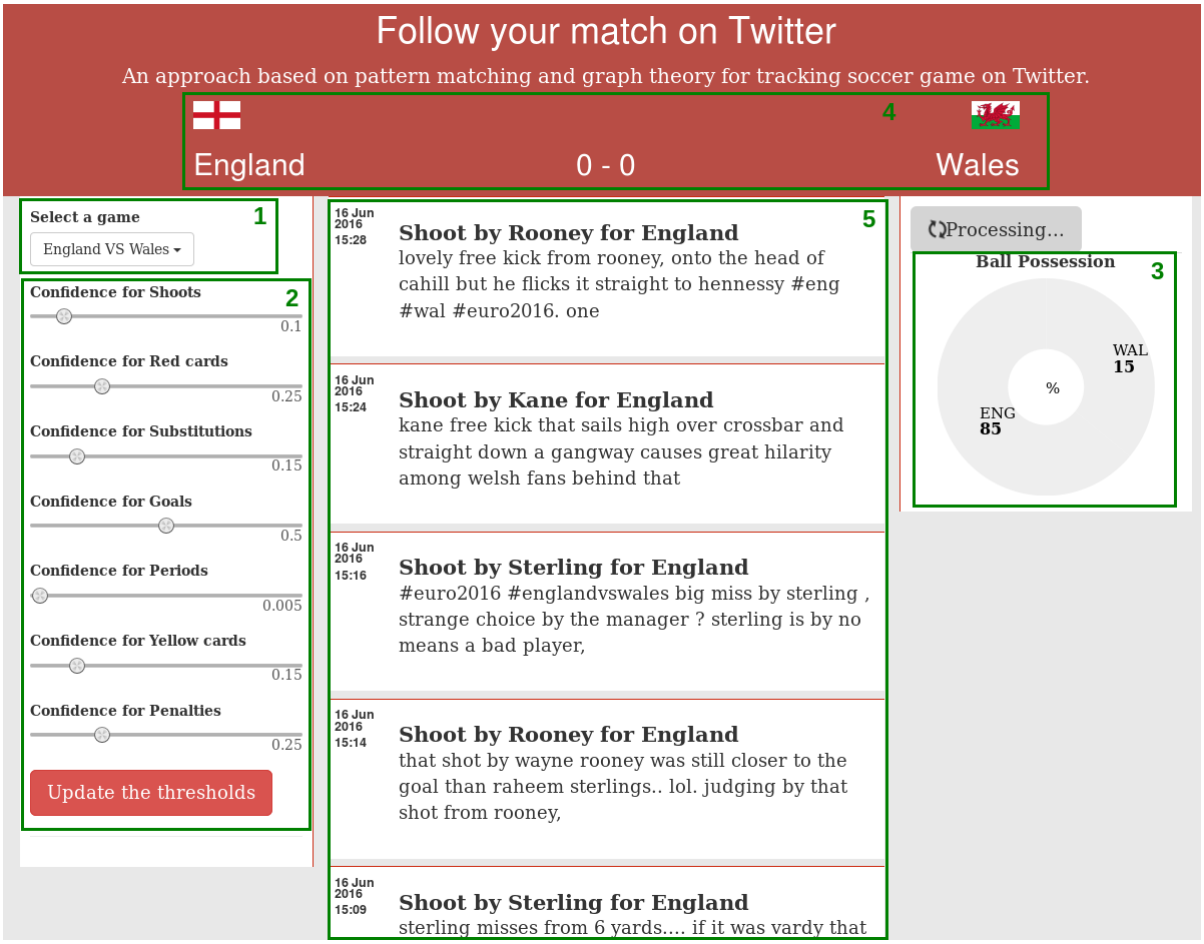

Figure 3: Screenshot of the timeline generated for the soccer game between England and Wales during the first stage of the Euro 2016 championship. The screenshot contains five main blocks : *i)* game selection, *ii)* threshold settings for actions selection, *iii)* ball possession, *iv)* salient actions, and *v)* updated score of the match

Ball possession. This functionality displays the ball possession for each team during a soccer game, i.e. the amount of time a team possesses the ball during a match, expressed as a percentage. To calculate that, we consider the amount of tweets describing actions of a certain team during a time window, i.e. the fact that a team or a player is mentioned in a tweet increases the ball possession score for that team. Ball possession is displayed as a pie chart at the bottom of the page and constantly updated (partition 3, Figure 3). The percentages of ball possession we obtain are very close to those reported by sports media. For instance, at the end of the England vs Wales match, The Guardian [8] reports 64/36 as the ratio of ball possession of the two teams, while we obtain a very similar ratio of 69/31.

Current score The panel at the top of the web page (partition 4, Figure 3) displays the current score of the match, as well as the scorers. When a player scores a goal, the match score is updated and the player name is displayed under the team he scored for, together with the time (see Figure 4). We retrieve the current score of the game from the tweets.

Salient actions. Actions detected in the tweets are displayed in a reversed chronological order to the end user (partition 5, Figure 3). For each action, its type (e.g. goal), the participants (e.g. Gareth Bale, Wales) and the time (e.g. 44 min) are provided. In addition, when an event is confirmed by our approach, we retrieve the content of the tweets that were used to detect this event and apply the approach proposed by (Mihalcea and Tarau, 2004) to produce a summary of the event. For instance, for the action "Shoot by Ram-

[8] E.g. https://goo.gl/MWc6dN Actions there are manually reported by sports journalists.

Figure 4: The score is constantly updated, and the players' name are added below the teams for which they score.

sey for Wales", we provide the following textual paragraph extracted from the tweets, also to show the supporters reaction to a certain action: *"aaron ramsey is by far a better player when he's playing for wales than arsenal #euro2016. aaronramsey ramsey shot from range"*. Or, for the action "Half time! England 2 - 1 Wales", we display *"i hope that's a blessing in disguise, now hodgson has to make a change at half time #engwal #euro2016 #optimistic"*.

3.1 Conference demo session

A video presenting the above described demo can be found at the following address: https://goo.gl/EsI8Db. During the demo session at the conference, participants are invited to test the different components of the web application live: they can select a game from the list of soccer games, click on "Start tracking the game", and experience with the creation of the timeline reporting the salient actions in the selected game. They are also invited to tune the threshold parameters, and compare the obtained results.

References

International Press Telecommunications Council. 2017. SportsML: A solution for sharing sports data. https://iptc.org/standards/sportsml-g2/. [Accessed 03-01-2017].

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Amosse Edouard, Elena Cabrio, Sara Tonelli, and Le Thanh Nhan. 2017. You'll never tweet alone - building sports match timelines from microblog posts. In *Proceedings of RANLP 2017 - Recent Advances in Natural Language Processing conference*.

C. Fellbaum. 1998. *WordNet. An Electronic Lexical Database*. MIT Press.

Erwin Kreyszig. 2007. *Advanced engineering mathematics*. John Wiley & Sons.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. Association for Computational Linguistics.

Barak Naveh et al. 2008. Jgrapht. *Internet: http://jgrapht.sourceforge.net* .

Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. 2012. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. ACM, pages 189–198.

Wei Xu, Ralph Grishman, Adam Meyers, and Alan Ritter. 2013. A preliminary study of tweet summarization using information extraction. *NAACL 2013* page 20.