https://sites.google.com/site/icqqmeas 2015

HUMAN SOCIAL SENSORS: DELAY-DRIVEN DETECTION OF EVENTS IN SOCIAL MEDIA FOR ADVERTISING APPLICATIONS

Klimis Ntalianis[1]*, Maria Tsirintani[1] and Spyridon Binioris[1]

[1]Athens University of Applied Sciences, Faculty of Management and Economics

28, Agiou Spyridonos str., Egaleo 12210, Athens, Greece

*e-mail: kntal@teiath.gr

ABSTRACT

Humans can be regarded as geographically distributed, multimodal sensors, who publish what they see and feel through blogs, forums, product reviews, and several available social networking sites. In this direction, Facebook or Twitter have short delays in reflecting what their users perceive, compared to other venues such as blogs and product reviews, which usually have much longer delays. And the question are: how good are the large social networks at near real-time sensing ? What are the lowest time limits of detecting a new event ? Towards this target, we devise efficient data collection and event recognition solutions over the most famous social networks (like Facebook, Twitter etc), so that content is rapidly extracted. These new capabilities are very useful not only e.g. for creating program guides for live broadcasting, but also for providing a better, more effective pricing mechanism for selling advertisement slots, typically by sensing instantaneous popularity of a segment. Preliminary experimental results over real data illustrate the promising performance of the proposed scheme.

Keywords: Social Media, Social Computing, Segment Popularity, Social Latent Semantic Analysis, Advertisement Slots.

JEL Classification: M37

1. Introduction

Publications (posts, twits etc.) in social media can be considered as the "readings" of human sensors that contain information about the physical world that we sense. In this direction, social networks have expanded rapidly during the last decade and till now their content has already been used in a variety of applications such as: (a) for ranking news stories (based on the interest they attract), (b) for building profiles of user preferences (based on implicit/explicit interactions), (c) even for products' recommendations (based on recorded activity). However this type of conversational, user-generated content found in social networks, might add great value and open amazing new horizons to marketing.

In this direction, Facebook or Twitter are considered to be the fastest to reflect what their users perceive. While many have demonstrated Twitter can provide insights into major social and physical events like earthquakes, celebrity deaths and presidential elections, in this work, we focus on the problem of near real-time sensing, for less significant but more frequent events such as sport games. Towards this target, we devise efficient data collection and event recognition solutions also over rule-stringent social networks (like Facebook), so that content is automatically annotated. These capabilities are very useful not only for creating program guides for live broadcasting, but also for providing a better pricing mechanism for selling advertisement slots, typically by sensing instantaneous popularity of a segment.

Here it should be mentioned that there are multiple challenges toward game event recognition using e.g. Facebook, Twitter or other social media. Firstly, posts should be detected and separated. Secondly, we should be able to distinguish which game a post is referencing to, when many games are played simultaneously. Finally, this analysis should be done in near real-time, having as short delays as possible. For example, to be useful for advanced advertising auctions, the event has to be recognized within seconds in order to auction and consequently display a customized advertisement. Otherwise, the information is meaningless when people are no longer interested in the event. In our case, two sources contribute to the delay: users and the proposed algorithm. More specifically, between perceiving an event and posting it, a fraction of time passes. This human delay is mainly determined by how fast social media users perceive the event, how fast they react to it, and how fast they type the post. Additionally, our data collection and analysis also introduces delays. To efficiently tackle this challenge, we incorporate a fast filtering method that discards content when it is out of time and incorporate a simple lexicon-based method. Preliminary experimental results illustrate the promising performance of the proposed scheme.

2. Related Work

Social media have been used to detect different social and physical events. Sakaki et al [1] and Qu et al [2] investigated the earthquakes detection. Vieweg et al [3] studied the grassfire and floods, on microblogs. TwitterStand [4] identifies current news topics and clusters the corresponding tweets into news stories. Hannon et al [5] used post rate of tweets to produce video highlights of the World Cup off-line. They did not recognize game events nor did they produce highlights in real-time. Chakrabarti and Punera [6] assumed that a game event is already recognized and focused on describing the event using Hidden Markov Models trained with tweets collected from events happened in the past. Ekin et al [7] employed visual features analysis to summarize soccer videos. Petridis et al [8] used MPEG-7 and webcast text to extract sports events.

On the other hand, latest works include [9]-[14] . In [9] a social software platform is proposed to detect a number of meaningful events from information diffusion patterns on social network services. The platform has been applied to fetch and cluster tweets from Twitter into relevant categories to reveal hot topics. In [10] the problems of identifying key trends via social media sources and of verifying the stories that emerge this way have been examined. Four key needs have been identified: (a) journalists require the ability to search for stories across multiple social networks in order to filter and spot key trends, (b) it is increasingly important that they find relevant video content and pictures as well as text-based material, (c) they should be able to customize the data in order to make it relevant to their own concerns and (d) they require help with verification (this includes identifying who is a reliable source, filtering out fake pictures and video content, and using geo-location to cross-check where individuals actually are). In [11] real-time and deferred interactions with video content are incorporated as rich vocabulary. In [12] the case that Twitter is a particularly promising sensor for TV is considered, while it is noted that TV watching and tweeting cuts across genres, but is particularly intense in sports. In [13] four questions are addressed: (a) can the event lexicon of a sport be detected with the speed and accuracy required for a real-time TV experience from the Twitter ? (b) how can the most capable Human Sensors be identified for a particular event in the event lexicon ? (c) how is the total effort reduced in aggregating the optimal set of Human Sensors and (d) what is the quality of sensing (per event and in aggregate) that can

be expected ? Finally in [14] the spatial and temporal characteristics of the twitter feed activity responding to a 5.8 magnitude earthquake is analyzed. In this framework, it is argued that these feeds represent a hybrid form of a sensor system that allows for the identification and localization of the impact area of the event and the potential of the use of harvested social media content for event monitoring is assessed.

3. Delay Issues

Delay is very important for any sensing framework. Many applications require game events and game popularity information to be detected in real-time, so that end users (e.g. advertisers) have all information on time. Otherwise, the information is meaningless when people are no longer interested in the event. For example, the Electronic Program Guide (EPG) needs the game information in real time.

In open social media that provide a communication API, three sources contribute to the delay: (a) the delay introduced by users (time between perceiving an event and writing a post), (b) delay in receiving content through the communication API (this delay is generally affected by the workload of the site, the index mechanism etc.) and (c) delay to perform data analysis. On the other hand in rule-stringent social media, there is also a delay for content wrapping, since data are not provided through a communication API.

3.1 Delay Introduced by the Users

Human delay expresses the interval between an event (e.g. as it happens on our TV) and the time stamp that is associated to a post referring to this event. It was found in [15] that the shortest human delay is about 13 seconds for non-mobile devices, while the longest is 27 seconds, with an average of 17 seconds. Interestingly, touchdowns saw shorter human delays than other less significant ones, indicating that users post faster for more significant events.

It should also be noted that there is a short delay in broadcasting live materials used to prevent profanity, bloopers, violence, or other undesirable material from making it to air. This delay may vary in different locations and is approximately 7 to 12 seconds. As a result, tweets by game watchers who are in the stadium could be posted earlier than those from homes. However, location analysis is not among the objectives of this paper and should be further investigated in future experiments.

Additionally even though it may be expected posts from mobile devices to present shorter delays (mainly due to the fact that the effort to switch from game watching to tweeting seems to be lower on mobile devices than on PC or laptops), this is not the case. Relevant results show the opposite: by examining the source clients, it was observed that nearly 40% of game-related posts were from clients on recognizable mobile devices, i.e. iPhone, BlackBerry, Android, txt, mobile, HTC, MOTO, and iPad. Furthermore non-mobile users react 3 to 5 seconds faster in several different events. The possible reasons may have to do with typing speed (since most users write much faster on a PC keyboard than on a mobile device, either with a small or with a virtual keyboard). Furthermore usually PCs are more powerful in performing computations than mobile devices. Finally network delay (in case of protocols supporting reliable, ordered, and error-checked delivery of information) has to do with signal quality. In case of mobile devices, wireless transmission is considered, which is more error prone than wired transmissions. Thus the signal may be transmitted two or more times in order to be successfully decoded at the receiver.

3.2 Delay Introduced by the Communication API

A characteristic example of communication API is the Twitter Streaming API, which allows keyword-based retrieval in real time. In this case the delay can be defined as the difference between the Twitter timestamp of a tweet and the time the same tweet has been obtained from Twitter. The delay has been calculated to about 30 seconds for tweets that only contain custom, random keywords and about 1 second for tweets with “Twitter promoted”, e.g., Twitter promoted #sb45 during the 2011 Super Bowl. Second, the delay seems fairly independent of the post rate. However it is surprising that the choice of keywords makes such a huge difference in Twitter delay: 30 seconds vs. 1 second ! This may be explained by taking into consideration that Twitter maintains its Streaming API results quality by applying the status quality metrics, in some instances, in combination with other metrics, to filter the tweets [16]. The goal is to eliminate spams, inappropriate, or repetitious tweets. Therefore, tweets that contain rarely used keywords may take longer to pass the process. In addition, developers in the Twitter development talk, an online forum in Google groups [17], mentioned that Twitter may index interesting tweets to improve the search speed. Since the Streaming API with filter predicates, i.e., follow, track, or locations, involves search, it may be speculated that such indexing helps significantly reducing the Twitter delay for Twitter promoted keywords and popular keywords such as team names [18]-[19]. It was also observed that the post rate does not impact the Twitter delay noticeably.

3.3 Delay Introduced by the Analysis Module

The analysis module to recognize events, also introduces a delay. We have implemented an intelligent content wrapping and analysis architecture so that this kind of delay is kept at a minimum level. In Section 4, details of this architecture are provided. Here we report the introduced delay. According to our estimations, the overall analysis takes less than 9 seconds on average. Among them, wrapping and token segmentation needs about 3 seconds, while analysis (including all computation tasks) takes about 6 seconds on average. As it can be observed, the longest delay is introduced by the communication API and for tweets that only contain custom, random keywords (~ 30 seconds). Thus the usual shortest delay can be about 23 seconds while the usual longest can reach 66 seconds. Here it should be mentioned that the ESPN web page has 90 seconds lag in updating the latest score changes. It should also be mentioned that 23 seconds may be tolerable for the advertising industry (since about 40 - 60 seconds are needed for the replay and fans jubilation), but after 66 seconds it may be too late.

4. Intelligent Wrapping and Content Analysis for Delay Reduction

As it was previously mentioned, 66 seconds is a long time for events detection e.g. in a sports game, when there is not a "Twitter promoted" tag. In this case, the Twitter's communication API cannot be effectively incorporated and other faster solutions should be proposed. In this paper an intelligent wrapping submodule is incorporated, which segments a Twitter web page and gathers content much faster compared to the communication API. In particular the proposed wrapper exploits the format of social media web sites to discover the underlying structure in order to finally infer and extract posts and corresponding associated information from the web pages. The system first identifies the section of the web page that contains the post to be extracted and then extracts it by using clustering techniques and other tools of statistical origin.

The proposed system is based on STAVIES [20] and comprises of two modules: namely the transformation and the extraction module. These modules are further subdivided into components, each one responsible for a different task. The overall system architecture, which extracts the information from the data source is composed of three distinct phases: Preparation phase, Segmentation phase and Information Retrieval phase (see Figure 1).

During the first phase the web document is prepared for the extraction that will follow. This is done by inserting the HTML document into the Transformation module. This module parses the HTML document and transforms it into well-formed XHTML, correcting any existing irregularities that would misguide the extraction process. Then the produced XHTML document is used to generate a tree representation of the initial web page (see Figure 2).

Figure 1: Overal architecture of the intelligent wrapper

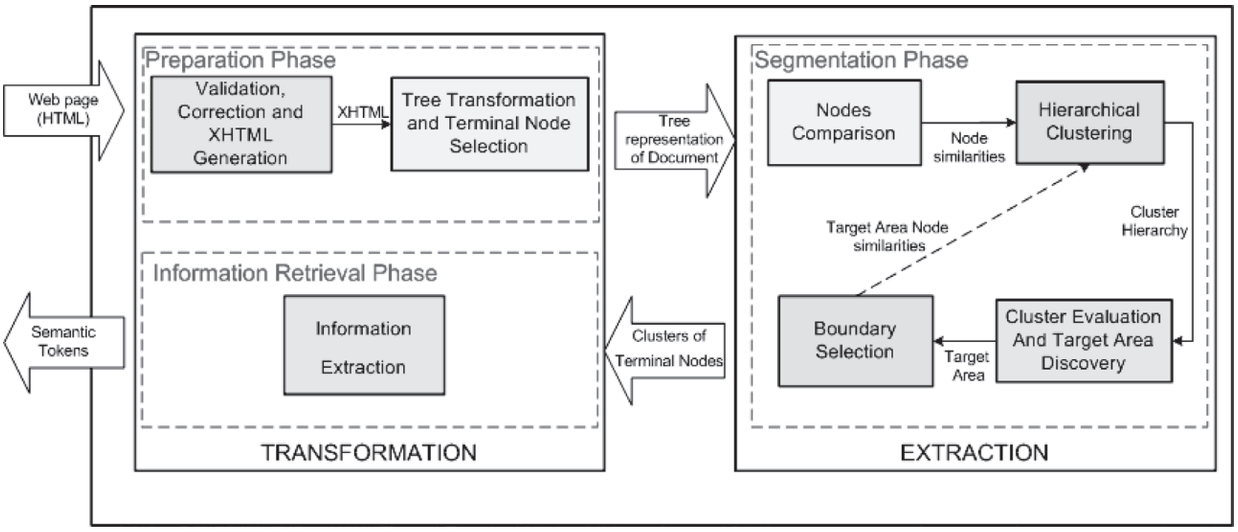
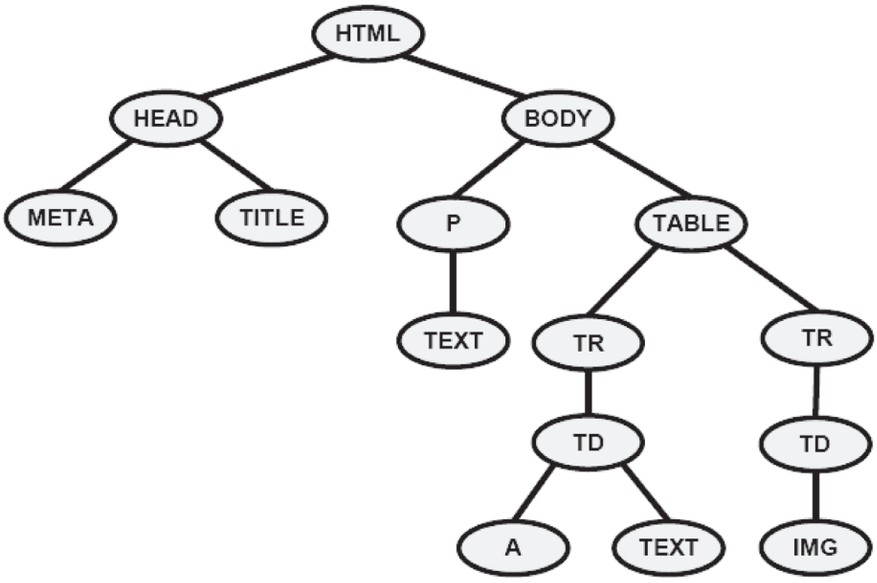


Figure 2: Tree representation of a web page, produced by an XHTML document

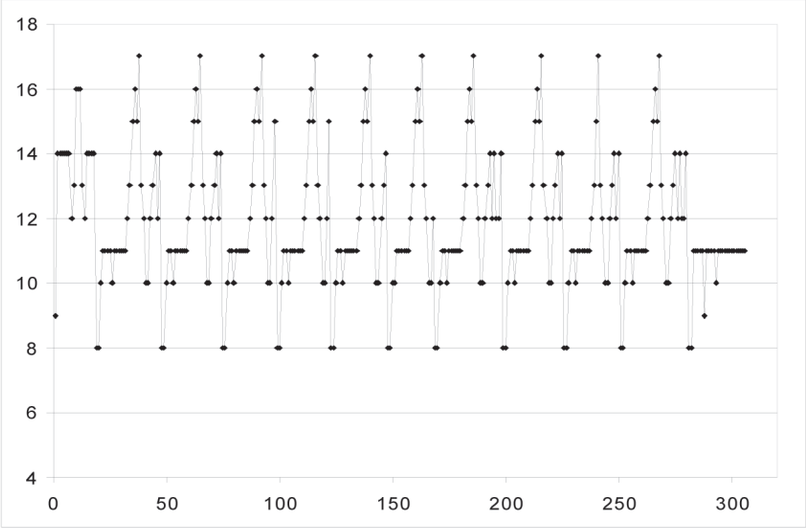


The terminal nodes of this tree correspond to elements of the web page among (namely images, links and text) where the useful information resides, while the intermediate nodes correspond to layout descriptive elements. Therefore the terminal nodes are the ones needed during the extraction process. At this point it is also important to note that the selection of the terminal nodes is performed in a way that preserves their ordering in the web page. The ordering of the terminal nodes is critical because it constitutes one of the criteria used to infer the semantic relationships between elements of the HTML page, since lack of locality in the ordering of these elements is translated into semantic irrelevance. Once the terminal nodes have been selected, the second phase (Segmentation phase) performs a clustering to identify the terminal nodes of interest and segment them into appropriate subsets. Finally the last phase concludes the operation by mapping the generated terminal node clusters to elements in the initial HTML web document. This task is straightforward, since there is one-to-one relationship between the terminal nodes of the tree and the elements of the initial web document. In this way the desired information is identified and can be retrieved.

4.1 Preparation Phase

The first component acting during this phase is the "Validation, Correction and XHTML Generation Component". Given a web document, like those shown in Figure 1, this component performs a syntactical correction to the source's HTML by transforming it into XHTML. This is necessary because, due to the leniency in HTML parsing by modern web browsers, a major portion of the web page is not well-formed. Data sources often either contain invalid tags or their tags are placed in a wrong manner. These irregularities could cause problems in the tree representation of the web page or, even worse, misguide the extraction process. Therefore this component's usage for cleaning and normalizing the HTML page is imperative. The cleaned and normalized page is then fed into the "Tree Transformation and Terminal Node Selection Component", which generates a tree representation of the page. The root of this tree corresponds to the whole document. The intermediate nodes represent HTML tags (e.g. <table>, <ol>, <tr>, <p>, etc) that determine the layout of the page. Finally, the terminal nodes (leaf nodes) correspond to visual elements on the web page, namely images, links and/or text. Once the tree construction is completed, the terminal nodes are selected. We only select the terminal nodes for further processing since they represent the elements of the page among which the useful information resides. The non-terminal nodes are not in our interest, since they represent layout descriptive elements, in other words the way the information is displayed in a web browser. We must note here that the selection of the terminal nodes happens in a way that preserves their ordering in the web page. The ordering of the terminal nodes is critical in our application. It is one of the criteria we use to infer the semantic relationships between elements of the HTML page since lack of locality in the ordering of these elements is translated into semantic irrelevance.

Figure 3: A plot of the "target area" for a sample web page. Each "semantic token" represents a multimedia file



4.2 Segmentation Phase

First the sections in the input page are identified and one of them is characterized as the target area, i.e. the area of the input page where the semantic tokens reside (post area). In this way, irrelevant elements in the page e.g. logos, advertisements etc. are eliminated. The second step is to further segment the target area, thus extracting the semantic tokens. The segmentation phase is performed in two different steps: the "Target Area Discovery" and the "Target Area Segmentation" step.

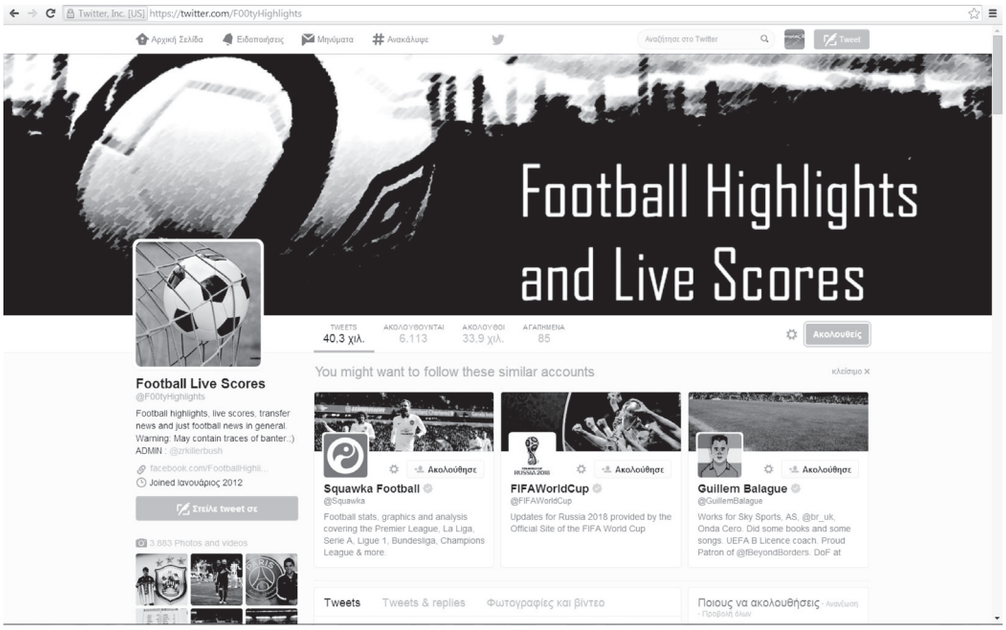
Given the list of terminal nodes n1, n2, ..., nN, the "Target Area Discovery" process aims at selecting a subset of these nodes corresponding to the elements belonging to the target area. This is achieved by applying hierarchical clustering. The outcome of the hierarchical clustering will reveal the region in the web document that contains the semantic tokens.

In the "Target Area Segmentation" step, a segmentation of the target area into further segments that represent the semantic tokens, is performed by locating a "cut-off" level. Once the cut-off level is located, the system performs a clustering similar to the one performed before. Each output cluster now represents a semantic token. A characteristic example of a target area is presented in Figure 3, where clusters (semantic tokens) corresponding to different information objects can be easily observed. Finally the text of each extracted token (text of a specific post) is analyzed using the sports dictionary of http://www.sportsdefinitions.com/, so that events are detected.

5. Experimental Results

We have implemented a prototype system to extract information from a variety of social networks using Java 8 and third-party libraries, like Tidy [21] for web page normalization (in phase one). For testing purposes we implemented a complementary component, which we call "Remote Page Fetching and Next Link Following Component". This component feeds the system with web pages in an uninterrupted way. It accepts a URL, issues an HTTP request to the remote web data source and fetches the corresponding web page containing the results. Each result is a structural token and has to be extracted. In most cases the results spread into more than one pages with every page containing a link to the next page. This component passes pages to the system by following through the chain of links.

Figure 4: The Twitter page of Football Live Scores (@F00tyHighlights).



Next and in order to evaluate our system's performance in real-life scenarios, we have tested the proposed architecture on several popular social networks such as Facebook. Twitter, LinkedIn etc. For presentation reasons and since we are interested in live games, only results from Twitter are presented.

In particular in Figure 4 the page Football Live Scores (@F00tyHighlights) of twitter.com is depicted, which was followed for a period of one month (1st of March till 31st of March 2015). During this period, the content of the page was gathered and analyzed in offline mode, once a day. This is justified, since the offline performance is not different to the online performance for our scheme. In particular, the delays introduced by users and the communication API do not play any role in our experiments. Furthermore the delay introduced by the analysis module is the same both in online and offline mode. Additionally we aimed at minimizing the time of automatically recognizing an event, so that auctions for advertisement slots are more effective. This is useful especially in cases of web transmission of non-famous games, since in traditional transmissions (e.g. DVB-T) and for important games, the event is manually recognized by human advertisers with almost no delay. However these non-famous games may attract several people to a website and banners may change based on auction procedures.

Now regarding the proposed content gathering mechanism, in total 1,611 twits have been gathered, a characteristic one of which can be seen in Figure 5 (retrieved on the 31st of March). In this twit, the second goal of Japan vs Uzbekistan at the 54th minute of the game is reported. The wrapper has segmented this twit into tokens and extracted the following information: "Football Live Scores", "@F00tyHighlights", "Japan 2 – 0 Uzbekistan. 54' Shinji Okazaki scores for Japan!", "RETWEETS 4", "FAVORITES 2", "5.04 π.μ. – 31 Μαρ 2015". Based on the sports dictionary, the word "scores" is mapped to the word "goal", which is recognized as an important event by the proposed algorithm.

Here it should be mentioned that in order to keep to a minimum the wrap time, only twits of the current day were gathered and analyzed (by checking the post date of each twit). The average time to wrap a twitter page (current day) with our system was less than 0.5 second in a machine with typical configuration (Core i5 5200U 2.2 GHz, PC with 4GB RAM), while tokens segmentation took about 2.5 seconds on average. Furthermore, a major advantage of the algorithm is its tolerance to the dissimilarities among the structural tokens, especially since these dissimilarities occur very often in these types of web pages. For example in the typical case of a social media site where the structural tokens are the posts contained in the web page, different numbers of comments, contained links etc exist.

Finally, in order to estimate the down limit of our scheme we have calculated the delay between events and their automatic detection by the proposed scheme. Towards this direction we have manually assigned to 300

out of the 1,611 twits the exact time instances they have occurred and compared them to the time assigned to each respective twit by the intelligent analysis mechanism. There were twitter pages that posted events very fast (less than 10 seconds), but there were also other pages with much larger delays (more than one minute). Thus our scheme achieved shortest delay of 18.51 seconds. However, very short delays, which are the desirable ones, depend on following proper sites and this is a major problem that should be investigated in future research.

Figure 5: A characteristic post from Football Live Scores (@F00tyHighlights).

6. Conclusion

A novel fully automated data collection and event recognition scheme over the most famous social networks (like Facebook and Twitter) has been presented. The scheme provides new capabilities to the advertisements industry since it supports better pricing for selling advertisement slots. Additionally it can also support several other applications such as creation of program guides for live broadcasting. Preliminary experimental results over real data illustrate the promising performance of the proposed scheme, which achieves a shortest delay of about 18 seconds. In the future, social media pages with the shortest delays should be found, followed and analyzed. Additionally, ways of finding percentages and demographics of fans for each specific game should be examined.

Acknowledgements

The authors would like to thank very much all members of the Online Computing Group for participating in the scientific experiments of this paper. Especially Mr. Georgios Nikopoulos and Mr. Nikolaos Manolis for their advices and support.

References

T. Sakaki, M. Okazaki, and Y. Matsuo, “Earthquake shakes Twitter users: real-time event detection by social sensors,” in Proc. ACM WWW, 2010.

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.

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, 2010.

J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling, “TwitterStand: news in tweets,” in Proc. ACM SIGSPATIAL, 2009.

J. Hannon, K. McCarthy, J. Lynch, and B. Smyth, “Personalized and automatic social summarization of events in video,” in Proc. ACM IUI, 2011.

D. Chakrabarti and K. Punera, “Event Summarization using Tweets,” in Proc. AAAI ICWSM, 2011.

A. Ekin, A. M. Tekalp, and R. Mehrotra, “Automatic soccer video analysis and summarization,” Image Processing, IEEE Transactions on, vol. 12, pp. 796-807, 2003.

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 -, vol. 153, pp. 255-262, 2006.

D. T. Nguyen and J. E. Jung, “Privacy-Preserving Discovery of Topic-Based Events from Social Sensor Signals: An Experimental Study on Twitter,” The Scientific World Journal, Hindawi Publishing Corporation, Vol. 2014, Article ID 204785, Apr. 2014.

S. Schifferes, N. Newman, N. Thurman, D. Corney, A.S. Goker, and C. Martin,C., "Identifying and verifying news through social media: Developing a User-Centred Tool for Professional Journalists," Digital Journalism, 2014.

David A. Shamma, Lyndon Kennedy, and Elizabeth F. Churchill. 2012. Watching and talking: media content as social nexus. In Proceedings of the 2nd ACM International Conference on Multimedia Retrieval (ICMR '12). ACM, New York, NY, USA, , Article 12 , 8 pages.

Arkaitz Zubiaga, Damiano Spina, Enrique Amigó, and Julio Gonzalo. 2012. Towards real-time summarization of scheduled events from twitter streams. In Proceedings of the 23rd ACM conference on Hypertext and social media (HT '12). ACM, New York, NY, USA, 319-320.

Venu Vasudevan, Jehan Wickramasuriya, Siqi Zhao, Lin Zhong "Is Twitter a Good Enough Social Sensor for Sports TV?," Workshop on Pervasive Collaboration and Social Networking, San Diego, March 2013.

A. Crooks, A. Croitoru, A. Stefanidis and J. Radzikowski, " #Earthquake: Twitter as a Distributed Sensor System," Transactions in GIS, Vol. 17, No. 1, p.p. 124–147, 2013.

S. Zhao, L. Zhong, J. Wickramasuriya and V. Vasudevan, "Human as real-time sensors of social and physical events: A case study of Twitter and sports games," Rice University Technical Report No. TR0620, Houston, TX, 2011.

Twitter API Documentation, "Streaming API Concept, http://dev.twitter.com/pages/streaming_api_methods."

"Twitter Development Talk, http://groups.google.com/group/twitter-development-talk."

Twitter, "Twitter Search Best Practices, http://support.twitter.com/groups/32-something-s-not-working/topics/118-search-problems/articles/42646-twitter-search-best-practices."

Twitter Development Talk, "Rate limit for streaming API, http://groups.google.com/group/twitter-development-talk," 2010.

N. K. Papadakis, D. Skoutas, K. Raftopoulos, and T. A. Varvarigou, "STAVIES: A System for Information Extraction from Unknown Web Data Sources through Automatic Web Wrapper Generation Using Clustering Techniques," IEEE Trans. on Knowledge and Data Engineering, Vol. 17, No. 12, Dec. 2005.

Tidy, http://tidy.sourceforge.net/