Event detection and analysis on short text messages

Amosse Edouard

UNIVERSITÉ CÔTE D'AZUR

ED STIC

THÈSE

présentée en vue de l'obtention du grade de

Docteur en Sciences
DE L'UNIVERSITÉ CÔTE D'AZUR
Mention: Informatique

Soutenue le *02 Octobre 2017* **par :**

Amosse EDOUARD

Event detection and analysis on short text messages

Devant le jury composé de:

Bernardo MAGNINI	Directeur de Recherche	Rapporteur
Sylvie DESPRES	Professeur des Universités	Rapporteur
Frédéric PRECIOSO	Professeur des Universités	Président du Jury
Elena CABRIO	Maître de Conférences	Co-directeur de thèse
Nhan LE THANH	Professeur des Universités	Invité

École doctorale et spécialité :
STIC : Sciences et Technologies de l'Information et de la Communication
Unité de Recherche :
Laboratoire d'Informatique Signaux et Système, Equipe SPARKS
Directeur(s) de Thèse :
Nhan LE THANH et *Elena CABRIO*

Abstract

In the latest years, the Web has shifted from a read-only medium where most users could only consume information to an interactive medium allowing every user to create, share and comment information. The downside of social media as an information source is that often the texts are short, informal and lack contextual information. On the other hand, the Web also contains structured Knowledge Bases (KBs) that could be used to enrich the user-generated content. This dissertation investigates the potential of exploiting information from the Linked Open Data KBs to detect, classify and track events on social media, in particular Twitter. More specifically, we address 3 research questions: i) How to extract and classify messages related to events? ii) How to cluster events into fine-grained categories? and 3) Given an event, to what extent user-generated contents on social medias can contribute in the creation of a timeline of sub-events? We provide methods that rely on Linked Open Data KBs to enrich the context of social media content; we show that supervised models can achieve good generalisation capabilities through semantic linking, thus mitigating overfitting; we rely on graph theory to model the relationships between NEs and the other terms in tweets in order to cluster fine-grained events. Finally, we use in-domain ontologies and local gazetteers to identify relationships between actors involved in the same event, to create a timeline of sub-events. We show that enriching the NEs in the text with information provided by LOD KBs improves the performance of both supervised and unsupervised machine learning models.

Résumé

Les réseaux sociaux ont transformé le Web d'un mode lecture, où les utilisateurs pouvaient seulement consommer les informations, à un mode interactif leur permettant de les créer, partager et commenter. Un défi majeur du traitement d'information dans les médias sociaux est lié à la taille réduite des contenus, leur nature informelle et le manque d'informations contextuelles. D'un autre côté, le web contient des bases de connaissances structurées à partir de concepts d'ontologies, utilisables pour enrichir ces contenus. Cette thèse explore le potentiel d'utiliser les bases de connaissances du Web de données, afin de détecter, classifier et suivre des événements dans les médias sociaux, particulièrement Twitter. On a abordé 3 questions de recherche: i)Comment extraire et classifier les messages qui rapportent des événements? ii)Comment identifier des événements précis? iii)Étant donné un événement, comment construire un fil d'actualité représentant les différents sous-événements? Les travaux de la thèse ont contribué à élaborer des méthodes pour la généralisation des entités nommées par des concepts d'ontologies pour mitiger le sur-apprentissage dans les modèles supervisés; une adaptation de la théorie des graphes pour modéliser les relations entre les entités et les autres termes et ainsi caractériser des événements pertinents; l'utilisation des ontologies de domaines et les bases de connaissances dédiées, pour modéliser les relations entre les caractéristiques et les acteurs des événements. Nous démontrons que l'enrichissement sémantique des entités par des informations du Web de données améliore la performance des modèles d'apprentissages supervisés et non supervisés.

Rezime

Rezo sosyal yo chanje fason moun itilize intènèt, yo fèl pase de yon mòd kote moun te ka li sèlman a yon moun pi ouvè kote tout itilizatè gen dwa kreye, pataje, e komante. Pi gwo defi pou trete enfòmasyon ki pataje nan rezo sosyal yo, se paske souvan yo pa respekte ankenn fòma men anplis yo pa genyen ase enfòmasyon sou kontèks yo, sa vle di ki sijè prensipal mesaj yo. Yon lòt bò, entènèt la genyen konesans ki konstwi a pati de tèm ki ekziste nan ontoloji ki ka itilize pou rann pi rich, pi konplè mesaj ki pibliye sou rezo sosyal yo. Nan tèz sa a nou analize koman konesans ki ekziste sou entènèt ka pèmèt nou detekte, klasifye et swiv evènman ki ekziste nan rezo sosyal, plis patikilyèman Twitter. Pi jeneralman, nou atake 3 keksyon rechèch: i) Kouman nou ka idantify e klasifye mesaj sou Twitter ki pale de evènman? ii) Kouman nou ka idantify evènman ekzat twit yo ap pale? iii) Si nou konnen yon evènman ekziste, koman nou ka trete twit ki pale de li pou nou kreye yon rezime de chak ti evènman ki pwodwi? Travay nou fè nan tèz sa a pèmèt nou kreye metòd ki pèmèt nou jereralize antite poun ranplase yo pa konsèp ki ekziste nan onloloji, sa ki pèmèt nou anpeche yon modèl aprann twòp bagay sou tèm ki plis repete nan moman kote lap aprann. Konsa tou, nou kreye yon modèl ki baze sou teyori graph pou nou konstwi relasyon ki ekziste ant antite e lòt tèm ki mansyone nan mesaj yo, sa ki pèmèt nou idantifye evènman presi ki ka enterese itilizatè yo. Yon lòt bò, nou itili baz ki gen done sou yon domèn patikilye pou nou detekte relasyon ki ekziste ant sa ki pase nan yon evènman ak moun ki patisipe ladann. Finalman, nou moutre ke enfòmasyon ki ekziste sou entènèt, ka itilize pou rann pi rich mesak ke itilizatè pibliye sou Twitter; konsa tou nou moutre ke enfòmasyon sa yo pèmèt nou amelyore pèfòmans modèl nou yo pou detekte, klasifye et swiv evènman ke se swa modèl siveye ou modèl non siveye.

Acknowledgements

I would like to thank my supervisors Elena Cabrio, Sara Tonelli and Nhan Le-Than, for their guidance and encouragement throughout this PhD. Particularly, thanks to Elena and Sara for being there anytime I needed, thank you so much for your patience. Many thanks to my examiners Bernardo Magnini and Sylvie Despres for their valuable feedbacks.

I am grateful to all my colleagues at WIMMICS and specially my fellow doctoral students, for their feedback, cooperation and of course friendship. Thanks to Emilie Palagi, Amel Ben Othman and Abdul Macina for their unconditional support.

A very special gratitude goes out to all my friends for lending an ear and taking my mind ofthings when needed. Specially, thanks to Yvan, Kimia, Melissa, Emeric and Julia. Also, I am grateful to Serge Miranda, Tayana Etienne and Gabriel Mopolo for their support and advice during my studies.

More importantly, many thanks to my family for their support and encouragement. I would like to express my gratitude to Mom and Dad for the many years of support for my study, through school and my undergraduate years as well as through this PhD. Finally, thanks to my wife, Kate, for her understanding and support over the last few years.

To Kate.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Introduction

This Chapter is intended to introduce readers to the context and the motivations underlying the present research work, and provide its positioning in the task of detecting events from Twitter.

1.1 The Context

Since its launch in 2006, Twitter[1] has become the most popular microblog platforms that allows registered users to quickly broadcast information about daily activities as well as sharing or commenting latest news to a wide range of users worldwide (Java, Song, Finin, & Tseng, 2007; Kwak, Lee, Park, & Moon, 2010). Central to the notion of news is the concept of events, commonly defined as "something that happens at a specific time and place" (Allan, Carbonell, Doddington, Yamron, & Yang, 1998). Nowadays, common citizens who witness events usually turn on Twitter to share observations within their communities. Traditional news sources (e.g. CNN, New York Times) also use Twitter to quickly broadcast breaking news or links to the latest articles. Until December 2016, the @BreadkingNews account, used by hundreds of journalists to broadcast recent news around the world, was followed by 9.6M+ people. Also, public administrations use Twitter to provide official information regarding the latest or on-going events.

It has been observed that several major events are first mentioned on Twitter

[1]Twitter `http://twitter.com`

than on newswire websites. Prominent examples include the death of Osama Bin Laden [2], which has been largely discussed on Twitter prior to the official confirmation by the White House and media news. Other examples include the death of Michael Jackson, the explosions at the Boston Marathon 2013, or the death of the former British Prime Minister Margaret Thatcher in April 2013, which were first reported by social media and later picked up by former media news. In this context, Twitter has emerged as a powerful source of timely ordered information covering various topics from every corner of the world. More importantly, such information are generally produced by common citizen[3] as soon as they observe (or hear about) them.

The capability to understand and analyze the stream of messages on Twitter is therefore an effective way to monitor what people think, what trending topics are emerging, and which main events are affecting people's lives. This is crucial for companies interested in social media monitoring, as well as for public administrations and policy makers. Also companies and organizations monitor tweets in order to report or confirm recent events. For instance, journalists from the storyful Company [4] use Twitter as information source for retrieving latest events. (Sakaki et al., 2010) describe an automatic approach for detecting recent earthquakes in Japan, simply by monitoring tweets. Also, (Abel, Hauff, Houben, Stronkman, & Tao, 2012) use tweets to evaluate the impact of events reported in an incident report systems. These examples among many others show that the analysis of tweets is a very relevant task that has generated a lot of interest in the latest years.

1.2 The Problem

In the Topic Detection and Tracking (TDT) community, the task of analyzing textual documents for detecting events is called Event Detection (ED). ED is generally defined as a discovery problem, i.e., mining a set of documents for new patterns recognition (Y. Yang, Pierce, & Carbonell, 1998). It mainly consists in discovering new or tracking previously identified events. Early works that address

[2] `http://www.huffingtonpost.com/2011/05/02/osama-bin-laden-death-twitter-leak_n_856121.html`

[3] Also referred to as citizen journalists or human sensors (Sakaki, Okazaki, & Matsuo, 2010)

[4] `https://storyful.com/publishers/`

ED from texts were mainly focused on finding and following events using conventional media sources such as a stream of broadcast news stories (Pustejovsky et al., 2003; Schilder, Katz, & Pustejovsky, 2007).

In the latest years, however, NLP researchers have shown growing interest in mining knowledge from social media data, specially Twitter, to detect and extract structured representations and summarize newsworthy events (McMinn & Jose, 2015; Katragadda, Benton, & Raghavan, 2017). However, this task is challenging for three main reasons:

1. Most of the tweets are not related to events (Java et al., 2007), thus locating the information of interest is a challenge requiring content classification.

2. The amount of documents present in social media streams exceeds by many orders of magnitude the number of documents produced in newswire (Petrovic, 2013), while at the same time there might be a high volume of redundant messages referring to the same issue or event. Many events are produced everyday and depending on their popularity and relevance, the number of tweets that discuss them may vary from a few to thousands. Thus, simple approaches that observe spikes in the volume of tweets will more likely fail to capture "small" events. On the other hand, since anyone is able to report an event, not all reported events are actually newsworthy (Van Canneyt et al., 2014) or even true at all. Hence, ED algorithms should take into account the real-time aspect of the Twitter stream and its changing nature.

3. The third challenge is related to the definition of "event" in social media streams, since the different definition efforts carried out within the Natural Language Processing (NLP) community do not seem to fully capture the peculiarities of social media content (Sprugnoli & Tonelli, 2017). Instead, more operational definitions taking into consideration the specific nature of such contents should be considered (Dou, Wang, Ribarsky, & Zhou, 2012).

Existing approaches to the task are either close-domain or open-domain. The former focus on detecting particular event types (e.g. earthquakes), and mainly use keywords related to the target events to retrieve event-related tweets from the stream (Sakaki et al., 2010). Although such approaches have been found effective

for detecting very specific event types, they are not general purpose since they require that one knows the keywords for all possible events in tweets, which is practically unfeasible.

General purpose or open-domain approaches, instead, attempt to detect events in tweets by creating clusters around event-related keywords (Parikh & Karlapalem, 2013), or NEs (McMinn & Jose, 2015) or monitor spikes in the volume of tweets (Nichols, Mahmud, & Drews, 2012). However, such approaches fail *i)* to capture events that do not generate high volume of tweets, for instance "Richard Bowes, victim of London riots, dies in hospital"; and *ii)* to distinguish between different events that involve the same NEs and keywords, as for instance "the shoot of Malala Yousafzai, the 14-year old Pakistani activist" and "her successful surgery" later on. Other approaches model the relationships between terms contained in the tweets relying on a graph representation (Katragadda, Virani, Benton, & Raghavan, 2016), and retain the nodes with the highest number of edges as event candidates. However, the main drawbacks of these approaches are that *i)* they generate highly dense graphs, and *ii)* trending terms not related to events may be considered as event candidates.

1.3 The Solution

In this thesis we focus on studying methods for detecting, classifying and tracking events on Twitter. Based on the previous discussions, we tackle the aforementioned challenges in a three different and complementary tasks.

Supervised Model Since most tweets are not related to events (Java et al., 2007), the task of separating tweets related to events from the rest of tweets is important. We propose a supervised model that, given a set of tweets as input returns only those related to events. In addition, we classify the event-related tweets into event categories according to the categories defined in the TDT community.

Unsupervised Model Once tweets are identified as related to events or belonging to an event categories, it is important to know what are the different events and their characteristics such as the type of the event, the location, and

the participants (people or organizations). We propose an unsupervised method that exploit local context of NEs mentioned in tweets to create event graphs; then based on graph theory we split the event graph into sub-graphs from which we extract the events by observing relationships between nodes. Also, exploiting the semantic class of NEs involved in the tweets, we extract meaningful properties of the events such that their geographical location or participants.

Semi Supervised Model Twitter is also used to discuss existing events. The ability to monitor the tweets that discuss a particular event is helpful to understand the reaction of the users to what is happening. For that reason, we propose a semi-supervised method, that given an event, monitors related tweets in order to build a timeline. We use in-domain vocabulary and Knowledge Base (KB) to extract related sub-events, again we exploit graph theory to determine relationships between the sub-events and NE mentions. This latter approach is applied to the soccer domain.

1.4 Thesis Contributions

The main contributions of this thesis are as follows:

Event classification In order to classify event-related tweets, we explore the impact of entity linking and of the generalization of NEs using DBpedia and YAGO ontologies. The approach we propose is applied to build a supervised classifier to separate event-related from non event-related tweets, as well as to associate to event-related tweets the event categories defined by the Topic Detection and Tracking community (TDT). We compare Naive Bayes (NB), Support Vector Machines (SVM) and Long Short-Term Memory (LSTM) classification algorithms, showing that NE linking and replacement improves classification performance and contributes to reducing overfitting, especially with Recurrent Neural Networks (RNN).

This work was partially published at: i) ESSLLI 2016 student session (Edouard, 2016), and i) CICLING 2017 international conference (Edouard, Cabrio, Tonelli, & Nhan, 2017c).

Event clustering Detecting which tweets describe a specific event and clustering them is one of the main challenging tasks related to Social Media currently addressed in the NLP community. Existing approaches have mainly focused on detecting spikes in clusters around specific keywords or Named Entities (NE). However, one of the main drawbacks of such approaches is the difficulty in understanding when the same keywords describe different events. As a second contribution, we propose a novel approach that exploits NE mentions in tweets and their local context to create a temporal event graph. Then, using simple graph theory techniques and a PageRank-like algorithm, we process the event graphs to detect clusters of tweets describing the same events. Additionally, exploiting semantic classes of the NE in ontologies, we detect meaningful attributes of the detected events such as the where they occur, their types as well as person or organizations involved. Experiments on two gold standard datasets show that our approach achieves state-of-the-art results both in terms of evaluation performances and the quality of the detected events.

This work has been accepted for publication in the Proceedings of the international conference RANLP2017(Edouard, Cabrio, Tonelli, & Nhan, 2017b).

Event tracking and summary The third contribution of this thesis is an approach to build a timeline with actions in a sports game based on tweets. We combine information provided by external knowledge bases to enrich the content of the tweets, and apply graph theory to model relations between actions and participants in a game. We demonstrate the validity of our approach using tweets collected during the EURO 2016 Championship and evaluate the output against live summaries produced by sports channels.

This work has been accepted for publication in the Proceedings of the international conference RANLP2017 (Edouard, Cabrio, Tonelli, & Nhan, 2017d), together with a demo at the same conference (Edouard, Cabrio, Tonelli, & Nhan, 2017a).

1.5 Outlines

The remainder of the dissertation is as follows:

Chapter 2 provides an overview on the main concepts addressed in the thesis. Since the thesis focuses on event detection, it introduces existing state-of-the-art methods for the task. It also discusses machine learning models for text classification and specifically for tweets.

Chapter 3 describes the first contribution of the thesis. It mainly describes the proposed method to separate tweets related to events from the rest of tweets as well as to classify them into event categories. It also describes extensive experiments we carried out to validate the method and provides detailed explanations of the obtained results.

Chapter 4 investigates the problem of identifying fine-grain events from tweets. It describes an unsupervised method for clustering tweets using the local context of NE mentioned to build event graphs and exploit graph cutting theory enriched with a PageRank-like algorithm to extract events from the event graphs. Also, it details the experiments on gold-standard data sets and comparison against state-of-the-art methods evaluated on the same task.

Chapter 5 describes a method for tracking sport events from tweets. The proposed method is based on an in-domain vocabulary and external KB to extract actions and participants involved in a given game as well as relationships between them. The approach is evaluated on a gold-standard dataset. It also describes a live demo, demonstrating the feasibility of the method.

Chapter 6 concludes the thesis by summarising the work and explaining how approaches presented in Chapters 3, 4 and 5 can be used in combination or as a standalone component for detecting and tracking events in tweets. It also discusses potential areas for future research.

Chapter 2

Background

This chapter is intended to provide a more in-depth understanding on the main concepts addressed throughout the thesis. These include concepts related both to social media and to the NLP techniques adopted in this work.

2.1 Twitter

Twitter is a micro-blogging site currently ranked 11^{th} world wide and 8^{th} in the United States according to Alexa traffic rankings [1]. Twitter allows public discussions about various topics, using short messages that are no longer than 140 characters, called tweets. At the time of writing, the Internet live stats website[2] reports that 400M+ tweets are sent every day. Since its creation in 2006, Twitter is the most popular microblogging platform that, with its followings/followers structure, allows users to quickly share information about their personal activities, report recent news and events or comment information shared by other users within their communities (Java et al., 2007).

Unlike other social network services such as Facebook, in which social relations are symmetric, the Twitter network is asymmetric and can be assimilated as a directed social network or follower network (Brzozowski & Romero, 2011). A user can follow any other user without requiring an approval or a reciprocal connection from the followed users.

[1] Alexa traffic rankings `http://www.alexa.com/siteinfo/twitter.com`
[2] Internet live stats `http://www.internetlivestats.com/`

Twitter has evolved over time and adopted suggestions originally proposed by users to make the platform more flexible. It currently provides different ways for users to converse and interact by referencing each other in a well-defined markup vocabulary. These include retweeting, user mentions, hyperlinks and hashtags.

Retweets

Twitter users often "retweet" other user's status updates to spread a message to their own followers. The original and unofficial convention for retweeting was to copy and paste the original content of the tweet and prefix it with "RT @username:", optionally adding the retweeter's own comment beforehand. Twitter has since introduced an official method of retweeting, where users simply press a button to perform a re-tweet. These retweets appear with a special icon beside them and are also annotated as retweets in the API output. According to (Boyd, Golder, & Lotan, 2010), retweeting may have various social motivations such as entering a specific audience, commenting someone's tweet or publicly agreeing with the message published by another user. In previous studies, retweets have been used to measure the popularity of a tweet and users (Kwak et al., 2010).

User mentions

Tweets are generally conveyed to a public audience, typically the followers of the user who creates them. Users may address a status to a specific user by referring their username prefixed by a @ symbol. Although such messages can be viewed by all the followers, their contents are generally addressed to the target user. Tweets that contain a user mention are considered as a reply or communication directed to that user (Honey & Herring, 2009). On the other hand, Twitter makes a hyperlink back to that user's personal timeline, allowing an informal direct communication.

Hyperlinks

Sharing links is a central practice in Twitter, allowing users to link their posts to external web pages, which usually contain additional information about the content of the tweets. Since tweets are limited to 140 characters, hyperlinks can be considered as a mechanism to extend the content of the tweets by liking them

to external resources. Because URLs are typically long, people usually short-hand the URLs using "URL shorteners" services (e.g., http://bit.ly).

Hashtags

Hashtags are commonly used by Twitter users to bookmark the content of tweets, participate in a community graph concentrating on the same topic (L. Yang, Sun, Zhang, & Mei, 2012) or to link their tweets to ongoing discussions. They can be of various forms including single words (*#winner*) or a combination of multiple words (*#ripAmyWinehouse*). Previous studies have shown that hashtags may contain useful information that can be used to improve various NLP tasks on tweets such as sentiment analysis (Wang, Wei, Liu, Zhou, & Zhang, 2011) or Named Entity Recognition.

Twitter as Information Source

In recent years, Twitter has emerged as one of the most popular information sources for practical applications and academic research. There are numerous examples of practical applications of Twitter data, ranging from stock forecasting (Arias, Arratia, & Xuriguera, 2013), through real-time event detection (Sakaki et al., 2010), trend analysis (Mathioudakis & Koudas, 2010), crisis management (Abel et al., 2012). Also, Twitter data has been found useful for public safety applications (Ritterman, Osborne, & Klein, 2009).

In order to help researchers, practitioners and organizations in exploiting its data, Twitter offers a public API[3] that facilitates its integration in external applications. The Twitter API is available as Search or Streaming APIs allowing the collection of Twitter data using different types of queries, including keywords and user profiles.

2.2 Tweet Classification

Classification is a supervised data mining technique that involves assigning a label to a set of unlabeled input objects. Based on the number of classes present, classification tasks can be broadly divided into two types:

[3] `https://dev.twitter.com/rest/public`

1. Binary Classification: The classification model is trained to classify input documents into two classes. For instance, a model can be trained to classify tweets as related or not related to events.

2. Multi-class classification: The model is trained to classify input documents into multiple classes. Example of such classification tasks are news classification as Politics, Sports or Economy.

Typically, the classes are handpicked by domain experts regarding the target applications. For instance, for detecting event-related tweets , one may define "event" and "non event" as labels (Ilina, Hauff, Celik, Abel, & Houben, 2012) while for classifying tweets related to events as event categories, one may define finer-grain categories such as "politic", "enocomy" or "sports" (McMinn, Moshfeghi, & Jose, 2013).

More formally, let $D = (d_i)_{i\in[n]}$ and $C = (c_i)_{i\in[k]}$, denoting the class of documents in D. Text classification is the task of learning a function $f : D \to C$ that maps every document $d_i \in D$ to its corresponding class $c_i \in C$.

For this purpose, the input documents (or learning corpus) are divided into training and testing set. The training set contains documents that are already labeled from which a classification algorithm will train (or teach) the model. On the other hand, documents in the testing set are unlabelled; thus the model is used to predict the class label for the test data accurately.

Text classification is generally performed in four main stages, namely : Text representation, Classification algorithms, and Evaluation.

2.2.1 Text Representation

Before any classification task, one of the most fundamental tasks that needs to be accomplished is that of document representation and feature selection. In the particular case of tweet classification, the tweets are merely string of texts. Hence, there is a need to represent them in a structured manner. There exists several techniques for representing the documents including feature vectors such as bag-of-words (BoW).

Bag of words

The bag-of-words is perhaps the simplest feature vector model for text classification. In this technique, the document is broken down as a sequence of terms. Formally, a document d is represented as a vector $X = (x_i)_{i \in [n]}$, where n is the number of features and x_i is either a number or categorical value quantifying some evidence pertaining to the document. In in very basic usage, the text is represented as a vector of tokens (a.k.a features), such tokens may be of various forms including unigrams (or words), bigrams (pair of words) or n-grams (a combination of $n > 2$ words). The input document D is represented as $n \times m - matrix$, where n is the number of documents in the training set and m, the number of distinct features in documents. The combination of features extracted from the input documents is called a feature vector. Since feature vectors can be very large, text preprocessing techniques are applied to prune the feature vector and thus reduce its dimensionality. This includes, for example, stop words removal or word stemming. Alternate techniques include weighing the features by using a TF-IDF model (Manning, Raghavan, Schütze, et al., 2008).

Despite the fact that bag-of-words representation is widely used, it has some drawbacks that can negatively impact the performance of ML models, especially text classification. A first problem is related to the dimension of the feature space, which is equal to the size of the dictionary. As a consequence, this may lead easily to the curse of dimensionality. Another drawback is that it does not take into account the possible semantic links that exist between words, since it is mainly based on the presence of terms in the documents. It will for example make a high distinction between the sentences "The president of the United States is visiting Europe" and "Donald Trump is visiting Europe", while for a human-being these sentences are related even the terms are slightly different. In the recent years, alternatives to incorporate semantic in vector representation model have been proposed including for instance the Bag-of-entities models (BoE) (Bordes, Usunier, Chopra, & Weston, 2015) or Word Embeddings (WE) (X. Yang, Macdonald, & Ounis, 2016). We focus in the following subsection on word embeddings, since they will be used to represent tweets in the experiments presented in this thesis.

Word Embeddings

Word embeddings (WE) are dense vectors which represent word contexts as low-dimensional weighted average vectors, originally coined by (Bengio, Ducharme, Vincent, & Jauvin, 2003). (Collobert & Weston, 2008) introduced first a unified architecture, in which they establish WE as a highly effective tool in different NLP tasks.

A word embedding $W : word \rightarrow \mathbb{R}^n$ is a parametrized function mapping words in a continuous vector space, where semantically similar words are mapped to nearby points. Thus, in a WE model, each word is represented with real-valued vectors and words that appear in similar contexts have closer vectors in the embedding space. Moreover, word vectors enable to capture many linguistic and semantic regularities, and because of the compositionality of the space, we can conduct vector arithmetic operations, such as $W(\text{“}women\text{”}) - W(\text{“}men\text{”}) \simeq W(\text{“}aunt\text{”}) - W(\text{“}uncle\text{”})$ or $W(\text{“}France\text{”}) - W(\text{“}Paris\text{”}) \simeq W(\text{“}Italy\text{”}) - W(\text{“}Rome\text{”})$ relationships.

Word embeddings are usually trained on very large datasets using different algorithms. The word2vec (Mikolov, Chen, Corrado, & Dean, 2013) toolkit proposes two variants for training embedding models: Continuous Bag of Words (CBOW), which uses the context to predict a target word, and Skip-gram, which predicts the context of each word.

In recent studies, WE vectors have shown a good generalisation power for representing features in many NLP tasks such as Named Entity Recognition (S. Miller, Guinness, & Zamanian, 2004; Sienčnik, 2015), dependency parsing (Bansal, Gimpel, & Livescu, 2014), machine translation (Vaswani, Zhao, Fossum, & Chiang, 2013) and text classification (X. Yang et al., 2016), as they provide a more nuanced representation of words than a simple indicator vector into a dictionary. Previous works have shown that WE-based features outperform traditional BOW models in different machine learning tasks (Forgues, Pineau, Larchevêque, & Tremblay, 2014; Jin, Zhang, Chen, & Xia, 2016).

2.3 The Semantic Web and Linked Open Data

In 2006, Sir Tim Berners-Lee introduced the "Semantic Web" as a way to facilitate "automatic" data sharing on the WWW (Berners-Lee, 2006). The Semantic Web (SW) is based on two main principles: i) a common framework that allows data to be shared and reused across application, enterprise, and community boundaries; and ii) a common language for recording how the data relates to real world objects. From the SW point of view, resources on the WWW are represented as semantic resources rather than raw texts. This semantic representation is based on concepts in ontologies.

(Gruber, 1993) defines an **ontology** as "an explicit representation of a conceptualization". From this definition, one should retain two main aspects of an ontology: i) a conceptualization, which describes the meanings of terms in a domain through the description of entities (or concepts); and ii) an explicit representation, which means that relations that link entities and their properties are clearly defined. In fact, once an ontology is created and accepted by a community, it can be used to describe resources that could be exploit by different applications. Furthermore, thanks to the conceptualization, applications can use the ontologies to perform inference or reasoning on the resources. Also, ontologies are used to link resources on the Web (Laublet, Reynaud, & Charlet, 2002).

In the SW community, the **Resource Description Framework** (RDF) is adopted as the standard representation of semantic resources on the Web. The main idea of RDF is to describe the semantics of the data by expressing simple statements of the form Subject – Predicate – Object, called triplet, which can also be considered as simple sentences involving a subject, a verb, and a complement. Note that when several statements are grouped together, the objects of some statements may also act as a subject of other statements which lead to a graph representation (Gandon, Corby, & Faron-Zucker, 2012). Indeed, a set of RDF statements can be considered as a directed labelled graph where Subjects and Objects define nodes and Predicates define labelled directed edges.

The RDF standard describes resources using a **Unique Resource Identifier** (URI), which enables the description of resources unambiguously, where a resource can be about almost everything (e.g. person, books, movies, geographical places, ...). In fact, URIs might be abstract or concrete, and they

are used to locate resources in an unambiguous way over the Internet. For instance "Paris" is an ambiguous name for various geographic places including the capital of France or a city in New York. But using URIs, each of such geographical places can be represented unambiguously with their unique resource ID, `http://dbpedia.org/page/Paris` for Paris the capital of France or `http://dbpedia.org/page/Paris,_New_York` for Paris the city in New York.

On the other hand, data published following the Web semantic principles can be easily queried using **SPARQL**[4]. SPARQL is a standardized query language for RDF data, which offers developers and end users a way to write and to consume the results of queries across this wide range of information. Used with a common protocol, usually HTTP, applications can access and combine information from linked open data (LOD) sets across the Web.

2.3.1 Linked Open Data Knowledge Bases

Following the SW principles, several linked data knowledge bases have been created and made publicly available. The Linking Open Data community project continuously monitors available data sets and fosters their publication in compliance with the Linked Open Data principles. As in February 2017, the latest cloud diagram (see 2.1) lists 1139 interlinked datasets. The cloud contains data sets with very diverse topics such as life sciences (light red circles), geographical data (blue circles), social networks (grey circles) or cross-domain sources such as Freebase[5] or DBpedia[6].

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web as a semantic knowledge graph (Auer et al., 2007). DBpedia is interlinked with various other data sets (as shown in Figure 2.1), such as Geonames (Wick, 2006) for geographical resources or DBLP Bibliography for scientific publications[7]. Resources in DBpedia are described according to concepts in the DBpedia ontology or various other ontologies such as **YAGO** (Mahdisoltani, Biega, & Suchanek, 2014) and FOAF (Brickley & Miller, 2007). The DBpedia Ontology is a cross-domain ontology,

[4]The SPARQL Query Language `https://www.w3.org/TR/sparql11-overview/`
[5]Freebase: `http://freebase.com/`
[6]DBpedia: `http://wiki.dbpedia.org/`
[7]DBLP Bibliography: `http://dblp.uni-trier.de/`

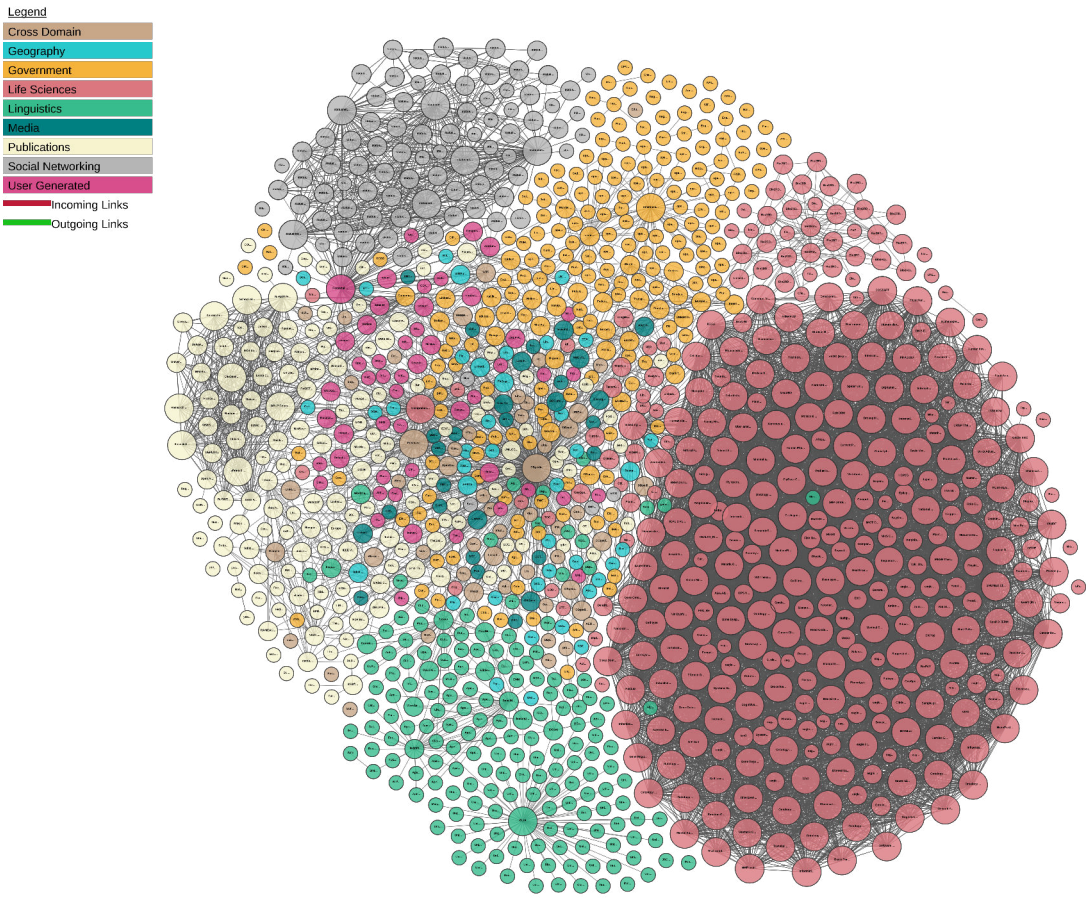

Figure 2.1: Linked Open Data cloud diagram, generated on February 20, 2017.

which has been manually created based on the most commonly used infoboxes within Wikipedia. The ontology currently covers 685 classes forming a subsumption hierarchy, which are described by 2,795 different properties. In its latest version, the ontology is modeled as a directed-acyclic graph instead of a tree. Classes in the ontology may have multiple superclasses linked by an "is-a" relation. YAGO is a larger ontology containing concepts or categories automatically extracted from Wikipedia infoboxes, WordNet (G. A. Miller, 1995) and Geonames (Wick, 2006), which contains approximately 350.000 classes. In the linking experiments presented in the remainder of this thesis, we will focus on DBpedia and Yago.

2.4 Named Entity Recognition

Named Entity Recognition(NER) is a sub-task of Information Extraction aimed at finding "atomic elements in text" that belong to a set of predefined categories. Listing 2.1 is an example from Mikheev (1999), marked up with four entity types: Date, Person, Organization, and Location.

```
On <Date>Jan 13th</Date>, <Person>John Briggs</Person> contacted <
    Organisation> Stockbrockers Inc</Organisation> in <Location>New
    York</Location> and instructed them to sell all his shares in <
    Organisation>Acme</Organisation>.
```

Listing 2.1: Example of Named Entities in unstructured text

NER encompasses two main tasks: i) The identification of names [8] such as "John Briggs", "Stockbrockers Inc", "New York" and ii) the classification of these names into a set of predefined types (or NE types), such as "Person", "Organization" and "Location". Generally, NER task tries to identify 3 types of concepts (Poibeau & Kosseim, 2001): i) Proper Names (PN) such as person, organization, and location, ii) temporal expressions and dates, and iii) numerical expressions such as number, currency, money and percentage. In addition, in-domain systems can define fine grained entity types such as names of drugs (Zhai et al., 2013), biomedical entities (Settles, 2004; Tang, Cao, Wu, Jiang, & Xu, 2013) or brands names (Bick, 2004; Hallili, 2014).

Existing NER tools are implemented with several approaches, which are broadly classified into three main categories including rule-based models, statistical models or machine learning and hybrid models (Nadeau, 2007).

Rule-based systems are mainly based on handcrafted rules to detect mentions of NE in texts. Generally, such rules are provided as language patterns using grammatical (e.g. part of speech), syntactic (e.g. word precedence) and orthographic features (e.g. capitalization) in combination with dictionaries (Budi & Bressan, 2003). The downside of this type of approach is that they are unable to detect entities that are not in the dictionary (Sampson, 1989), specially emergent PNs (Coates-Stephens, 1992) and ambiguous NEs (Nadeau & Sekine, 2007). The use of morphological, syntactic and semantic constraints has been exploited as an alternative to improve the performance of dictionary-based approaches on

[8] Usually referred as Named Entities.

new entities (Coates-Stephens, 1992). However, the varied and complex constructions in which PN can occur still present difficulties to a system processing it. Methods based on handcrafted rules involve designing and implementing lexical-syntactic extraction patterns. They are efficient for domains where there is certain formalism in the construction of terminology, for instance the biology domain (Chiticariu, Krishnamurthy, Li, Reiss, & Vaithyanathan, 2010). However, the major limitation of these systems is that they require significant domain knowledge from the developers. Besides, they are often expensive to build and maintain and are not transferable across domains. Consequently these approaches suffer from limited or no portability (Chiticariu et al., 2010).

Supervised Learning (SL) methods typically consist in training algorithms to learn discriminative features from large annotated corpora and apply them to annotate unseen text. Typically, SL systems are trained to study features of positive and negative examples of NE over a collection of annotated samples. SL methods exploit various learning techniques including Hidden Markov Models (HMM) (Bikel, Miller, Schwartz, & Weischedel, 1997), Decision Trees (Sekine et al., 1998), Maximum Entropy Models (Borthwick, Sterling, Agichtein, & Grishman, 1998), Support Vector Machines (SVM) (Asahara & Matsumoto, 2003), Maximum Entropy (ME) (Curran & Clark, 2003), Conditional Random Fields (CRF) (Finkel, Grenager, & Manning, 2005; McCallum & Li, 2003). The main drawback with SL methods is that they usually require a remarkable effort to provide large training data. Besides, they hardly adapt to new domains.

Semi-Supervised Learning (SSL) exploits both labelled and un-labelled data and requires a small degree of supervision to start the learning process. It proves useful when labelled data are scarce and hard to construct, while un-labelled data is abundant and easy to access. Generally, SSL methods use a bootstrapping-like technique to strengthen the models, starting from small sets of seed data (Nadeau & Sekine, 2007). In general, SL methods outperfosm SSL ones, but are more expensive to build in terms of resources.

In contrast to the previous methods, where sample of annotated documents or dictionary of terms are required, **Unsupervised Learning** (UL) methods typically create clusters that gather similar entities without supervision. UL methods have been found as an alternative when labelled data is missing during the learning process. In NLP tasks, such approaches can take advantage of lin-

1	#AMY IS DEAD, CAN'T BELIEVE IT :(
2	amy winehouse passed away, she was so young

Table 2.1: Example of Named Entities in tweets.

guistic resources to extract relations between the target entities. For instance, Alfonseca and Manandhar (n.d.) used Wordnet synsets in the process of labelling an input word with an appropriate NE.

2.4.1 Named Entity Recognition in Tweets

Past research in NER has been mostly interested in detecting NEs from standard texts such as newspapers. However, when it comes to tweets, available NE recognition tools face new challenges due to the specific nature of tweets, such as the presence of misspelled words or jargon (Han & Baldwin, 2011), and the fact that tweets contain very little contextual information (Kinsella, Passant, & Breslin, 2010). For instance, Derczynski et al. (2015) demonstrated that the performance of various state-of-the-art NER software (e.g., Stanford NER and ANNIE) is typically lower than 50% F-score for tweets.

In addition to non-standard words, informal writing style further reduce the accuracy of NER tools on tweets (Derczynski, Ritter, Clark, & Bontcheva, 2013). As an example, capitalization is a conventional feature for both machine learning and rule-based methods. However, in tweets capitalization is used in a very unconventional manner: Twitter users rely on capitalization for emphasis rather than identifying proper names (Han, Yepes, MacKinlay, & Chen, 2014). Table 2.1 shows two tweets related to the death of "Amy Winehouse". Conventional NLP tools would fail to extract NE mentions from these tweets due to an improper usage of capitalization in both tweets.

In recent years, several approaches for detecting NE in tweets have been investigated, resulting in various NER tools specially built for tweets. (Rizzo & Troncy, 2011) propose NERD as a a misture of several NER tools including : AlchemyAPI, DBpedia Spotlight, Extractiv, Lupedia, OpenCalais, Saplo, SemiTags, Wikimeta, Yahoo! Content Analysis, and Zemanta, currently focusing on results that include named entity recognition and resolution, and sense tagging. The main idea behind the NERD service is to combine the output of various NER

tools in order to improve the detection of NE in tweets. In a recent experimental study, (Derczynski et al., 2013) conducted an empirical analysis of named entity recognition and disambiguation, investigating the robustness of a number of NER tools on noisy texts, specifically tweets. Among the different tools, NERD-ML has been found to outperform the other tools, specifically for the NER task.

2.5 Event Processing in Tweets

In recent years, several approaches to build event-related services applied to Twitter data have been investigated. The task is of practical significance since it has been shown that Twitter reacts to news events faster compared with traditional media (Hermida, 2010). The capability to understand and analyze the stream of messages on Twitter is an effective way to monitor what people think, what trending topics are emerging, and which main events are affecting people's lives. For this reason, several automated ways to track and categorize *events* on Twitter have been proposed in the literature. In this section, we summarize some challenges related to the detection of events from tweets and the state-of-the-art techniques for the task.

2.5.1 Event definition and detection

While the issue of defining an event in text is still open (Sprugnoli & Tonelli, 2017), researchers working on tweets have generally agreed on the proposal made by (Dou et al., 2012), who define an event in the context of social media as *"An occurrence causing change in the volume of text data that discusses the associated topic at a specific time. This occurrence is characterized by topic and time, and often associated with entities such as people and location"*. This definition, which we also adopt throughout the thesis, highlights a strong connection between events in the context of social media and the NEs involved in such events (corresponding to events' participants, typically persons, organizations and locations).

In order to automatically identify events in tweets, approaches usually start by collecting data from the Twitter Streaming API and the process them using text preprocessing techniques including, Part-of-Speech (POS) tagging, stop-word re-

moval, NER, semantic enrichment.

As for event identification, a recent survey (Atefeh & Khreich, 2015) classifies existing works into two main categories namely *close domain* and *open domain*. The first ones are mainly focused on detecting very specific event types such as earthquakes (Sakaki et al., 2010), influenza epidemics (Ritterman et al., 2009). The second ones instead monitor the Twitter stream in order to detect new events without prior knowledge on the target events (Ritter, Clark, Etzioni, et al., 2011).

2.5.2 Event Tracking

Event Tracking (ET) is a sub task of event detection which aims at monitoring tweets related to existing events over time. Depending on the type of event, ET systems might identify also sub-events related to a target event such as actions in a soccer game; others might focus on monitoring how a particular event affects people's lives, for instance tweets reacting to a natural disaster. Since single tweets that report an event are usually sparse and do not necessarily provide a good summary of the event, instead of displaying a list of tweets to the end users, ET systems attempt to extract a summary from the set of tweets related to an event.

Early systems performing this task required that end users provide a list of keywords related to the event to track (Sharifi, Hutton, & Kalita, 2010a; Marcus et al., 2011a). Other, more recent approaches, instead, do not require keywords as input, since they automatically group tweets related to the same events into clusters before trying to summarize them (Chakrabarti & Punera, 2011). This research line is the one we are pursuing in this thesis.

Most of the existing ET systems focus on monitoring sport games on Twitter, particularly soccer games (Nichols et al., 2012; Kubo, Sasano, Takamura, & Okumura, 2013; Jai-Andaloussi, El Mourabit, Madrane, Chaouni, & Sekkaki, 2015). In order to compare our approach with existing ones, we also perform part of our event processing experiments related to soccer games.

Chapter 3

Event-Based Classification of Tweets*

Goal of this chapter is to investigate the problem of detecting and classifying event-related tweets. To address this issue, we present our supervised approach for detecting event related tweets and to classify them into event categories.

3.1 Introduction

Detecting which tweets are related to events and classifying them into categories is a challenging task due to the peculiarities of Twitter language discussed in the previous Chapter, and to the lack of contextual information. Following the definition of event provided by (Dou et al., 2012) that highlights a strong connection between events in the context of social media and the NEs involved in such events (corresponding to events' participants, typically persons, organizations and locations), we propose to face the above mentioned challenge by taking advantage of the information that can be automatically acquired from external knowledge bases.

Despite their importance, however, using entity mentions as features in a supervised setting to classify event-related tweets does not generalize well, and may affect classification performance across different event categories. We investigate this issue in the present chapter, and we analyze the effect of replacing entity

*Most of the work described in this chapter has been published in (Edouard et al., 2017c).

mentions in tweets with specific or generic categories automatically extracted from external knowledge bases. Specifically, we compare the classification performance linking named entities to DBpedia (Lehmann et al., 2014) and YAGO (Mahdisoltani et al., 2014) in order to classify tweets in different event categories defined by the TDT community.

The main contributions of this chapter are as follows : *i)* we propose and evaluate an approach for detecting tweets related to events as well as classifying them into event categories; *ii)* we show that supervised models can achieve good generalization capabilities through semantic linking; *iii)* we evaluate how generic and specific types of NE affect the output of supervised models.

In the following, Section 3.2 reports on relevant related literature, Section 3.3 describes the proposed approach to detect and classify event-related tweets, while Section 3.4 presents the experimental settings and discusses the obtained results.

3.2 Related Work

In recent years, several approaches to build event-related services applied to social media have been investigated. Existing approaches to event detection on Twitter have been classified into two main categories: *closed domain* and *open domain* (Atefeh & Khreich, 2015). The first ones are mainly focused on detecting specific fine-grained event types (e.g. earthquakes, influenza epidemics), while the second ones do not target a specific event type, but try to detect real-world events belonging to various categories as they happen. Works in the closed domain scenario mostly rely on keywords to extract event-related messages from Twitter (Sakaki et al., 2010), recognise event patterns (Popescu, Pennacchiotti, & Paranjpe, 2011) or define labels for training classifiers (Anantharam, Barnaghi, Thirunarayan, & Sheth, 2014).

The open domain scenario is more challenging. Its first step is the separation between event-related and non event-related tweets (Ilina et al., 2012), a task that we also tackle in the present paper. (Becker, Naaman, & Gravano, 2011) apply an online clustering and filtering framework to distinguish between messages about real-life events and non-events. The framework clusters streaming tweets using their similarity with existing clusters. In this framework, a set of event features

including temporal, social, topical and Twitter-centric features is defined.

In (Ritter, Etzioni, Clark, et al., 2012), events are modelled using a 4-tuple representation including NEs, temporal expressions, event phrases and event types. The system recognizes event triggers as a sequence labelling task using Conditional Random Field. In addition, the authors measure the association strength between entities and calendar dates, which is used as key feature to separate event and non event-related tweets. Nevertheless, this assumption restricts the approach to tweets that explicitly contain temporal expressions.

More recently, researches have explored the usage of external knowledge sources to enrich the content of tweets. Genc et al. (Genc, Sakamoto, & Nickerson, 2011) introduced a Wikipedia-based classification technique to construct a latent semantic model that maps tweets to their most similar articles on Wikipedia. Similarly (Song, Wang, Wang, Li, & Chen, 2011) proposed a probabilistic framework to map terms in tweets to concept in the Probase[1] knowledge base.

Cano et al. (Cano, Varga, Rowe, Ciravegna, & He, 2013) exploit information obtained from different knowledge sources for tweet classification and evaluate their approach in the violence and emergency response domains. The evaluation shows that extracting semantic features from external knowledge sources outperform state-of-the-art techniques such as bag of words, bag of entities or part of speech features.

In a very recent work, (Nigel & Rachel, 2017) exploit orthographic features (e.g. emoticons, URL), NE (e.g. Person, Location), syntactic features (e.g. POS tags) and frequency features (e.g. tf-idf) to train a supervised model for tweets classification to maintain portability across different datasets. They found that the combination of all such features contribute in improving the performance of the model compared to traditional bag-of-words unigram baseline when training and test instances coming from different dataset. While their approach has been evaluated on the task of separating tweets as related and not related to events, we propose a pipeline where the event related tweets are further classified into event categories. In addition, we train and test our approach on data coming from dataset collected over two distinct periods.

In our work, we exploit information acquired from external knowledge bases to enrich NEs mentioned in the tweets with additional information. Then en-

[1] `https://www.microsoft.com/en-us/research/project/probase/`

riched content is used to extract features for building word-embedding vectors which serve as feature model for training supervised models in the aim of identifying tweets related to events as well as classifying event-related into fine-grained event categories. Furthermore, while previous approaches have been evaluated on datasets collected during a short time period (Becker et al., 2011; Cano et al., 2013; Petrović, Osborne, & Lavrenko, 2010), we evaluate ours on two datasets collected over two different periods and covering different event types.

3.3 Detecting and Classifying Event Tweets

This section describes the approach we propose to identify event-related tweets and classify them into categories. Given a set of tweets as input, the main steps of our framework include a **Preprocessing** step, to clean the input data, **Named Entity replacement**, based on NE recognition and linking, and **Tweet classification**. The goal of the last step is to classify the input tweets related to events into categories such as *Sports* and *Politics*. We propose two framework configurations: the first one carries out two steps in a row, in which the input tweets are first classified into event-related and not event-related, and the event-related ones feed the second classifier labelling them with different categories. The second configuration relies on a one-step solution, in which tweets that are not related to events are considered as an additional category together with the events categories in a multi-class classification step. Figure 3.1 shows the proposed framework architecture, detailed in the following sections.

3.3.1 Tweet Pre-processing

The first step of the pipeline consists in cleaning the input data in order to remove noise. Due to the peculiarity of being at most 140-characters long, tweets contain medium-specific expressions and abbreviations, that are not present in standard texts. We make the assumption that emoticons, user mentions, URLs and re-tweets are not discriminating features for our event classification purpose, therefore we remove them from the contents of the tweets. Tweets in input are tokenized with a Twitter-specific POS tagger (Owoputi et al., 2013), that extends (Gimpel et al., 2011)'s rule-based tokenizer for informal texts. In addition

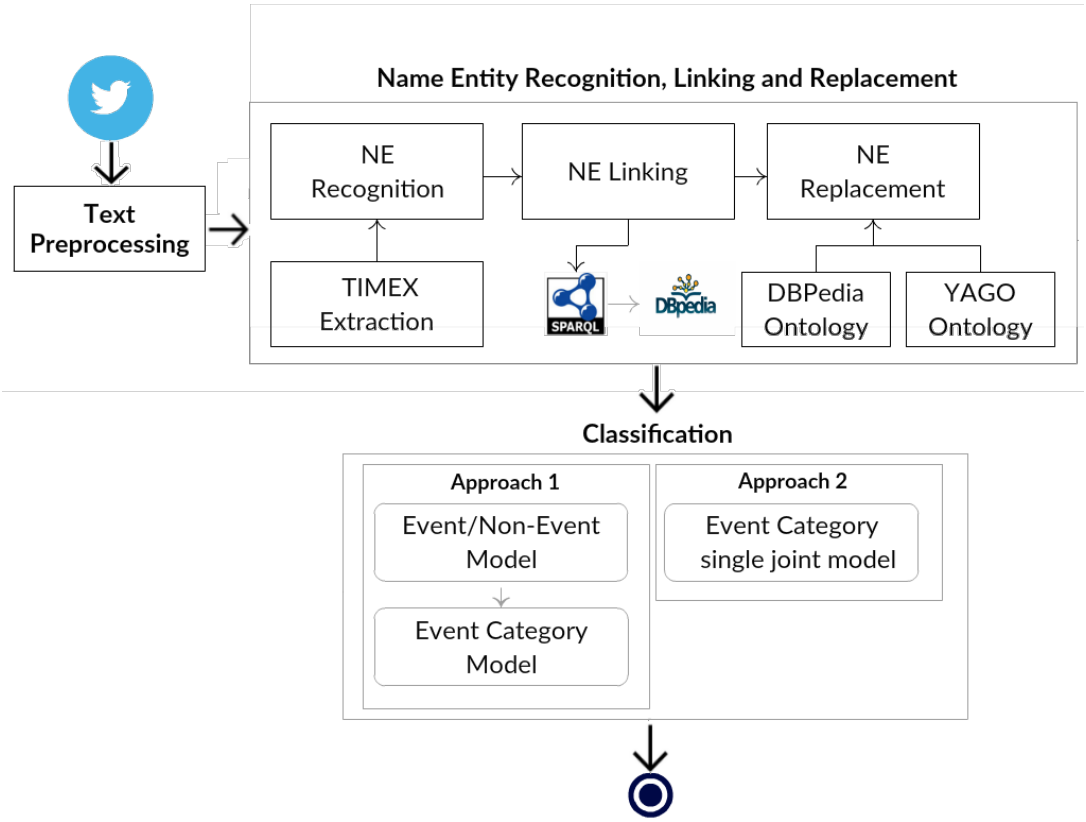

Figure 3.1: The event detection pipeline (tweets are the input source). Rectangles are conceptual stages in the pipeline with data flowing in the order of the arrows. The solid circle at the bottom represents the output, i.e. tweets related to events and classified into categories.

to common POS tags, the tokenizer also defines additional Twitter ad-hoc PoS tags such as URL, hashtag, user mention, emoticons or re-tweets.

Hashtag Segmentation As discussed in Chapter 2, hashtags are commonly used by Twitter users to bookmark the content of tweets, participate in a community graph concentrating on the same topic (L. Yang et al., 2012) or to link their tweets to ongoing discussions. Previous studies have shown that hashtags may contain useful information that can be used to improve various NLP tasks on tweets such as sentiment analysis (Wang et al., 2011) or Named Entity Recognition. We process the hashtags in the input tweets applying hand-crafted rules. A first set of rules aims at breaking hashtags into meaningful words, mainly based on the presence of capital letters commonly used by Twitter users while creating

hashtags that contain several words (e.g. #RIPAmyWinehouse becomes "RIP Amy Winehouse"; #presidentialElection becomes 'presidential election"). However, in most cases hashtags do not follow any particular predefined pattern (e.g. "#presidentialdebate"). For such cases, we define a dictionary-based method (relying on WordNet (Fellbaum, 1998)) for hashtag segmentation. Algorithm 1 describes our approach, that consists in three steps: *i)* the "#" symbol is firstly removed and the resulting word is searched in the dictionary; if found, such word is returned; *ii)* otherwise, the algorithm checks if the input tag satisfies a predefined pattern; if so, the identified words are returned; *iii)* otherwise, the hashtag is broken into random terms, starting with the k first letters until a valid term is found in the dictionary. When a term is found, the algorithm checks if also the remaining string is in the dictionary, otherwise it performs again step iii) with the remaining term as input parameter. At the end of this process, the longest set of candidates is retained. For instance, for the hashtag "#presidentialdebate" the algorithm outputs "preside", "president" and "presidential,debate" as candidates: among them, "presidential" and "debate" as retained as replacement terms.

Tweet Correction Since tweets often contain misspelled terms, we apply the Symmetric Delete Spelling Correction algorithm (SymSpell)[2] to match misspelled tokens in tweets to their correct spelling using a dictionnary language (SymSpell is six order of magnitude faster than (Norvig, 2009)'s spelling corrector). A token is considered as misspelled if it is an out-of-vocabulary term, where the vocabulary is built from the Wordnet's synsets (Fellbaum, 1998). We adapt the original version of Symspell to pre-generate all terms in the language dictionary with an edit distance ≤ 2, which we use as support vocabulary for the Symspell algorithm. During the correction phase, SymSpell generates all edit variations of the input token and tries to match them with those in the support vocabulary; if an edit variation is found, the misspelled term in the tweet is replaced with the term suggested by SymSpell with the highest probability. For example, $delete(president, 1) = \{'pesident', 'prsident', 'preident', ...\}$, where 1 is the edit distance. Note that if no matching term in the dictionary is found, we do not remove the input token, given that it could correspond to a NE. For instance, "fb" is often used for "Facebook" or "mbp" for "Mac Book Pro" (Ritter et al.,

[2]https://github.com/wolfgarbe/symspell

Algorithm 1 Algorithm to process hashtags.

```
1: function TAG_PROCESSING(h)                                  ▷ h is a hashtag
2:     Remove "#" h if present
3:     Let D be the language dictionary
4:     if hashtag ∈ D then
5:         return h
6:     end if
7:     parts ← IS_REGEX()h
8:     if parts not ∅ then return parts
9:     end if
10:    result ← ∅
11:    n ← length(h)                                  ▷ Number of characters in h
12:    for k ∈ {2, ..., n} do
13:        term1 ← h[0, k]
14:        term2 ← h[k + 1, n]
15:        temp ← ∅
16:        if term1 ∈ D then
17:            Assign term1 to temp
18:        end if
19:        if term2 ∈ D then
20:            Assign TAG_PROCESSING(term2) to temp
21:        end if
22:        if temp not ∅ then
23:            Assign temp to result
24:        end if
25:    end for
26:    Sort result by the number of candidates and the length of the terms
27:    return result[0] or h
28: end function
```

2011).

3.3.2 NE Linking and Replacement

We first remove Twitter-specific features such as user mentions, emoticons and URLs, identified with a Twitter-specific Part-Of-Speech tagger (Owoputi et al., 2013). Identical re-tweets are removed, as well as hashtags, if they cannot be mapped to a NE.

Then, we run the Entity Replacement module in our pipeline (Figure 3.1, Named Entity Recognition step) by calling the API of NERD-ML (Van Erp,

Rizzo, & Troncy, 2013) to first recognise NEs in tweets, and then to link them to DBpedia[3] for the entity linking task.

As a comparison, we also rely on the YAGO ontology (Mahdisoltani et al., 2014), to assess how the two resources impact on our task. Since resources in DBpedia are also mapped to concepts in YAGO, linking tweets' NEs to this resource starting from the DBpedia categories labeled with NERD-ML is pretty straightforward. We rely on NERD-ML because it proved to outperform other NER systems on Twitter data[4].

As for the knowledge bases, we focus on YAGO and DBpedia because they are among the most widely used general-purpose knowledge bases in the Semantic Web and NLP community, each having its own peculiarities. DBpedia relies on an ontology with around 685 classes and 2,795 properties, which have been manually mapped to Wikipedia infobox types. YAGO, instead, contains approximately 350,000 classes and is based on a much larger and deeper hierarchy derived partly from Wikipedia categories (the lower levels) and WordNet (the most general layers of the hierarchy) in a semi-automated fashion. More recently, it has also been enriched with concepts in GeoNames. This reflects the different coverage of the two resources: while DBpedia covers only the Wikipedia pages with an infobox that was mapped to its ontology, YAGO includes all Wikipedia pages having at least one category, thus it has a broader coverage. Such difference emerges also in our experiments, since we found that DBpedia URIs cover approximately 56% of the NEs detected in our tweet corpus, while YAGO accounts for 62% of the entities.

The NE linking submodule (Figure 3.1) relies on the DBpedia URI provided by NERD-ML to retrieve the categories to which an entity belongs in the order in which they appear in the hierarchy of the considered ontology (i.e DBpedia or YAGO). For example, for the geographical entity "New York", though the sparql

[3]When using DBpedia as external KB for entity linking, our approach is not limited to proper names (i.e. persons, location or organisations) but considers any term that has an associated URI in DBpedia (e.g. Nobel Prize).

[4]A recent evaluation study on NE recognition and linking (Derczynski et al., 2015) tools, proves that NERD-ML is among the best performing tool both for Named Entity Recognition (NER) and linking (NEL) in tweets, outperforming systems such as Stanford NER (Manning et al., 2014), DBpedia Spotlight (Mendes, Jakob, García-Silva, & Bizer, 2011) or Ritter T-NER (Ritter et al., 2011). NERD combines several extractors including AlchemyAPI, TextRazor, Zemanta or OpenCalais in order to detect mentions of NE in tweets and map them to concepts in the NERD ontology (Rizzo & Troncy, 2011).

query reported in Listing 3.1 we retrieve from DBpedia ontology the following categories:

Administrative Region → *Region* → *Populated Place* → *Place*.

```
PREFIX rdf: <http://www.w3.org/1999/02/22 rdf syntax ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf schema#>
PREFIX dbp: <http://dbpedia.org/resource/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT GROUP_CONCAT(?parent;separator='>') where{
dbp:New_York rdf:type ?type.
?type rdfs:subClassOf* ?parent.
}
GROUP BY ?type
```

Listing 3.1: Sparql query to retrieve the ontological superclasses of an entity (identified by its URI).

We then apply NE replacement, to investigate the impact of NE generalization on event classification. We compare two strategies that replace the NEs in tweets with *i)* the first element in the hierarchy, i.e. *the most specific category* (e.g. "Administrative Region" in the example above); *ii)* the last element, i.e. the *most generic category* (e.g. "Place") of the entity. The rationale behind this replacement is to generalize over single mentions and to make classification more robust across different domains.

Beside NEs, also temporal information is relevant to event recognition and classification. Therefore, the Entity Replacement module extracts temporal expressions in the content of the tweets with the SUTime tool (Chang & Manning, 2012) and replaces them with one of the TIMEX3 types assigned by the tool: Date, Time, Duration and Set (Pustejovsky, Knippen, Littman, & Saurí, 2005). SUTime maps temporal expressions to unambiguous calendar dates depending on a reference time, which in our case is the tweet timestamp. Although SUTime was trained on news texts, it is expected to be precise enough on tweets, given that temporal expressions are relatively unambiguous (Ritter et al., 2012). Table 3.1 reports two example tweets in the original format and the version after replacement.

Original Tweet	Generic Categories	Specific Categories
Cambodia's ex-King Norodom Sihanouk dead at 89 http://q.gs/2IvJk #FollowBack	`[Place]` ex-king `[Person]` die at `[number]`	`[Country]` ex-king `[Royalty]` die at `[number]`
Amy Winehouse, 27, dies at her London flat http://bit.ly/nD9dy2 #amyWinehouse	`[Person]`, `[number]`, die at her `[Place]` flat `[Person]`	`[Person]`, `[number]`, die at her `[Settlement]` flat `[Person]`

Table 3.1: Output of the Entity Replacement module on two example tweets with DBpedia categories.

3.3.3 Tweet classification

As shown in Figure 3.1, the third module is devoted to tweet classification. Two classification steps are performed. The first one is aimed at separating tweets that are related to real-world events from tweets that do not talk about events, and we cast it as a binary supervised classification. The second step aims at classifying tweets related to events into a set of categories. We cast the problem of classifying event-related tweets into categories as a supervised multi-class classification problem. Let $T = \{t_1, t_2, t_3, \ldots, t_n\}$ be a collection of tweets where t_i is a tweet. Let $C = \{c_1, c_2, c_3, \ldots, c_n\}$ be a set of target classed, where c_i is a class to which a document t belongs. The tweet classification problem is defined as a function $f : T \rightarrow C$ that maps a tweet t to its corresponding class c. The learning function f learns discriminant concepts for each class c from the feature set D_T. In previous studies, it has been widely observed that feature selection can be a powerful tool for simplifying or speeding up computations, and when employed appropriately it can lead to little loss in classification quality (Forman, 2007; Friedman, Hastie, & Tibshirani, 2001; Liu & Motoda, 2007; Y. Yang & Pedersen, 1997).

We limit the scope of this work to the eight event categories from (McMinn et al., 2013) (see Table 3.5).

3.4 Experimental setting and Results

This section presents the experiments we carried out to validate the proposed framework for event detection and classification on Twitter. We first describe the datasets (Section 3.4.1), then we present the algorithms and the results obtained on event detection (Sect. 3.4.3), and on event classification (Sect. 3.4.4).

3.4.1 Dataset

For the purpose of the evaluation, we rely on two gold-standard corpora: the Event 2012 (EVENT2012) corpus (McMinn et al., 2013) and First Story Detection Corpus (FSD) corpus (Petrović, Osborne, & Lavrenko, 2012).

The Event2012 Corpus

A total of 120 million tweets were collected from October to November 2012 from the Twitter streaming API[5], of which 159,952 tweets were labeled as event-related. 506 event types were gathered from the Wikipedia Current Event Portal, and Amazon Mechanical Turk was used to annotate each tweet with one of such types. Besides, each event was also associated with an event category among Science, Armed Conflicts, Politics, Economy, Culture, Sports, Accidents and Miscellaneous following the Topic Detection and Tracking (TDT) annotation manual (*Topic Detection and Tracking, TDT (2004): Annotation Manual*, n.d.) (see Table 3.5). Events covered by this dataset include for example the US presidential debate between Barack Obama and Mitt Romney, the US presidential election results or the cancellation of the New York Marathon due to the hurricane Sandy.

According to Twitter policies, only tweet identifiers can be released. Therefore, we use these identifiers to download the original contents of the tweets from the Twitter platform. After removing duplicated tweets and those that are no-longer available, we are left with 152,768 tweets related to one out of 506 events. Table 3.2 reports the 10 major events in the corpus.

[5] `https://dev.twitter.com/streaming/overview`

Event	# tweets
During US presidential debate, President Barack Obama tells candidate Mitt Romney he is "the last person to get tough on China".	18542
Paul Ryan spoke for 40 of the 90 minutes during Thursday night's vice presidential debate	9889
Barack Obama And Mitt Romney Went Head-To-Head In The Final Presidential Debate	7565
Felix Baumgartner flew many miles into the air above the south-western U.S. and then jumped, breaking several world records.	4344
People react to incoming election results, threatening to leave the country if their favored candidate does not win.	3336
The San Francisco Giants defeat the Detroit Tigers in game four	2577
Tweets discussing about Rondo	2353
Pride of Britain awards.	2224
Tony Parker from the San Antonio Spurs shoot a buzzer beater shot over Russell Westbrook from the Oklahoma City Thunder.	1992
The cancellation of the New York City Marathon by the mayor due to Hurricane Sandy	1867

Table 3.2: Top 10 events reported in the Event2012 corpus.

The FSD Corpus

This corpus contains 50 million Twitter messages collected from July 2011 until September 2011 using the Twitter API. Human annotators annotated the tweets related to events with one out of 27 event types extracted through the Wikipedia Current Event Portal. In total, 3,035 tweets were labeled as related to events and annotated with a corresponding event topic (e.g. 'death of Amy Winehouse', 'earthquake in Virginia' or 'plane crash of the Russian hockey team'). After removing tweets that are no more available, we are left with ∼31 million tweets from which 2,250 are related to events. Table 3.3 reports the 10 major events discussed in the corpus.

Data characteristics

Contrary to the Event 2012 corpus, the events in the FSD corpus are not associated with event categories. Therefore, in order to merge the two corpora in a single dataset, we extended the FSD corpus by labelling each event topic

Event	# tweets
Death of Amy Winehouse	710
S&P downgrades US credit rating	271
Earthquake in Virginia	265
Plane carrying Russian hockey team Locomotive crashes, 44 dead	212
Explosion in French nuclear power plant Marcoule	133
Google announces plans to buy Motorola Mobility	128
NASA announces discovery of water on Mars	105
US increases debt ceiling	71
Famine declared in Somalia	69
Trevor Ellis (first victim of London riots) dies	65

Table 3.3: Top 10 events reported in the FSD corpus.

with one of the event categories of the Event2012 corpus. The task was manually performed by three annotators: the labels were first assigned independently, and then adjudicated by majority vote in case of disagreements (Inter Annotator Agreement: Krippendorff's α=0.758).

Tables 3.2 and 3.3 show a list of examples of the events included in the Event 2012 and FSD corpus, respectively. It can be observed that the two datasets contain different events: in the Event 2012 dataset, most of them concern the presidential election in the US, while in the FSD corpus most of the tweets report on the death of the singer Amy Winehouse. Moreover, in the Event 2012 corpus the topics of the events are mostly related (i.e. the top 3 events concern the US presidential elections), while events in the FSD dataset are more heterogeneous.

Since both corpora contain much more non-event related than event related tweets, resulting in a very skewed class distribution, we built our evaluation dataset by randomly selecting a sample of non event-related tweets. Tables 3.4 and 3.5 report the final amount of tweets in the two dataset.

	Event-related	Non event-related	Total
Event2012	152,768	1,232,000	1,384,768
FSD	2,250	3,040	5,290

Table 3.4: Total number of tweets per dataset

Event Category	Event2012	FSD
Arts, Culture & Entertainment	15792	710
Armed Conflicts & Attacks	9813	56
Law, Politics & Scandals	57285	58
Sports	50444	0
Business & Economy	4691	342
Science & Technology	2850	296
Disasters & Accidents	5036	778
Miscellaneous	6857	10
Total	152,768	2250

Table 3.5: Tweets in each event category

3.4.2 Experimental Setup

We compare two external knowledge sources to generalize over NE mentions: *1)* the DBpedia ontology, and *2)* the YAGO ontology. We also test the integration of YAGO for missing categories in DBpedia ontology, but this configuration did not improve the performance of the classifiers compared to DBpedia or YAGO alone. For the two knowledge bases, we also analyze which generalisation strategy works better by replacing NEs in the tweets either with the most generic or the most specific category in each ontology. As a baseline, we compute the classification performance without entity replacement, using the entity mentions as features. To simulate a real scenario, where large streams of tweets to be classified may describe events and domains different from those in the training data, results presented in this paper are obtained by training the models on the Event 2012 corpus via cross-validation, and testing them on the FSD corpus. For sake of completeness, results obtained with cross-validation on the Events2012 corpus are presented in Tables 3.7 and 3.8.

Classification algorithms: We compare different classification algorithms including Naive Bayes (NB), Support Vector Machines (SVM) trained with a degree-2 polynomial kernel, and Recurrent Neural Networks (RNN), which have recently shown to advance state of the art in several NLP tasks (Socher et al., 2013). We use the implementations included in the sckit-learn library (Pedregosa et al., 2011) to train NB and SVM models. As for RNN, we use a multi-layered feed-forward NN with Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997).

Feature Representation: We represent tweets through word embeddings, which have shown a good generalization power, outperforming feature models such as Bag of Words in many NLP tasks (Kim, 2014; X. Yang et al., 2016). Following the approach proposed in (X. Yang et al., 2016), we use tweets in the training set to build word embeddings using the Word2Vec tool with 300-dimensional vectors, context window of size 2 and minimum word frequency of 10. Before building the embeddings, we apply the preprocessing and entity replacement steps (see Section 3.3.2) to clean up the dictionary and replace NEs by their semantic categories. Thus, we build three variants of the word embedding vectors: *i)* NEs are replaced by their *generic* category; *ii)* NEs are replaced by their *specific* category; *iii)* no NE replacement (i.e. our baseline). We use the same embeddings as features for all the three classification algorithms. Concerning RNN, we train a 5-layer model with 128 hidden nodes consisting in one input layer, three hidden layers and one output layer. In the input layer, tweets are represented by concatenating the vector in the word embeddings corresponding to each word in the input tweets. Words in tweets that do not appear in the word embedding vectors are initialized randomly (Kim, 2014). The model takes mean of the outputs of all LSTM cells to form a feature vector, and then uses logistic regression and tangent transformation as activation function for the feature vector. We use LSTM as our first hidden layer with batches of 512 examples using Logistic regression as activation function. We use recurrent sum and dropout as second and third layer, respectively. The dropout layer is considered as regularization method for the network. Finally, we use Softmax as the output layer and compute the cost with cross entropy. The implementation is done with Neon[6], a Python-based deep learning library with a GPU back-end.

3.4.3 Task 1: Results

The first task is the detection of event-related tweets 3.3. We cast the problem as a binary classification task, in which event-related tweets are considered as positive instances and non-event related ones are negative instances. We carry out approximate randomization test to evaluate the statistical significance of our results, allowing us to validate our hypothesis. The results reported in Table 3.6

[6] `http://neon.nervanasys.com/`

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.79	0.79	0.79	0.80	0.78	0.79	0.88	0.87	0.87
dbp specific	0.79	0.79	0.79	0.79	0.78	0.77	0.87	0.86	0.86
yago:generic	0.80	0.80	0.80	0.83	0.82	0.82	**0.88**	**0.88**	**0.88**
yago:specific	0.78	0.78	0.78	0.82	0.82	0.82	0.85	0.84	0.84
Baseline (no NER)	0.68	0.67	0.67	0.69	0.67	0.67	0.71	0.71	0.71

Table 3.6: Results of Task 1: tweet classification as event-related or non event-related (weighted average).

show that, for the three classification algorithms, entity linking and replacement is effective and always contributes to outperform the baseline. Moreover, the replacement strategy using the most generic ontological category achieves always a better performance than the most specific option. More importantly, the best results are obtained using YAGO to extract NE categories, i.e. relying on WordNet synsets that represent the upper level of YAGO ontology. We got highly significant results ($P < 0.001$) when comparing YAGO vs. DBpedia both for the generic and for the specific replacement strategies (RNN), and significant results ($P < 0.009$) for YAGO vs DBpedia in both replacement strategies (SVM). Finally, we observe that LSTM-RNNs yield better results compared to SVM (highly significant for yago:spec (SVM) vs. yago:spec (RNN) and $P < 0.06$ for yago:gen (SVM) vs. yago:gen (RNN)).

As a comparison, we run the same classification task using cross-validation only on the Event 2012 dataset. In this setting, the baseline outperforms all the other approaches with every algorithm considered. For example, the F-measure of the baseline for the RNN classifier is 0.94 compared to 0.93 when the NEs are replaced by their generic class in YAGO. The difference in performance between the two settings shows that classification based only on in-domain data can be affected by overfitting. This confirms the importance of evaluating the task in a more realistic setting, with training and test data coming from different domains.

3.4.4 Task 2: Results

We evaluate the proposed approach on the task of classifying event-related tweets into event categories. For this task, we consider only tweets related to events

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.87	0.87	0.87	0.92	0.92	0.92	0.92	0.92	0.92
dbp specific	0.87	0.87	0.87	0.93	0.93	0.93	0.93	0.93	0.93
yago:generic	0.87	0.87	0.87	0.92	0.92	0.92	0.93	0.93	0.93
yago:specific	0.87	0.87	0.87	0.92	0.92	0.92	0.93	0.93	0.93
Baseline (no NER)	0.87	0.87	0.87	0.93	0.93	0.93	**0.94**	**0.94**	**0.94**

Table 3.7: Results of Task 1: tweet classification as event-related or non event-related (weighted average) on the Event2012 dataset using cross validation (k=10).

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.87	0.87	0.87	0.90	0.90	0.90	0.92	0.91	0.92
dbp specific	0.90	0.90	0.90	0.89	0.88	0.88	0.95	0.94	0.94
yago:generic	0.85	0.85	0.85	0.93	0.92	0.92	0.90	0.91	0.90
yago:specific	0.90	0.90	0.90	0.93	0.92	0.92	0.94	0.92	0.93
Baseline (no NER)	0.92	0.92	0.92	0.95	0.95	0.95	**0.96**	**0.94**	**0.95**

Table 3.8: Results of Task 2: classification of event-related tweets into event categories on the Event2012 dataset using cross validation (k=10).

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.75	0.38	0.50	0.72	0.25	0.37	0.82	0.70	0.75
dbp:specific	0.73	0.30	0.43	0.72	0.23	0.35	0.85	0.74	0.79
yago:generic	0.75	0.42	0.54	0.75	0.35	0.48	0.87	0.74	0.80
yago:specific	0.71	0.39	0.50	0.74	0.32	0.45	**0.87**	**0.75**	**0.81**
Baseline (no NER)	0.63	0.22	0.33	0.61	0.22	0.32	*0.72*	*0.62*	*0.67*

Table 3.9: Results of Task 2: classification of event-related tweets into event categories.

in each of the datasets presented in Section 3.4.1. Table 3.9 shows the results obtained for the 8 event categories listed in Table 3.5. In line with the findings on the previous task, LSTM-RNNs outperform the other classifiers in all settings.

However, contrary to the findings of Task 1, the classifier performance is higher when using specific categories in DBpedia and YAGO (see for instance, the difference between yago:gen vs. yago:spec, which is highly significant, $p < 0.001$). Although the difference between specific and generic categories is in some cases very small, the setting in which NEs are replaced by their most specific category seems more suitable for classifying tweets into event categories, while the generic setting targets better the binary classification of Task 1. The yago:specific categories are more fine-grained than dbp:specific ones, yielding better results in this scenario (results of this comparison are statistically significant, $p < 0.01$). Among the different event categories, the worst results are obtained for *Miscellaneous*, that in most cases are assigned to the categories *Politics* and *Economy*.

If we classify only in-domain data using cross-validation with the Event 2012 dataset, the baseline always outperforms the other approaches, like in the binary classification task. Again, this may be due to overfitting and shows the importance of evaluating the task in a different scenario and choosing an approach that generalizes over specific entity mentions.

Combining Task 1 and Task 2

In the experiments described so far, the identification and the classification of event-related tweets have been carried out separately. Specifically, for event classification (Task 2) we considered only tweets related to events. In a real scenario,

a combination of the two tasks would be needed if, given a set of tweets, the goal is to understand which event categories can be detected in the data. In this section, we compare two models, one combining Task 1 and 2 in a pipeline, and the other based on a single classification step. We train the classifiers on the Event 2012 dataset and we test on the FSD dataset.

Pipeline model

A model for classifying tweets as event-related or non event-related provides the input to a second model, which classifies event-related tweets into categories (see the box 'Approach 1' in Figure 3.1). We consider all tweets labelled as event-related by the binary classification algorithm (i.e. both true positive and false positive instances) as input for the second model.

In Table 3.10, we report the performances of the complete pipeline (the performance of the binary model remains the same as in Table 3.6). As expected, combining Task 1 and Task 2 in a pipeline yields a performance drop compared with Task 2 in isolation, due to error propagation. Nevertheless, the drop is only around 0.03 points F-measure, that can still be considered as satisfactory. The main issue is precision: since the second model is not trained to handle non event-related tweets, all misclassified instances in the first model are also misclassified by the second one, which lowers precision. However, the recall of the pipeline model is higher than the classification recall in Task 2, due to a lower number of tweets per category, because of event-related tweets misclassified as non-event related tweets by the binary model (i.e. false negatives).

Similar to the results reported in table 3.9, LSTM-RNNs with entity replacement using the most specific YAGO category outperform all the other settings (the results obtained comparing YAGO vs. DBpedia both for the generic and for the specific replacement strategies (RNN) are significant at $p < 0.01$, while the difference between yago:spec vs. yago:gen (RNN) is not significant $p < 0.216$). Again, LSTM-RNN baseline achieves a better performance.

Single joint model

We compare the pipeline model with a single joint model trained on 9 classes, including the 8 event categories plus a non event-related class (a single multi-

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.64	0.48	0.55	0.76	0.29	0.42	0.64	0.82	0.72
dbp:specific	0.56	0.39	0.46	0.72	0.31	0.44	0.60	0.81	0.69
yago:generic	0.57	0.40	0.47	0.73	0.33	0.46	0.72	0.84	0.77
yago:specific	0.59	0.47	0.52	0.77	0.36	0.49	**0.73**	**0.83**	**0.78**
Baseline (no NER)	0.48	0.29	0.36	0.57	0.21	0.30	0.59	0.69	0.64

Table 3.10: Experimental results of the pipeline model.

	NB			SVM			RNN		
Approach	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
dbp:generic	0.63	0.41	0.49	0.71	0.23	0.34	0.78	0.64	0.71
dbp:specific	0.55	0.33	0.42	0.70	0.20	0.31	0.79	0.63	0.70
yago:generic	0.56	0.36	0.44	0.71	0.27	0.39	0.76	0.64	0.70
yago:specific	0.51	0.33	0.40	0.72	0.23	0.35	**0.81**	**0.67**	**0.73**
Baseline (no NER)	0.45	0.23	0.30	0.58	0.11	0.18	0.70	0.58	0.63

Table 3.11: Experimental results of the single joint model.

class classification step). The input and output are the same as those used for the pipeline experiment. Table 3.11 reports the evaluation of the joint model (we consider the weighted average Precision, Recall and F1-measure for the event-related classes only). The results between the specific and the generic replacement strategies for YAGO are significant with RNN, while they are not significant when comparing the two replacement strategies using DBpedia.

A comparison between the two approaches shows that the single joint model yields lower results than the pipeline (results are highly significant, $p < 0.001$). Recall is particularly affected by several event-related tweets that are classified as non event-related. For example, 60% of the tweets of the category *Economy* were classified as non event-related by the SVM classifier.

3.5 Conclusions

In this chapter, we have presented a framework for identifying and classifying event-related tweets by exploiting information automatically leveraged from DBpedia and YAGO. We evaluated the approach in different classification tasks. In

general, we observed that information extracted from YAGO contributes better to improve classification performances than DBpedia. Possible reasons for that are: i) the better coverage of YAGO, and ii) YAGO class hierarchy is deeper than the DBpedia ontology, which has an impact especially when using specific categories for the multi-class classification task. The fact that DBpedia ontology was manually created, while YAGO was built semi-automatically does not affect much our experiments. In all the experiments, LSTM-RNNs outperform SVM and NB, confirming previous findings on the effectiveness of RNNs when applied to several NLP tasks (Socher et al., 2013). Our experiments on different classification tasks show that performing binary classification first and then passing the output to the second classification step in a pipeline is more accurate than the single-step model. A possible future extension of this work could be to exploit domain-specific ontologies for certain categories, e.g. geographical names.

Finally, the proposed approach can be considered as a first step in the event detection and classification pipeline. Goal of the next chapter is to explore further this research direction, proposing an approach to further specifying event types inside each event category using unsupervised methods. We show that using event categories collected from diverse knowledge bases could be beneficial also in this case (Bryl, Tonelli, Giuliano, & Serafini, 2012).

Chapter 4

Fine-Grained Events Extraction from Tweets*

Goal of this chapter is to investigate the problem of detecting which tweets describe a specific event, and clustering them. To address this issue, we propose a novel approach that exploits NE mentions in tweets and their entity context to create a temporal event graph. Then, using simple graph theory techniques and a PageRank-like algorithm, we process the event graphs to detect clusters of tweets describing the same events.

4.1 Introduction

In the previous chapter we have presented a framework for identifying and classifying event-related tweets into coarse-grained categories (e.g. *Sport, Politics*) by exploiting information automatically leveraged from DBpedia and YAGO. In this chapter we take a step forward focusing on finer-grained event extraction from Twitter, consisting in the automated clustering of tweets related to the same event based on relevant information such as time and participants.

Existing approaches to the task create clusters of tweets around event-related keywords (Parikh & Karlapalem, 2013), or Named Entities (NE) (McMinn & Jose, 2015). However, such approaches fail *i)* to capture events that do not generate spikes in the volume of tweets, for instance "Richard Bowes, victim of

*Most of the work presented in this chapter has been accepted for publication in (Edouard et al., 2017b)

London riots, dies in hospital"; and *ii)* to distinguish between events that involve the same NEs and keywords, as for instance "the shoot of Malala Yousafzai, the 14-year old Pakistani activist" and "her successful surgery". Other approaches model the relationships between terms contained in the tweets relying on a graph representation (Katragadda et al., 2016), and retain the nodes with the highest number of edges as event candidates. However, the main drawbacks of these approaches are that *i)* they generate highly dense graphs, and *ii)* trending terms not related to events may be considered as event candidates. To address such limitations, we propose an unsupervised approach to detect open-domain events on Twitter, where the stream of tweets is represented through temporal event graphs, modeling the relations between NEs and the terms that surround their mentions in the tweets.

In the following, Section 4.2 presents related work; Section 4.3 describes the approach we propose for event detection on Twitter, while Section 4.4 reports on the experiments we carried out to evaluate our work, and to compare it with existing approaches.

4.2 Related Work

As discussed in Chapter 2 (Section 2.5), event detection approaches are generally classified into two main categories: *closed-domain* and *open-domain*. Our work belongs to the latter category, since we are not interested in detecting a particular type of event, but in detecting different event types without prior knowledge on the events. In this section, we compare the approach we propose to state-of-the-art approaches in the open-domain scenario.

Among the works applying an unsupervised approach, Petrović et al. (2010) address the First Story Detection task by analyzing solely the contents of tweets. Their approach is based on local sensitive hashing, a randomized technique that reduces the time needed to find a nearest neighbor in a vector space. Each new tweet is assigned to the thread that contains the most similar tweets, where similarity is based on cosine similarity. The growth rate of the thread is used to eliminate non-event related threads, such that threads that grow fastest are considered as event-related. Since the size of the Twitter vocabulary can be quite large, clusters can be driven by different keywords. McMinn and Jose (2015)

propose to create event clusters from tweets using NE mentions as central terms driving the clusters. Thus, tweets mentioning the same entities are grouped together in a single cluster. Such works do not consider the temporal aspect of events and will more likely fail to capture terms or entities involved in different events at different time periods, as for instance the events related to the shoot of *Malala Yousafzai* and her surgery later on.

Ritter et al. (2012) model events on Twitter as a 4-tuple representation including NEs, temporal expressions, event phrases and event type. NEs and temporal expressions are extracted using Twitter specific tools (Ritter et al., 2011) while event phrases are extracted using a supervised method. The system recognizes event triggers as a sequence labeling task using Conditional Random Field; then an unsupervised approach is used to classify the events into topics. In addition, the authors consider the association strength between NEs and temporal expressions to decide whether or not a tweet is related to an event. However, this assumption restricts the approach to tweets that explicitly contain temporal expressions and NEs.

Zhou, Zhang, and He (2017) use a non-parametric Bayesian Mixture Model leveraged with word embeddings to create event clusters from tweets. In this approach, events are modeled as a 4-tuple $\langle y, l, k, d \rangle$ modeling non-location NEs, location NEs, event keywords and date. Each component of the quadruple is generated from a multinomial distribution computed with Dirichlet process. The work was focused on detecting events given a set of event-related tweets, which is however not applicable to a real scenario, where the stream of tweets can also contain messages that are not event-related.

There are some approaches that model the relationships between terms contained in the tweets relying on graph representation, and retain the nodes with the highest number of edges as event candidates. Katragadda et al. (2016) use graphs to model relationships between terms in tweets at different time windows. First, a graph is created based on the set of links between terms in tweets, where terms are considered as connected according to the order of their appearance in the text independently of their syntactic or semantic relations. Then, the graph is pruned to remove terms that are less frequent than a given threshold. Finally, clusters in the graph are evaluated in order to determine whether or not they are credible, where the credibility of an event is determined by their presence or

not in other time windows. Differently from them, we create event graphs from terms that appear in the NE context, which contributes in reducing the density of the event graphs by considering event-related features. However, the main drawbacks of these approaches are that *i)* they generate highly dense graphs, and *ii)* trending terms not related to events may be considered as event candidates.

Most of the existing works on open-domain event detection on Twitter rely on the speed according to which the clusters are growing: clusters that grow faster are considered as event-related (Ritter et al., 2012; Xie, Zhu, Jiang, Lim, & Wang, 2013). Although this assumption helps in discovering large-scale events (Osborne, Petrovic, McCreadie, Macdonald, & Ounis, 2012), it is less suitable for events with a small audience on Twitter. Moreover, clusters may be driven by non-event related terms that could negatively impact the quality of the event clusters.

4.3 Approach Description

In this section, we describe our approach for detecting open-domain events on tweets. The proposed approach is based on graph theory to model relations between terms in tweets. The pipeline consists of the following components: *i)* **Tweet preprocessing**, *ii)* **Named Entity recognition and linking**, *iii)* **graph creation**, *iv)* **graph partitioning**, *v)* **event detection**, and *vi)* **event merging**. Figure 4.1 shows the pipeline of the proposed model, where each step is described in the following subsections.

4.3.1 Tweet Pre-processing, NE recognition and linking

As a first step, we carry out the pre-processing steps described in the previous Chapter (Section 3.3.1) on the tweets in input.

We then detect NE mentions in tweets applying a modified version of the approach presented in Section 3.3.2. More precisely, in this case we do want to generalize the entity by replacing it with its semantic category (differently from the previous task, here the goal is to detect fine-grained events, therefore we need to keep the NEs as they are). To account for language variability, we search for the NEs detected in the tweets in the DBpedia ontology with the goal of

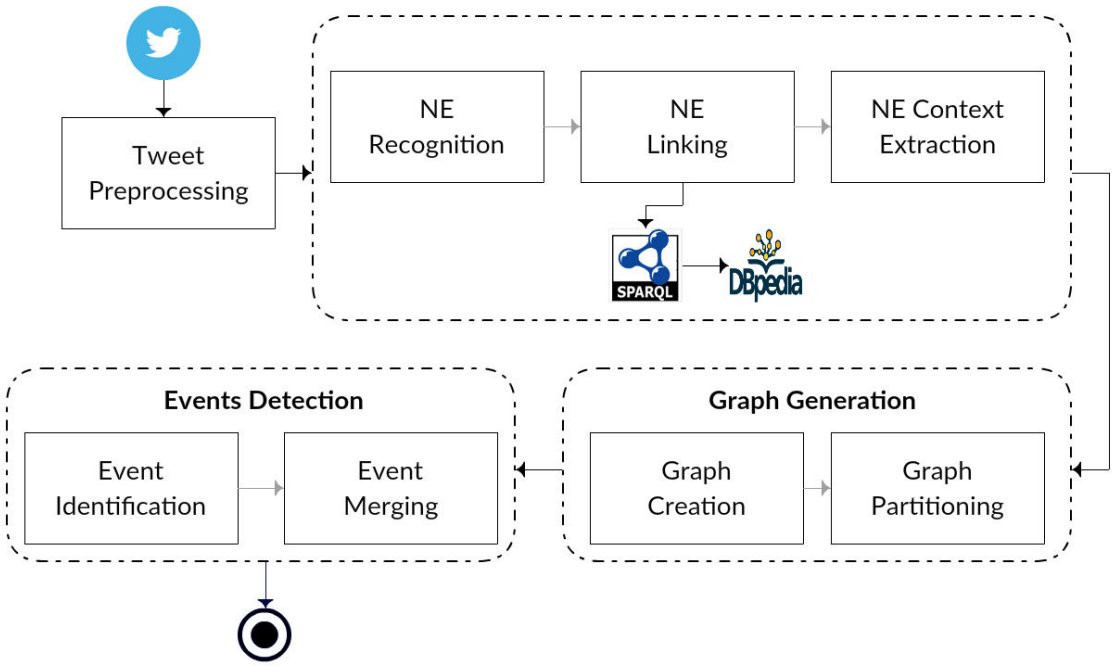

Figure 4.1: The event detection pipeline (tweets are the input source). Rectangles are conceptual stages in the pipeline with data flowing in the order of the arrows. The solid circle at the bottom represents the output, i.e. tweets related to the same event grouped into event clusters.

extracting for such resources the values of the properties *dbp:name* or *rdfs:label*, to normalize the entity mentions in the tweet. Table 4.1 reports on two examples where NEs surface forms are normalized using their labels from DBpedia.

At the end of the pre-processing phase, each input tweet is annotated with a set of NE labels, including the type of the NEs according to the DBpedia ontology (its most generic category/ies). It is important to notice that, in this approach we use concepts in DBpedia ontology instead of YAGO. Our motivation is twofold: *i)* as shown in Section 3.4.3, generalizing over generic concepts in DBpedia and YAGO results in similar performances, and *ii)* the DBpedia ontology has a more consistent terminology and classes than YAGO.

4.3.2 Graph Generation

Previous works using graph-based methods to model relations between terms in text considered all terms in the input document as nodes and used their position

The EU has won a noble peace prize! I'm guessing merkel will go and accept it
The `European_Union` has won a `Noble_Peace_Prize`! I'm guessing `Angela_Merkel` will go and accept it

Table 4.1: Output of the Entity Recognition and Linking module on tweets: entity mentions are normalized after linking.

in text to set edges (Andersen, Chung, & Lang, 2006; Xu, Grishman, Meyers, & Ritter, 2013). Such approaches may generate a dense graph, which generally requires high computational costs to be processed. Instead, we assume that the terms surrounding the mention of a NE in a tweet define its local context (Nugroho et al., 2015). Thus, we rely on the local NE context to create the event graphs, built as follows:

- *Nodes* : We consider NE and k terms in its local context (i.e. terms that precede and succeed its mention in tweets) as nodes, where $k > 1$ is the number of terms surrounding the NE.

- *Edges* : Nodes in the graph are connected by an edge if they co-occur in the local context of a NE.

- *Weight* : The weight of the edges is the number of co-occurrences between terms in the NE context. In addition, each edge maintains as a property the list of tweets from which the relationship is observed.

Formally, let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a directed graph (or digraph) with a set of vertices $\mathcal{V}$ and edges $\mathcal{E}$, such that $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. For any $\mathcal{V}_i \in \mathcal{V}$, let $In(\mathcal{V}_i)$ be the set of vertices that point to $\mathcal{V}_i$ (i.e. predecessors), and $Out(\mathcal{V}_i)$ be the set of vertices that $\mathcal{V}_i$ points to (i.e. successors). Let $\mathcal{E}_i = (\mathcal{V}_j, \mathcal{V}_k)$ be an edge that connects node $\mathcal{V}_j$ to $\mathcal{V}_k$, we define ω_{ij} as the weight of $\mathcal{E}_i$, which is represented by the number of times relationships between $\mathcal{V}_j$ and $\mathcal{V}_k$ are observed in tweets published during a time window. An example of the graph created on 2011-07-07 with tweets related to the famine in Somalia and space shuttle to Mars is shown in Figure 4.2.

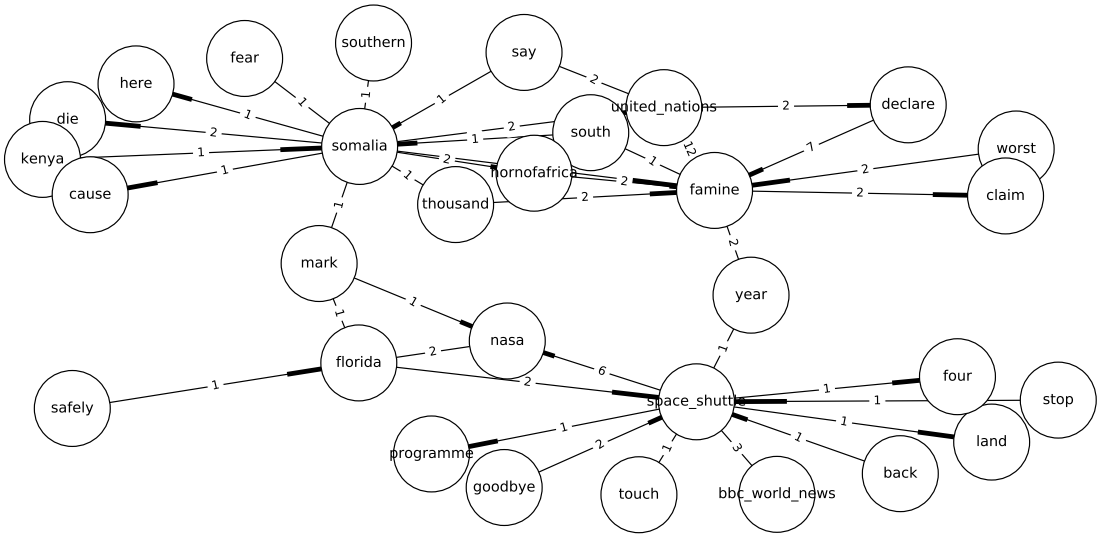

Figure 4.2: Example of event graph about the famine in Somalia and the space shuttle to Mars.

4.3.3 Graph Partitioning

At this stage, an event graph is generated to model relationships between terms in the NE contexts. We apply graph theory to partition the graph into sub-graphs, which will be considered as event candidates. Tweets related to the same events usually share a few common keywords, while tweets that are not related to events or those related to different events are usually characterized by different keywords (McMinn & Jose, 2015). In the event graphs, this phenomenon is expressed by stronger links between nodes related to the same event. In other words, the weight of edges that connect terms from tweets related to similar events are higher than edges between nodes that connect terms from tweets related to different events. The graph partitioning purpose is to identify such edges that, if removed, will split the large graph G into sub-graphs.

Often, the resulted graphs at a given time window is disconnected, i.e, all the nodes in the graph are not connected to each other. We first process the generated graph by analyzing the connection between different nodes, thus those that are connected with few other nodes are simply discarded from the graph.

Experiments show that nodes connected to less than 3 other nodes that have weight lower than 3 are not useful for the event identification purpose, thus we discard such nodes. In a second phase, if the resulted graphs is still disconnected, we process each components separately. Next, we use graph partitioning to split the resulting components of $\mathcal{G}$ into sub-graphs.

Let $\mathcal{E} = \{(\mathcal{V}_1, \mathcal{W}_1), (\mathcal{V}_2, \mathcal{W}_2), \ldots, (\mathcal{V}_n, \mathcal{W}_n)\}$ be a set of pair of vertices in a strongly connected graph $\mathcal{G}$. We define λ as the least number of edges whose deletion from $\mathcal{G}$ would split $\mathcal{G}$ into sub-graphs. Similarly, we define the edge-connectivity $\lambda(\mathcal{G})$ of $\mathcal{G}$ of an edge set $\mathcal{S} \subset \mathcal{E}$ as the least cardinality $|\mathcal{S}|$ such that $\mathcal{G} - \mathcal{S}$ is no longer strongly connected. Thus, we have:

$$\lambda(\mathcal{G}) = min\{\lambda(\mathcal{V}, \mathcal{W})\} \tag{4.1}$$

where $\lambda(\mathcal{G})$ is the minimum number of edges to remove from $\mathcal{G}$ so that it is disconnected in sub-graphs. Given equation 4.1, to compute $\lambda(G)$, it is necessary to compute $\lambda(v, w)$ for any pair of nodes in G. We do so using max-flow min-cut theorem (Hoffman, 1974) using Algorithm 2 as described by (Even, 2011).

Algorithm 2 Algorithm to select the cutting nodes.

function GRAPH_CUT($\mathcal{G}$) ▷ $\mathcal{G}$ is a strongly connected graph
  Let $\mathcal{V}_\rangle \in \mathcal{V}$ ▷ Select an arbitrary vertex in $\mathcal{V}$
  $X \leftarrow \mathcal{V} - \{\mathcal{V}_\rangle\}$
  for $\mathcal{X}_\rangle \in \mathcal{X}$ **do**
    Assign V_i as source and $\mathcal{X}_\rangle$ as the sink vertex
    Assign the capacity of of each arc to 1
    $f \leftarrow maxflow(H)$ (Hoffman, 1974) ▷ $\mathcal{H}$ is the resulting network
    $\lambda(\mathcal{V}_\rangle, \mathcal{X}_\rangle) \leftarrow total_flow(f)$.
  end for
  $\lambda(\mathcal{G}) \leftarrow min(\lambda(v, w))$
  return $\lambda(\mathcal{G})$
end function

For instance, given the graph in Figure 4.2 as input, the Algorithm 2 would return edges "(mark,somalia)" and "(year,space shuttle)", such that the deletion of these edges from $\mathcal{G}$ will brake $\mathcal{G}$ in two sub-graphs $\mathcal{G}_\infty$ and $\mathcal{G}_\in$, where the first one contains keywords related to "famine in Somalia" and the other contains keywords related to "The space shuttle to Mars".

4.3.4 Event detection

At this stage, the initial event graph is divided into a set of sub-graphs that contain highly related keywords. In our event detection approach, we assume that events from different sub-graphs are not related to each other. Thus, in the event detection sub-module, each sub-graph is processed separately. In a study on local partitioning, (Andersen et al., 2006) show that a good partition of a graph can be obtained by separating high-ranked vertices from low-ranked ones, if the nodes in the graph have distinguishable values. Similar to (Mihalcea & Tarau, 2004), we use a PageRank-like algorithm (Brin & Page, 1998) to rank vertices in the event-graph as follows :

$$S(V_i) = ((1-d) + d \sum_{v_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_k)} \omega_{jk}} S(V_j))\epsilon_i \tag{4.2}$$

where ω_{ij} is the weight of edge connecting V_i to V_j, d a dumping factor usually set to 0.85 (Brin & Page, 1998) and ϵ_i a personalization parameter for node i. While in (Mihalcea & Tarau, 2004) the personalization parameter is considered as a uniform distribution, we define the personalization parameter of a node according to its tf-idf score. Due to redundant nature of tweets, the score of the nodes can be biased by the trending terms in different time windows. Thus, we use the tf-idf score to reduce the impact of trending terms in the collection of tweets. Before computing the score with Equation 4.2, we assign an initial value $\tau = 1/n$ to each vertex in the graph, where n is the total number of nodes in the graph. Then, for each node, the computation iterates until the desired degree of convergence is reached. The degree of convergence of a node can be obtained by computing the difference between the score at the current iteration and at the previous iteration, which we set to 0.0001 (Brin & Page, 1998). Notice that the final salience score of each node is not affected by the choice of the initial value assigned to each node in the graph, but rather by the weight of the edges (Mihalcea & Tarau, 2004).

As shown in Algorithm 3, we start by splitting the vertex set into high-ranked and low-ranked vertices based on a gauged parameter α (Line 3). Next, we process the vertices in the high-ranked subset starting from the highest ones, and for each candidate we select the highest weighted predecessors and successors as keywords for event candidates (Lines 4-9). After removing the edges between

the keywords from the graph, if it becomes disconnected, we also consider the disconnected nodes as keywords for the event candidate (Lines 10-13). Based on the semantic class provided by the NER tool (see Section 4.3.1), we divide the keywords related to an event in the following subsets: *what* (i.e., the type of the event), *where* (i.e., the location where the event happens), *who* (i.e., the person or organization involved) and *when* (i.e., the date). As for the date, we select the creation date of the oldest tweets that report the event. We recall that each edge holds a list of tweets from which the relationship is obtained.

In the second stage, Algorithm 4 is used to further process the event candidates to remove noise and duplicate events. First, we merge duplicate event candidates (Lines 4-12). Event candidates are considered as duplicate if they share common terms, location and/or participants. When two event candidates are found as duplicate, they are merged into a new event built from the combination of terms and entities of the two event candidates. Finally, an event is considered as valid if at least a NE is involved and occurs in a minimum number of tweets provided as input parameter.

4.3.5 Event Merging

It is common to observe in the Twitter stream mentions of the same event in different time slices (e.g. hours, days). Thus, we found it useful to detect and merge duplicated events. We consider events in different time-windows as duplicate if they are driven by the same keywords and involved the same entities (e.g. person, organization, location) in an interval of k days, where k is an input parameter. When a new event is found as duplicate, we merge it with the previous detected event.

4.4 Experiments

In this section, we describe the experiments carried out to validate our approach. Given a set of tweets, the goal of these experiments is to cluster such tweets so that each cluster corresponds to a fine-grained event such as "Death of Amy Winehouse" or "Presidential debate between Obama and Romney during the US presidential election". We first describe the datasets, then we present the experi-

Algorithm 3 Algorithm to detect event candidates from event graphs.

function GRAPH_PROCESSING(G, α)
  $E \leftarrow \emptyset$
  $H \leftarrow \{v_i \in vertex(G)/score(v_i) >= \alpha\}$ ▷ Equation 4.2
  while $H \neq \emptyset$ **do**
    $G' \leftarrow G.copy()$
    $v_i \leftarrow H.pop()$
    $p \leftarrow max(W_j \in In(v_i))$
    $s \leftarrow max(W_j \in Out(v_i))$
    $keywords = set(p, v_i, s)$
    $G'.remove_edges((p, v_i), (v_i, s))$
    if $notG'.connected()$ **then**
      $append(keywords, disc\_vertices(G'))$
    end if
    $who \leftarrow \{person||organization \in keywords\}$
    $where \leftarrow \{location \in keywords\}$
    $what \leftarrow keywords - who - where\}$
    $tweets \leftarrow tweet\_from(keywords)$
    $when \leftarrow oldest(tweets, date)$
    $event \leftarrow < what, who, where, when >$
    $append(E, event)$
  end whilereturn E
end function

mental setting. This section ends with a comparison of the obtained experimental results with state-of-the-art approaches.

4.4.1 Dataset

For the evaluation, we rely on the same two gold standard datasets used to evaluate the event identification and categorization tasks (see Chapter 3 for statistics on these datasets), i.e. the First Story Detection (FSD) corpus (Petrović et al., 2012), and the Event2012 corpus (McMinn et al., 2013).

More precisely, to reproduce the same dataset used by the state-of-the-art systems to which we compare for the fine-grained event extraction task, we consider only those events in the FSD corpus mentioned in more than 15 tweets. Thus, the final FSD dataset contains 2,295 tweets, describing 20 events in total. Concerning the Event2012 dataset, all tweets related to events are considered, i.e. 152,758 tweets.

Algorithm 4 Algorithm to extract fine-grained events from event candidates.

```
1: function EXTRACT_EVENTS(E, P)   ▷ E candidate events, P previously detected events
2:     for e ∈ E do
3:         for e′inE do
4:             if what(e) ⊂ what(e′) then
5:                 if who(e) ∩ who(e′) or where(e) ∩ where(e′) then
6:                     merge(e, e′)
7:                 end if
8:             else if what(e) ∩ what(e′) then
9:                 if who(e) ∩ who(e′) and where(e) ∩ where(e′) then
10:                    merge(e, e′)
11:                end if
12:            end if
13:        end for
14:        if not who(e) or not where(e) then
15:            discard(E, e)
16:        end if
17:        if not who(e) or not where(e) or len(tweets(e) < min_tweets then
18:            discard(E, e)
19:        end if
20:        for p ∈ P do
21:            if interval_day(p, e) > min_day then
22:                merge(p, e)
23:                discard(E, e)
24:            end if
25:        end for
26:    end for
       return E
27: end function
```

4.4.2 Experimental settings

For each dataset, we compare our approach with state-of-the-art approaches. For the FSD2O11 dataset, we compare with LEM Bayesian model (Zhou, Chen, & He, 2011) and DPEMM Bayesian model enriched with word embeddings (Zhou et al., 2017). For the Event2012 dataset, we compare our results with Named Entity-Based Event Detection approach (NEED) (McMinn & Jose, 2015) and Event Detection Onset (EDO) (Katragadda et al., 2016).

In order to simulate a real scenario where tweets are continuously added to a stream, we simulate the Twitter stream with a client-server architecture which

pushes tweets according to their creation date. We evaluate our approach in two different scenarios: in the first scenario, we consider tweets from the FSD dataset that are related to events and we classify them into fine-grained event clusters. In the second scenario, we adopt a more realistic approach in that we consider all the tweets from the EVENT2012 dataset (i.e event-related and not event-related ones), and we classify them into event clusters, discarding those that are not related to events.

Our approach requires a few parameters to be provided as input. In the experiments reported in this chapter, we process the input stream with fixed time-window $w = 1$ hour. The minimum number of tweets for event candidates is set to $n = 5$. Finally, we empirically choose $t = 3$ days as the interval of validity for the detected events.

4.4.3 Results

Performance is evaluated both in terms of precision, recall and f-score, and the quality of the detected events, i.e. cluster purity.

Precision is computed as the ratio between the number of events (i.e. tweet clusters) correctly classified and the number of detected events. *Recall* is calculated by taking the ratio between the number of events correctly classified and the number of events in the ground truth. The *purity* of the detected events for the event identification model is calculated following Equation 4.3.

$$P_e = \frac{n_e}{N_e} \tag{4.3}$$

where $P_e \in [0, 1]$ is the purity, n_e the number of tweets correctly classified and N the total number of tweets in the cluster.

Results on the FSD dataset

In this scenario, we consider an event as correctly classified if *all* the tweets in that cluster belong to the same event in the gold standard, otherwise the event is considered as misclassified. In addition, due to the low number of tweets, we set the gauged parameter $\alpha = 0.5$ as the minimum score for nodes in the graph to be considered as useful for events. Table 4.2 shows the experimental results yielded

by our approach in comparison to state-of-the-art approaches. Our approach outperforms the others, improving the F-score by 0.07 points w.r.t. DPEMM and by 0.13 w.r.t. LEM. Example of events detected by our approach are shown in Table 4.3. For most event, the keywords are highly informative; for instance event #1002 about the "space shuttle on Mars by NASA on 2011-07-21" is identified with informative terms (e.g. space, shuttle, Atlantis), location (e.g. Kennedy space center) and participant involved (e.g. NASA). Moreover, our method is able to detect events driven by common terms occurred in different locations or periods. For instance, events reported in tweets #1010, #1024 and #1025 in Table 4.3, all related to the *London riot*, are detected in different locations (i.e. Croydan, Birmingham and Tottenham).

Among the 20 events in the ground truth, we fail to detect two events related to *A children's camp attack in Utoya, Norway* and *A car bomb explosion in Oslo, Norway*. These events occurred both on *July, 7th 2011* and were reported by a few tweets only. Most of such tweets mentioned both events at the same time (*e.g. After the Oslo bomb, now reports of shooting at Norweigan summer camp. What's happening? How can anyone attack Norway?*). Thus, tweets related to these events were wrongly classified in the same cluster.

Approach	Precision	Recall	F-measure
LEM	0.792	0.850	0.820
DPEMM	0.862	0.900	0.880
Our Approach	**0.950**	**0.950**	**0.950**

Table 4.2: Evaluation results on the FSD dataset.

ID	WHAT	WHERE	WHO	WHEN
1022	independent, nation	south sudan	congrats	2011-07-09
1021	famine, United Nations	Somalia	hornofafrica,	2011-07-15
1002	Space shuttle Atlantis	Kenedy Space Center	nasa	2011-07-21
-1	Summer camp	Norway, Oslo, Utoya Islands	terrorism	2011-07-22
1001	Drug overdose	London	Amy Winehouse	2011-07-23
1025	Riot	Tottenham	–	2011-08-06
1010	Riot	Croydon	BBC	2011-08-09
1024	Riot	Birmingham	–	2011-08-10
1016	Android	–	Motorola Mobility	2011-08-15
1014	radioactive, explosion	France	nuclear power	2011-09-12
1020	explosion	Kenya	Kenya pipeline	2011-09-12

Table 4.3: Example of event detected by our model on the FSD dataset.

Figure 4.3 reports on the purity of the events detected by our approach compared to LEM and DPEMM, where each point (x, y) denotes the percentage of events having purity less than x. It can be observed that 5% of the events detected as well as DPEMM have purity less than 0.65 compared to 25% for LEM, while 95% of the events detected have purity higher than 0.95 compared to 75% for DPEMM and 55% for LEM.

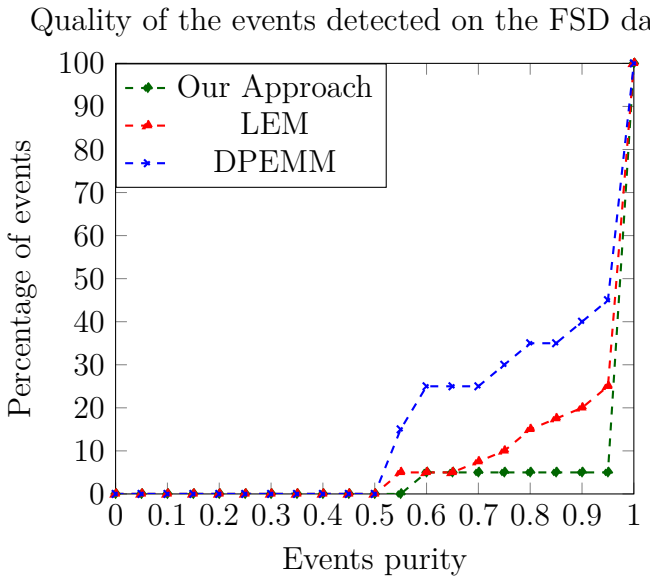

Figure 4.3: Purity of the events detected by our approach, LEM and DPEMM on the FSD dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.

To test a more realistic scenario in which all the tweets (i.e. events-related and non-event related tweets) are considered, we build a pipeline model that applies as a first step the classification algorithm described in Chapter 3 to separate tweets related to events from those that are not, followed by the clustering strategy described above. For the first step, we choose the best performing classification model on FSD dataset (see Table 3.6). Again, for evaluation, a cluster is considered as related to an event in the ground truth if all the tweets in that cluster belong to the same event in the ground truth, otherwise is is considered as misclassified. We set the cutting parameter $\alpha = 0.5$ as the minimum score of nodes in the graph to be considered as important for events.

Table 4.4 reports on the obtained results (we could not compare to existing

approaches, given that both LEM and DPEMM are tested on tweets related to events only, while here we test the full pipeline). The pipeline obtains satisfactory results. Precision only drops by 0.02 points compared to the results in Table 4.2, while, as expected, recall is impacted by the mistakes of the first module. A manual inspection of the detected events show that the approach fails to detect events that are less represented in tweets such as those related to "to the London Riot" (they are wrongly classified as non event by the first module of the pipeline). Figure 4.4 shows the purity of the clusters created by the pipeline model. We can see that less than 10% of the clusters have purity lower than 0.75, while more than 90% have purity equal to 1. In total, we detect 15 events from which 14 have purity equal to 1 and 1 has purity equals to 0.73.

Approach	Precision	Recall	F-measure
Pipeline	0.93	0.70	0.80

Table 4.4: Evaluation results of the pipeline model. Tweets labeled as related to events by the supervised model described in Chapter 3 are provided as input to the event clustering model.

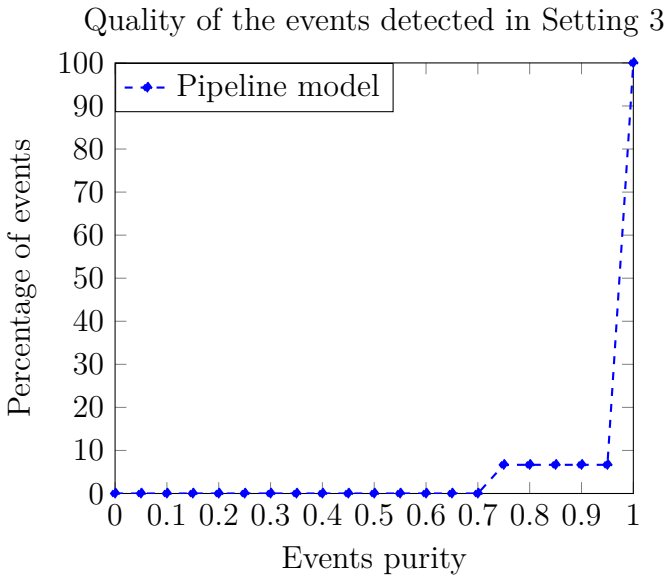

Figure 4.4: Purity of the events detected by the pipeline model. The y-axis denotes the percentage of events and the x-axis the purity of the clusters.

Results on the Event2012 dataset

We also evaluate our approach on the Event2012 dataset considering all the tweets (i.e. events related and non-event related tweets). Compared to the FSD dataset, this dataset has more events and tweets and thus a larger vocabulary. We set the cutting parameter $\alpha = 0.75$ as the minimum score of nodes in the graph to be considered as important for events. We further detail the importance of the parameters α in Section 4.4.4. Also, since we include both event-related and not event-related tweets, we consider an event detected by our approach as correct if 80% of the tweets belong to the same event in the ground truth. We recall that a cluster is considered as related to the event in the ground truth which contains the majority of tweets contained in that cluster.

Table 4.5 reports on the experimental results compared to the NEED and EDO approaches. In general, our approach improves the f-score by 0.07 points w.r.t. EDO and 0.23 points w.r.t. NEED. Recall is particularly affected by events in the ground truth that contain a few tweets, for instance we found that 28 events in the ground truth have less than 10 tweets. This is particular due to tweets that were no longer available at the time we built the dataset.

Similar to Setting 1, we also evaluate the purity of the events detected by our approach (Figure 4.5). More than 20% of the detected events have purity *lower* than 0.7. As expected, event purity is mainly affected by the inclusion in the clusters of non event-related tweets. After a manual check of the output, we noticed that some issues with precision may depend on the quality of the dataset due to errors in the annotated tweets. For example, we found that *9,010* tweets related to "BET hip hop award" were not correctly annotated in the ground truth. The same was found for many major events including "the Presidential debate between Obama and Romney" or the "shooting of Malala Yousafzai, the 14-year old activist for human rights in Pakistan". Consequently, the cluster detected by our approach was considered as misclassified, yielding in lowering the precision and the quality of the clusters.

4.4.4 Effect of the Cutting Parameter

We further experiment on the impact of the dangling parameter on the output of our model. The dangling parameter α is used to separate the nodes of the event

Approach	Precision	Recall	F-measure
NEED	0.636	0.383	0.478
EDO	0.754	0.512	0.638
Our Approach	0.750	0.668	0.710

Table 4.5: Evaluation results on the EVENT2012 dataset.

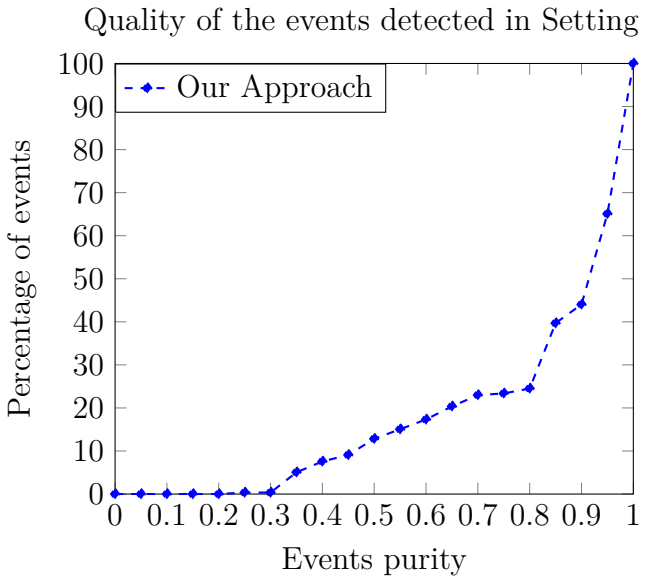

Figure 4.5: Purity of the events detected by our approach on the Event2012 dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.

graph into high-ranked and low-ranked nodes, where the high-ranked nodes are used to extract keywords related to event candidates. We experiment different values for "α" and we evaluate their impact on the performance of our approach on both datasets.

In Figure 4.6 we show the performance of our model for $0 < \alpha \leq 4$ on the FSD dataset. We observe that higher value of α gives higher precision while lowering the recall. More specifically, for $\alpha \geq 3$ we obtain 100% precision and recall lower than 50%. On the other hand, the best performance is obtained for $\alpha \leq 0.5$. Since the FSD dataset contains $\sim 6,000$ unique words, at each time window the generated graph is strongly connected, thus the average minimum score of the nodes is higher than 0.5. For values higher than 0.5, important terms referring to events are ignored, mainly when they are related to events that do not generate

a high volume of tweets. In our experiments, we also observe that higher values of α mostly affect the recognition of events with low number of tweets.

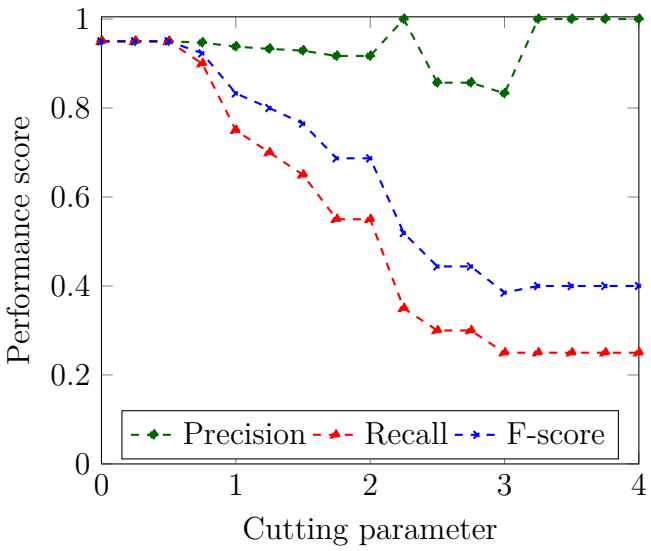

Figure 4.6: Purity of the events detected by our approach on the FSD dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.

The performance of the pipeline model for different values of α is depicted in Figure 4.7. Again, higher values of α increase the precision of the model while decreasing the recall (recall is dramatically impacted for high values of α). This is due to the fact that the number of events in this configuration is smaller w.r.t the other settings, thus higher values of the cutting parameter discard too many event-related terms (mainly those that are not well represented in the data).

Figure 4.8 shows the performance of our model for different values of α on the Event2012 dataset. We observe that for different values of α, both precision and recall are affected. More specifically, the recall of the model tends to decrease for lower values of α. Without edge cutting (i.e. $\alpha = 0$), the recall of our model is similar to EDO. Overall, the impact of α is bigger on the Event2012 dataset than on FSD dataset. The variation of precision and recall curves is smaller for consecutive values of α w.r.t. to FSD, because *i)* the Event2012 dataset has a richer vocabulary, and *ii)* events in the Event2012 dataset are more similar to each other.

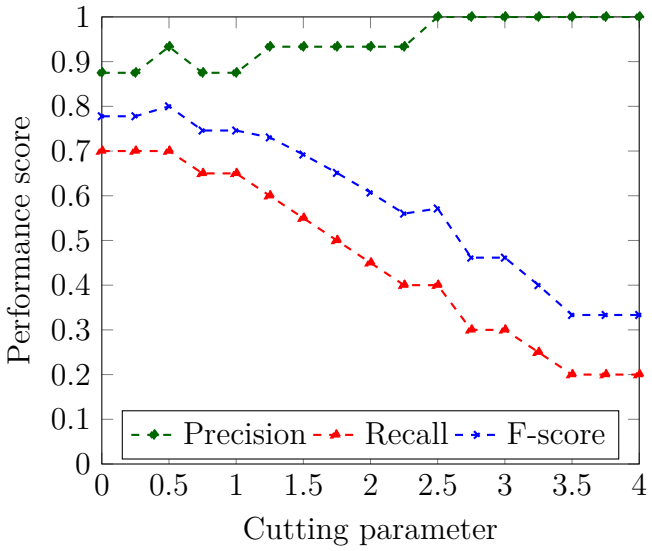

Figure 4.7: Purity of the events detected by the pipeline model. A model to classify tweets as events and non events provides input to the event cluster model. The y-axis denotes the percentage of events and the x-axis the purity of the events.

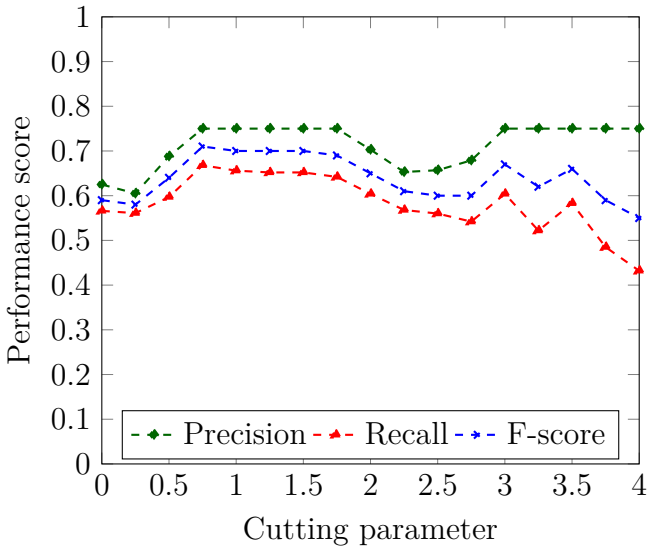

Figure 4.8: Purity of the events detected by our approach on the Event2012 dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.

4.5 Conclusion

In this chapter, we have described a model for detecting open-domain events from tweets by modeling relationships between NE mentions and terms in a directed graph. The proposed approach is unsupervised and can automatically detect fine-grained events without prior knowledge of the number or type of events. Our experiments on two gold-standard datasets show that the approach yields state-of-the-art results.

This approach could be improved by investigating whether linking terms to ontologies (e.g. DBpedia, YAGO) can help in detecting different mentions of the same entity, for instance "German chancellor" and "Angela Merkel". This can be done by exploiting some properties of the DBpedia ontology such as $owl : sameAs$ to extract relationship between different resources. This can be used to reduce the density of the event graph. Another possible improvement would be to enrich the content of the tweets with information from external web pages resolving the URLs in the tweets.

Chapter 5

Tracking and Summarizing Events in Twitter*

Goal of this chapter is to investigate the problem of monitoring the stream of tweets discussing a particular event. As a use case, we select the sport domain, and we propose a semi-supervised method that tracks an event discussed on Twitter and builds a timeline that summarizes its salient points. Information provided by external knowledge bases is used to enrich the content of the tweets, and graph theory is applied to model the relations between actions and participants in a game.

5.1 Introduction

Historically, sports fans have watched matches either at the stadium or on TV, or have listened to them on the radio. In the latest years, however, social media platforms, in particular microblogs, have become a new communication channel also to share information and comment on sports events, thus creating online communities of sports fans around the world. Microblogs are particularly suitable for this, thanks to their coverage and speed, making them a successful channel to follow and comment on events in real time. Also sports teams and medias have benefited from these platforms, using them to extend their contact networks, increase their popularity and exchange information with fans (Gibbs & Haynes,

*Most of the work presented in this chapter has been accepted for publication in (Edouard et al., 2017d) and (Edouard et al., 2017a)

2013; Özsoy, 2011). The need to monitor, categorize and organize information about the matches is particularly relevant during large events like the Olympic Games or FIFA World Cup: several matches take place in a limited time span, sometimes in parallel, and summaries are manually made by journalists who take notes of the main actions during the matches. A few approaches have recently tried to perform this task automatically by recognizing actions in multimedia data such as videos, transcripts of matches or news (Hannon, McCarthy, Lynch, & Smyth, 2011; Snoek, Worring, et al., 2003; Snoek & Worring, 2005).

In this chapter, we investigate whether the same task can be performed relying only on user-generated content from microblogs. In fact, opinions shared by fans during sports matches are usually reactions to what is happening in the game, implicitly conveying information on the ongoing events. Existing works aimed at building complete summaries of sports games from tweets, e.g. (Nichols et al., 2012; Xu et al., 2013) used simple approaches based on the observation of peaks in the tweets' volume. Even though such approaches effectively detect the most salient actions in games (e.g. goals), they fail to capture actions that are not reported by many users (e.g. shoots). Moreover, they focus only on specific information related to the events in sports games. For example, (Löchtefeld, Jäckel, & Krüger, 2015) are interested in detecting only goals, yellow and red cards in soccer games, ignoring the players involved in the actions, while (Alonso & Shiells, 2013) only detect time and keywords describing sub-events, ignoring the players that are involved.

In this chapter we perform a more complex task, which aims at creating a fine-grained, real-time summary of the sub-events occurring in sports games using tweets. We define a sub-event in a match as an action that involves one or many participants (e.g. a player, a team) at a given time, as proposed by (Dou et al., 2012). More specifically, we want to address the following research questions:

- Is it possible to build detailed sports games summaries in a unsupervised fashion, relying only on a controlled vocabulary?
- To what extent can Twitter be used to build a complete timeline of a game? Is information retrieved via Twitter reliable and sufficient?

The rest of the chapter is organized as follows. Section 3.2 reviews existing literature on the topic; Section 5.3 presents the approach we propose, and Section

5.5 outlines the experimental setting and the obtained results. Moreover, in order to demonstrate the feasibility and the validity of the approach, the chapter ends with the presentation of a web demo application that displays in nearly real-time the actions detected from tweets posted by users for a given match of Euro 2016.

5.2 Related Work

In the latest years, there has been a bulk of work on event tracking on Twitter. This section discusses works that analyze the content of tweets for tracking major events, and more specifically sports events.

Most of the approaches to track sports events are based on spike detection on the stream of messages, in order to detect sub-events. To summarize event streams, Nichols et al. (2012) propose a method that identifies spikes in Twitter feed and selects tweets from a sub-event by scoring each of them based on phrase graph (Sharifi, Hutton, & Kalita, 2010b). This method may produce unexpected summary if most of the tweets published during the spike are not related to the sub-event. Kubo et al. (2013) live sports summary are generating by prioritizing tweets published by good reporters (defined a users who posts informative tweets right after an important event has occurred in the event stream of an identified event). First, they identify spikes in the stream of an event as indicators of sub-events, and then the system tries to generate a summary by measuring the explanatory of the tweet by the presence of player's names, team names and terms related to the event. Similarly, when a spike is detected, Alonso and Shiells (2013) analyzed the tweets published during the period to identify the most frequent terms which they use to describe spikes in a tweets' histograms (spikes are considered as sub-events).

To summarize tweets related to football games, Jai-Andaloussi et al. (2015) create event clusters with similar documents (according to cosine similarity), that are then automatically classified as relevant to football actions. This method requires training data for cluster classification.

In the peculiar case of sports games, spikes do not necessarily characterize a sub-event. For example, when the crowd disagrees with the referees or a player, emotional tweets to express disagreement are published. On the other hand,

actions with low importance (e.g. a shoot) or actions produced by non-popular teams or players (e.g. Albania) may not produce peaks in the volume of tweets. Thus, approaches solely based on spikes detection will be unable to capture those actions. In our approach, we rely on Named Entities (NEs) to identify whether or not a tweet is related to a sports event. Besides, we rely on an adaptive threshold tuned according to the actions and the team (or player) of interest to evaluate whether or not the actions should be added to the timeline.

5.3 Proposed Approach

This section describes the approach we propose to detect sub-events in sport games and to build a timeline (Figure 5.1). Although the approach is general-purpose, we take as an example soccer games, so that we can use a consistent terminology (e.g. teams, penalties, players, etc.). The pipeline can be applied to any sport as long as it is represented in the Sports Markup Language.

First, a module for information extraction identifies actions (e.g. goals, penalties) and participants (e.g. player's names, teams) mentioned in tweets, setting relations between them (see examples in Table 5.1). Then, participants, actions and relations are modelled together in a temporal event-graph, taking into account also the time of the tweet. This leads to the creation of a timeline where actions and participants are connected and temporally ordered. The modules of this pipeline are described in detail in the following Sections.

Tweets	Action	Participants
kick off.... #engwal #euro2016 #teamengland	D1P	england, wales
how has ramsey not got a yellow card yet every attempt to tackle has been a foul.	CJA	ramsey
goaaaaaaaaaaal from bale woah #eng 0-1 #wal	BUT	bale, wales

Table 5.1: Example of input tweets and detected actions and participants in the game played on June 16, 2016 between England and Wales. D1P: First period begins, CJA: Yellow card, BUT: Goal.

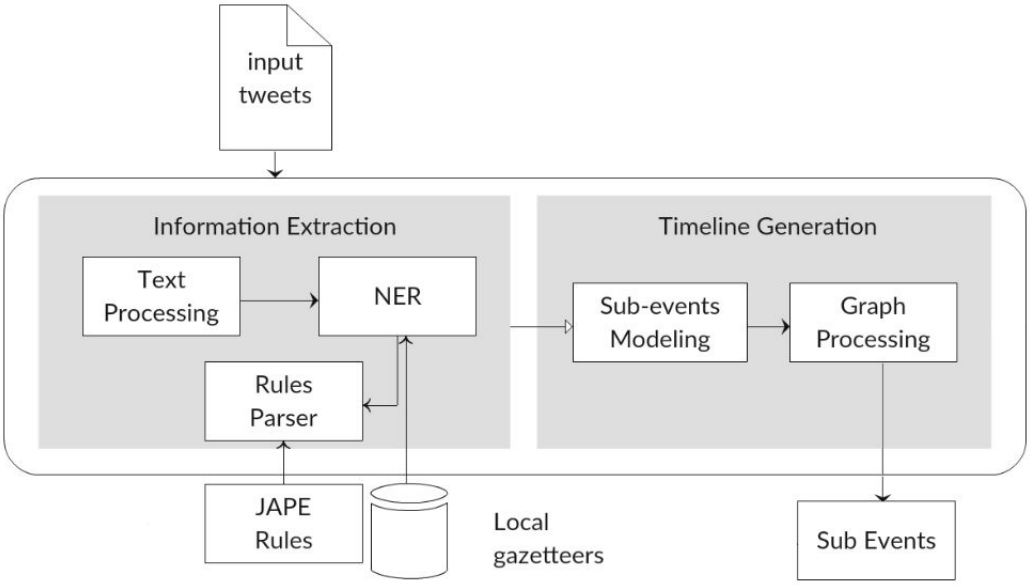

Figure 5.1: Sub-events extraction pipeline in which data is flowing in the sense of the arrows. The output are the sub-events detected from the input tweets.

5.3.1 Information Extraction

The first module of the timeline extraction pipeline retrieves participants and sub-events (or actions)[1] from tweets, and sets relations between them. In the case of soccer, actions are defined by FIFA (Fédération Internationale de Football Association), e.g. goals, penalties, yellow/red cards, etc. Participants are the actors who induce the actions. For soccer games, they are players and teams.

Text preprocessing

The input tweets are tokenized with the TweetMotifs Tokenizer (Owoputi et al., 2013), a Twitter specific tokenizer which treats hashtags, mentions, URLs, and emoticons as single tokens. In this phase, we remove URLs, non ASCII characters and re-tweets (given that re-tweets often appear a few minutes later than the original tweet (Boyd et al., 2010; Nagarajan, Purohit, & Sheth, 2010), we make the assumption that they are not very helpful for detecting real-time actions in a soccer game). Note that at this stage, we do not remove stop words since they are needed in a subsequent step (i.e. for pattern matching, described later on).

[1]We use interchangeably the terms actions and sub-events to refer to actions in a sports game.

Entity extraction

For entities recognition, we use GATE (Cunningham, Maynard, Bontcheva, & Tablan, 2002), because such tool allows for the integration of custom gazetteers. Indeed, in order to detect mentions of *actions* and *participants* in tweets we update the GATE's gazetteers using two distinct strategies namely, *offline update* and *online update*.

Offline update In the offline mode, we update the gazetteers 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 or penalties. Furthermore, we enrich the list of actions with synonyms extracted from Wordnet (Fellbaum, 1998). Then, we create appropriate major and minor labels for the extracted actions following the GATE naming convention standard and update the local gazetteers accordingly. For instance, `foot_action` and `sport_action` are respectively major and minor types for actions related to soccer. Concretely, while parsing a tweet, GATE will annotate any token in the tweets that is an action with the labels `foot_action` and `sport_action`. Updating Gate in the offline mode is performed once for each sport that we need to track.

Online update While the actions remain the same for all soccer games, participants vary according to the names of the teams and players involved in the match. Thus, we employ an online strategy to update the local gazetteers with teams' and players' names as well as their appropriate labels.

First, we use football-data API[2] that, given a soccer game in input, returns the name of the teams and their players, to extract participants involved in the game. In addition to the former names, short names and surnames of the participants are also provided by the football-data API. Also, we apply some heuristics so as to associate different spelling variations to players' and teams' names. This is done by considering separately or by combining the different parts

[2] `http://api.football-data.org`

of the players' names. For instance, *"giroud"*, *"oliviergiroud"* or *"olivier_giroud"* are all associated with *"Olivier Giroud"*, a player in the French national team. Finally, as for the offline mode, we dynamically create minor and major types for players and teams before adding them to the gazetteers. For instance, we update the gazetteer with football_player and FRA_player as minor and major types for *Oliver Giroud*. The minor label characterizes *Oliver Giroud* as a football player while the major type qualifies him as a player if the French national team.

When launching GATE, we first pre-process the data using the in-built tweet normalizer, tokenizer and PoS-tagger. Then, we apply the NER module including the two custom gazetteers that we created as described before. We also set links representing relations between actions and participants by means of JAPE (Java Annotation Pattern Engine) rules, a GATE-specific format to define regular expressions needed for pattern matching. As an example, we report below the JAPE rule matching all tokens whose type is "action" and "football_player", separated by any preposition or subordinating conjunction, all matching patterns are labeled as a "participate". This rule enables the detection of relation such as "what a goal by bale", "bale, goal" or "goaaaaaaaaaaal from bale woah".

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

): participate
```

Listing 5.1: JAPE rule to detect the relation between actions and participants in tweets exploiting preposition and subordinating conjunctions.

Since relations detected through JAPE rules tend to be very accurate, we assign a weight = 2 to edges extracted from such rules. If an action and a participant appear in the same tweet but are not matched through a JAPE rule, we set a link with a lower weight = 1, to account for a lower precision.

5.4 Timeline creation

This section describes how we build a timeline describing a match from the extracted list of actions, participants and their relationships.

5.4.1 Modeling sub-events

The output of the information extraction module (Figure 5.1) is a list of tuples $\langle a, p, t, \omega \rangle$, where a is a sport action, t the timestamp of the tweet and p the set of participants involved and ω is the weight of the edge connecting a and p. These quadruples are used to build a temporal event graph (see Figure 5.1). To retain temporal information on the sub-events, 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 for each time window. We refer to such graphs as *temporal graphs* (Verhagen et al., 2007) and we build them as follows:

- *Nodes*: Actions and participants are represented by nodes in the event-graph. First, we retrieve the nodes of the actions, and then we add the connected participants' nodes;

- *Edges*: Nodes are connected by an edge if a relation can be set in the tweets published during the time-window. The occurrence of this relation is used to increase the weight of the edges. Relationships between participants are created for actions involving 2 or more participants (e.g. a substitution).

Figure 5.2 shows a temporal graph at time-window 22 of the game between England and Wales (Game #16 on June 16, 2016). In this example, we observe edges linking participants, e.g. connecting the node "Sterling" and "Vardy", retrieved from tweets requesting the substitution of "Sterling" by "Vardy". These are both linked also to the node "England", i.e. their team.

5.4.2 Processing the Event-Graphs

At this stage, the weighted relations between actions and participants are considered as sub-event candidates. We cannot automatically include them in the timeline because they could represent opinions or wishes of the fans: when the

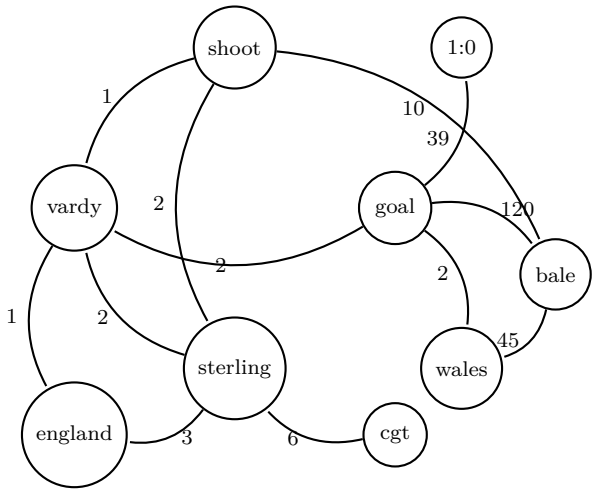

Figure 5.2: Example of the event-graph for the game between England and Wales at time-window 22.

supporters disagree with a call by the referees, they usually express their disagreement by tweeting the actions that should have been called. For example, users may ask for penalties or a yellow card after a fault by a player, as in the following tweet: *"how has ramsey not got a yellow card yet every attempt to tackle has been a foul"*. In general terms, we may assume that real sub-events in a game are reported by many users, while, on the contrary, an action reported only by a few users is more likely to be a subjective post reflecting a user's opinion (for example, s/he thinks that a player could have done a better choice).

In most of the existing work, an empirical threshold is set to measure the importance of the actions reported in tweets (Alonso & Shiells, 2013; Marcus et al., 2011b). However, we observe that the number of tweets generated for a given action is highly dependant on the game and the team or player involved. For instance, the number of tweets reporting the goal scored by Romania against France (match #1: June 10, 2016) was twice lower than the number of tweets reporting a shoot by Rooney in the beginning og the match between England and Wales. Thus, we find it useful to tune the thresholds by taking into account both the type of the action and the popularity of the teams involved in the game.

For each action belonging to a certain sport, we manually define an empirical threshold according to the importance of the action. For soccer, we can assume that a goal will trigger a higher number of tweets than a shoot. These empirical

values can be defined by domain experts for each category of the sports we want to track. Based on the predefined thresholds, the interest of the games for people and the popularity of the opponent teams, we adjust the empirical thresholds using Kreyszig standard score formula (Kreyszig, 2007) as follows :

$$\varphi_{a,t} = \epsilon_a * \frac{\eta_{g,t} - \overline{\eta}_g}{\sigma_g} \tag{5.1}$$

where $\varphi_{a,t}$ is the threshold for action a at time t of the game, ϵ_a the empirical threshold for a, $\eta_{g,t}$ the count of tweets related to the game at time t, $\overline{\eta_g}$ the mean count, σ_g the standard deviation of tweets related to the game in the past time windows.

5.4.3 Ranking Sport Actions

Let $A = \langle a, p, t, \omega \rangle$ be a quadruplet modeling an action a at time t, involving participants p and weighted by ω (i.e. the number of edges connecting a and p in the event graph). For each participant, we compute a standard score as follows:

$$z_{a,p,t} = \frac{\eta_{\omega_i} - \overline{\eta}_{\omega_i}}{\sigma_{\omega_i}} \tag{5.2}$$

where η_ω is the weight of the edge in graph G that connects nodes a and p, $\overline{\eta}_\omega$ is the mean count of all the actions of type a induced by p, and σ_ω is the standard deviation of relationship between a and p over all past time windows. Thus, we evaluate the action by taking the ratio between the standard score for each participant and the total standard scores for all the participants as follows :

$$z_{a,t} = \frac{z_{a,p_i,t}}{\sum\limits_{p_i \in P} z_{a,p_i,t}} \tag{5.3}$$

At a given time t an action is added to the timeline iff there exists at least a participant p such that $z_{a,t} \geq \varphi_{a,t}$.

As shown in Algorithm 5, we first merge the current event graph and the graph from the previous time window (Line 1). Then, from the merged graph, we collect all vertices of type $foot\_action$ and for each we retrieve all connected nodes as participants of the action (Lines 4-6). We compute the adaptive threshold for each action and a standard score for each participant using equation 5.1

and 5.2, respectively (Lines 7-9). Finally, sub-event candidates are created with participants that have a score higher than the threshold of the action (Lines 10-16). It is important to notice that, for some actions, the participants may not be required (e.g. beginning/end of periods in soccer), for such actions we consider both teams as participants in order to comply with equations (5.2 and 5.3). We remove from the event graph actions and participants involved in sub-events; on the other hand nodes that were not found as related to sub-events are kept to be processed in the next time-window. However, if a node cannot be confirmed as related to sub-events in two consecutive time windows, we consider it as noise and simply discarded.

Before putting sub-events on a timeline, we perform a final check to see whether they have not been validated in the previous time window. If yes, it means that an action overlaps two time-windows, and the timestamp of the event must be updated, matching the time of the first occurrence. We consider two events as identical if: *i)* they mention the same action and participants; *ii)* the number of tweets reporting the newest action is lower than the number of tweets on the oldest.

5.5 Experiments

This section reports on the experiments we carried out to evaluate the proposed framework. We first present the dataset, then we describe the experimental setting and we discuss the obtained results.

5.5.1 Dataset

We experiment our framework on the Hackatal 2016 dataset[3], collected during the Euro 2016 Championship. A set of keywords were manually defined, including hashtags (#Euro, #Euro2016, #football) and the names of the teams involved in the competition (e.g. France) as well as their short names (e.g. #FRA) and hashtags related to current games (e.g. #FRAROM for the game between France and Romania). For each game, tweets were collected for a two-hour time

[3] `http://hackatal.github.io/2016/`.

Algorithm 5 Algorithm to process a given event-graph to retrieve important sub-events.

function GRAPH_PROCESSING(G_t, G_{t-1}, t) ▷ G_t - Event graph at time t, G_{t-1} - Event graph at t-1, t - current time
  $G =$ **merge** (G_t, G_{t-1})
  $E = \emptyset$
  for $vertex \in G.vertices()$ **do**
    if $vertex is foot\_action$ **then**
      $P = G.neighbors(node)$
      $a = node.action$
      $\varphi_{a,t} =$ **compute** (a, t) ▷ equation 5.1
      $z_{a,t} =$ **compute** (a, P, t) ▷ equation 5.3
      for $z \in z_{a,t}$ **do**
        if $z \geq \varphi_{a,t}$ **then**
          $event = (a, p, t)$
          E **append** (a, p, t)
          G **delete** (a, p)
        end if
      end for
    end if
  end for
end function

span, starting at the beginning of the game. For comparisons and to limit the complexity of the processing pipeline, we limit our analysis to tweets in English.

Figure 5.3 shows the average number of tweets per game. Most of the tweets in the dataset were collected during matches involving teams such as France, England or Germany. Given the old tradition of soccer in these countries and the high number of supporters, such matches where strongly commented on Twitter. Table 5.2 shows the Top-50 keywords observed in the dataset after removing stop words, teams and players' names. It can be observed that the keywords distribution is driven by a few terms related to soccer actions (goal, half or penalty). It is also interesting to observe keywords that express negative sentiments among the most frequent terms ("fraud", "spam") or terms that express wishes ("would", "could"). Such terms mainly appear in tweets that express the supporters' disagreements towards players, referees or managers.

The dataset also contains the summary of the salient actions in each game, retrieved from journalistic reports (e.g. LeFigaro[4]). We consider these summaries

[4] `http://sport24.lefigaro.fr`

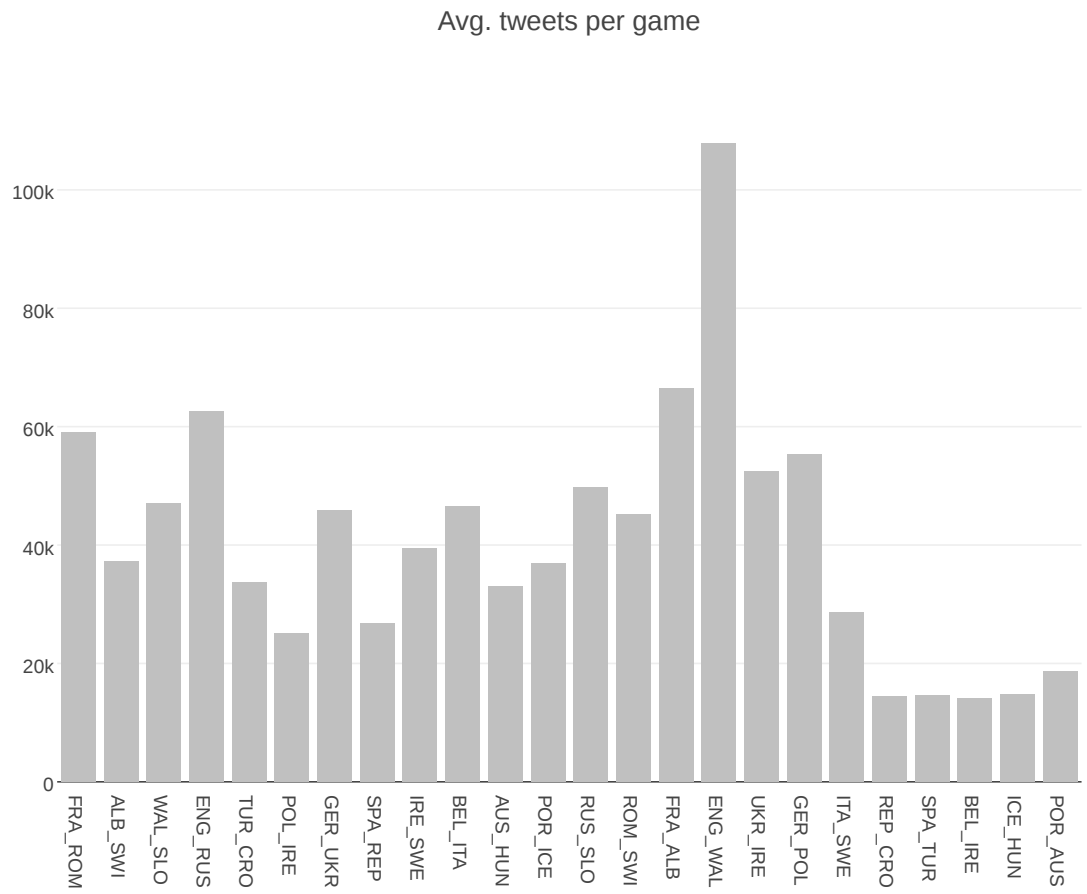

Figure 5.3: Average number of tweets per game in the dataset. Matches are reported on the X-axis and the number of tweets on the Y-axis. Matches are temporally ordered according to the time they were played in the competition.

as the ground truth while evaluating our approach. These summaries are defined as a set of triples ⟨`time, action, participant`⟩ where "time" is the time the sub-event occurs, the "action" is the type of the sub-event and "participants" are players or teams involved in the action. The sub-events in the ground truth are listed in Table 5.3. As an example, we report in Table 5.4 a few examples of the sub-events in the game between "England" and "Wales".

Terms	Freq.	Terms	Freq.	Terms	Freq.	Terms	Freq.
goal	367471	half	64832	first	53625	fraud	46273
hack	40586	game	38658	penalty	36751	like	34636
injury	33049	spam	33000	time	32126	second	31054
today	29469	good	27978	great	25772	foul	25199
flood	23645	scam	23270	reach	20550	would	20477
come	20052	need	19184	score	18969	well	18623
match	17996	different	17351	card	17228	virus	16501
ways	16471	team	16462	several	16034	earthquake	15980
back	15877	libra	15627	football	15058	best	15005
last	14901	still	14746	another	14505	free	14336
could	13992	think	13597	watch	13474	play	13435
make	13305	going	13290	life	12859	know	12798
people	12224	tournament	12163	really	12085	ball	11989

Table 5.2: Top keywords in the Euro 2016 dataset.

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 5.3: Ground truth actions in the Euro 2016 dataset.

Time	Action	Participants	Summary
15:02	D1P	–	Beginning of the first period
15:09	TIR	Sterling	Shot by Sterling for England
...	...	...	...
15:44	BUT	Bale	First goal by Bale for Wales
15:48	F1P	–	End of the first period
16:04	CGT	Sterling;Vardy	Vardy substitutes Sterling
16:18	BUT	Vardy	Goal by Vardy for England
...	...	...	...

Table 5.4: A few examples of the sub-events that occurred in the game between England and Wales.

5.5.2 Experimental Settings

We simulate the Twitter stream by grouping the tweets related to a game in intervals of two minutes, which we refer to as *time-windows*. Thus, we collect all the tweets published in a time-window in a single document which we give in input to our algorithm. In the preprocessing phase, we remove re-tweets if the original tweet is already in the collection, and we consider one tweet per user in a time window. The input tweets are then analyzed with GATE. We use the JGraph library (Naveh et al., 2008) to create the event-graph. At each time-window, we create a new graph to model the relation between actions and participants detected in tweets. We process the event-graph with Algorithm 5 to detect real sub-events found in tweets.

5.5.3 Evaluation Strategies

We report on two different evaluation strategies. In the first one, we compare the output of our framework against the state of the art approach described in (Alonso & Shiells, 2013). There, the authors detect sub-events by identifying spikes in the Twitter stream. Since they do not detect participants, in this first comparison we also limit our evaluation to the action timeline, letting out additional information. We also compare the results with the gold standard timeline from manually created summaries by sports journalists. We show the results through a graphical representation for three sample matches (Figures 5.5, 5.6 and 5.7).

In the second evaluation strategy, we evaluate our approach against the gold standard data (see above) in term of precision, recall and f-measure. This time we include also the sub-event type, the time and participants information. We adopt three evaluation strategies, namely *complete* matching, *partial* matching and *loose* matching. In the complete matching mode, we evaluate each sub-event detected by our system by taking into account the type of the sub-event, the participants and the time. A sub-event is considered correct if all three elements are correctly identified. In the partial mode, we consider the time and the type of the sub-events; and in the loose mode, we only consider the type. We set the error margin to 2 minutes while comparing the time, since this is the duration of the time-windows used to build the temporal graphs. We report P/R/F1 for the

	Loose			Partial			Complete		
actions	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
goal	0.745	0.512	0.549	0.670	0.456	0.493	0.623	0.405	0.444
card	0.758	0.560	0.622	0.693	0.506	0.568	0.600	0.433	0.516
subt	0.859	0.629	0.693	0.627	0.460	0.510	0.501	0.374	0.438
shoot	0.643	0.203	0.292	0.571	0.185	0.264	0.548	0.167	0.243
period	0.814	0.656	0.706	0.655	0.517	0.562	0.585	0.462	0.523

Table 5.5: Experimental results of our approach for 24 games in the first stage of the Euro 2016 dataset

same sample matches described above, as well as an average of the scores for 24 matches in the first stage of the competition (Table 5.5).

5.5.4 Results and discussion

The overall evaluation concerning the first 24 games in the Euro 2016 Championship (Table 5.5) shows that the approach is very accurate in some cases, while it suffers from low performance, especially recall, in other settings. If we compare the different actions (left-most columns in the table), we observe that the best performance is obtained when recognizing the start and the end of the match (last line in the table). For other actions, the performance varies across the three evaluation modes. For example, when considering participants to *shoot* actions, the approach fails to identify the correct player, probably because other players such as the defender and the goalkeeper are likely to be mentioned in the same tweet. In Figure 5.4 we provide a global overview of Precision and Recall obtained on the whole dataset with the different evaluation strategies, with each dot corresponding to a match.

We further focus on three sample matches, which were selected to compare our approach with (Alonso & Shiells, 2013). We plot in Figures 5.5, 5.6 and 5.7 the sub-events detected by (Alonso & Shiells, 2013), those detected by our approach as well as those present in the gold standard. We also report in Tables 5.6, 5.7 and 5.8 P/R/F1 measures according to the loose, partial and complete evaluation strategy.

The first game considered was played between England and Wales and gained particular attention on Twitter. Figure 5.5 shows the distribution of tweets during the game (in gray), distinguishing between tweets explicitly mentioning England

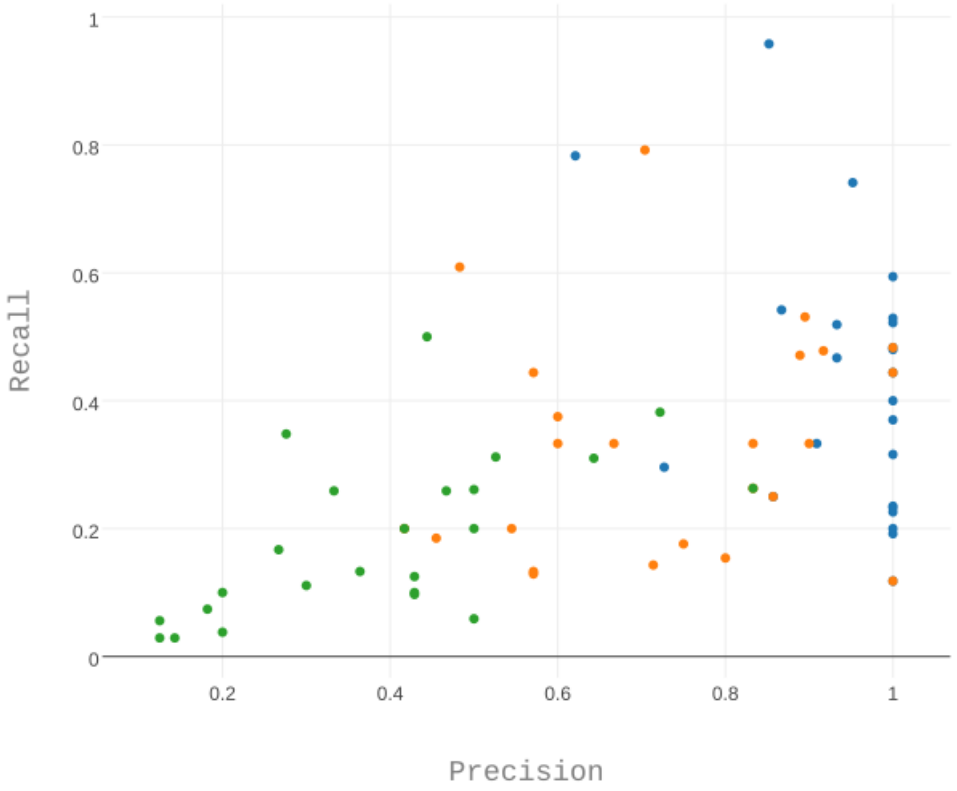

Figure 5.4: Precision Recall chart of the performances of our approach. X-axis is the average precision and Y-axis the average recall. Blue dots represent the loose matching, orange dots the partial matching and green dots the complete matching.

(red line) and Wales (green). The blue dots correspond to the sub-events identified by (Alonso & Shiells, 2013)'s approach, while those detected by our approach and the ground truth are represented with yellow and green dots, respectively. The graphical representation shows that there is a significant correspondence between the sub-events detected by our approach and the gold standard ones. We can also observe that (Alonso & Shiells, 2013) fails to detect sub-events that do not produce spikes in the volume of tweets (e.g. shoots).

Table 5.6 shows for the same match the average performance (P/R/F1) of our approach compared to the ground truth. In this case, our performance is affected by problems in detecting actions of type *substitution* and *shoots* (tweets mostly contain complains by England fans against *Kane* and *Sterling* who seemed to have missed a lot of opportunities to score for England in the first period).

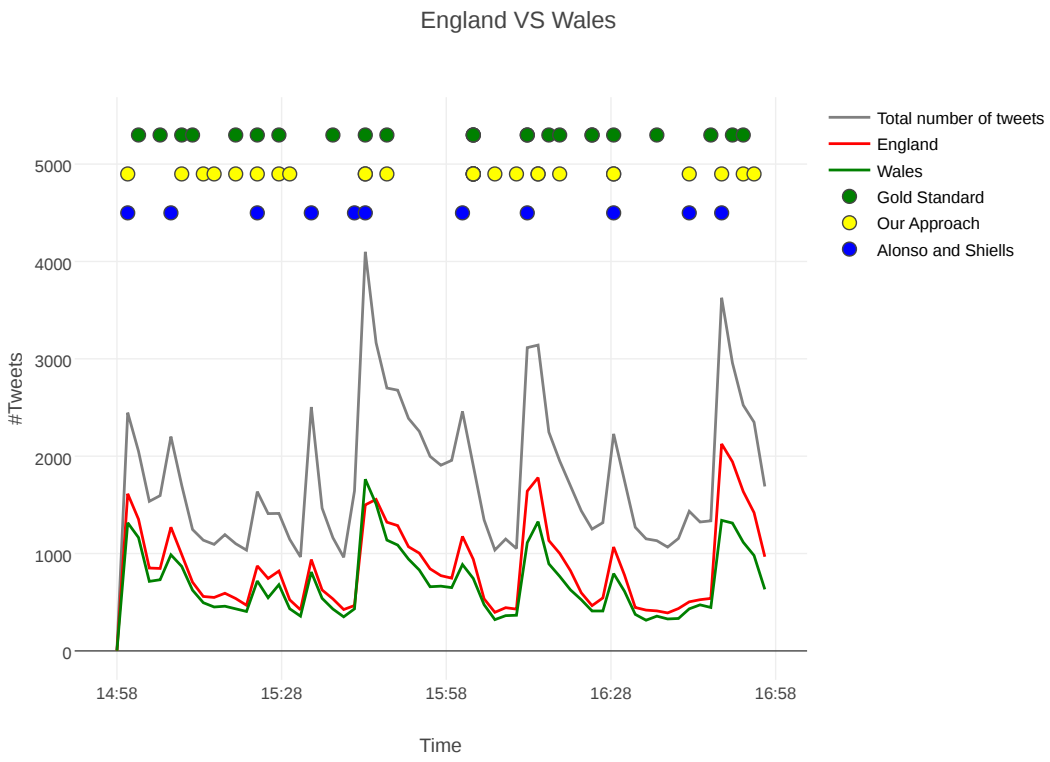

Figure 5.5: Sub-events for the game England vs Wales.

A second example is the match between France and Romania, represented in Figure 5.6. Although the game was quite debated on Twitter, a few spikes were detected in the stream. In fact, during the first period the teams were barely mentioned, as indicated by the red and green curves on the graph. Instead, other teams were mentioned, which were not directly involved in the game. The second period seemed to be more interesting in terms of sub-events. In Table 5.7, we show the performance of our approach on this game. We obtain a 91.3% precision in the loose mode, since we detect 23 out of 34 sub-events in the game compared to 9 identified by (Alonso & Shiells, 2013), and 21 of the detected sub-events were associated to the correct actions. However, the latency between the sub-events detected by our approach compared to the ground truth contributes in decreasing the performance of our approach in both intermediate and complete matching. For example, there is a huge peak at time 22:24 when the player *Stancu* equalizes for Romania, but we detect this action four minutes later since most of the tweets

Methods	Prec	Rec	F-score
loose	0.852	0.958	0.902
partial	0.630	0.708	0.667
complete	0.444	0.500	0.470

Table 5.6: Evaluation performance for the game between England and Wales.

in that time span discuss the penalty issue rather than the goal. Many sub-events in the game, mostly actions by Romania, were not mentioned in any tweet in the dataset. For example, no tweets mentioned the shoot by Pintilii at time 21:04.

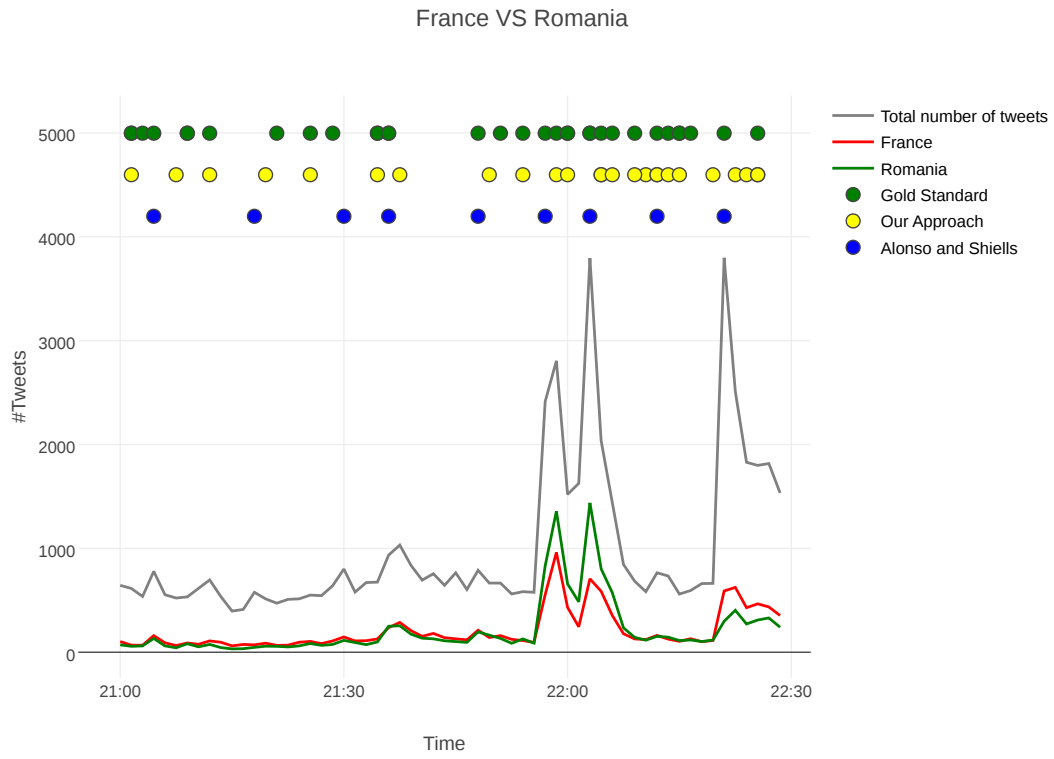

Figure 5.6: Sub-events for the game France vs Romania.

As a third example, we consider the game between Belgium and Italy, that was less popular in terms of tweets than the ones described so far. A few peaks are detected in the game, as shown in Figure 5.7. This affects negatively the number of sub-events found by (Alonso & Shiells, 2013), while our approach proves to have a better coverage, even if recall is on average lower than for the

Methods	Prec	Rec	F-score
loose	0.913	0.656	0.763
partial	0.696	0.500	0.582
complete	0.609	0.438	0.510

Table 5.7: Evaluation performance for the game between France and Romania.

other matches. In most cases, we detect mentions of the actions, but we fail to detect the participants. Table 5.8 shows the overall performance of our approach. In the ground truth there were only a few tweets related to this game, and $\sim 50\%$ of them were shoots. Our approach failed to identify such events, impacting on the recall. On the other hand, all the events detected were correct, accounting for 100% precision in the loose mode, and $\sim 85\%$ in the complete mode.

Methods	Prec	Rec	F-score
loose	1.000	0.448	0.619
partial	0.923	0.414	0.572
complete	0.846	0.379	0.523

Table 5.8: Evaluation performance for the game between Belgium and Italy.

5.6 DEMO: "Follow Your Game on Twitter"

This section describes a system that implements the approach described in the previous section, that builds a timeline with salient actions of a soccer games discussed on Twitter. It results 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. The system is implemented as a client-server application, where the server component implements the framework for processing the tweets described before (Section 5.6.1), sends the result to a client component, that displays them to final users (Section 5.6.2). In the remainder of this section, we describe the technical implementation of each component.

5.6.1 The Server Component

The server component has been implemented following the pipeline described in the previous section, and displayed in Figure 5.1.

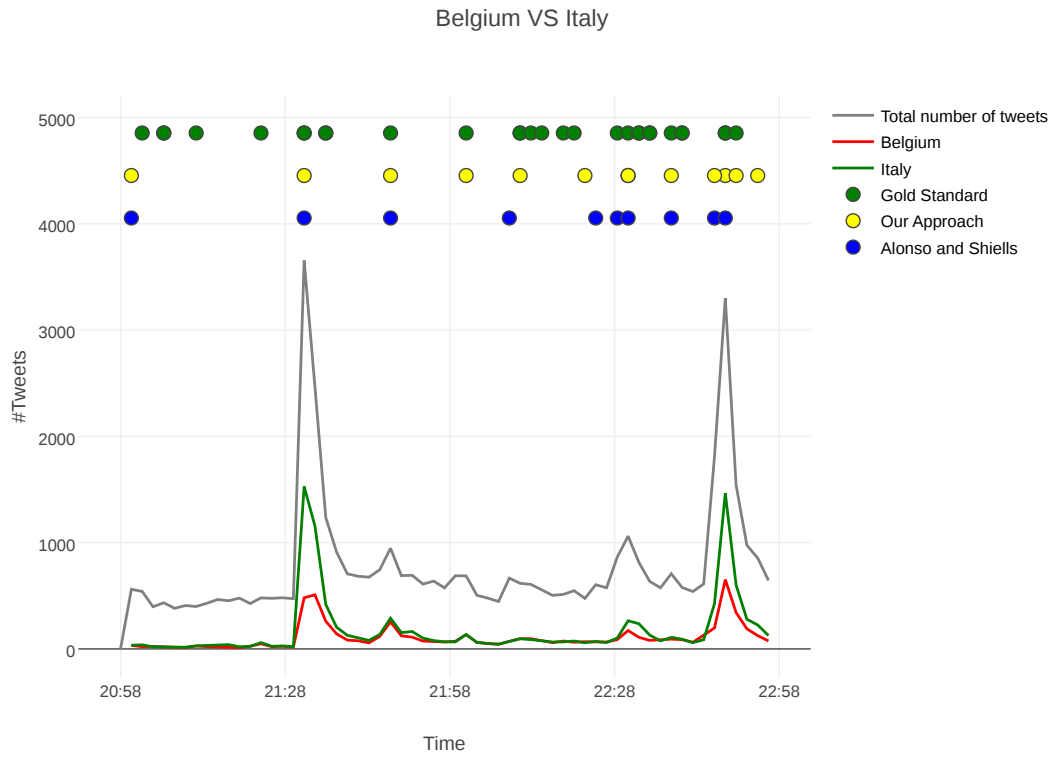

Figure 5.7: Sub-events for the game Belgium vs Italy.

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 opponent teams to be provided (e.g. England, 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. While posting tweets concerning a soccer game, fans tend to use abbreviations for teams' names (e.g. ENG or WAL) more often than full names (e.g. England, Wales). Also, fans may use common patterns to refer to soccer games by combining teams' names into single hashtags (e.g. #engwal, #waleng, #englandwales,...). Based on these observations, we define heuristics to create keywords using different combinations of the team names. For example eng, wal, engwal, waleng and many others can be used to query tweets related to the match between England and Wales. Furthermore, terms related to a given competition (e.g. euro, euro2016) or soccer actions (e.g. goal) are used as parameters to retrieve tweets related to a game.

Input tweets are pre-processed in order to remove noise and redundant information as previously explained. Then, For detecting mentions of actions and participants in tweets, we rely on GATE (Cunningham et al., 2002) (Section 5.3.1), updating the gazetteer both in a offline and an online mode. In the offline mode, we extract actions related to soccer games from the SportsML vocabulary, which we enrich with synonyms extracted from Wordnet. In addition, we manually inspect the list of actions to remove out-of-context terms, for instance, "finish" is retrieved as a synonym for "goal", but we do not consider it as a valid term for soccer actions. For soccer games, we extract actions such as goal, substitution, yellow/red cards, or penalty. The online mode is unsupervised and is devoted to update the local gazetteer with participants (e.g. players and teams) involved in the game. To this aim, we dynamically query the football-data API to obtain the list of players in each team (see Section 5.3.1). Once the players are obtained, we update the gazetteer with their names, surnames and spelling variations using a set of heuristics as explained in Section 5.3.1. We initialize GATE with the custom gazetteers and use its in-built NER to identify mentions of actions and participants in the tweets. We process all the tweets published in a time-window (e.g. every 2 minutes) as a single XML document, which we provide as input to GATE. We define an internal routine to parse the annotated documents in order to extract reported actions and their relation with the players and/or teams involved in the game, which we model in a graph. Whenever actions and participants occur in the same tweet, we extract them and connect them through an edge in the graph.

We split the timeline in fixed time windows of two minutes and at each span we create a temporal graph that models relations between actions and participants observed in the tweets published during the time-window. Specifically, nodes in the graphs are either actions or players and edges represent relations between the nodes. Edges are weighted by the number of times the relationship has been observed in the current time window. In addition, the edges hold metadata about the relations such as the list of tweets that report the action or the weight. Also nodes hold metadata on the type of participants, which can be a player, a team or a soccer action.

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 a threshold. An action is confirmed iff its pound is above an adaptive threshold. We computed the pound of an action with Equation 5.3 and the adaptive threshold is tuned by taking into account both the type of the action and the popularity of the teams involved in the game (Equation 5.2).

Technical implementation

We implement the server component in the Java programming language. As local database, we use MongoDB[5] to store tweets and we use Twitter4J[6] to collect tweets from the Twitter streaming API. We use the JGraph library (Naveh et al., 2008) to generate and process the event-graph. The server component also implements a socket listener allowing a bidirectional communication with the client component.

5.6.2 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 (e.g. goals, penalties) in the match and the statistics (e.g. the current score, the ball possession) related to the selected game. 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[7], CSS3 [8] and Java Script libraries such as angularJS[9]. Data between the client and the server are exchanged using web socket. The web interface is inspired by the website of The Guardian[10] 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.

[5]https://www.mongodb.com/
[6]http://twitter4j.org/en/index.html
[7]HTML5: `https://www.w3.org/TR/html5/`
[8]`https://www.w3.org/Style/CSS/current-work`
[9]Angular JS `https://angularjs.org/`
[10]The Guardian `https://goo.gl/MWc6dN`

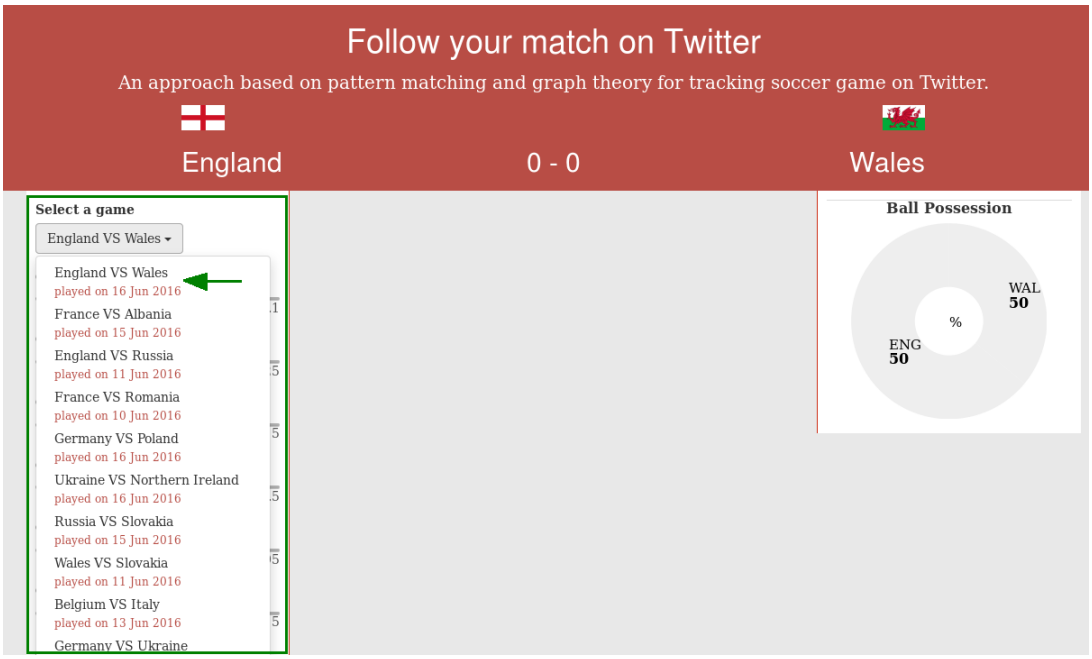

Figure 5.8: Screenshot of the web demo. To start, the user can select from a list the matches he wishes to track (for demo purposes, the list contains the matches from the first stage of the Euro 2016 championship.

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. The web client also implements a socket listener that enables it to automatically update the interface, thanks to angularJS data binding.

In order to better describe the different demo components, we mark in the screenshot of the web platform (see Figure 5.9) 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 collected during the first stage of the Euro 2016 championship[11]. In the current implementation, the server can track

[11]EURO 2016 dataset `https://github.com/HackaTAL/2016/tree/master/Tweets`

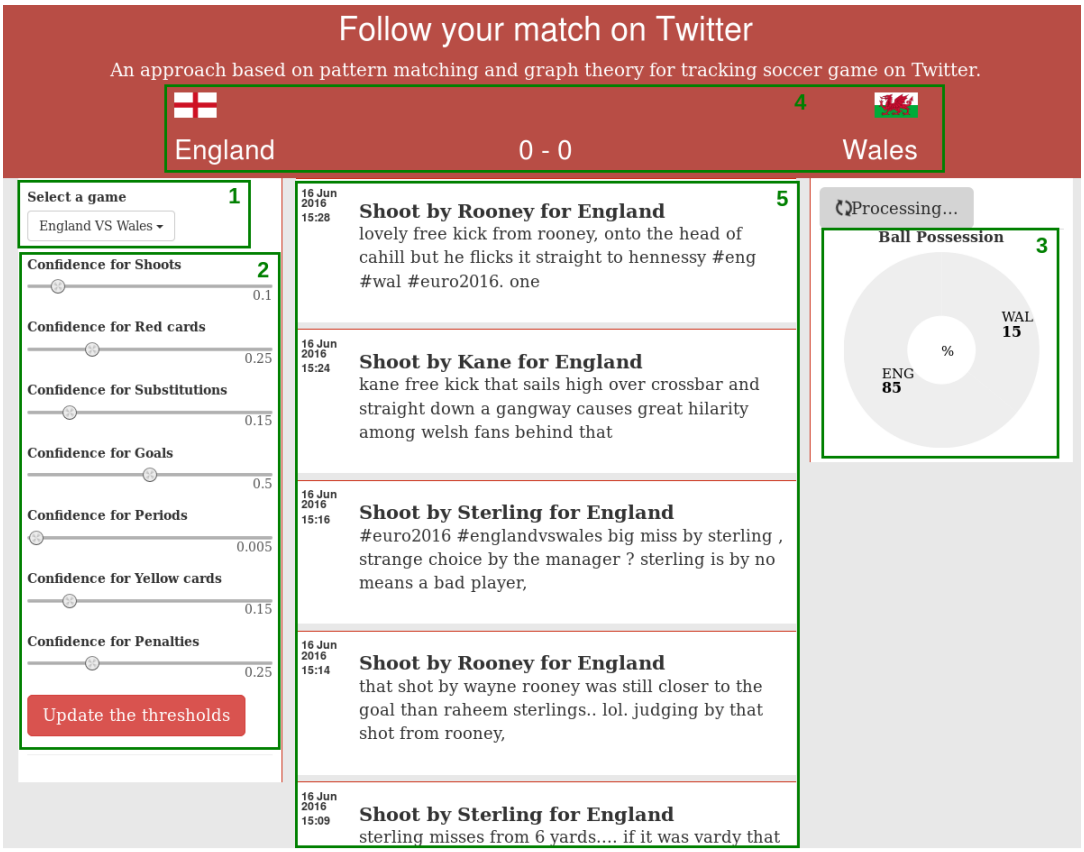

Figure 5.9: Screenshot of the timeline generated for the soccer game between England and Wales during the first stage of the Euro 2016 championship. The screenshot contains five main blocks : *i)* game selection, *ii)* threshold settings for actions selection, *iii)* ball possession, *iv)* salient actions, and *v)* updated score of the match

a single game at a time, which can be selected from the list at the top of the left panel (see Figure 5.8 for the game selection list, and partition 1, Figure 5.9). Example matches are Switzerland-Albania or England-Wales.

Action thresholds

As introduced before, the action threshold is the minimal confidence value, based on frequency, 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 in a detected action (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.

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 certain 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 top right corner of the page and constantly updated (partition 3, Figure 5.9). 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 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 5.9) 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 5.10). We retrieve the current score of the game from the tweets.

Figure 5.10: The score is constantly updated, and the players' names are added below the teams for which they score.

Salient actions

Actions detected in the tweets are displayed in a reversed chronological order to the end user (partition 5, Figure 5.9). 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 & Tarau, 2004) to produce a summary of the event. For instance, for the action "Shoot by Ramsey 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"*.

5.7 Conclusions

In this Chapter we have described a framework to generate timelines of salient sub-events in sports games exploiting information contained in tweets. We use the GATE system enriched with information provided by domain knowledge bases to detect mentions of actions and participants, as well as their relations in the sports domain (e.g. players and teams). Exploiting the self-contained nature of tweets, we made the hypothesis that entities that appear in the same tweets can be considered as related. We model the relationships between the entities in a temporal graph and use adaptive thresholds to measure the veracity of actions reported in tweets.

Experiments on a dataset of tweets collected during the EURO 2016 Championship proved that our approach is able to accurately detect sub-events in sports games when compared to news on the same events reported by sports media. While previous approaches focused only on detecting the type of the most important sub-events, we extract and model a richer set of information, including almost every type of sub-event and participants involved in the actions.

Also, we have described a demo that demonstrates the effectiveness of the

approach. This demo implemented as a client-server application allows end users to select a game and track the most salient actions through a dedicated web interface. In addition to the sub-events, the demo also computes statistical information regarding a given soccer game such as the ball possession, the current score as well as the scorers.

As for future work in this direction, our approach could be extended to cover other sports such as American football and basketball. To this end, it would be possible to extend our rules to detect relations between action and participants according to the rules that govern the games. Then, our framework can be configured to collect data from knowledge bases that provide information for these sports categories.

Chapter 6

Conclusions and Perspectives

This Thesis presents and discusses the more relevant results of our research in event detection, classification and tracking from short text messages on Twitter.

More precisely, we have presented a set of methods that rely on the identification of NE mentions in tweets and the relationships among them to define an event, with the goal of separating tweets that discuss about an event from the others, classifying them into event categories, extracting fine-grained events, and track them in Twitter. Such methods take advantage of information available in knowledge bases in the Linked Open Data to enrich the context of the NE mentioned in the tweets.

As a first contribution of this Thesis, we have presented a framework for identifying and classifying event-related tweets by exploiting information automatically leveraged from DBpedia and YAGO. We evaluated the supervised approach we propose in different classification tasks. We observed that information extracted from YAGO contributes better to improve classification performances than DBpedia. Possible reasons for that are: i) the better coverage of YAGO, and ii) YAGO class hierarchy is deeper than the DBpedia ontology, which has an impact especially when using specific categories for the multi-class classification task. In all the experiments, LSTM-RNNs outperform SVM and NB, confirming previous findings on the effectiveness of RNNs when applied to several NLP tasks (Socher et al., 2013). Our experiments on different classification tasks show that performing binary classification first and then passing the output to the second classification step in a pipeline is more accurate than the single-step model.

As a second contribution, we have described a model for detecting open-

domain events from tweets by modeling relationships between NE mentions and terms in a directed graph. The proposed approach is unsupervised and can automatically detect fine-grained events without prior knowledge of the number or type of events. More specifically, we have exploited the local contexts of the NEs, i.e. the words that surround their mention in tweets, to create event graphs. We have then used a PageRank-like algorithm to split the event graphs in sub-graphs and exploited the connections between nodes to identify events. Furthermore, by exploiting the semantic classes of the NEs, we have extracted meaningful properties of the events such as their types, geographical locations, times and people involved. Our experiments on two gold-standard datasets show that the approach yields state-of-the-art results.

As a third contribution, we have proposed a semi supervised method to generate timelines of salient sub-events in sports games exploiting information contained in tweets. We use the GATE system enriched with information provided by domain knowledge bases to detect mentions of actions and participants, as well as their relations in the sports domain (e.g. players and teams). Exploiting the self-contained nature of tweets, we made the hypothesis that entities that appear in the same tweets can be considered as related. We model the relationships between the entities in a temporal graph and use adaptive thresholds to measure the veracity of actions reported in tweets. Experiments on a dataset of tweets collected during the EURO 2016 Championship proved that our approach is able to accurately detect sub-events in sports games when compared to news on the same events reported by sports media. While previous approaches focused only on detecting the type of the most important sub-events, we extract and model a richer set of information, including almost every type of sub-event and participants involved in the actions. Also, we have described a demo to demonstrate the validity of the approach. This demo implements a client-server application that allows end users to select a game and track the most salient actions through a dedicated web interface. In addition to the sub-events, the demo also computes statistical information regarding a given soccer game such as the ball possession, the current score as well as the scorers.

To summarize the big picture of our research on Event detection, classification and tracking on Twitter messages, the three contributions described in this Thesis correspond to three steps of a pipeline, that takes in input tweets from the Twitter

API, classifies them as event or non event applying a supervised model, clusters event-related tweets into fine-grained events, and allows to the user to track such event on Twitter thanks to the event tracking model. We hope that the analysis of the different dimensions of the problem we provided may bring interesting elements to enhance future work in this direction.

6.1 Perspectives

The methods described in this Thesis have achieved promising results, as demonstrated in the different experiments on standard datasets. However, improvements can be obtained by incorporating more semantic information in the framework. Short-term improvements in this direction on the three tasks addressed in this Thesis can be outlined as follows: *i)* in the event classification model, exploiting domain-specific ontologies for certain categories, e.g. for geographical names. Indeed, domain-specific KB such as Geonames (for geographic entities) contain much more resources and generally better organized than general-purpose KB; *ii)* the event clustering approach could be improved by investigating whether linking terms to ontologies (e.g. DBpedia, YAGO) can help in detecting different mentions of the same entity, for instance "German chancellor" and "Angela Merkel". This can be used to reduce the density of the event graph while augmenting the dangling score of the NEs; *iii)* finally, the event tracking method could be extended to cover other sports such as American football and basketball. To this end, the hand-crafted rules should be extended in order to detect relations between action and participants according to the rules that govern the games. Also, the model should be configured to collect data from knowledge bases that provide information for other sports categories.

As for long-term improvements, our work could benefit from the use of location information in tweets to approximate the geographical place where events take place, to suggest to the end users the place of an event (e.g. a concert or a soccer game). For events such as natural disasters or accidents, the geographical location is critical for a better coordination of the rescue activities. However this task is not trivial due to ambiguity on the mentions of geographic terms in text, specifically in tweets. There are two types of ambiguity: geo/geo ambiguity and geo/non-geo ambiguity. Geo/geo ambiguity arises when different places have the

same name (e.g., there are 63 different resources for Washington in Geonames). Geo/non-geo has to do with place names that can have non geographical meaning (e.g., Washington can be either a place or a person (e.g. Denzel Washington)). The entity linking approach adopted in Chapters 3 and 4 has used the most common sense to perform disambiguation, which might lead to unexpected results for ambiguous locations.

Moreover, it could be interesting to enrich the events on Twitter with additional information as photos and videos provided by other social media platforms, by extracting the content from the related hyperlinks or query other platforms (as Youtube or Instagram) with relevant keywords extracted from tweets.

Last but not least, also linking new events to related events in the past would be a valuable task to investigate. For instance, "Mala Yousafzai graduated from high school (2017) " and "Mala Yousafzai, a 14-year old activist girl shoot in Taliban (2012)". However linking simple occurrences of the NEs would not be sufficient for this task, otherwise all events mentioning e.g. Barack Obama would be considered as related, which would not be pertinent. Instead, linking events related to Obama's election and his investiture would be more interesting. As a first attempt, hyperlinks in tweets could be exploited for this task, since web pages describing an event generally contain cross-references to previous related event. For instance the NY Times article[1] that reports on the event related to "Malala Yousafzai graduated from highschool" lists a set of links to past events that involved Malala including her peace Noble Price or her shooting. However, the very final goal would be to generate such lists automatically.

[1] `https://www.nytimes.com/2017/07/07/world/middleeast/malala-yousafzai-graduates.html?smid=tw-nytimes&smtyp=cur`

Appendix A

Résumé étendu de la thèse en français

Les réseaux sociaux comme Twitter, Facebook ou Google Plus ont été adoptés par les communautés en ligne pour partager, consulter et commenter des informations sur les faits et événements à travers le monde. La liberté offerte à tout utilisateur de pouvoir créer et partager des informations, contribue à la génération massive de données et en temps quasi réel sur le Web. Ainsi, les réseaux sociaux, plus particulièrement Twitter, sont considérés comme une source importante de données.

Dans les récentes années, plusieurs recherches ont été consacrées à l'étude d'approches permettant de traiter les données des réseaux sociaux afin d'extraire des informations utiles et pertinentes dans le but de développer de nouveaux services comme l'analyse de sentiment, de tendance des utilisateurs ou la détection d'événement. Cependant, la nature informelle de ces messages dû à l'usage d'un vocabulaire non standard comme l'utilisation de raccourcis, les fautes d'orthographes ou l'utilisation de jargons, constitue un défi majeur dans le traitement de ces données.

D'un autre côté, le Web contient des bases de connaissances structurées à partir de concepts extraits des ontologies permettant de représenter sémantiquement les ressources.

Dans cette thèse, nous avons étudié un ensemble d'approches permettant de traiter automatiquement les contenus de Twitter (ou tweets) pour la classification, l'identification et le suivi d'événements. Nous avons adopté la définition

proposée par Dou et al., qui dans le contexte des médias sociaux, définit un événement comme “Une occurrence causant un changement dans le volume de données qui discute un sujet donné à un temps donné. Cette occurrence, est caractérisée par le sujet, le temps, et souvent associé à des entités telles que personnes et les lieux”. Cette définition montre une relation entre les événemetns et les entités nommées que sont les personnes, organisations, considérées comme acteurs de l'événement. Nous avons décrit un ensemble de méthodes qui permettent d'exploiter la présence des entités nommées et les informations contenues dans les bases de connaissances du Web pour enrichir le contexte des entités nommées afin d'améliorer la performances des modèles d'apprentissage à la fois supervisés et non supervisés.

Extraire les tweets relatifs aux événements

La première contribution de la thèse est une approche permettant d'identifier les tweets qui discutent des événements et de les classifier en catégorie d'événement se basant sur les catégories adoptées dans la communauté de détection et de suivi d'événement à partir de contenus textuels. Cette démarche est primordiale pour la détection d'évènement car la plupart des tweets ne traitent pas d'événements, mais de préférence d'autres aspects relatifs aux activités personnelles des utilisateurs (Java et al., 2007).

Cette contribution a permis de répondre à trois principales questions de recherches : i) Comment maintenir la performance d'un modèle supervisé quand les données d'entrainement et de tests proviennent des sources différentes ? ii) Comment réduire l'impact du sur-apprentissage dans les modèles supervisés. iii) Comment les informations contenues dans les bases de connaissances du Web sémantique peuvent contribuer à l'amélioration des modèles supervisés?

Pour répondre à ces questions, nous avons proposé une approche de généralisation sur les entités nommées par leurs types sémantiques dans des ontologies. Plus concrètement, notre modèle se construit en deux étapes : i) La première étape consiste au traitement et à l'enrichissement des tweets en utilisant des techniques du Traitement Automatique de la Language Naturelle (TALN) et du Web sémantique ; ii) nous avons utilisé le contenu enrichi pour entrainer un modèle supervisé afin de détecter les tweets qui se rapportent à des événements et de les

classifier en catégories d'événements.

Afin d'identifier les entités nommées dans les tweets, nous avons utilisé NERD-ML, qui, pour un tweet donné en entrée, identifie les mentions d'entités nommées et les associe à des ressources de DBpedia. Puis, exploitant le langage SPARQL nous interrogeons DBpedia pour extraire les types des entités, en fonction de la hiérarchie de concepts des ontologies DBpedia ou YAGO. Finalement, nous avons remplacé les entités nommées par leur concept se basant sur différente stratégies : i) les entités nommées sont remplacées par leurs types génériques (ex. Obama est remplacé par Person) ; ii) les entités sont remplacées par leurs types spécifiques (e.g. Obama est remplacé par Président).

La deuxième étape de notre approche consiste à entrainer un modèle supervisé à partir du contenu des tweets résultant de la phase d'enrichissement. Pour cela, nous avons représenté les tweets en utilisant les Word Embeddings. Nous avons créé cinq variantes de Word Embedding en fonction des stratégies de remplacement d'entités nommées de la phase d'enrichissement. Finalement, pour chaque stratégie de remplacement, nous avons expérimenté plusieurs algorithmes d'apprentissage comme : Naive Bayes, Machine à Vecteur Support (SVM) et les réseaux de neurones. Nous avons exploité ce modèle dans quatre scenarios :

1. Modèle binaire : les tweets sont classifiés en événements ou non événements ;

2. Modèle multi-classe : les tweets rapportant des événements sont classifiés en catégorie d'événements ;

3. Modèle combiné : le modèle binaire classifie des tweets en événement ou non événement et le modèle multi-classe les classifie en catégorie d'événement ;

4. Modèle unique : un modèle unique classifie les tweets en catégorie d'événement, incluant une catégorie supplémentaire pour les tweets qui ne rapportent pas des événements.

Nous avons conduit des expériences sur deux jeux de données de l'état de l'art collectées durant deux périodes différentes, l'un en 2011 et l'autre en 2012. Dans un premier temps, nous avons entrainé et testé nos modèles sur des tweets provenant du même jeu de données, en utilisant le cross-validation. Dans cette

configuration, notre approche a obtenu des résultats légèrement inférieurs à ceux de notre méthode de référence. Dans un second temps, nous avons évalué le modèle résultant sur des tweets provenant d'un jeu de données différents de celui de l'entrainement. Les expériences ont montré que lorsque les données d'entrainement et de test proviennent de sources différentes, notre modèle obtient de meilleurs résultats que le modèle de référence. En général, le remplacement des entités nommées par leurs types sémantiques permet de réduire l'impact du sur-apprentissage dans les modèles supervisés.

Identification d'événements dans les tweets

La deuxième contribution de cette thèse est une approche permettant d'identifier et de caractériser des événements sur Twitter comme par exemple, les élections Américaines ou la mort de Amy Winehouse. Suivant la définition d'événement qu'on a adopté dans cette thèse, notre approche consiste à déterminer le type de l'événement (quoi), sa date (quand), le lieu (où) et éventuellement les entités concernées (qui).

Les approches existantes permettant d'identifier des événements sur Twitter sont généralement basés sur le regroupement de tweets autour de mots-clés relatifs aux événements (Parikh & Karlapalem, 2013) ou autour des entités nommées (McMinn & Jose, 2015). Bien que ces approches permettent de détecter des événements qui génèrent un volume important de tweets et pour lesquels des mots-clés peuvent être facilement identifés, cependant, elles sont moins performantes sur des événements qui ne produisent pas de pics dans le volume de tweets. D'un autre côté, ces approches ne permettent pas de détécter d'identifier des événenements différents partageant des mots-clés identiques ou impliquant les mêmes entités; comme par exemple les événements liés à "l'attaque sur Malala" et "son opération chirurgicale" se sont produits dans la même période et concerne la même personne.

Nous avons exploité la théorie de graphe pour construire des graphes temporels modélisant les relations entre les termes des tweets. Les travaux existants ayant utilisé les graphes pour la détection d'événement dans les tweets utilisent la position des termes dans les tweets pour créer les graphes, ce qui résulte en la création de graphes denses et générant un coût de traitement élevé. De préférence, nous

avons exploité le contexte des entités nommées dans les tweets pour déterminer les nœuds du graph ainsi que les relations entre eux. Les graphes sont générés comme suit :

1. Nous considérons les entités nommées et k termes qui entourent leurs mentions dans les tweets comme nœuds du graphe.

2. Les nœuds d'un graphe sont connectés s'ils apparaissent dans le contexte d'une entité nommée.

3. Le poids d'un lien est défini par le nombre de fois les deux termes apparaissent dans les tweets.

La seconde phase de notre approche consiste à analyser le graphe résultant pour y extraire les nœuds décrivant des événements. Nous avons utilisé la théorie de "partionnement de graphe" afin de diviser le graphe en sous graphes. Cette approche est motivée par les observations de plusieurs études de l'état de l'art montrant que les tweets se rapportant à un même événement partagent des termes communs (McMinn et al., 2013). Dans les graphes, cette observation se traduit par des liens plus forts entre les nœuds extraits à partir de tweets qui discutent le même événement. A l'inverse, les nœuds extraits de tweets qui discutent d'événements différents sont connectés par des liens faibles ou la plupart du temps ne sont pas connectés. Nous avons appliqué la théorie de max-flow min-cut (Hoffman, 1974) pour partitionner le graphe d'événement en sous graphes.

La dernière phase de notre approche consiste à analyser les sous-graphes afin d'y extraire des événements. Nous avons utilisé un algorithme dérivé de PageRank (Brin & Page, 1998) afin de déterminer le poids des nœuds de chaque sous graphe résultant du partionnement. Afin de reduire l'impact des termes de tendance sur le poids des noeuds, nous avons modifié l'algorithme de PageRank pour prendre en compte le score tf-idf des termes. Nous avons analysé les nœuds du graphe du plus faible au moins faible afin d'extraire des événements. Pour chaque nœud considéré, l'algorithme extrait les nœuds le succédant ou le précédant et ayant les liens les plus forts, puis les supprime du graph. Si le graphe devient déconnecté, les nœuds isolés sont considérés comme appartenant au même événement. Finalement, nous avons caractérisé les événements en exploitant les types sémantiques des nœuds permettant de le définir.

Nous avons validé notre approche en comparant les résultats obtenus sur deux jeux de données de l'état de l'art par rapport à des méthodes existantes. Dans un premier scénario, nous avons considéré uniquement des tweets relatifs à des événements. Notre approche a obtenu de meilleurs performances par rapport à deux approches de l'état de l'art. Dans une scénario plus réaliste, nous avons évalué notre approche sur un jeu de données contenant à la fois des tweets relatifs à des événements et d'autres tweets ne traitant pas d'événements. Notre approche a obtenu de meilleurs scores que des méthodes de l'état de l'art évalué selon les mêmes critères. De manière générale, nos expériences ont montré que notre approche a obtenu des résultats compétitifs par rapport aux approches de l'état de l'art à la fois en terme de précision, rappel et f-mesure ainsi qu'en terme de pureté des événements identifiés.

Suivi d'événement sur Twitter

La troisième et dernière contribution de la thèse est une approche qui consiste à suivre l'évolution des événements sur Twitter. Cette approche est motivée par le fait que certains événements surtout sportifs sont largement discutés sur Twitter, et la capacité de traiter ces messages permettrait de mesurer l'impact d'un événement sur les utilisateurs. Plus particulièrement, dans le cadre des événements sportifs, l'analyse des tweets peut permettre d 'évaluer la réaction des fans par rapport au match. Ainsi, nous avons proposé une approche de suivi d'événement sportif sur Twitter, plus particulièrement le football.

Premièrement, nous avons décrit une méthode d'extraction d'information utilisant GATE enrichi d'informations extraites d'une base de connaissances de domaine (Football Data) et un vocabulaire contrôlé (SportsML) pour extraire les actions, les acteurs ainsi que les relations entre elles. En effet, afin d'obtenir les informations sur les joueurs et les équipes impliquées dans un match, nous avons utilisé Football Data, une base de connaissance qui permet d'obtenir des informations relatives à un match comme par exemple le nom des équipes ainsi que les joueurs d'une équipe.

D'une autre coté, afin de déterminer les actions autorisées dans un match, nous avons utilisé SportsML, qui définit un vocabulaire contrôlé permettant de décrire les sous-événements de la plupart des sports comme le football, le basket

ball ou le tennis. Aussi, dans le but de détecter les relations entre les actions et les participants, nous avons utilisé JAPE, un langage à base de règles permettant d'inférer des règles dans GATE pour la détection d'entité dynamiques.

Nous avons développé deux stratégies de mise à jour de GATE: une stratégie hors ligne permettant d'intégrer les actions relatives à un sport particulier. Cette stratégie se base sur les informations extraites à partir du vocabulaire SportML pour chaque type de sport supporté par notre approche. La seconde stratégie, dite en ligne, permet d'inclure les informations obtenues à partir de football Data comme ressources dans GATE afin de permettre l'identification des joueurs et des équipes.

Le deuxième composant de notre approche est dédié à la modélisation des relations entre les joueurs et les actions dans des graphes temporels. Ces graphes sont construits de telle sorte que les nœuds sont les actions ou participants du match et les liens sont définies soit en fonction des relations extraites à partir des règles JAPE ou par rapport à la co-occurence des actions ou les participants dans un même tweet. Le poids d'un lien représente le nombre de fois qu'une relation est observée dans les tweets.

Les approches existantes utilisent généralement un seuil fixe pour déterminer les actions à présenter à l'utilisateur. Cependant, nous avons observé qu'en fonction de la popularité d'une compétition ou des équipes impliquées, le nombre de tweets discutant un match peut varier énormément. En conséquence, une action de faible importance dans un match populaire peut générer plus de tweets qu'une action de grandes importance dans un match faiblement discuté sur Twitter. Nous avons adapté la formule de Kreyszig (Kreyszig, 2007) pour déterminer le seuil à partir duquel une action peut être considérée. Finalement, en utilisant la théorie de graphes et le score affecté à chaque nœud, nous avons développé un algorithme pour l'identification des nœuds représentant les actions ainsi que les joueurs ou les équipes afin d'identifier les actions du match.

Nous avons évalué cette approche dans le suivi d'événement dans les matchs joués durant la première phase de l'EURO 2016. Nos expériences sur différents matchs de l'EURO 2016 ont montré l'efficacité de notre approche. Aussi, nous avons développé un système permettant de visualiser en temps réel les actions du match.

Conclusion générale

Dans cette thèse, nous avons étudié des approches permettant d'identifier et de suivre des événements sur Twitter. Les approches proposées sont basées sur la définition d'événement dans le contexte des réseaux sociaux qui a montré une dépendance forte entre les événements et les entités nommées. D'un autre côté les bases de connaissances du Web contiennent des informations sur la sémantique des entités nommées, par exemple leurs types ou leurs relations avec d'autres entités. Nos différentes méthodes ont exploité les bases de connaissances dans l'objectif d'enrichir le contexte des entités nommées dans les tweets. Ainsi, cette approche d'enrichissement et de généralisation a été utilisé pour dans la classification de tweets relatifs à des événements, l'identification des événements ainsi que leurs caractéristiques et finalement le suivi de l'évolution d'événements sur Twitter.

Bibliography

Abel, F., Hauff, C., Houben, G.-J., Stronkman, R., & Tao, K. (2012). Twitcident: fighting fire with information from social web streams. In *Proceedings of the 21st international conference companion on world wide web* (pp. 305–308).

Alfonseca, E., & Manandhar, S. (n.d.). An unsupervised method for general named entity recognition and automated concept discovery..

Allan, J., Carbonell, J. G., Doddington, G., Yamron, J., & Yang, Y. (1998). Topic detection and tracking pilot study final report.

Alonso, O., & Shiells, K. (2013). Timelines as summaries of popular scheduled events. In *Proceedings of the 22nd international conference on world wide web* (pp. 1037–1044).

Anantharam, P., Barnaghi, P., Thirunarayan, K., & Sheth, A. (2014). Extracting city traffic events from social streams. *ACM Transactions on Intelligent Systems and Technology*, *9*(4).

Andersen, R., Chung, F., & Lang, K. (2006). Local graph partitioning using pagerank vectors. In *Foundations of computer science, 2006. focs'06. 47th annual ieee symposium on* (pp. 475–486).

Arias, M., Arratia, A., & Xuriguera, R. (2013). Forecasting with twitter data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *5*(1), 8.

Asahara, M., & Matsumoto, Y. (2003). Japanese named entity extraction with redundant morphological analysis. In *Proceedings of the 2003 conference of the north american chapter of the association for computational linguistics on human language technology-volume 1* (pp. 8–15).

Atefeh, F., & Khreich, W. (2015). A survey of techniques for event detection in twitter. *Computational Intelligence*, *31*(1), 132–164.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. *The semantic web*, 722–735.

Bansal, M., Gimpel, K., & Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. In *Acl (2)* (pp. 809–815).

Becker, H., Naaman, M., & Gravano, L. (2011). Beyond trending topics: Real-world event identification on twitter. *ICWSM*, *11*, 438–441.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, *3*(Feb), 1137–1155.

Berners-Lee, T. (2006). Linked data-design issues. *http://www. w3. org/DesignIssues/LinkedData. html*.

Bick, E. (2004). A named entity recognizer for danish. In *Lrec.*

Bikel, D. M., Miller, S., Schwartz, R., & Weischedel, R. (1997). Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on applied natural language processing* (pp. 194–201).

Bordes, A., Usunier, N., Chopra, S., & Weston, J. (2015). Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Borthwick, A., Sterling, J., Agichtein, E., & Grishman, R. (1998). Nyu: Description of the mene named entity system as used in muc-7. In *In proceedings of the seventh message understanding conference (muc-7.*

Boyd, D., Golder, S., & Lotan, G. (2010). Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *System sciences (hicss), 2010 43rd hawaii international conference on* (pp. 1–10).

Brickley, D., & Miller, L. (2007). *Foaf vocabulary specification 0.91.* Technical report, ILRT.

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, *30*(1), 107–117.

Bryl, V., Tonelli, S., Giuliano, C., & Serafini, L. (2012). A Novel Framenet-based Resource for the Semantic Web. In *Proceedings of the 27th annual acm symposium on applied computing* (pp. 360–365).

Brzozowski, M. J., & Romero, D. M. (2011). Who should i follow? recommending people in directed social networks. In *Icwsm.*

Budi, I., & Bressan, S. (2003). Association rules mining for name entity recognition. In *Web information systems engineering, 2003. wise 2003. proceedings of the fourth international conference on* (pp. 325–328).

Cano, A. E., Varga, A., Rowe, M., Ciravegna, F., & He, Y. (2013). Harness-

ing linked knowledge sources for topic classification in social media. In *Proceedings of the 24th acm conference on hypertext and social media* (pp. 41–50).

Chakrabarti, D., & Punera, K. (2011). Event summarization using tweets. *ICWSM*, *11*, 66–73.

Chang, A. X., & Manning, C. D. (2012). Sutime: A library for recognizing and normalizing time expressions. In *Lrec* (pp. 3735–3740).

Chiticariu, L., Krishnamurthy, R., Li, Y., Reiss, F., & Vaithyanathan, S. (2010). Domain adaptation of rule-based annotators for named-entity recognition tasks. In *Proceedings of the 2010 conference on empirical methods in natural language processing* (pp. 1002–1012).

Coates-Stephens, S. (1992). The analysis and acquisition of proper names for the understanding of free text. *Computers and the Humanities*, *26*(5), 441–456.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on machine learning* (pp. 160–167).

Council, I. P. T. (2017). *SportsML: A solution for sharing sports data.* `https://iptc.org/standards/sportsml-g2/`. ([Accessed 03-01-2017])

Cunningham, H., Maynard, D., Bontcheva, K., & Tablan, V. (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).*

Curran, J. R., & Clark, S. (2003). Language independent ner using a maximum entropy tagger. In *Proceedings of the seventh conference on natural language learning at hlt-naacl 2003-volume 4* (pp. 164–167).

Derczynski, L., Maynard, D., Rizzo, G., van Erp, M., Gorrell, G., Troncy, R., ... Bontcheva, K. (2015). Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, *51*(2), 32–49.

Derczynski, L., Ritter, A., Clark, S., & Bontcheva, K. (2013). Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *Proceedings of the international conference on recent advances in natural language processing.* Association for Computational Linguistics.

Dou, W., Wang, K., Ribarsky, W., & Zhou, M. (2012). Event detection in social media data. In *Ieee visweek workshop on interactive visual text analytics-*

task driven analytics of social media content (pp. 971–980).

Edouard, A. (2016). *Using Named Entities to Discover Heterogeneous Events on Twitter*. Retrieved from `https://hal.inria.fr/hal-01354088`

Edouard, A., Cabrio, E., Tonelli, S., & Nhan, L. T. (2017a). Building timelines of soccer matches from twitter. In *to appear in proceedings of ranlp 2017 - recent advances in natural language processing conference (demo paper).*

Edouard, A., Cabrio, E., Tonelli, S., & Nhan, L. T. (2017b). Graph-based event extraction from twitter. In *to appear in proceedings of ranlp 2017 - recent advances in natural language processing conference.*

Edouard, A., Cabrio, E., Tonelli, S., & Nhan, L. T. (2017c). Semantic linking for event-based classification of tweets. In *Proceedings of the 18th international conference on computational linguistics and intelligent text processing (cicling).*

Edouard, A., Cabrio, E., Tonelli, S., & Nhan, L. T. (2017d). You'll never tweet alone - building sports match timelines from microblog posts. In *to appear in proceedings of ranlp 2017 - recent advances in natural language processing conference.*

Even, S. (2011). *Graph algorithms*. Cambridge University Press.

Fellbaum, C. (1998). *WordNet. an electronic lexical database*. MIT Press.

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics* (pp. 363–370).

Forgues, G., Pineau, J., Larchevêque, J.-M., & Tremblay, R. (2014). Bootstrapping dialog systems with word embeddings. In *Nips, modern machine learning and natural language processing workshop.*

Forman, G. (2007). Feature selection for text classification. *Computational methods of feature selection*, *1944355797*.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). Springer series in statistics Springer, Berlin.

Gandon, F., Corby, O., & Faron-Zucker, C. (2012). *Le web sémantique: Comment lier les données et les schémas sur le web?* Dunod.

Genc, Y., Sakamoto, Y., & Nickerson, J. V. (2011). Discovering context: classifying tweets through a semantic transform based on wikipedia. In *Interna-*

tional conference on foundations of augmented cognition (pp. 484–492).

Gibbs, C., & Haynes, R. (2013). A phenomenological investigation into how twitter has changed the nature of sport media relations. *International Journal of Sport Communication*, *6*(4), 394–408.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., ... Smith, N. A. (2011). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies: short papers-volume 2* (pp. 42–47).

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, *5*(2), 199–220.

Hallili, A. (2014). Toward an ontology-based chatbot endowed with natural language processing and generation. In *26th european summer school in logic, language & information.*

Han, B., & Baldwin, T. (2011). Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1* (pp. 368–378).

Han, B., Yepes, A. J., MacKinlay, A., & Chen, Q. (2014). Identifying twitter location mentions. In *Australasian language technology association workshop 2014* (p. 157).

Hannon, J., McCarthy, K., Lynch, J., & Smyth, B. (2011). Personalized and automatic social summarization of events in video. In *Proceedings of the 16th international conference on intelligent user interfaces* (pp. 335–338).

Hermida, A. (2010). From tv to twitter: How ambient news became ambient journalism.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hoffman, A. J. (1974). A generalization of max flow—min cut. *Mathematical Programming*, *6*(1), 352–359. Retrieved from `http://dx.doi.org/10.1007/BF01580250` doi: 10.1007/BF01580250

Honey, C., & Herring, S. C. (2009). Beyond microblogging: Conversation and collaboration via twitter. In *System sciences, 2009. hicss'09. 42nd hawaii international conference on* (pp. 1–10).

Ilina, E., Hauff, C., Celik, I., Abel, F., & Houben, G.-J. (2012). Social event detection on twitter. In *Web engineering* (pp. 169–176). Springer.

Jai-Andaloussi, S., El Mourabit, I., Madrane, N., Chaouni, S. B., & Sekkaki, A. (2015). Soccer events summarization by using sentiment analysis. In *2015 international conference on computational science and computational intelligence (csci)* (pp. 398–403).

Java, A., Song, X., Finin, T., & Tseng, B. (2007). Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th webkdd and 1st sna-kdd 2007 workshop on web mining and social network analysis* (pp. 56–65).

Jin, P., Zhang, Y., Chen, X., & Xia, Y. (2016). Bag-of-embeddings for text classification. In *International joint conference on artificial intelligence* (pp. 2824–2830).

Katragadda, S., Benton, R., & Raghavan, V. (2017). Framework for real-time event detection using multiple social media sources. In *Proceedings of the 50th hawaii international conference on system sciences.*

Katragadda, S., Virani, S., Benton, R., & Raghavan, V. (2016). Detection of event onset using twitter. In *Neural networks (ijcnn), 2016 international joint conference on* (pp. 1539–1546).

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kinsella, S., Passant, A., & Breslin, J. G. (2010). Using hyperlinks to enrich message board content with linked data. In *Proceedings of the 6th international conference on semantic systems* (p. 1).

Kreyszig, E. (2007). *Advanced engineering mathematics*. John Wiley & Sons.

Kubo, M., Sasano, R., Takamura, H., & Okumura, M. (2013). Generating live sports updates from twitter by finding good reporters. In *Proceedings of the 2013 ieee/wic/acm international joint conferences on web intelligence (wi) and intelligent agent technologies (iat)-volume 01* (pp. 527–534).

Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on world wide web* (pp. 591–600).

Laublet, P., Reynaud, C., & Charlet, J. (2002). Sur quelques aspects du web sémantique. *Assises du GDR I*, *3*, 59–78.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P., ... Bizer, C. (2014). DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*.

Liu, H., & Motoda, H. (2007). *Computational methods of feature selection*. CRC Press.

Löchtefeld, M., Jäckel, C., & Krüger, A. (2015). Twitsoccer: knowledge-based crowd-sourcing of live soccer events. In *Proceedings of the 14th international conference on mobile and ubiquitous multimedia* (pp. 148–151).

Mahdisoltani, F., Biega, J., & Suchanek, F. (2014). Yago3: A knowledge base from multilingual wikipedias. In *7th conference on innovative data systems research.*

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval* (Vol. 1) (No. 1). Cambridge university press Cambridge.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *Acl (system demo)* (pp. 55–60).

Marcus, A., Bernstein, M. S., Badar, O., Karger, D. R., Madden, S., & Miller, R. C. (2011a). Twitinfo: Aggregating and visualizing microblogs for event exploration. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 227–236). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/1978942.1978975` doi: 10.1145/1978942.1978975

Marcus, A., Bernstein, M. S., Badar, O., Karger, D. R., Madden, S., & Miller, R. C. (2011b). Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 227–236).

Mathioudakis, M., & Koudas, N. (2010). Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 acm sigmod international conference on management of data* (pp. 1155–1158).

McCallum, A., & Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on natural language learning at hlt-naacl 2003-volume 4* (pp. 188–191).

McMinn, A. J., & Jose, J. M. (2015). Real-time entity-based event detection for

twitter. In *International conference of the cross-language evaluation forum for european languages* (pp. 65–77).

McMinn, A. J., Moshfeghi, Y., & Jose, J. M. (2013). Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22nd acm international conference on conference on information & knowledge management* (pp. 409–418).

Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems* (pp. 1–8).

Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into texts..

Mikheev, A. (1999). A knowledge-free method for capitalized word disambiguation. In *Proceedings of the 37th annual meeting of the association for computational linguistics on computational linguistics* (pp. 159–166).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Miller, S., Guinness, J., & Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *Hlt-naacl* (Vol. 4, pp. 337–342).

Nadeau, D. (2007). *Semi-supervised named entity recognition: learning to recognize 100 entity types with little supervision* (Unpublished doctoral dissertation). University of Ottawa.

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, *30*(1), 3–26.

Nagarajan, M., Purohit, H., & Sheth, A. P. (2010). A qualitative examination of topical tweet and retweet practices. *ICWSM*, *2*(010), 295–298.

Naveh, B., et al. (2008). Jgrapht. *Internet: http://jgrapht.sourceforge.net*.

Nichols, J., Mahmud, J., & Drews, C. (2012). Summarizing sporting events using twitter. In *Proceedings of the 2012 acm international conference on intelligent user interfaces* (pp. 189–198).

Nigel, D., & Rachel, C. (2017). Just the facts:winnowing microblogs for newsworthy statements using non-lexical features. In *Proceedings of the 18th international conference on computational linguistics and intelligent text processing.*

Norvig, P. (2009). *Natural language corpus data.* O'Reilly Media.

Nugroho, R., Zhao, W., Yang, J., Paris, C., Nepal, S., & Mei, Y. (2015). Time-sensitive topic derivation in twitter. In *International conference on web information systems engineering* (pp. 138–152).

Osborne, M., Petrovic, S., McCreadie, R., Macdonald, C., & Ounis, I. (2012). Bieber no more: First story detection using twitter and wikipedia. In *Sigir 2012 workshop on time-aware information access.*

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., & Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters..

Özsoy, S. (2011). Use of new media by turkish fans in sport communication: Facebook and twitter. *Journal of Human Kinetics*, *28*, 165–176.

Parikh, R., & Karlapalem, K. (2013). Et: events from tweets. In *Proceedings of the 22nd international conference on world wide web* (pp. 613–620).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Petrovic, S. (2013). Real-time event detection in massive streams.

Petrović, S., Osborne, M., & Lavrenko, V. (2010). Streaming first story detection with application to twitter. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 181–189).

Petrović, S., Osborne, M., & Lavrenko, V. (2012). Using paraphrases for improving first story detection in news and twitter. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 338–346).

Poibeau, T., & Kosseim, L. (2001). Proper name extraction from non-journalistic texts. *Language and computers*, *37*(1), 144–157.

Popescu, A.-M., Pennacchiotti, M., & Paranjpe, D. (2011). Extracting events and event descriptions from twitter. In *Proceedings of the 20th international conference companion on world wide web* (pp. 105–106).

Pustejovsky, J., Castano, J. M., Ingria, R., Sauri, R., Gaizauskas, R. J., Setzer, A., ... Radev, D. R. (2003). Timeml: Robust specification of event and temporal expressions in text. *New directions in question answering*, *3*,

28–34.

Pustejovsky, J., Knippen, R., Littman, J., & Saurí, R. (2005). Temporal and event information in natural language text. *Language resources and evaluation*, *39*(2), 123–164.

Ritter, A., Clark, S., Etzioni, O., et al. (2011). Named entity recognition in tweets: an experimental study. In *Proceedings of the conference on empirical methods in natural language processing* (pp. 1524–1534).

Ritter, A., Etzioni, O., Clark, S., et al. (2012). Open domain event extraction from twitter. In *Proceedings of the 18th acm sigkdd international conference on knowledge discovery and data mining* (pp. 1104–1112).

Ritterman, J., Osborne, M., & Klein, E. (2009). Using prediction markets and twitter to predict a swine flu pandemic. In *1st international workshop on mining social media* (Vol. 9, pp. 9–17).

Rizzo, G., & Troncy, R. (2011). Nerd: evaluating named entity recognition tools in the web of data.

Sakaki, T., Okazaki, M., & Matsuo, Y. (2010). Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on world wide web* (pp. 851–860).

Sampson, G. (1989). How fully does a machine-usable dictionary cover english text? *Literary and Linguistic Computing*, *4*(1), 29–35.

Schilder, F., Katz, G., & Pustejovsky, J. (2007). Annotating, extracting and reasoning about time and events. In *Annotating, extracting and reasoning about time and events* (pp. 1–6). Springer.

Sekine, S., et al. (1998). Nyu: Description of the japanese ne system used for met-2. In *Proc. message understanding conference.*

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications* (pp. 104–107).

Sharifi, B., Hutton, M.-A., & Kalita, J. (2010a). Summarizing microblogs automatically. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 685–688). Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved from `http://dl.acm.org/citation.cfm?id=1857999`

.1858099

Sharifi, B., Hutton, M.-A., & Kalita, J. (2010b). Summarizing microblogs automatically. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics* (pp. 685–688).

Sienčnik, S. K. (2015). Adapting word2vec to named entity recognition. In *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, 2015, vilnius, lithuania* (pp. 239–243).

Snoek, C. G., & Worring, M. (2005). Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, *7*(4), 638–647.

Snoek, C. G., Worring, M., et al. (2003). Time interval based modelling and classification of events in soccer video. In *Proceedings of the 9th annual conference of the advanced school for computing and imaging (asci), heijen.*

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of emnlp* (Vol. 1631, p. 1642).

Song, Y., Wang, H., Wang, Z., Li, H., & Chen, W. (2011). Short text conceptualization using a probabilistic knowledgebase. In *Proceedings of the twenty-second international joint conference on artificial intelligence - volume volume three* (pp. 2330–2336). AAAI Press. Retrieved from http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-388 doi: 10.5591/978-1-57735-516-8/IJCAI11-388

Sprugnoli, R., & Tonelli, S. (2017). One, no one and one hundred thousand events: Defining and processing events in an inter-disciplinary perspective. *Natural Language Engineering*, *23*(4), 485–506. Retrieved from https://doi.org/10.1017/S1351324916000292

Tang, B., Cao, H., Wu, Y., Jiang, M., & Xu, H. (2013). Recognizing clinical entities in hospital discharge summaries using structural support vector machines with word representation features. *BMC medical informatics and decision making*, *13*(1), S1.

Topic Detection and Tracking, TDT (2004): Annotation Manual. (n.d.). https://catalog.ldc.upenn.edu/docs/LDC2006T19/TDT2004V1.2.pdf. ([Accessed 03-03-2016])

Van Canneyt, S., Feys, M., Schockaert, S., Demeester, T., Develder, C., &

Dhoedt, B. (2014). Detecting newsworthy topics in twitter. In *Data challenge (snow 2014)* (pp. 1–8).

Van Erp, M., Rizzo, G., & Troncy, R. (2013). Learning with the web: Spotting named entities on the intersection of nerd and machine learning. In *# msm* (pp. 27–30).

Vaswani, A., Zhao, Y., Fossum, V., & Chiang, D. (2013). Decoding with large-scale neural language models improves translation. In *Emnlp* (pp. 1387–1392).

Verhagen, M., Gaizauskas, R., Schilder, F., Hepple, M., Katz, G., & Pustejovsky, J. (2007). Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th international workshop on semantic evaluations* (pp. 75–80).

Wang, X., Wei, F., Liu, X., Zhou, M., & Zhang, M. (2011). Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach. In *Proceedings of the 20th acm international conference on information and knowledge management* (pp. 1031–1040).

Wick, M. (2006). *Geonames*. GeoNames.

Xie, W., Zhu, F., Jiang, J., Lim, E.-P., & Wang, K. (2013). Topicsketch: Real-time bursty topic detection from twitter. In *Data mining (icdm), 2013 ieee 13th international conference on* (pp. 837–846).

Xu, W., Grishman, R., Meyers, A., & Ritter, A. (2013). A preliminary study of tweet summarization using information extraction. *NAACL 2013*, 20.

Yang, L., Sun, T., Zhang, M., & Mei, Q. (2012). We know what@ you# tag: does the dual role affect hashtag adoption? In *Proceedings of the 21st international conference on world wide web* (pp. 261–270).

Yang, X., Macdonald, C., & Ounis, I. (2016). Using word embeddings in twitter election classification. *arXiv preprint arXiv:1606.07006*.

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Icml* (Vol. 97, pp. 412–420).

Yang, Y., Pierce, T., & Carbonell, J. (1998). A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international acm sigir conference on research and development in information retrieval* (pp. 28–36).

Zhai, H., Lingren, T., Deleger, L., Li, Q., Kaiser, M., Stoutenborough, L., & Solti,

I. (2013). Web 2.0-based crowdsourcing for high-quality gold standard development in clinical natural language processing. *Journal of medical Internet research*, *15*(4), e73.

Zhou, D., Chen, L., & He, Y. (2011). A simple bayesian modelling approach to event extraction from twitter. *Atlantis*, 0.

Zhou, D., Zhang, X., & He, Y. (2017, April). Event extraction from Twitter using Non-Parametric Bayesian Mixture Model with Word Embeddings. In *Proceedings of the 15th conference of the european chapter of the association for computational linguistics* (pp. 808–817). Valencia, Spain.