Real-Time Detection of Traffic From Twitter Stream Analysis

Eleonora D'Andrea, Pietro Ducange, Beatrice Lazzerini, *Member, IEEE*, and Francesco Marcelloni, *Member, IEEE*

***Abstract*—Social networks have been recently employed as a source of information for event detection, with particular reference to road traffic congestion and car accidents. In this paper, we present a real-time monitoring system for *traffic* event detection from Twitter stream analysis. The system fetches tweets from Twitter according to several search criteria; processes tweets, by applying text mining techniques; and finally performs the classification of tweets. The aim is to assign the appropriate class label to each tweet, as related to a *traffic* event or not. The traffic detection system was employed for real-time monitoring of several areas of the Italian road network, allowing for detection of traffic events almost in real time, often before online traffic news web sites. We employed the support vector machine as a classification model, and we achieved an accuracy value of 95.75% by solving a binary classification problem (traffic versus nontraffic tweets). We were also able to discriminate if traffic is caused by an external event or not, by solving a multiclass classification problem and obtaining an accuracy value of 88.89%.**

***Index Terms*—Traffic event detection, tweet classification, text mining, social sensing.**

I. Introduction

SOCIAL network sites, also called micro-blogging services (e.g., Twitter, Facebook, Google+), have spread in recent years, becoming a new kind of real-time information channel. Their popularity stems from the characteristics of portability thanks to several social networks applications for smartphones and tablets, easiness of use, and real-time nature [1], [2]. People intensely use social networks to report (personal or public) real-life events happening around them or simply to express their opinion on a given topic, through a public message. Social networks allow people to create an identity and let them share it in order to build a community. The resulting *social network* is then a basis for maintaining social relationships, finding users with similar interests, and locating content and knowledge entered by other users [3].

The user message shared in social networks is called Status Update Message (SUM), and it may contain, apart from the text, meta-information such as timestamp, geographic coordinates (latitude and longitude), name of the user, links to other resources, hashtags, and mentions. Several SUMs referring to a certain topic or related to a limited geographic area may provide, if correctly analyzed, great deal of valuable information about an event or a topic. In fact, we may regard social network users as social sensors [4], [5], and SUMs as sensor information [6], as it happens with traditional sensors.

Recently, social networks and media platforms have been widely used as a source of information for the detection of events, such as traffic congestion, incidents, natural disasters (earthquakes, storms, fires, etc.), or other events. An *event* can be defined as a real-world occurrence that happens in a specific time and space [1], [7]. In particular, regarding traffic-related events, people often share by means of an SUM information about the current traffic situation around them while driving. For this reason, event detection from social networks is also often employed with Intelligent Transportation Systems (ITSs). An ITS is an infrastructure which, by integrating ICTs (Information and Communication Technologies) with transport networks, vehicles and users, allows improving safety and management of transport networks. ITSs provide, e.g., real-time information about weather, traffic congestion or regulation, or plan efficient (e.g., shortest, fast driving, least polluting) routes [4], [6], [8]–[14].

However, event detection from social networks analysis is a more challenging problem than event detection from traditional media like blogs, emails, etc., where texts are well-formatted [2]. In fact, SUMs are unstructured and irregular texts, they contain informal or abbreviated words, misspellings or grammatical errors [1]. Due to their nature, they are usually very brief, thus becoming an incomplete source of information [2]. Furthermore, SUMs contain a huge amount of not useful or meaningless information [15], which has to be filtered. According to Pear Analytics,[1] it has been estimated that over 40% of all Twitter[2] SUMs (i.e., *tweets*) is pointless with no useful information for the audience, as they refer to the personal sphere [16]. For all of these reasons, in order to analyze the information coming from social networks, we exploit text mining techniques [17], which employ methods from the fields of

Manuscript received July 2, 2014; revised October 7, 2014 and December 16, 2014; accepted February 10, 2015. Date of publication March 10, 2015; date of current version July 31, 2015. This work was carried out in the framework of and was supported by the SMARTY project, funded by "Programma Operativo Regionale (POR) 2007–2013"—objective "Competitività regionale e occupazione" of the Tuscany Region. The Associate Editor for this paper was Q. Zhang.

E. D'Andrea is with the Research Center "E. Piaggio," University of Pisa, 56122 Pisa, Italy (e-mail: eleonora.dandrea@for.unipi.it).

P. Ducange is with the Faculty of Engineering, eCampus University, 22060 Novedrate, Italy (e-mail: pietro.ducange@uniecampus.it).

B. Lazzerini and F. Marcelloni are with the Dipartimento di Ingegneria dell'Informazione, University of Pisa, 56122 Pisa, Italy (e-mail: b.lazzerini@iet.unipi.it; f.marcelloni@iet.unipi.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2015.2404431

[1] http://www.pearanalytics.com/, 2009.

[2] https://twitter.com.

data mining, machine learning, statistics, and Natural Language Processing (NLP) to extract meaningful information [18].

More in detail, text mining refers to the process of automatic extraction of meaningful information and knowledge from unstructured text. The main difficulty encountered in dealing with problems of text mining is caused by the vagueness of natural language. In fact, people, unlike computers, are perfectly able to understand idioms, grammatical variations, slang expressions, or to contextualize a given word. On the contrary, computers have the ability, lacking in humans, to quickly process large amounts of information [19], [20].

The text mining process is summarized in the following. First, the information content of the document is converted into a structured form (vector space representation). In fact, most of text mining techniques are based on the idea that a document can be faithfully represented by the set of words contained in it (*bag-of-words* representation [21]). According to this representation, each document j of a collection of documents is represented as an M-dimensional vector $V_j = \{w(t_{j1}), \ldots, w(t_{ji}), \ldots, w(t_{jM})\}$, where M is the number of words defined in the document collection, and $w(t_{ji})$ specifies the weight of the word t_i in document j. The simplest weighting method assigns a binary value to $w(t_{ji})$, thus indicating the absence or the presence of the word t_i, while other methods assign a real value to $w(t_{ji})$. During the text mining process, several operations can be performed [21], depending on the specific goal, such as: i) linguistic analysis through the application of NLP techniques, indexing and statistical techniques, ii) text filtering by means of specific keywords, iii) feature extraction, i.e., conversion of textual features (e.g., words) in numeric features (e.g., weights), that a machine learning algorithm is able to process, and iv) feature selection, i.e., reduction of the number of features in order to take into account only the most relevant ones. The feature selection is particularly important, since one of the main problems in text mining is the high dimensionality of the feature space $\Re^M$. Then, data mining and machine learning algorithms (i.e., support vector machines (SVMs), decision trees, neural networks, etc.) are applied to the documents in the vector space representation, to build classification, clustering or regression models. Finally, the results obtained by the model are interpreted by means of measures of effectiveness (e.g., statistical-based measures) to verify the accuracy achieved. Additionally, the obtained results may be improved, e.g., by modifying the values of the parameters used and repeating the whole process.

Among social networks platforms, we took into account Twitter, as the majority of works in the literature regarding event detection focus on it. Twitter is nowadays the most popular micro-blogging service; it counts more than 600 million active users,[3] sharing more than 400 million SUMs per day [1]. Regarding the aim of this paper, Twitter has several advantages over the similar micro-blogging services. First, tweets are up to 140 characters, enhancing the real-time and news-oriented nature of the platform. In fact, the life-time of tweets is usually very short, thus Twitter is the social network platform that is best suited to study SUMs related to real-time events [22]. Second, each tweet can be directly associated with meta-information that constitutes additional information. Third, Twitter messages are public, i.e., they are directly available with no privacy limitations. For all of these reasons, Twitter is a good source of information for real-time event detection and analysis.

In this paper, we propose an intelligent system, based on text mining and machine learning algorithms, for real-time detection of traffic events from Twitter stream analysis. The system, after a feasibility study, has been designed and developed from the ground as an event-driven infrastructure, built on a Service Oriented Architecture (SOA) [23]. The system exploits available technologies based on state-of-the-art techniques for text analysis and pattern classification. These technologies and techniques have been analyzed, tuned, adapted, and integrated in order to build the intelligent system. In particular, we present an experimental study, which has been performed for determining the most effective among different state-of-the-art approaches for text classification. The chosen approach was integrated into the final system and used for the on-the-field real-time detection of traffic events.

The paper has the following structure. Section II summarizes related work about event detection from social Twitter stream analysis. Section III outlines the architecture of the proposed system for traffic detection, by describing the methodology used to collect, elaborate, and classify SUMs, with particular reference to SUMs extracted from the Twitter stream. Section IV describes the setup of the system. Section V presents the results achieved with different classification models and provides a comparison with similar works in the literature. Section VI presents the real-world monitoring application for real-time detection of traffic events. Finally, Section VII provides concluding remarks.

II. Related Work

With reference to current approaches for using social media to extract useful information for event detection, we need to distinguish between *small-scale* events and *large-scale* events. Small-scale events (e.g., traffic, car crashes, fires, or local manifestations) usually have a small number of SUMs related to them, belong to a precise geographic location, and are concentrated in a small time interval. On the other hand, large-scale events (e.g., earthquakes, tornados, or the election of a president) are characterized by a huge number of SUMs, and by a wider temporal and geographic coverage [24]. Consequently, due to the smaller number of SUMs related to small-scale events, small-scale event detection is a non-trivial task. Several works in the literature deal with event detection from social networks. Many works deal with large-scale event detection [6], [25]–[28] and only a few works focus on small-scale events [9], [12], [24], [29]–[31].

Regarding large-scale event detection, Sakaki *et al.* [6] use Twitter streams to detect earthquakes and typhoons, by monitoring special trigger-keywords, and by applying an SVM as a binary classifier of positive events (earthquakes and typhoons) and negative events (non-events or other events). In [25], the authors present a method for detecting real-world events,

[3] http://www.statisticbrain.com/twitter-statistics

such as natural disasters, by analyzing Twitter streams and by employing both NLP and term-frequency-based techniques. Chew *et al.* [26] analyze the content of tweets shared during the H1N1 (i.e., swine flu) outbreak, containing keywords and hashtags related to the H1N1 event to determine the kind of information exchanged by social media users. De Longueville *et al.* [27] analyze geo-tagged tweets to detect forest fire events and outline the affected area.

Regarding small-scale event detection, Agarwal *et al.* [29] focus on the detection of fires in a factory from Twitter stream analysis, by using standard NLP techniques and a Naive Bayes (NB) classifier. In [30], information extracted from Twitter streams is merged with information from emergency networks to detect and analyze small-scale incidents, such as fires. Wanichayapong *et al.* [12] extract, using NLP techniques and syntactic analysis, traffic information from microblogs to detect and classify tweets containing place mentions and traffic information. Li *et al.* [31] propose a system, called TEDAS, to retrieve incident-related tweets. The system focuses on Crime and Disaster-related Events (CDE) such as shootings, thunderstorms, and car accidents, and aims to classify tweets as CDE events by exploiting a filtering based on keywords, spatial and temporal information, number of followers of the user, number of retweets, hashtags, links, and mentions. Sakaki *et al.* [9] extract, based on keywords, real-time driving information by analyzing Twitter's SUMs, and use an SVM classifier to filter "noisy" tweets not related to road traffic events. Schulz *et al.* [24] detect small-scale car incidents from Twitter stream analysis, by employing semantic web technologies, along with NLP and machine learning techniques. They perform the experiments using SVM, NB, and RIPPER classifiers.

In this paper, we focus on a particular small-scale event, i.e., road traffic, and we aim to detect and analyze traffic events by processing users' SUMs belonging to a certain area and written in the Italian language. To this aim, we propose a system able to fetch, elaborate, and classify SUMs as related to a road traffic event or not. To the best of our knowledge, few papers have been proposed for traffic detection using Twitter stream analysis. However, with respect to our work, all of them focus on languages different from Italian, employ different input features and/or feature selection algorithms, and consider only binary classifications. In addition, a few works employ machine learning algorithms [9], [24], while the others rely on NLP techniques only. The proposed system may approach both binary and multi-class classification problems. As regards binary classification, we consider traffic-related tweets, and tweets not related with traffic. As regards multi-class classification, we split the traffic-related class into two classes, namely traffic congestion or crash, and traffic due to *external* event. In this paper, with *external* event we refer to a scheduled event (e.g., a football match, a concert), or to an unexpected event (e.g., a flash-mob, a political demonstration, a fire). In this way we aim to support traffic and city administrations for managing scheduled or unexpected events in the city.

Moreover, the proposed system could work together with other traffic sensors (e.g., loop detectors, cameras, infrared cameras) and ITS monitoring systems for the detection of traffic difficulties, providing a low-cost wide coverage of the road network, especially in those areas (e.g., urban and suburban) where traditional traffic sensors are missing.

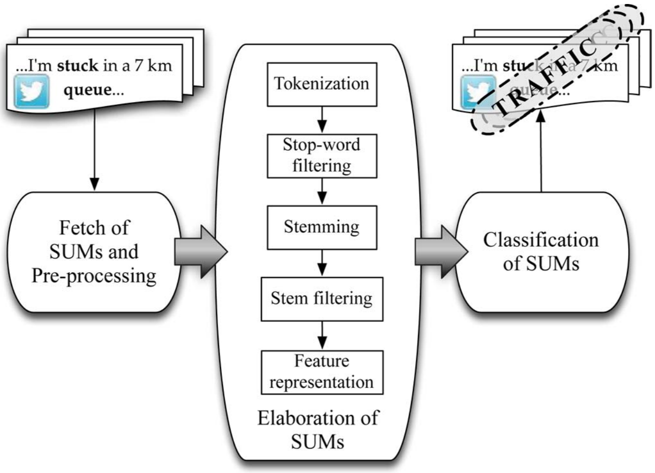


Fig. 1. System architecture for traffic detection from Twitter stream analysis.

Concluding, the proposed ITS is characterized by the following strengths with respect to the current research aimed at detecting traffic events from social networks: i) it performs a multi-class classification, which recognizes *non-traffic*, *traffic due to congestion or crash*, and *traffic due to external events*; ii) it detects the traffic events in real-time; and iii) it is developed as an event-driven infrastructure, built on an SOA architecture. As regards the first strength, the proposed ITS could be a valuable tool for traffic and city administrations to regulate traffic and vehicular mobility, and to improve the management of scheduled or unexpected events. For what concerns the second strength, the real-time detection capability allows obtaining reliable information about traffic events in a very short time, often before online news web sites and local newspapers. As far as the third strength is concerned, with the chosen architecture, we are able to directly notify the traffic event occurrence to the drivers registered to the system, without the need for them to access official news websites or radio traffic news channels, to get traffic information. In addition, the SOA architecture permits to exploit two important peculiarities, i.e., scalability of the service (e.g., by using a dedicated server for each geographic area), and easy integration with other services (e.g., other ITS services).

III. Architecture of the Traffic Detection System

In this section, our traffic detection system based on Twitter streams analysis is presented. The system architecture is service-oriented and event-driven, and is composed of three main modules, namely: i) "Fetch of SUMs and Pre-processing", ii) "Elaboration of SUMs", iii) "Classification of SUMs". The purpose of the proposed system is to fetch SUMs from Twitter, to process SUMs by applying a few text mining steps, and to assign the appropriate class label to each SUM. Finally, as shown in Fig. 1, by analyzing the classified SUMs, the system is able to notify the presence of a traffic event.

The main tools we have exploited for developing the system are: 1) Twitter's API,[4] which provides direct access to the

[4] http://dev.twitter.com

public stream of tweets; 2) Twitter4J,[5] a Java library that we used as a wrapper for Twitter's API; 3) the Java API provided by Weka (Waikato Environment for Knowledge Analysis) [32], which we mainly employed for data pre-processing and text mining elaboration.

We recall that both the "Elaboration of SUMs" and the "Classification of SUMs" modules require setting the optimal values of a few specific parameters, by means of a supervised learning stage. To this aim, we exploited a training set composed by a set of SUMs previously collected, elaborated, and manually labeled. Section IV describes in greater detail how the specific parameters of each module are set during the supervised learning stage.

In the following, we discuss in depth the elaboration made on the SUMs by each module of the traffic detection system.

A. Fetch of SUMs and Pre-Processing

The first module, "Fetch of SUMs and Pre-processing", extracts raw tweets from the Twitter stream, based on one or more search criteria (e.g., geographic coordinates, keywords appearing in the text of the tweet). Each fetched raw tweet contains: the user id, the timestamp, the geographic coordinates, a retweet flag, and the text of the tweet. The text may contain additional information, such as hashtags, links, mentions, and special characters. In this paper, we took only Italian language tweets into account. However, the system can be easily adapted to cope with different languages.

After the SUMs have been fetched according to the specific search criteria, SUMs are pre-processed. In order to extract only the text of each raw tweet and remove all meta-information associated with it, a Regular Expression filter [33] is applied. More in detail, the meta-information discarded are: user id, timestamp, geographic coordinates, hashtags, links, mentions, and special characters. Finally, a case-folding operation is applied to the texts, in order to convert all characters to lower case. At the end of this elaboration, each fetched SUM appears as a string, i.e., a sequence of characters. We denote the jth SUM pre-processed by the first module as SUM_j, with $j = 1, \ldots, N$, where N is the total number of fetched SUMs.

B. Elaboration of SUMs

The second processing module, "Elaboration of SUMs", is devoted to transforming the set of pre-processed SUMs, i.e., a set of strings, in a set of numeric vectors to be elaborated by the "Classification of SUMs" module. To this aim, some text mining techniques are applied in sequence to the pre-processed SUMs. In the following, the text mining steps performed in this module are described in detail:

a) *tokenization* is typically the first step of the text mining process, and consists in transforming a stream of characters into a stream of processing units called *tokens* (e.g., syllables, words, or phrases). During this step, other operations are usually performed, such as removal of punctuation and other non-text characters [18], and normalization of symbols (e.g., accents, apostrophes, hyphens, tabs and spaces). In the proposed system, the *tokenizer* removes all punctuation marks and splits each SUM into tokens corresponding to words (*bag-of-words* representation). At the end of this step, each SUM_j is represented as the sequence of words contained in it. We denote the jth tokenized SUM as $SUM_j^T = \{t_{j1}^T, \ldots, t_{jh}^T, \ldots, t_{jH_j}^T\}$, where t_{jh}^T is the hth token and H_j is the total number of tokens in SUM_j^T;

b) *stop-word filtering* consists in eliminating *stop-words*, i.e., words which provide little or no information to the text analysis. Common stop-words are articles, conjunctions, prepositions, pronouns, etc. Other stop-words are those having no statistical significance, that is, those that typically appear very often in sentences of the considered language (language-specific stop-words), or in the set of texts being analyzed (domain-specific stop-words), and can therefore be considered as noise [34]. The authors in [35] have shown that the 10 most frequent words in texts and documents of the English language are about the 20–30% of the tokens in a given document. In the proposed system, the stop-word list for the Italian language was freely downloaded from the Snowball Tartarus website[6] and extended with other *ad hoc* defined stop-words. At the end of this step, each SUM is thus reduced to a sequence of *relevant tokens*. We denote the jth stop-word filtered SUM as $SUM_j^{SW} = \{t_{j1}^{SW}, \ldots, t_{jk}^{SW}, \ldots, t_{jK_j}^{SW}\}$, where t_{jk}^{SW} is the kth relevant token and K_j, with $K_j \leq H_j$, is the total number of relevant tokens in SUM_j^{SW}. We recall that a *relevant token* is a token that does not belong to the set of stop-words;

c) *stemming* is the process of reducing each word (i.e., token) to its *stem* or root form, by removing its suffix. The purpose of this step is to group words with the same theme having closely related semantics. In the proposed system, the *stemmer* exploits the *Snowball Tartarus Stemmer*[7] for the Italian language, based on the Porter's algorithm [36]. Hence, at the end of this step each SUM is represented as a sequence of stems extracted from the tokens contained in it. We denote the jth stemmed SUM as $SUM_j^S = \{t_{j1}^S, \ldots, t_{jl}^S, \ldots, t_{jL_j}^S\}$, where t_{jl}^S is the lth stem and L_j, with $L_j \leq K_j$, is the total number of stems in SUM_j^S;

d) *stem filtering* consists in reducing the number of stems of each SUM. In particular, each SUM is filtered by removing from the set of stems the ones not belonging to the set of relevant stems. The set of F *relevant stems* $RS = \{\hat{s}_1, \ldots, \hat{s}_f, \ldots, \hat{s}_F\}$ is identified during the supervised learning stage that will be discussed in Section IV.

At the end of this step, each SUM is represented as a sequence of relevant stems. We denote the jth filtered SUM as $SUM_j^{SF} = \{t_{j1}^{SF}, \ldots, t_{jp}^{SF}, \ldots, t_{jP_j}^{SF}\}$, where

[5] http://twitter4j.org

[6] http://snowball.tartarus.org/algorithms/italian/stop.txt

[7] http://snowball.tartarus.org/algorithms/italian/stemmer.html

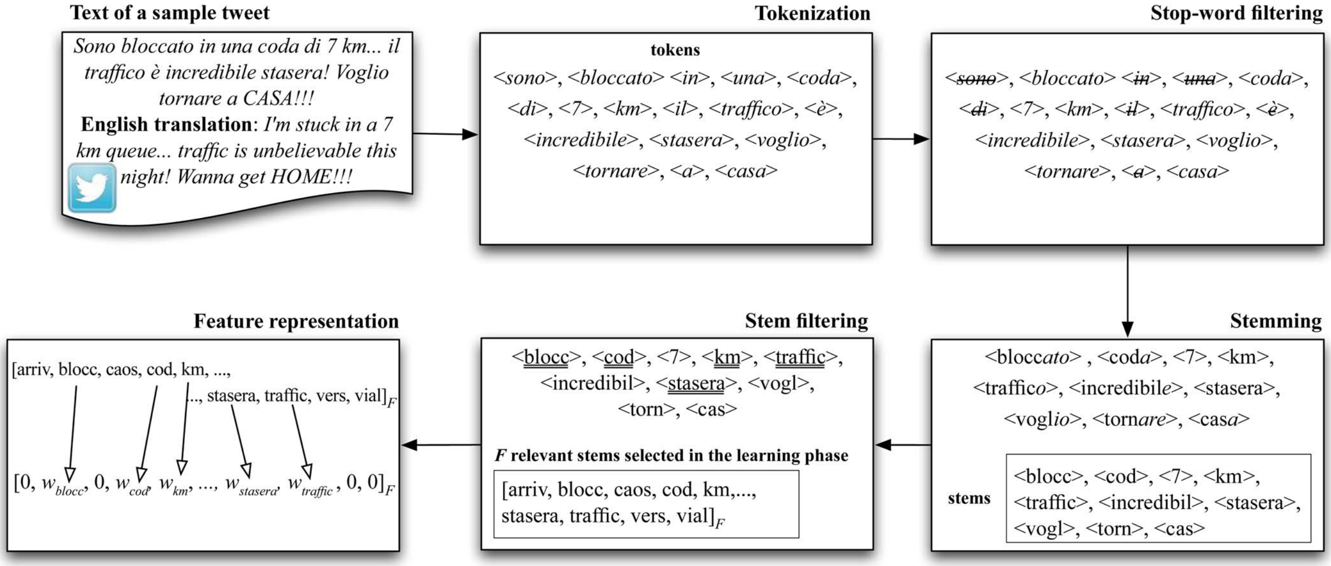

Text of a sample tweet
Sono bloccato in una coda di 7 km... il traffico è incredibile stasera! Voglio tornare a CASA!!!
English translation: I'm stuck in a 7 km queue... traffic is unbelievable this night! Wanna get HOME!!!
Tokenization
tokens
<sono>, <bloccato> <in>, <una>, <coda>, <di>, <7>, <km>, <il>, <traffico>, <è>, <incredibile>, <stasera>, <voglio>, <tornare>, <a>, <casa>
Stop-word filtering
<~~sono~~>, <bloccato> <~~in~~>, <~~una~~>, <coda>, <~~di~~>, <7>, <km>, <~~il~~>, <traffico>, <~~è~~>, <incredibile>, <stasera>, <voglio>, <tornare>, <~~a~~>, <casa>
Stemming
<bloccato>, <coda>, <7>, <km>, <traffico>, <incredibile>, <stasera>, <voglio>, <tornare>, <casa>
stems
<blocc>, <cod>, <7>, <km>, <traffic>, <incredibil>, <stasera>, <vogl>, <torn>, <cas>
Stem filtering
<blocc>, <cod>, <7>, <km>, <traffic>, <incredibil>, <stasera>, <vogl>, <torn>, <cas>
F relevant stems selected in the learning phase
[arriv, blocc, caos, cod, km,..., stasera, traffic, vers, vial]F
Feature representation
[arriv, blocc, caos, cod, km, ..., ..., stasera, traffic, vers, vial]F
[0, wblocc, 0, wcod, wkm, ..., wstasera, wtraffic, 0, 0]F

Fig. 2. Steps of the text mining elaboration applied to a sample tweet.

$t_{jp}^{SF} \in RS$ is the pth relevant stem and P_j, with $P_j \le L_j$ and $P_j \le F$, is the total number of relevant stems in SUM_j^{SF};

e) *feature representation* consists in building, for each SUM, the corresponding vector of numeric features. Indeed, in order to classify the SUMs, we have to represent them in the same feature space. In particular, we consider the F-dimensional set of features $X = \{X_1, \ldots, X_f, \ldots, X_F\}$ corresponding to the set of relevant stems. For each SUM_j^{SF} we define the vector $x_j = \{x_{j1}, \ldots, x_{jf}, \ldots, x_{jF}\}$ where each element is set according to the following formula:

$$x_{jf} = \begin{cases} w_f & \text{if stem } \hat{s}_f \in SUM_j^{SF} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

In (1), w_f is the numeric weight associated to the relevant stem $\hat{s}_f$: we will discuss how this weight is computed in Section IV.

In Fig. 2, we summarize all the steps applied to a sample tweet by the "Elaboration of SUMs" module.

C. Classification of SUMs

The third module, "Classification of SUMs", assigns to each elaborated SUM a class label related to traffic events. Thus, the output of this module is a collection of N labeled SUMs. To the aim of labeling each SUM, a classification model is employed. The parameters of the classification model have been identified during the supervised learning stage. Actually, as it will be discussed in Section V, different classification models have been considered and compared. The classifier that achieved the most accurate results was finally employed for the real-time monitoring with the proposed traffic detection system. The system continuously monitors a specific region and notifies the presence of a traffic event on the basis of a set of rules that can be defined by the system administrator. For example, when the first tweet is recognized as a traffic-related tweet, the system may send a warning signal. Then, the actual notification of the traffic event may be sent after the identification of a certain number of tweets with the same label.

IV. Setup of the System

As stated previously, a supervised learning stage is required to perform the setup of the system. In particular, we need to identify the set of *relevant stems*, the weights associated with each of them, and the parameters that describe the classification models. We employ a collection of N_{tr} labeled SUMs as training set. During the learning stage, each SUM is elaborated by applying the tokenization, stop-word filtering, and stemming steps. Then, the complete set of stems is built as follows:

$$CS = \left(\bigcup_{j=1}^{N_{tr}} SUM_j^S \right) = \{s_1, \ldots, s_q, \ldots, s_Q\}. \tag{2}$$

CS is the union of all the stems extracted from the N_{tr} SUMs of the training set. We recall that SUM_j^S is the set of stems that describes the jth SUM after the stemming step in the training set.

Then, we compute the weight of each stem in CS, which allows us to establish the importance of each stem s_q in the collection of SUMs of the training set, by using the Inverse Document Frequency (IDF) index as

$$w_q = IDF_q = \ln(N_{tr}/N_q), \tag{3}$$

where N_q is the number of SUMs of the training set in which the stem s_q occurs [37]. The IDF index is a simplified version of the TF-IDF (Term Frequency-IDF) index [38]–[40], where the TF part considers the frequency of a specific stem within each SUM. In fact, we heuristically found that the same stem seldom appears more than once in an SUM. On the other hand, we performed several experiments also with the TF-IDF index and we verified that the performance in terms of classification accuracy is similar to the one obtained by using only the IDF index. Thus, we decided to adopt the simpler IDF index as weight.

In order to select the set of relevant stems, a *feature selection* algorithm is applied. SUMs are described by a set $\{S_1, \ldots, S_q, \ldots, S_Q\}$ of Q features, where each feature S_q corresponds to the stem s_q. The possible values of feature S_q are w_q and 0.

Then, as suggested in [41], to evaluate the quality of each stem s_q, we employ a method based on the computation of the Information Gain (IG) value between feature S_q and output $C = \{c_1, \ldots, c_r, \ldots, c_R\}$, where c_r is one of the R possible class labels (two or three in our case). The IG value between S_q and C is calculated as $IG(C, S_q) = H(C) - H(C|S_q)$, where $H(C)$ represents the entropy of C, and $H(C|S_q)$ represents the entropy of C after the observation of feature S_q.

Finally, we identified the set of relevant stems RS by selecting all the stems which have a positive IG value. We recall that the stem selection process based on IG values is a standard and effective method widely used in the literature [40], [42].

The last part of the supervised learning stage regards the identification of the most suited classification models and the setting of their structural parameters. We took into account several classification algorithms widely used in the literature for text classification tasks [43], namely, i) SVM [44], ii) NB [45], iii) C4.5 decision tree [46], iv) k-nearest neighbor (kNN) [47], and v) PART [48]. The learning algorithms used to build the aforementioned classifiers will be briefly discussed in the following section.

V. Evaluation of the Traffic Detection System

In this section, we discuss the evaluation of the proposed system. We performed several experiments using two different datasets. For each dataset, we built and compared seven different classification models: SVM, NB, C4.5, kNN (with k equal to 1, 2, and 5), and PART. In the following, we describe how we generated the datasets to complete the setup of the system, and we recall the employed classification models. Then, we present the achieved results, and the statistical metrics used to evaluate the performance of the classifiers. Finally, we provide a comparison with some results extracted from other works in the literature.

A. Description of the Datasets

We built two different datasets, i.e., a 2-class dataset, and a 3-class dataset. For each dataset, tweets in the Italian language were collected using the "Fetch of SUMs and Pre-processing" module by setting some search criteria (e.g., presence of keywords, geographic coordinates, date and time of posting). Then, the SUMs were manually labeled, by assigning the correct class label.

1) 2-Class Dataset: The first dataset consists of tweets belonging to two possible classes, namely i) road traffic-related tweets (*traffic* class), and ii) tweets not related with road traffic (*non-traffic* class). The tweets were fetched in a time span of about four hours from the same geographic area. First, we fetched candidate tweets for *traffic* class by using the following search criteria:

— geographic area of origin of the tweet: Italy. We set the center of the area in Rome (latitude and longitude equal to 41° 53' 35" and 12° 28' 58", respectively) and we set a radius of about 600 km to cover approximately the whole country;

— time and date of posting: tweets belong to a time span of four evening hours of two weekend days of May 2013;

— keywords contained in the text of the tweet: we apply the **or**-operator on the set of keywords S_1, composed by the three most frequently used traffic-related keywords, $S_1 = \{$"traffico" (traffic), "coda" (queue), "incidente" (crash)$\}$, with the aim of selecting tweets containing at least one of the above keywords. The resulting condition can be expressed by:

$Cond_A$: "traffico" **or** "coda" **or** "incidente".

Then, we fetched the candidate tweets for *non-traffic* class using the same search criteria for geographic area, and time and date, but without setting any keyword. Obviously, this time, tweets containing traffic-related keywords from set S_1, already found in the previous fetch, were discarded.

Finally, the tweets were manually labeled with two possible class labels, i.e., as related to road traffic event (*traffic*), e.g., accidents, jams, queues, or not (*non-traffic*). More in detail, first we read, interpreted, and correctly assigned a *traffic* class label to each candidate *traffic* class tweet. Among all candidate *traffic* class tweets, we actually labeled 665 tweets with the *traffic* class. About 4% of candidate *traffic* class tweets were not labeled with the *traffic* class label. With the aim of correctly training the system, we added these tweets to the *non-traffic* class. Indeed, we collected also a number of tweets containing the traffic-related keywords defined in S_1, but actually not concerning road traffic events. Such tweets are related to, e.g., illegal drug trade, network traffic, or organ trafficking. It is worth noting that, as it happens in the English language, several words in the Italian language, e.g., "traffic" or "incident", are suitable in several contexts. So, for instance, the events "traffico di droga" (drug trade), "traffico di organi" (organ trafficking), "incidente diplomatico" (diplomatic scandal), "traffico dati" (network traffic) could be easily mistaken for road traffic-related events.

Then, in order to obtain a balanced dataset, we randomly selected tweets from the candidate tweets of *non-traffic* class until reaching 665 *non-traffic* class tweets, and we manually verified that the selected tweets did not belong to the *traffic* class. Thus, the final 2-class dataset consists of 1330 tweets and is balanced, i.e., it contains 665 tweets per class.

Table I shows the textual part of a selection of tweets fetched by the system with the corresponding, manually added, class label. In Table I, tweets #1, #2 and #3 are examples of *traffic* class tweets, tweet #4 is an example of a *non-traffic* class tweet, tweets #5 and #6 are examples of tweets containing traffic-related keywords, but belonging to the *non-traffic* class. In the table, for an easier understanding, the keywords appearing in

TABLE I
SOME EXAMPLES OF TWEETS AND CORRESPONDING CLASSES FOR THE 2-CLASS DATASET

#	Text of tweet	Class
1	Original tweet: *"A tutti gli amici catanesi incidente in tangenziale! Coda da San Gregorio a Misterbianco!"* English translation: *"To all the friends of Catania, crash on the bypass! Queue from San Gregorio to Misterbianco!"*	Traffic
2	Original tweet: *"Via Sperone 44 incidente stradale - traffico rallentato - procedere con prudenza"* English translation: *"Crash in Via Sperone 44 - traffic slowed - proceed with caution"*	Traffic
3	Original tweet: *"Trento, in zona sud, tamponamento convolti 7 autoveicoli, forti rallentamenti e lunghe code in direzione nord"* English translation: *"Trento, in the southern area, rear-ending with 7 cars involved, long delays and long queues northward"*	Traffic
4	Original tweet: *"Ogni tanto è bello anche un Sabato tranquillo tra pizza e risate, le migliori serate!"* English translation: "*Every now and then is nice even a peaceful Saturday of pizza and laughs, the best nights!*"	Non-traffic
5	Original tweet: *"Sono felice di vedere che questo grande scrittore stia bene, dopo l'incidente che ha avuto!"* English translation: *"I am happy to see that this great writer is well after the crash he had!"*	Non-traffic
6	Original tweet: *"Fermati 2 giovani per spaccio di droga e traffico d'armi"* English translation: *"Stopped 2 young men for drug dealing and arms trafficking"*	Non-traffic

TABLE II
SIGNIFICANT FEATURES RELATED TO THE *TRAFFIC* CLASS

Feature (stem)	Meaning	
	Source word	English translation
traffic	*traffico, trafficato*, etc.	*traffic, busy*
cod	*coda, code*	*queue, queues*
blocc	*blocco, bloccato*, etc.	*jam, stuck*
direzion	*direzione, direzioni*	*direction, directions*
km	*km*	*km*
incident	*incidente, incidenti*	*crash, crashes*
rallent	*rallentamento, rallentamenti*	*slowdown, slowdowns*
segnal	*segnalazione, segnale, segnalato*, etc.	*signal, sign, signaled*
strad	*strada, strade*	*road, roads*

the text of each tweet are underlined. Table II shows some of the most important textual features (i.e., stems) and their meaning, related to the *traffic* class tweets, identified by the system for this dataset.

2) 3-Class Dataset: The second dataset consists of tweets belonging to three possible classes. In this case we want to discriminate if traffic is caused by an *external* event (e.g., a football match, a concert, a flash-mob, a political demonstration, a fire) or not. Even though the current release of the system was not designed to identify the specific event, knowing that the traffic difficulty is caused by an external event could be useful to traffic and city administrations, for regulating traffic and vehicular mobility, or managing scheduled events in the city. More in detail, we took into account four possible *external events*, namely, i) matches, ii) processions, iii) music concerts, and iv) demonstrations. Thus, in this dataset the three possible classes are: i) *traffic due to external event*, ii) *traffic congestion or crash*, and iii) *non-traffic*. The tweets were fetched in a similar way as described before. More in detail, first, we fetched candidate road traffic-related tweets due to an external event (*traffic due to external event* class) according to the following search criteria:

— geographic area of origin of the tweet: Italy, parameters set as in the case of the 2-class dataset;

— time and date of posting: parameters set as in the case of the 2-class dataset, but different hours of the same weekend days are used;

— keywords contained in the text of the tweet: for each external event aforementioned, we took into account only one keyword, thus obtaining the set $S_2 = \{$"partita" (match), "processione" (procession), "concerto" (concert), "manifestazione" (demonstration)$\}$. Next we combined each keyword representing the external event with one of the traffic-related keywords from set $S_3 = \{$"traffico" (traffic), "coda" (queue)$\}$. Finally, we applied the **and**-operator between each keyword from set S_2 and the condition

$Cond_B$ expressed as:

$Cond_B$: "traffico" **or** "coda",

thus obtaining the following conditions:

$Cond_C$: $Cond_B$ **and** "partita",

$Cond_D$: $Cond_B$ **and** "processione"

$Cond_E$: $Cond_B$ **and** "concerto",

$Cond_F$: $Cond_B$ **and** "manifestazione".

Then, we fetched candidate tweets related to traffic congestion, crashes, and jams (*traffic congestion or crash* class) by using the following search criteria:

— geographic area of origin of the tweet: Italy, parameters set as as in the case of the 2-class dataset;

— time and date of posting: parameters set as in the case of the 2-class dataset, but different hours of the same weekend days are used;

— keywords contained in the text of the tweet: we combined the mentioned above keywords from set S_1 in three possible sets: $S_4 = \{$"traffico" (traffic), "incidente" (crash)$\}$, $S_5 = \{$"incidente" (crash), coda(queue)$\}$, and the already defined set S_3. Then we used the **and**-operator to define the exploited conditions as follows:

$Cond_G$: "traffico" **and** "incidente",

$Cond_H$: "traffico" **and** "coda",

$Cond_I$: "incidente" **and** "coda".

Obviously, as done before, tweets containing external event-related keywords, already found in the previous fetch, were discarded. Further, we fetched the candidate tweets of *non-traffic* class using the same search criteria for geographic area, and time and date, but without setting any keyword. Again, tweets already found in previous fetches were discarded.

Finally, the tweets were manually labeled with three possible class labels. We first labeled the candidate tweets of *traffic due to external event* class (this set of tweets was the smaller one), and we identified 333 tweets actually associated with this class. Then, we randomly selected 333 tweets for each of the

TABLE III
SOME EXAMPLES OF TWEETS AND CORRESPONDING CLASSES FOR THE 3-CLASS DATASET

#	Text of tweet	Class
1	Original tweet*: “Firenze, processione verso S. Maria Novella. traffico bloccato!”* English translation*: “Florence, procession towards St. Maria Novella. jammed traffic!”*	Traffic due to external event
2	Original tweet*: “Partita Milan - Lazio, allo stadio non si passa, un traffico atroce”* English translation*:* “*Milan - Lazio match, stucked near the stadium, a terrible traffic!”*	Traffic due to external event
3	Original tweet*: “Tutto esaurito e traffico enorme per il concerto dei Duran Duran”* English translation*: “Sold out and huge traffic for Duran Duran concert”*	Traffic due to external event
4	Original tweet*: “Manifestazione dei camionisti. Parcheggiano i camion bloccando la strada. Si è formata una coda chilometrica!”* English translation*: “Manifestation of truckers. They park the trucks blocking the road. A huge queue has formed!”*	Traffic due to external event
5	Original tweet: *“Incidente sulla FI-PI-LI tra Cascina e Navacchio direzione Pisa. Coda”* English translation: *“Crash on the FI-PI-LI highway between Cascina and Navacchio towards Pisa. Queue"*	Traffic congestion or crash
6	Original tweet: *“Milano piove di nuovo. Traffico inesistente ancora tutti in ferie?”* English translation*: “it’s raining again in Milan. Traffic’s absent still everyone on holiday?”*	Non-traffic
7	Original tweet*: “Ho appena scoperto che il bar sotto l'ufficio a breve avrà la crema al caffè....che bello!”* English translation*: “I just found out that the bar under the office soon will have the coffee cream… it’s great!”*	Non-traffic

two remaining classes. Also, in this case, we manually verified the correctness of the labels associated to the selected tweets. Finally, as done before, we added to the *non-traffic* class also tweets containing keywords related to traffic congestion and to external events but not concerning road-traffic events. The final 3-class dataset consists of 999 tweets and it is balanced, i.e., it has 333 tweets per class.

Table III shows a selection of tweets fetched by the system for the 3-class dataset, with the corresponding, manually added, class label. In Table III, tweets #1, #2, #3 and #4 are examples of tweets belonging to the class *traffic due to external event*: in more detail, #1 is related to a procession event, #2 is related to a match event, #3 is related to a concert event, and #4 is related to a demonstration event. Tweet #5 is an example of a tweet belonging to the class *traffic congestion* or *crash*, while tweets #6 and #7 are examples of *non-traffic* class tweets. Words underlined in the text of each tweet represent involved keywords.

B. Employed Classification Models

In the following we briefly describe the main properties of the employed and experimented classification models.

SVMs, introduced for the first time in [49], are discriminative classification algorithms based on a separating hyper-plane according to which new samples can be classified. The best hyper-plane is the one with the maximum margin, i.e., the largest minimum distance, from the training samples and is computed based on the support vectors (i.e., samples of the training set). The SVM classifier employed in this work is the implementation described in [44].

The NB classifier [45] is a probabilistic classification algorithm based on the application of the Bayes’s theorem, and is characterized by a probability model which assumes independence among the input features. In other words, the model assumes that the presence of a particular feature is unrelated to the presence of any other feature.

The C4.5 decision tree algorithm [46] generates a classification decision tree by recursively dividing up the training data according to the values of the features. Non-terminal nodes of the decision tree represent tests on one or more features, while terminal nodes represent the predicted output, namely the class. In the resulting decision tree each path (from the root to a leaf) identifies a combination of feature values associated with a particular classification. At each level of the tree, the algorithm chooses the feature that most effectively splits the data, according to the highest information gain.

The kNN algorithm [50] belongs to the family of “lazy” classification algorithms. The basic functioning principle is the following: each unseen sample is compared with a number of pre-classified training samples, and its similarity is evaluated according to a simple distance measure (e.g., we employed the normalized Euclidean distance), in order to find the associated output class. The parameter k allows specifying the number of neighbors, i.e., training samples to take into account for the classification. We focus on three kNN models with k equal to 1, 2, and 5. The kNN classifier employed in this work follows the implementation described in [47].

The PART algorithm [48] combines two rule generation methods, i.e., RIPPER [51] and C4.5 [46]. It infers classification rules by repeatedly building partial, i.e., incomplete, C4.5 decision trees and by using the *separate-and-conquer* rule learning technique [52].

C. Experimental Results

In this section, we present the classification results achieved by applying the classifiers mentioned in Section V-B to the two datasets described in Section V-A. For each classifier the experiments were performed using an n-fold stratified cross-validation methodology. In n-fold stratified cross-validation, the dataset is randomly partitioned into n folds and the classes in each fold are represented with the same proportion as in the original data. Then, the classification model is trained on $n-1$ folds, and the remaining fold is used for testing the model. The procedure is repeated n times, using as test data each of the n folds exactly once. The n test results are finally averaged to produce an overall estimation. We repeated an n-fold stratified cross-validation, with $n = 10$, for two times, using two different seed values to randomly partition the data into folds.

TABLE IV
STATISTICAL METRICS

Name	Equation
Accuracy	$Acc = \dfrac{TP+TN}{TP+FP+FN+TN}$
Precision	$Prec = \dfrac{TP}{TP+FP}$
Recall	$Rec = \dfrac{TP}{TP+FN}$
F-score	$F_\beta\text{-}score=\left(1+\beta^2\right)\cdot\dfrac{Prec\cdot Rec}{\left(\beta^2\cdot Prec\right)+Rec}$

We recall that, for each fold, we consider a specific training set which is used in the supervised learning stage to learn both the pre-processing (i.e., the set of *relevant stems* and their weights) and the classification model parameters.

To evaluate the achieved results, we employed the most frequently used statistical metrics, i.e., precision, accuracy, recall, and F-score. To explain the meaning of the metrics, we will refer, for the sake of simplicity, to the case of a binary classification, i.e., *positive* class versus *negative* class. In fact, in the case of a multi-class classification, the metrics are computed per class and the overall statistical measure is simply the average of the per-class measures. The correctness of a classification can be evaluated according to four values: i) *true positives* (TP): the number of real positive samples correctly classified as positive; ii) *true negatives (TN)*: the number of real negative samples correctly classified as negative; iii) *false positives (FP)*: the number of real negative samples incorrectly classified as positive; iv) *false negatives (FN)*: the number of real positive samples incorrectly classified as negative.

Based on the previous definitions, we can now formally define the employed statistical metrics and provide, in Table IV, the corresponding equations. Accuracy represents the overall effectiveness of the classifier and corresponds to the number of correctly classified samples over the total number of samples. Precision is the number of correctly classified samples of a class, i.e., positive class, over the number of samples classified as belonging to that class. Recall is the number of correctly classified samples of a class, i.e., positive class, over the number of samples of that class; it represents the effectiveness of the classifier to identify positive samples. The F-score (typically used with $\beta = 1$ for class-balanced datasets) is the weighted harmonic mean of precision and recall and it is used to compare different classifiers.

In the first experiment, we performed a classification of tweets using the 2-class dataset $(R = 2)$ consisting of 1330 tweets, described in Section V-A. The aim is to assign a class label (*traffic* or *non-traffic*) to each tweet.

Table V shows the average classification results obtained by the classifiers on the 2-class dataset. More in detail, the table shows for each classifier, the accuracy, and the per-class value of recall, precision, and F-score. All the values are averaged over the 20 values obtained by repeating two times the 10-fold cross validation. The best classifier resulted to be the SVM with an average accuracy of 95.75%.

As Table VI clearly shows, the results achieved by our SVM classifier appreciably outperform those obtained in similar works in the literature [9], [12], [24], [31] despite they refer to different datasets. More precisely, Wanichayapong *et al.* [12] obtained an accuracy of 91.75% by using an approach that considers the presence of place mentions and special keywords in the tweet. Li *et al.* [31] achieved an accuracy of 80% for detecting incident-related tweets using Twitter specific features, such as hashtags, mentions, URLs, and spatial and temporal information. Sakaki *et al.* [9] employed an SVM to identify heavy-traffic tweets and obtained an accuracy of 87%. Finally, Schulz *et al.* [24], by using SVM, RIPPER, and NB classifiers, obtained accuracies of 89.06%, 85.93%, and 86.25%, respectively. In the case of SVM, they used the following features: word n-grams, TF-IDF score, syntactic and semantic features. In the case of NB and RIPPER they employed the same set of features except semantic features.

In the second experiment, we performed a classification of tweets over three classes $(R = 3)$, namely, *traffic due to external event*, *traffic congestion or crash*, and *non-traffic*, with the aim of discriminating the cause of traffic. Thus, we employed the 3-class dataset consisting of 999 tweets, described in Section V-A. We employed again the classifiers previously introduced and the obtained results are shown in Table VII. The best classifier resulted to be again SVM with an average accuracy of 88.89%.

In order to verify if there exist statistical differences among the values of accuracy achieved by the seven classification models, we performed a statistical analysis of the results. We took into account the model which obtains the best average accuracy, i.e., the SVM model. As suggested in [53], we applied non-parametric statistical tests: for each classifier we generated a distribution consisting of the 20 values of the accuracies on the test set obtained by repeating two times the 10-fold cross validation. We statistically compared the results achieved by the SVM model with the ones achieved by the remaining models. We applied the Wilcoxon signed-rank test [54], which detects significant differences between two distributions. In all the tests, we used $\alpha = 0.05$ as level of significance. Tables VIII and IX show the results of the Wilcoxon signed-rank test, related to the 2-class and the 3-class problems, respectively. In the tables R^+ and R^- denote, respectively, the sum of ranks for the folds in which the first model outperformed the second, and the sum of ranks for the opposite condition. Since the p-values are always lower than the level of significance we can always reject the statistical hypothesis of equivalence. For this reason, we can state that the SVM model statistically outperforms all the other approaches on both the problems.

VI. REAL-TIME DETECTION OF TRAFFIC EVENTS

The developed system was installed and tested for the real-time monitoring of several areas of the Italian road network, by means of the analysis of the Twitter stream coming from those areas. The aim is to perform a continuous monitoring of frequently busy roads and highways in order to detect possible traffic events in real-time or even in advance with respect to the traditional news media [55], [56]. The system is implemented as

TABLE V
CLASSIFICATION RESULTS ON THE 2-CLASS DATASET (BEST VALUES IN BOLD)

Classifier	Accuracy (%)	Precision (%) by class		Recall (%) by class		F_1-score (%) by class	
		Traffic	*Non-traffic*	*Traffic*	*Non-traffic*	*Traffic*	*Non-traffic*
SVM	**95.75**	**95.3**	**96.3**	**96.5**	**95.0**	**95.8**	**95.7**
C4.5	95.15	94.4	96.1	96.1	94.2	95.2	95.1
1NN	91.87	93.2	91.2	90.9	93.3	92	92.2
3NN	91.69	93.3	90.3	89.9	93.5	91.5	91.8
5NN	91.61	93.7	89.9	89.3	93.9	91.4	91.8
NB	90.56	93.2	88.4	87.7	93.4	90.3	90.8
PART	94.66	94.1	95.4	95.3	94.0	94.7	94.6

TABLE VI
RESULTS OF THE CLASSIFICATION OF TWEETS IN OTHER WORKS IN THE LITERATURE

Classification algorithm	Considered tweets' classes	Classification results		Dataset	
		Measure	Value (%)	Size	Class balancing
NLP analysis [12]	*traffic-related* vs. *non-traffic-related*	Accuracy	91.75	1249 tweets	no
		Precision	91.39		
		Recall	87.53		
NLP analysis [31]	*CDE-related* vs. *non-CDE-related*	Accuracy	80.00	-	-
SVM [24]	*incident-related* vs. *non-incident-related*	Accuracy	89.06	640 tweets	yes
		Precision	89.10		
		Recall	89.10		
		F-score	89.10		
NB [24]	*incident-related* vs. *non-incident-related*	Accuracy	86.25	640 tweets	yes
		Precision	87.30		
		Recall	86.30		
		F-score	86.20		
RIPPER [24]	*incident-related* vs. *non-incident-related*	Accuracy	85.93	640 tweets	yes
		Precision	86.80		
		Recall	85.90		
		F-score	85.90		
SVM [9]	*heavy-traffic* vs. *non-heavy-traffic*	Precision	87.00	120 tweets	-
		Recall	67.00		
		F-score	75.00		

TABLE VII
CLASSIFICATION RESULTS ON THE 3-CLASS DATASET (BEST VALUES IN BOLD)

Classifier	Accuracy (%)	Precision (%) by class			Recall (%) by class			F_1-score (%) by class		
		Traffic cong. or crash	*Traffic due to ext. event*	*Non-traffic*	*Traffic cong. or crash*	*Traffic due to ext. event*	*Non-traffic*	*Traffic cong. or crash*	*Traffic due to ext. event*	*Non-traffic*
SVM	**88.89**	**81.4**	**93.5**	**93.9**	**92.8**	85.9	88.0	**86.6**	**89.5**	**90.8**
C4.5	86.03	77.6	90.4	92.5	89.0	80.3	**88.9**	82.8	84.9	90.5
1NN	80.53	73.3	84.6	86.0	79.3	77.9	84.5	76.2	80.7	85.0
3NN	81.13	70.3	93.8	85.5	85.7	71.6	86.2	77.2	80.7	85.6
5NN	80.28	68.2	94.6	87.5	90.1	65.0	85.7	77.5	76.8	86.4
NB	81.23	75.9	77.0	94.0	75.3	**86.1**	81.1	75.4	81.4	86.9
PART	85.04	77.1	89.5	91.5	87.8	79.9	87.4	81.8	84.1	89.4

TABLE VIII
RESULTS OF THE WILCOXON SIGNED-RANK TEST ON THE ACCURACIES OBTAINED ON THE TEST SET FOR THE 2-CLASS DATASET

Comparison	R^+	R^-	p-value	Hypothesis ($\alpha = 0.05$)
SVM vs. C4.5	170	40	$1.4 \cdot 10^{-2}$	Rejected
SVM vs. 1NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. 3NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. 5NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. NB	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. PART	167	23	$2 \cdot 10^{-3}$	Rejected

TABLE IX
RESULTS OF THE WILCOXON SIGNED-RANK TEST ON THE ACCURACIES OBTAINED ON THE TEST SET FOR THE 3-CLASS DATASET

Comparison	R^+	R^-	p-value	Hypothesis ($\alpha = 0.05$)
SVM vs. C4.5	197	13	$1.6784 \cdot 10^{-4}$	Rejected
SVM vs. 1NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. 3NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. 5NN	210	0	$1.9074 \cdot 10^{-6}$	Rejected
SVM vs. NB	208	2	$5.722 \cdot 10^{-6}$	Rejected
SVM vs. PART	202.5	7.5	$4.196 \cdot 10^{-5}$	Rejected

a service of a wider service-oriented platform to be developed in the context of the SMARTY project [23]. The service can be called by each user of the platform, who desires to know the traffic conditions in a certain area. In this section, we aim to show the effectiveness of our system in determining traffic events in short time. We just present some results for the

TABLE X
REAL-TIME DETECTION OF TRAFFIC EVENTS

#	Traffic event	Detection of the event		
		Our system	News channels	Time difference early (-)/late (+)
	5th September 2014			
1	Car crash and 3 km queue on A4 highway near Monza.	9:21 a.m.	10:27 a.m.	Early (- 66 min.)
	8th September 2014			
2	Car crash and 2 km queue on A1 highway between Calenzano and Roncobilaccio.	7:51 a.m.	7:35 a.m.	Late (+ 16 min.)
3	Traffic slowdown on A4 highway near Bresso.	8:43 a.m.	-	Early
	9th September 2014			
4	Two car crashes on A12 highway near Genoa.	8:24 a.m.	8:14 a.m.	Late (+ 10 min.)
5	Queue on A11 highway near Sesto Fiorentino.	9:12 a.m.	8:22 a.m.	Late (+ 50 min.)
	10th September 2014			
6	4 km queue on A1 highway between Sasso Marconi and Rioveggio.	8:04 a.m.	8:43 a.m.	Early (- 39 min.)
	12th September 2014			
7	Car on fire on A11 highway between Firenze and Prato.	3:08 p.m.	3:17 p.m.	Early (- 9 min.)
	15th September 2014			
8	Car crash and 2 km queue on FI-P-LI highway near Scandicci.	9:22 a.m.	9:00 a.m.	Late (+ 22 min.)
9	Traffic difficulties on A11 highway near Florence Airport.	9:08 a.m.	-	Early
10	Traffic difficulties on A1 highway near Altopascio.	10:11 a.m.	10:01 a.m.	Late (+ 10 min.)
11	Car crash and 2 km queue on FI-P-LI highway near Empoli.	5:08 p.m.	5:03 p.m.	Late (+ 5 min.)
12	Traffic slowdown on FI-P-LI highway near Scandicci.	3:57 p.m.	-	Early
13	Traffic on A11 highway near Prato.	9:19 a.m.	-	Early
	16th September 2014			
14	Car crash in Rome, Via Nomentana.	9:14 a.m.	9:17 a.m.	Early (- 3 min.)
15	Car crash and 2 km queue in Turin, Corso Dante.	9:15 a.m.	-	Early
	17th September 2014			
16	Car crash.in Milan, Viale Fulvio Testi.	10:12 a.m.	-	Early
17	Car crash and 1 km queue on Fi-PI-LI highway near Ginestra Fiorentina.	12:17 a.m.	12:09 a.m.	Late (+ 8 min.)
18	Car crash and 1 km queue in Rome, Via Casal del Marmo.	2:20 p.m.	2:06 p.m.	Late (+ 14 min.)
	22nd September 2014			
19	Car crash and queue in Rome, Piazza Cornelia.	9:48 a.m.	9:42 a.m.	Late (+ 6 min.)
20	Car crash in Rome, Via Flaminia.	8:59 a.m.	9:11 a.m.	Early (- 12 min.)
21	Car crash on A4 highway near Milano.	11:30 a.m.	11:24 a.m.	Late (+ 6 min.)
22	Traffic difficulties on SP166 near Domodossola.	11:15 a.m.	11:12 a.m.	Late (+ 3 min.)
	24th September 2014			
23	Traffic slowdown on SS148 in Rome, Via Pontina.	7:47 a.m.	7:50 a.m.	Early (- 3 min.)
24	Traffic difficulties on A11 highway near Florence Airport.	8:10 a.m.	9:03 a.m.	Early (- 53 min.)
25	Traffic slowdown in Rome, Via Appia.	7:56 a.m.	8:01 a.m.	Early (- 5 min.)
26	Traffic difficulties in Bologna near railway station.	8:25 a.m.	-	Early
27	Car crash on FI-PI-LI highway near Scandicci.	10:19 a.m.	9:59 a.m.	Late (+ 20 min.)
	25th September 2014			
28	Car crash in Rome, Via Battistini	8:24 a.m.	8:29 a.m.	Early (- 5 min.)
29	Car crash and 3 km queue on A11 highway between Pistoia and Prato.	9:31 a.m.	9:11 a.m.	Late (+ 20 min.)
30	Traffic slowdown for crash in Rome, Viale Damiano Chiesa.	9:59 a.m.	10:14 a.m.	Early (- 15 min.)
	26th September 2014			
31	Traffic slowdown on A1 highway between Barberino di Mugello and Roncobilaccio.	10:09 a.m.	9:51 a.m.	Late (+ 18 min.)
32	Traffic slowdown for crash in Rome, Viale America.	8:36 a.m.	8:32 a.m.	Late (+ 4 min.)
33	Traffic slowdown for crash in Rome, Via Flaminia.	8:29 a.m.	8:31 a.m.	Early (- 2 min.)
34	Traffic slowdown for crash in Rome, Via Righi.	5:03 p.m.	5:18 p.m.	Early (- 15 min.)
	29th September 2014			
35	Traffic slowdown for crash in Rome, Via Rossini.	9:09 a.m.	9:12 a.m.	Early (- 3 min.)
36	Traffic slowdown for crash in Rome, Via Appia Nuova.	7:42 a.m.	7:38 a.m.	Late (+ 4 min.)
37	Traffic difficulties on FI-PI-LI Highway between Lastra a Signa and Ginestra Fiorentina.	3:32 p.m.	2:44 p.m.	Late (+ 48 min.)
38	Long queue in Genoa, Lungomare Canepa.	5:12 p.m.	5:18 p.m.	Early (- 6 min.)
	30th September 2014			
39	Traffic difficulties and 5 km queue on FI-PI-LI Highway near Navacchio.	8:32 a.m.	-	Early
40	Traffic difficulties and 6 km queue on A1 Highway near Parma.	8:59 a.m.	8:10 a.m.	Late (+ 49 min.)
41	Car crash and queue on A12 Highway between Sestri Levante and Lavagna.	9:16 a.m.	9:09 a.m.	Late (+ 7 min.)
42	Car crash in Florence, via Filippo Strozzi.	9:15 a.m.	-	Early
43	Traffic slowdown on A11 Highway between Lucca and Capannori.	3:32 p.m.	3:28 p.m.	Late (+ 4 min.)
44	Traffic difficulties in Rome, Via di Boccea.	2:56 p.m.	2:57 p.m.	Early (-1 min.)

2-class problem. For the setup of the system, we have employed as training set the overall dataset described in Section V-A. We adopt only the best performing classifier, i.e., the SVM classifier. During the learning stage, we identified $Q = 3227$ features, which were reduced to $F = 582$ features after the feature selection step.

TABLE X
(*Continued.*) Real-Time Detection of Traffic Events

#	Traffic event	Detection of the event		
		Our system	News channels	Time difference early (-)/late (+)
45	Traffic difficulties for crash on Turin Ring Road, near Viale Regina Margherita.	1:59 p.m.	1:17 p.m.	Late (+ 42 min.)
	1st October 2014			
46	Traffic slowdown on A11 Highway near Florence.	7:52 a.m.	7:38 a.m.	Late (+ 14 min.)
47	Traffic slowdown on GRA Highway near Via Aurelia.	10:02 a.m.	9:53 a.m.	Late (+ 9 min.)
48	Traffic slowdown on Bologna Ring Road.	8:15 a.m.	-	Early
49	Car on fire and 6 km queue on A22 Highway near Verona.	11:00 a.m.	11:06 a.m.	Early (- 6 min.)
	2nd October 2014			
50	Truck crash and 5 km queue on A21 Highway between Torino and Brescia.	9:57 a.m.	9:19 a.m.	Late (+ 38 min.)
	3rd October 2014			
51	Truck crash in Rome, Via Casal del Marmo.	9:07 a.m.	9:19 a.m.	Early (- 12 min.)
52	Queue on GRA Highway between Via Appia and Via Aurelia.	8:36 a.m.	8:29 a.m.	Late (+ 7 min.)
53	Car crash and 6 km queue on A8 Highway between Milano and Varese.	9:02 a.m.	8:27 a.m.	Late (+ 35 min.)
54	Car crash and 5 km queue on A1 Highway near Piacenza.	11:12 a.m.	11:27 a.m.	Early (- 15 min.)
55	Car crash on A11 Highway near Prato.	12:44 a.m.	12:33 a.m.	Late (+ 11 min.)
56	Car crash on SS1 near Grosseto.	13:35 a.m.	13:15 a.m.	Late (+ 20 min.)
57	Car crash on A22 Highway between Reggiolo and Carpi.	13:25 a.m.	13:01 a.m.	Late (+ 24 min.)
58	Traffic difficulties in Rome, Viale Isacco Newton.	8:53 a.m.	-	Early
59	Car crash in Rome, Via Battistini.	8:17 a.m.	8:30 a.m.	Early (- 13 min.)
60	Traffic difficulties on A9 Highway near Como.	6:53 a.m.	6:38 a.m.	Late (+ 15 min.)
	6th October 2014			
61	Traffic difficulties on A1 Highway near Lodi.	9:07 a.m.	9:00 a.m.	Late (+ 7 min.)
62	Car crash and 2 km queue on A11 Highway, near Prato.	10:36 a.m.	10:31 a.m.	Late (+ 5 min.)
63	Car crash in Rome, Via Michelotti.	10:32 a.m.	10:21 a.m.	Late (+ 11 min.)
64	Car crash and 6 km queue on A1 Highway between Barberino di Mugello and Roncobilaccio.	1:29 p.m.	1:19 p.m.	Late (+ 10 min.)
65	Car crash in Rome, Via di Boccea.	12:47 a.m.	12:17 a.m.	Late (+ 30 min.)
66	Car crash in Rome, Via Appia Nuova.	2:54 p.m.	4:06 p.m.	Early (- 72 min.)
	7th October 2014			
67	Car crash and 2 km queue on A1 Highway near Guidonia.	8:09 a.m.	8:07 a.m.	Late (+ 2 min.)
68	Car crash in Rome, Corso Vittorio Emanuele II.	8:22 a.m.	8:18 a.m.	Late (+ 4 min.)
69	Car crash in Rome, Via della Pineta Sacchetti.	8:04 a.m.	8:02 a.m.	Late (+ 2 min.)
70	Car crash on A10 Highway near Genoa.	10:35 a.m.	10:28 a.m.	Late (+ 7 min.)

The system continuously performs the following operations: it i) fetches, with a time frequency of z minutes, tweets originated from a given area, containing the keywords resulting from $Cond_A$, ii) performs a real-time classification of the fetched tweets, iii) detects a possible traffic-related event, by analyzing the *traffic* class tweets from the considered area, and, if needed, sends one or more traffic warning signals with increasing intensity for that area. More in detail, a first low-intensity warning signal is sent when m *traffic* class tweets are found in the considered area in the same or in subsequent temporal windows. Then, as the number of *traffic* class tweets grows, the warning signal becomes more reliable, thus more intense. The value of m was set based on heuristic considerations, depending, e.g., on the traffic density of the monitored area. In the experiments we set $m = 1$. As regards the fetching frequency z, we heuristically found that $z = 10$ minutes represents a good compromise between fast event detection and system scalability. In fact, z should be set depending on the number of monitored areas and on the volume of tweets fetched.

With the aim of evaluating the effectiveness of our system, we need that each detected traffic-related event is appropriately validated. Validation can be performed in different ways which include: i) direct communication by a person, who was present at the moment of the event, ii) reports drawn up by the police and/or local administrations (available only in case of incidents), iii) radio traffic news; iv) official real-time traffic news web sites; v) local newspapers (often the day after the event and only when the event is very significant).

Direct communication is possible only if a person is present at the event and can communicate this event to us. Although we have tried to sensitize a number of users, we did not obtain an adequate feedback. Official reports are confidential: police and local administrations barely allow accessing to these reports, and, when this permission is granted, reports can be consulted only after several days. Radio traffic news are in general quite precise in communicating traffic-related events in real time. Unfortunately, to monitor and store the events, we should dedicate a person or adopt some tool for audio analysis. We realized however that the traffic-related events communicated on the radio are always mentioned also in the official real-time traffic news web sites. Actually, on the radio, the speaker typically reads the news reported on the web sites. Local newspapers focus on local traffic-related events and often provide events which are not published on official traffic news web sites. Concluding, official real-time traffic news web sites and local newspapers are the most reliable and effective sources of information for traffic-related events. Thus, we decided to analyze two of the most popular real-time traffic news web sites for the

Italian road network, namely "CCISS Viaggiare informati",[8] managed by the Italian government Ministry for infrastructures and transports, and "Autostrade per l'Italia",[9] the official web site of Italian highway road network. Further, we examined local newspapers published in the zones where our system was able to detect traffic-related events.

Actually, it was really difficult to find realistic data to test the proposed system, basically for two reasons: on the one hand, we have realized that real traffic events are not always notified in official news channels; on the other hand, situations of traffic slowdown may be detected by traditional traffic sensors but, at the same time, may not give rise to tweets. In particular, in relation to this latter reason, it is well known that drivers usually share a tweet about a traffic event only when the event is unexpected and really serious, i.e., it forces to stop the car. So, for instance, they do not share a tweet in case of road works, minor traffic difficulties, or usual traffic jams (same place and same time). In fact, in correspondence to minor traffic jams we rarely find tweets coming from the affected area.

We have tried to build a meaningful set of traffic events, related to some major Italian cities, of which we have found an official confirmation. The selected set includes events correctly identified by the proposed system and confirmed via official traffic news web sites or local newspapers. The set of traffic events, whose information is summarized in Table X, consists of 70 events detected by our system. The events are related both to highways and to urban roads, and were detected during September and early October 2014.

Table X shows the information about the event, the time of detection from Twitter's stream fetched by our system, the time of detection from official news websites or local newspapers, and the difference between these two times. In the table, positive differences indicate a late detection with respect to official news web sites, while negative differences indicate an early detection. The symbol "-" indicates that we found the official confirmation of the event by reading local newspapers several hours late. More precisely, the system detects in advance 20 events out of 59 confirmed by news web sites, and 11 events confirmed the day after by local newspapers. Regarding the 39 events not detected in advance we can observe that 25 of such events are detected within 15 minutes from their official notification, while the detection of the remaining 14 events occurs beyond 15 minutes but within 50 minutes. We wish to point out, however, that, even in the cases of late detection, our system directly and explicitly notifies the event occurrence to the drivers or passengers registered to the SMARTY platform, on which our system runs. On the contrary, in order to get traffic information, the drivers or passengers usually need to search and access the official news websites, which may take some time and effort, or to wait for getting the information from the radio traffic news.

As future work, we are planning to integrate our system with an application for analyzing the official traffic news web sites, so as to capture traffic condition notifications in real-time. Thus, our system will be able to signal traffic-related events in the worst case at the same time of the notifications on the web sites. Further, we are investigating the integration of our system into a more complex traffic detection infrastructure. This infrastructure may include both advanced physical sensors and social sensors such as streams of tweets. In particular, social sensors may provide a low-cost wide coverage of the road network, especially in those areas (e.g., urban and suburban) where traditional traffic sensors are missing.

VII. Conclusion

In this paper, we have proposed a system for real-time detection of traffic-related events from Twitter stream analysis. The system, built on a SOA, is able to fetch and classify streams of tweets and to notify the users of the presence of traffic events. Furthermore, the system is also able to discriminate if a traffic event is due to an external cause, such as football match, procession and manifestation, or not.

We have exploited available software packages and state-of-the-art techniques for text analysis and pattern classification. These technologies and techniques have been analyzed, tuned, adapted and integrated in order to build the overall system for traffic event detection. Among the analyzed classifiers, we have shown the superiority of the SVMs, which have achieved accuracy of 95.75%, for the 2-class problem, and of 88.89% for the 3-class problem, in which we have also considered the *traffic due to external event* class.

The best classification model has been employed for real-time monitoring of several areas of the Italian road network. We have shown the results of a monitoring campaign, performed in September and early October 2014. We have discussed the capability of the system of detecting traffic events almost in real-time, often before online news web sites and local newspapers.

Acknowledgment

We would like to thank Fabio Cempini for the implementation of some parts of the system presented in this paper.

References

[1] F. Atefeh and W. Khreich, "A survey of techniques for event detection in Twitter," *Comput. Intell.*, vol. 31, no. 1, pp. 132–164, 2015.

[2] P. Ruchi and K. Kamalakar, "ET: Events from tweets," in *Proc. 22nd Int. Conf. World Wide Web Comput.*, Rio de Janeiro, Brazil, 2013, pp. 613–620.

[3] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proc. 7th ACM SIGCOMM Conf. Internet Meas.*, San Diego, CA, USA, 2007, pp. 29–42.

[4] G. Anastasi *et al.*, "Urban and social sensing for sustainable mobility in smart cities," in *Proc. IFIP/IEEE Int. Conf. Sustainable Internet ICT Sustainability*, Palermo, Italy, 2013, pp. 1–4.

[5] A. Rosi *et al.*, "Social sensors and pervasive services: Approaches and perspectives," in *Proc. IEEE Int. Conf. PERCOM Workshops*, Seattle, WA, USA, 2011, pp. 525–530.

[6] T. Sakaki, M. Okazaki, and Y. Matsuo, "Tweet analysis for real-time event detection and earthquake reporting system development," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 919–931, Apr. 2013.

[7] J. Allan, *Topic Detection and Tracking: Event-Based Information Organization*. Norwell, MA, USA: Kluwer, 2002.

[8] K. Perera and D. Dias, "An intelligent driver guidance tool using location based services," in *Proc. IEEE ICSDM*, Fuzhou, China, 2011, pp. 246–251.

[8] http://www.cciss.it/

[9] http://www.autostrade.it/autostrade-gis/gis.do

[9] T. Sakaki, Y. Matsuo, T. Yanagihara, N. P. Chandrasiri, and K. Nawa, "Real-time event extraction for driving information from social sensors," in *Proc. IEEE Int. Conf. CYBER*, Bangkok, Thailand, 2012, pp. 221–226.

[10] B. Chen and H. H. Cheng, "A review of the applications of agent technology in traffic and transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 485–497, Jun. 2010.

[11] A. Gonzalez, L. M. Bergasa, and J. J. Yebes, "Text detection and recognition on traffic panels from street-level imagery using visual appearance," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 228–238, Feb. 2014.

[12] N. Wanichayapong, W. Pruthipunyaskul, W. Pattara-Atikom, and P. Chaovalit, "Social-based traffic information extraction and classification," in *Proc. 11th Int. Conf. ITST*, St. Petersburg, Russia, 2011, pp. 107–112.

[13] P. M. d'Orey and M. Ferreira, "ITS for sustainable mobility: A survey on applications and impact assessment tools," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 477–493, Apr. 2014.

[14] K. Boriboonsomsin, M. Barth, W. Zhu, and A. Vu, "Eco-routing navigation system based on multisource historical and real-time traffic information," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1694–1704, Dec. 2012.

[15] J. Hurlock and M. L. Wilson, "Searching twitter: Separating the tweet from the chaff," in *Proc. 5th AAAI ICWSM*, Barcelona, Spain, 2011, pp. 161–168.

[16] J. Weng and B.-S. Lee, "Event detection in Twitter," in *Proc. 5th AAAI ICWSM*, Barcelona, Spain, 2011, pp. 401–408.

[17] S. Weiss, N. Indurkhya, T. Zhang, and F. Damerau, *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Berlin, Germany: Springer-Verlag, 2004.

[18] A. Hotho, A. Nürnberger, and G. Paaß, "A brief survey of text mining," *LDV Forum-GLDV J. Comput. Linguistics Lang. Technol.*, vol. 20, no. 1, pp. 19–62, May 2005.

[19] V. Gupta, S. Gurpreet, and S. Lehal, "A survey of text mining techniques and applications," *J. Emerging Technol. Web Intell.*, vol. 1, no. 1, pp. 60–76, Aug. 2009.

[20] V. Ramanathan and T. Meyyappan, "Survey of text mining," in *Proc. Int. Conf. Technol. Bus. Manage.*, Dubai, UAE, 2013, pp. 508–514.

[21] M. W. Berry and M. Castellanos, *Survey of Text Mining*. New York, NY, USA: Springer-Verlag, 2004.

[22] H. Takemura and K. Tajima, "Tweet classification based on their lifetime duration," in *Proc. 21st ACM Int. CIKM*, Shanghai, China, 2012, pp. 2367–2370.

[23] The Smarty project. [Online]. Available: http://www.smarty.toscana.it/

[24] A. Schulz, P. Ristoski, and H. Paulheim, "I see a car crash: Real-time detection of small scale incidents in microblogs," in *The Semantic Web: ESWC 2013 Satellite Events*, vol. 7955. Berlin, Germany: Springer-Verlag, 2013, pp. 22–33.

[25] M. Krstajic, C. Rohrdantz, M. Hund, and A. Weiler, "Getting there first: Real-time detection of real-world incidents on Twitter" in *Proc. 2nd IEEE Work Interactive Vis. Text Anal.—Task-Driven Anal. Soc. Media IEEE VisWeek*," Seattle, WA, USA, 2012.

[26] C. Chew and G. Eysenbach, "Pandemics in the age of Twitter: Content analysis of tweets during the 2009 H1N1 outbreak," *PLoS ONE*, vol. 5, no. 11, pp. 1–13, Nov. 2010.

[27] B. De Longueville, R. S. Smith, and G. Luraschi, "OMG, from here, I can see the flames!: A use case of mining location based social networks to acquire spatio-temporal data on forest fires," in *Proc. Int. Work. LBSN*, 2009 Seattle, WA, USA, pp. 73–80.

[28] J. Yin, A. Lampert, M. Cameron, B. Robinson, and R. Power, "Using social media to enhance emergency situation awareness," *IEEE Intell. Syst.*, vol. 27, no. 6, pp. 52–59, Nov./Dec. 2012.

[29] P. Agarwal, R. Vaithiyanathan, S. Sharma, and G. Shro, "Catching the long-tail: Extracting local news events from Twitter," in *Proc. 6th AAAI ICWSM*, Dublin, Ireland, Jun. 2012, pp. 379–382.

[30] F. Abel, C. Hauff, G.-J. Houben, R. Stronkman, and K. Tao, "Twitcident: fighting fire with information from social web streams," in *Proc. ACM 21st Int. Conf. Comp. WWW*, Lyon, France, 2012, pp. 305–308.

[31] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang, "TEDAS: A Twitter-based event detection and analysis system," in *Proc. 28th IEEE ICDE*, Washington, DC, USA, 2012, pp. 1273–1276.

[32] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Jun. 2009.

[33] M. Habibi, *Real World Regular Expressions with Java 1.4*. Berlin, Germany: Springer-Verlag, 2004.

[34] Y. Zhou and Z.-W. Cao, "Research on the construction and filter method of stop-word list in text preprocessing," in *Proc. 4th ICICTA*, Shenzhen, China, 2011, vol. 1, pp. 217–221.

[35] W. Francis and H. Kucera, "Frequency analysis of English usage: Lexicon and grammar," *J. English Linguistics*, vol. 18, no. 1, pp. 64–70, Apr. 1982.

[36] M. F. Porter, "An algorithm for suffix stripping," *Program: Electron. Library Inf. Syst.*, vol. 14, no. 3, pp 130–137, 1980.

[37] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manage.*, vol. 24, no. 5, pp. 513–523, Aug. 1988.

[38] L. M. Aiello *et al.*, "Sensing trending topics in Twitter," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1268–1282, Oct. 2013.

[39] C. Shang, M. Li, S. Feng, Q. Jiang, and J. Fan, "Feature selection via maximizing global information gain for text classification," *Knowl.-Based Syst.*, vol. 54, pp. 298–309, Dec. 2013.

[40] L. H. Patil and M. Atique, "A novel feature selection based on information gain using WordNet," in *Proc. SAI Conf.*, London, U.K., 2013, pp. 625–629.

[41] M. A. Hall and G. Holmes. "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 6, pp. 1437–1447, Nov./Dec. 2003.

[42] H. Uğuz, "A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm," *Knowl.-Based Syst.*, vol. 24, no. 7, pp. 1024–1032, Oct. 2011.

[43] Y. Aphinyanaphongs *et al.*, "A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization," *J. Assoc. Inf. Sci. Technol.*, vol. 65, no. 10, pp. 1964–1987, Oct. 2014.

[44] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Learning*, B. Schoelkopf, C. J. C. Burges and A. J. Smola, Eds. Cambridge, MA, USA, MIT Press, 1999, pp. 185–208.

[45] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell.*, San Mateo, CA, 1995, pp. 338–345.

[46] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[47] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Mach. Learn.*, vol. 6, no. 1, pp. 37–66, Jan. 1991.

[48] E. Frank and I. H. Witten, "Generating accurate rule sets without global optimization," in *Proc. 15th ICML*, Madison, WI, USA, 1998, pp. 144–151.

[49] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[50] T. T. Cover and P. E. Hart, "Nearest neighbour pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.

[51] W. W. Cohen, "Fast effective rule induction," in *Proc. 12th ICML*, Tahoe City, CA, USA, 1995, pp. 115–123.

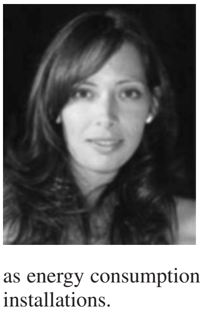
[52] G. Pagallo and D. Haussler, "Boolean feature discovery in empirical learning," *Mach. Learn.*, vol. 5, no. 1, pp. 71–99, Mar. 1990.

[53] J. Derrac, S. Garcia, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[54] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.* , vol. 1, no. 6, pp. 80–83, Dec. 1945.

[55] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on Twitter," in *Proc. 5th AAAI ICWSM*, Barcelona, Spain, 2011, pp. 438–441.

[56] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter, a social network or a news media?" in *Proc. ACM 19th Int. Conf. WWW*, Raleigh, NY, USA, 2010, pp. 591–600.



Eleonora D'Andrea received the M.S. degree in computer engineering for enterprise management and the Ph.D. degree in information engineering from University of Pisa, Pisa, Italy, in 2009 and 2013, respectively.

She is a Research Fellow with the Research Center "E. Piaggio," University of Pisa. She has coauthored several papers in international journals and conference proceedings. Her main research interests include computational intelligence techniques for simulation and prediction, applied to various fields, such as energy consumption in buildings or energy production in solar photovoltaic installations.

Pietro Ducange received the M.Sc. degree in computer engineering and the Ph.D. degree in information engineering from University of Pisa, Pisa, Italy, in 2005 and 2009, respectively.

He is an Associate Professor with the Faculty of Engineering, eCampus University, Novedrate, Italy. He has coauthored more than 30 papers in international journals and conference proceedings. His main research interests focus on designing fuzzy-rule-based systems with different tradeoffs between accuracy and interpretability by using multiobjective evolutionary algorithms. He currently serves the following international journals as a member of the Editorial Board: *Soft Computing* and *International Journal of Swarm Intelligence and Evolutionary Computation*.

Beatrice Lazzerini (M'98) is a Full Professor with the Department of Information Engineering, University of Pisa, Pisa, Italy. She has cofounded the Computational Intelligence Group in the Department of Information Engineering, University of Pisa. She has coauthored seven books and has published over 200 papers in international journals and conferences. She is a coeditor of two books. Her research interests are in the field of computational intelligence and its applications to pattern classification, pattern recognition, risk analysis, risk management, diagnosis, forecasting, and multicriteria decision making. She was involved and had roles of responsibility in several national and international research projects, conferences, and scientific events.

Francesco Marcelloni (M'06) received the Laurea degree in electronics engineering and the Ph.D. degree in computer engineering from University of Pisa, Pisa, Italy, in 1991 and 1996, respectively.

He is an Associate Professor with University of Pisa. He has cofounded the Computational Intelligence Group in the Department of Information Engineering, University of Pisa, in 2002. He is also the Founder and Head of the Competence Centre on MObile Value Added Services (MOVAS). He has coedited three volumes and four journal Special Issues and is the (co)author of a book and of more than 190 papers in international journals, books, and conference proceedings. His main research interests include multiobjective evolutionary fuzzy systems, situation-aware service recommenders, energy-efficient data compression and aggregation in wireless sensor nodes, and monitoring systems for energy efficiency in buildings. Currently, he is an Associate Editor for *Information Sciences* (Elsevier) and *Soft Computing* (Springer) and is on the Editorial Board of four other international journals.