Future Generation Computer Systems 95 (2019) 816–828

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Social information discovery enhanced by sentiment analysis techniques

Claudia Diamantini, Alex Mircoli [*], Domenico Potena, Emanuele Storti

Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, via Brecce Bianche, 60131 Ancona, Italy

HIGHLIGHTS

- Social networks provide a large amount of analyzable user-generated content (USG).
- Social data are mainly textual and their analysis is complex.
- Approach: a social information discovery system enhanced by sentiment analysis.
- Dedicated algorithms for word sense disambiguation and negation handling improve the accuracy of sentiment analysis.
- Experimental evidence of efficiency and effectiveness on synthetic and real data.

ARTICLE INFO

Article history:
Received 1 May 2017
Received in revised form 5 January 2018
Accepted 25 January 2018
Available online 6 February 2018

ABSTRACT

In recent years, the massive diffusion of social networks has made available a large amount of user-generated content, for the most part in the form of textual data that contain people's thoughts and emotions about a great variety of topics. In order to exploit these publicly available information, in this work we introduce a social information discovery system which elaborates simultaneously over more-than-one social network in an integrated scenario. The system is designed to ensure flexibility and scalability, thus enabling for (near-)real-time analysis even in case of high rates of content's creation and large amounts of heterogeneous data. Furthermore, a noise detection technique ensures a high relevance of analyzed posts/tweets to the domain of interest. We also propose a lexicon-based sentiment analysis algorithm to extract and measure users' opinion, in order to support collaboration and open innovation. Polysemous words and negations are typically challenging for lexicon-based approaches: for this reason, we introduce both a word sense disambiguation algorithm and a negation handling technique. Experiments on several datasets have proven that the combined use of both techniques improves the classification accuracy on 3-class sentiment analysis.

1. Introduction

In recent years, social networks have dramatically increased their popularity and have become part of everyday life for people of every culture and age. Enterprises already use social networks as an effective tool for marketing campaigns and to communicate with their customers. However, only few enterprises use social networks as an active source of information (e.g., for crowdsourcing and leveraging open innovation) or as a tool for collaborative product development and enhancement. Nevertheless, every day millions of social media data, in which people willingly express their opinions and emotions over a particular product or topic, are posted. Marketers could enormously benefit from their analysis: for instance, at the launch of a new marketing campaign, the analysis of social contents could provide immediate insights about people's reactions. To promote and simplify this innovative use of social networks, we believe that there is the need for an information discovery system which may enable the simultaneous analysis of multiple social networks in an integrated scenario. Such a system has to be flexible and scalable, in order to be able to handle the speed at which the contents of social network are generated, the huge amount of available data and dynamism at which networks evolve and new kinds of content are shared. As a consequence, classical Business Intelligence (BI) tools, such as data warehouses and data mining techniques, are unsuitable for this domain, as they respectively are too rigid to adapt to analysis that vary over time and too slow, if compared to the speed with which the contents of social networks evolve. To overcome these limitations and also exploit the great amount of text data available on social networks,

[*] Corresponding author.
E-mail addresses: c.diamantini@univpm.it (C. Diamantini), a.mircoli@pm.univpm.it (A. Mircoli), d.potena@univpm.it (D. Potena), e.storti@univpm.it (E. Storti).

https://doi.org/10.1016/j.future.2018.01.051
0167-739X/© 2018 Elsevier B.V. All rights reserved.

we believe that a social information discovery system should adopt Exploratory Data Analysis (EDA) techniques, in order to quickly get insights from data, and use sentiment analysis to evaluate users' opinions and monitor their progress over time.

Sentiment analysis is the automatic analysis of the attitude of a speaker/writer with respect to some topic; it encompasses a variety of Natural Language Processing (NLP) tasks, such as the detection of subjectivity and polarity, the classification of intensity, the extraction of opinion holder and targets. In our work, we are interested in the analysis of the polarity of a given text, that is the evaluation of positiveness or negativeness of the author's view towards a particular entity. Due to the intrinsic complexity of the human language, this task offers several challenges, such as the detection of the scope of negation, the interpretation of ironic sentences and the disambiguation of polysemous words (i.e., words having multiple meanings). These issues are even more significant in social networks, and in particular in microblogging platforms (e.g., Twitter), since the constraint on message length forces users to express themselves in a more creative (and less intelligible) way. As a consequence, there is the need for NLP techniques that can cope with the complexity of text analysis in social networks and hence enable for an effective analysis of users' opinions. Sentiment analysis is usually performed through lexicon-based or learning-based approaches. The latter have higher classification accuracy but they need to be trained on a large number of manually annotated samples and they do not generalize well (i.e., they have low performance on data belonging to different domains). For this reason we believe that a lexicon-based approach could be a good option for sentiment analysis of social contents, as they are very dynamic and cover a multitude of different topics.

The main contributions of this work are: (i) a methodology for the design of a (near-)real-time social information discovery system, according to the requirements of speed, flexibility and scalability; (ii) a novel set of features for noise detection in Twitter and (iii) the definition of sentiment analysis algorithms facing well-known issues of the text mining field, namely polysemy and negation, in order to increase the classification accuracy of the text polarity.

The present work is an extended version of [1], while a preliminary description of the system has been proposed in [2]. Major improvements to the previous work include the introduction and testing of a novel set of features for noise detection in Twitter, a broader evaluation of the proposed sentiment analysis algorithms on real world datasets and a deeper analysis of related work.

The rest of the paper is organized as follows: in Section 2 we present some related work on Exploratory Data Analysis, sentiment analysis and noise detection in Twitter. Section 3 introduces the requirements and the design methodology for the social information discovery system, while in Section 4 we discuss the sentiment analysis techniques used to extract users' opinions. Section 5 describes the implementation of the social information discovery system and Section 6 presents the results of an experimental evaluation of both the noise detection algorithm and the sentiment analysis techniques. Finally, Section 7 draws conclusions and discusses future work.

2. Related work

2.1. Exploratory data analysis

The EDA has been introduced in [3] and consists of a set of techniques for data analysis in absence of statistical hypotheses. EDA techniques are mainly based on data visualization, with the aim of finding interesting patterns through iterative data exploration. In order to display data in an intelligible form, EDA is based on data summarization and transformation techniques [4,5]. In the Knowledge Discovery in Databases (KDD) field, EDA techniques are mainly used in the first phase of the discovery process, in particular for data understanding and for supporting the choice of the right methodology to adopt [6]. In recent years, EDA has evolved in order to face the challenges posed by Big Data sources, such as social networks (e.g. [7,8]). In [7] advanced methods and tools for graph visualization are introduced with the purpose of studying interactions among the networks' users. An approach similar to ours is adopted in [9], where EDA techniques are used to analyze tweets.

2.2. Sentiment analysis

In recent literature, much work has been written on sentiment analysis of user-generated content [10], including reviews and social discussions [11,12]. Sentiment analysis can be performed at different levels: at document level [13], at sentence level [14] and at aspect level [15]. In our work we only focus on the analysis of sentiment at sentence level, since social contents usually consist of a single sentence, especially for microblogging platforms like Twitter. Unlike reviews, which are usually long and can be used to discuss several aspects of a topic (e.g., product, movie, hotel, and so on), a social content is formed by a short text; hence it is reasonable to assume that the writer discusses only one topic. Many different techniques have been presented to analyze sentence polarity, using both lexicon-based [16–18] and machine learning [19,20] approaches. The former involves the use of lexical resources (e.g., SentiWordNet [16]), while the latter use statistical models trained on human annotated datasets. An extensive survey on the existing sentiment analysis techniques can be found in [21]. In literature there is a debate about which technique should be used in order to obtain the better compromise between accuracy and generalizability of analysis. With respect to the former property, the state-of-the-art algorithms for sentiment analysis are actually based on deep neural networks (see [22,23]). In [24] the authors present a recursive deep tensor network that is able to model the effects of negation but it is very expensive in terms of human resources, as it requires training data to be manually annotated at several levels. An attempt to solve the problem of human annotation is proposed in [25], where automated means of labeling the training data are presented, basing on the emoticons found in the text. This method shows a good accuracy only when the models are trained on large datasets (more than 1,000,000 sentences). Felbo et al. [26] extend this approach by considering 64 emojis as noisy labels. They reach state-of-the-art performance on sentiment, emotion and sarcasm detection by training a bidirectional Long Short-Term Memory (biLSTM) model on 1.2 billion tweets with emojis. A different approach is presented in [27], where the authors propose a methodology for the automatic building of annotated corpora through the analysis of facial expressions in Youtube videos.

Since statistical approaches suffer from lack of generalization, their performance decrease when moving away from the domain on which they were trained. For this reason, they are less suitable for general purpose applications, in which we need to analyze sentences belonging to heterogeneous domains, and several authors (e.g., [28,29]) proposed to build domain-independent sentiment classifiers using lexicon-based approaches. A lexicon-based classifier for sentiment analysis is proposed in [30], where authors present the Semantic Orientation CALculator (SO-CAL), that computes sentiment polarity taking into account negations, intensifiers and irrealis markers. A limit of this work is that words are assumed to have just a single polarity, that is supposed to be independent from context. We differ from this work in that we also consider polysemous words and hence we introduce a context-based word-sense disambiguation algorithm (see Section 4.2). Our disambiguation algorithm has some similarities with [31], in that they both disambiguate a word on the basis of the glosses of nearby words. Nevertheless, our approach differs from [31] in the way the glosses are used for the disambiguation: while in [31] it is proposed to evaluate the overlaps between the glosses, our algorithm searches the contextual words in the polysemous words glosses.

Traditional supervised techniques are based on the bag-of-words approach, that is a vector representation of words' frequency in the sentence that does not take into account the context. These methods usually have less than 60% accuracy on 3-class sentiment classification. We use a different approach, since we consider the grammatical relations among words and their order of appearance, both for word sense disambiguation and negation handling. On the latter, a survey of the main techniques can be found in [32]. In [33] the authors propose to negatively label every word until the next punctuation mark. However, this simple approach is not suitable for complex sentences, since it does not consider the presence of different clauses expressing different opinions. Wilson et al. [34] extend the previous work by taking into account a fixed window of four words for negation. The approach seems to have a good accuracy, but it cannot be compared to [33], as it also considers other polarity shifters, such as intensifiers (e.g., very) and diminishers (e.g., barely). Choi and Cardie [35] present a classifier that calculates the sentiment of a negative sentence by using inference rules. Nevertheless, these approaches have several limitations, since they are not able to detect the right scope of negation. A first attempt to model the effects of negation taking into account grammatical dependencies among words can be found in [36], where a parser is considered to determine the scope of negation. This work is similar to ours, but the authors do not give information about how the parser is used, neither perform an experimental evaluation. Jia et al. [37] determine the scope of negation by considering static (e.g., because) and dynamic (e.g., for) clause delimiters and heuristic rules, while [38] rely on the combined use of semantic parsing and knowledge bases. These two approaches differ from ours in that we do not use knowledge bases or handcrafted lists of delimiters, which are often ambiguous and hence require the definition of complex disambiguation rules.

2.3. Noise detection in twitter

A general definition of noise in Twitter is not limited to spam, but it also includes all the tweets that would not be useful for the analysis, such as tweets only containing mentions, hashtags and/or links. At the moment, little research exists about noise detection in social networks and it is mainly focused in detecting spam content. In [39] a machine learning approach is proposed to detect spam bots in social networks. Wang [40] introduces the concept of user reputation and proposes to use both graph-based and content-based features in order to find spam tweets. In the paper many traditional machine learning techniques are evaluated and the Naïve Bayes classifier shows the best performance. The work is similar to ours in terms of the definition of some features and the use of machine learning techniques. A similar approach is used in [41] but results are different from [40], since the Naïve Bayes classifier shows poor performance in comparison to other classical machine-learning algorithms, such as Support Vector Machines and Random Forests.

3. The social information discovery system

3.1. Design methodology

Social network data adhere to the 3V model (Volume, Velocity and Variety) of Big Data, since they are characterized by big volume, high rate of growth and update, and a variety of (mainly unstructured) types. As a consequence, an effective and efficient Social Information Discovery project must be based on a fast and scalable system that allows for quick analysis, in order to cope with the dynamic nature of the domain. In order to take into account such desirable properties in the design of the system, we define a 4-step methodology that focuses on each critical component of the system:

- Data Definition
- ETL Design
- Text Analysis
- User Interface Design

A detailed description of each methodology step is presented in the following subsections, along with the discussion of the main issues of each phase, while the implementation of the system will be presented in Section 5.

3.2. Data definition

The main goal of an information discovery platform is to give fast answers to any analytical request of the user. Hence, a good data model should guarantee both flexibility and (near-)real time analysis. The system must be flexible to expand as soon as new demands of analysis are made. This means to adopt a simple data structure, where new data and attributes can be easily added. For instance, let us assume a system analyzing only Facebook posts. If later an analyst needs to analyze the comments to these posts, in a rigid model we need to alter the data model, adding new information regarding comments, and define new queries on that data. As a consequence, there is the need for the adoption of a versatile model, that is not subject to the rigid constraints needed to guarantee the ACID[1] (Atomicity, Consistency, Isolation and Durability) properties of transactional systems. Furthermore, since user requests are unpredictable, the design of the data structure cannot be based on well-known best practices of database design, such as database normalization. Moreover, note that, unlike classical business intelligence projects, where data sources are mainly internal to the enterprise, the analysis of social network data implies the access to external sources that could suddenly change during the project lifecycle. As a consequence, the resulting data structure will be redundant and not in normal form [42], and hence the query response time will be not optimized. However, the goal is to obtain a fast (on average) response time for any query that is executed in the system.

The above considerations also imply that the multidimensional model [43], which is the reference model for business intelligence projects, is unsuitable for this kind of analysis. In fact, the multidimensional model is designed to respond quickly to unpredictable queries, but it has a rigid structure. Therefore, adding new data to a multidimensional structure requires a redesign, which is a time-consuming activity.

In the definition of the data structure it is also important to take into account the evolution of the system. In fact, a social network could make new data available for the analysis or other data sources could become relevant for analysis. For instance, after a preliminary analysis of Facebook posts, a company could be also interested in analyzing LinkedIn discussions or YouTube comments, which have different characteristics with respect to Facebook posts. A good data design reduces evolutive maintenance costs, thus making possible the integration of new data sources without the need of a database redesign phase.

At the moment, non-relational databases (also known as NoSQL[2] or NoREL systems) are a possible solution to both the requirements of scalability and flexibility, since they usually offer simplicity of design, as they permit to store schema-less data, and horizontal scaling by distributing data over several cluster nodes.

1 https://en.wikipedia.org/wiki/ACID.
2 https://en.wikipedia.org/wiki/NoSQL.

3.3. ETL design

The Extraction, Transformation and Loading (ETL) phase plays a very important role in information discovery projects, especially in the analysis of social networks, since data extracted from these kinds of sources are characterized by high variability (i.e., large amounts of data are created or updated hourly). For this reason, the right design of the ETL process is a critical issue, that mainly consists in finding the right balance between data freshness and overall system performance. In fact, even if it is absolutely necessary to make recent and updated data available to analysts, it is also important to prevent the ETL process from requiring too many resources, so as to not penalize the performance of the analytical subsystem. For this reason, a preliminary estimation of the amount of data generated or updated (per unit time) in the considered data source is needed to properly choose the right frequency of the ETL process. The main steps of the ETL phase are:

- the extraction of meaningful information from social networks (e.g., posts, comments, number of likes)
- the filtering of unrelated content, such as spam posts
- the calculation of social metrics that are relevant for the specific application domain (e.g., number of shares, number of positive comments)
- the data integration and loading in a database

The choices made during the ETL design can be very critical as they impact on both the infrastructure (i.e., hardware choice and size) and the software architecture (e.g., structure of the ETL process, degree of parallelism of ETL activities, and so forth). Note that poor system performance may result in a loss of trust by analysts.

3.4. Text analysis

Textual data are the main source of data in social networks. Although users also share images and videos, text messages are considered to be more content-rich and, hence, more useful for analyzing what happens in the network. Messages often contain opinions and feelings about a specific topic; that information is considered authentic, since in the above contexts people usually feel free to express their thoughts, and hence its analysis is valuable. In our system we identify two main text analysis tasks: (i) *entity extraction*, that is the extraction of people, brands, products, places, hashtag and all other pieces of information that may be relevant for the analysis and (ii) *sentiment analysis*, that is the determination of the user's opinion about a particular topic. An important aspect to take into account when analyzing social text data is multilingualism, since social networks like Facebook and Twitter have users from different countries or cultures. Classical natural language processing tasks, such as tokenization and lemmatization, depend on the language and most tools in the literature are specialized for a single language. Hence, a system should use various tools in parallel (each for a different language) in order to take into account multilingualism. At the moment, in our system we focus on English messages. It should be also noted that, especially in informal messages, as those that are usually shared in social networks, people use slang and jargon to express an opinion. Hence, sentiment identification should also take into account cultural and ethnic differences. A more detailed discussion on our sentiment analysis process is presented in Section 4.

3.5. User interface design

As a result of the use of the EDA paradigm, the user interface should allow for a fast and flexible analysis. Therefore, it has to be highly interactive and it must enable users to explore data and to ask questions to the system. A solution is to provide users with different data perspectives, allowing them to navigate from one point-of-view to another one in the simplest possible way. Each perspective is based on a set of filters, that allows for an intuitive selection of data of interest, and a set of graphical tools (e.g. histograms, pie charts, geographical maps, tag clouds) to visually explore data. Since in the context of social network there are both structured and unstructured data, the system should have components able to display both kinds of data in the same user interface. Even if the user interface is provided with a set of predefined perspectives, the user has to be able to easily create a new custom view. Finally, the interface must be intuitive; in our idea of Exploratory Data Analysis, this means to allow the analyst to select (by clicking or touching) any displayed graphical element, in order to disaggregate data into its component data, and to re-aggregate them by removing filters.

4. Sentiment analysis

4.1. The sentiment analysis process

A social information discovery system must assure a fast analysis of social network contents, that are characterized by high variability of topics, and hence should be designed as a general purpose application. For this reason, performing sentiment analysis through machine learning techniques has two major drawbacks: (i) they are usually not able to generalize well and (ii) they require a costly manual annotation of a new training set and a time-consuming training phase for each domain of interest. As a consequence, in the perspective of a near-real-time analysis of social network contents, we perform sentiment analysis by means of a lexicon-based approach, that offers higher flexibility and speed in exchange for lower accuracy in sentiment detection. The chosen lexical resource is SentiWordNet 3.0 [16], that is an extended version of the WordNet ontology, where words are annotated with respect to their sentiment score. In SentiWordNet, terms are organized in synsets (synonym sets), which contain the terms that can be described by the same definition (also called gloss). Each synset has a set of attributes: an identifier, a *part-of-speech* (POS) tag, a gloss and three scores in [0,1], which represent the values of synset's positivity, negativity and objectivity. However, in this work we use a sentiment score with values in [-1,1], where -1 represents a totally negative term, 0 a neutral term and 1 a totally positive term.

The whole sentiment analysis process includes several steps, as depicted in Fig. 1. The first steps are typical NLP tasks, namely tokenization, POS tagging and lemmatization. In addition to them, we introduce a word sense disambiguation algorithm for polysemous words and a negation handling algorithm for negative sentences, that will be discussed in the next subsections. Finally, the sentiment score of the post/tweet is calculated as the mean value of its words' sentiment scores.

4.2. Word sense disambiguation

The meaning of a word is often not unique but it depends on the context, namely the surrounding words in the sentence. A large number of words in natural language has multiple different meanings: they are said to be polysemous. For example, the term "good" can be used in both the meanings of "fine" (positive sentiment) and "asset" (neutral sentiment). Although a human reader can easily choose the right meaning by analyzing the context of the word, polysemous words are an obstacle to the correct automatic evaluation of users' opinions, since their different meanings often have different sentiment scores. To this purpose, in our system we use a word sense disambiguation algorithm [44] that is based on the analysis of nearby words. For the sake of completeness, in

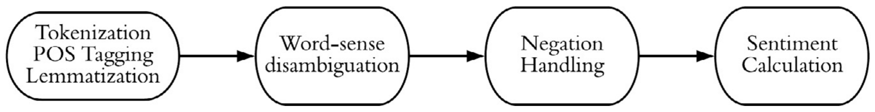


Fig. 1. The sentiment analysis process.

this subsection we describe the main steps of the algorithm. The technique disambiguates polysemous words by searching for the shortest path in a dictionary between a term and its surrounding terms. We define a *path* between two terms w_a and w_b of SentiWordNet as a sequence of glosses $\langle g_1, g_2, \ldots, g_n \rangle$ such that g_j (with $j > i$) is the gloss of $w_j \in g_i, g_1$ is the gloss of w_a and $w_b \in g_n$. The chosen dictionary is a variant of SentiWordNet, which has been denormalized by removing the synset-based aggregation: hence, each dictionary occurrence represents a different semantic variant of a single term (i.e., an occurrence of the term in a synset). In order to distinguish among the semantic variants of polysemous terms, we added a *variant number* attribute with progressive value.

The algorithm disambiguates a word w_1 by searching, in the w_1's glosses, for an occurrence of the word w_2 directly before or after w_1. Both the previous and following words give information about the contextual meaning of w_1. We first select the word following w_1 as w_2; if w_2 is not useful to disambiguate w_1, i.e., w_2 is not in SentiWordNet or w_2 is not in the glosses of w_1, then we assign to w_2 the word preceding w_1. If both previous and following words are not useful to disambiguate w_1, then the sentiment score of w_1 is calculated as the mean value of the sentiment scores of its semantic variants. In order to increase the chance for the selected w_2 to be useful for the disambiguation of w_1, we added a preprocessing phase of POS filtering, through which to delete from the sentence all words that are not names, pronouns, adjectives or verbs. If a matching between w_1 and the chosen w_2 exists, the algorithm stops and selects the semantic variant having the matching gloss as the most suitable meaning for w_1. Otherwise, the search is extended to every definition of each word included in w_1's glosses, then to each definition of each word of those definitions and so forth, up to a maximum search depth (defined on the basis of time and/or memory constraints). The result is a n-ary search tree, whose nodes contain glosses, that is explored using a breadth-first search (BFS) strategy. The BFS strategy guarantees a better accuracy than other strategies (e.g., depth-first search), because it first explores all the nodes that are closer (and hence more related) to w_1. A necessary, but not sufficient, condition for the convergence of the algorithm is that both words must be included in SentiWordNet, either as dictionary occurrences or as part of a gloss. The pseudo-code of the WSD algorithm is shown in Algorithm 1.

The inputs of the algorithm are w_1 (the term to be disambiguated), w_2 (the term used for disambiguation) and *depth* (the maximum search depth). The last parameter is a simple pruning criterion and it allows to stop the search at the desired depth. The output is the sentiment score, with values in [-1,1] of the most suitable meaning of w_1 on the basis of its context. A *null* value is returned in case of failure, that is if a path between w_1 and w_2 does not exist or is deeper than the chosen value of *depth*.

At the beginning of the algorithm, the root node, corresponding to w_1, is put into the queue (statement 2). In the loop condition, the node is extracted from the queue (statement 4) and its semantic variants are retrieved (statement 5), then a new child node is added for each definition of w_1 in SentiWordNet. Therefore, the algorithm searches for w_2 among glosses of existing nodes (statements 6–8): if it fails, new children nodes are added from actual nodes (statement 8) and the search starts again (statement 3), until w_2 is found or the maximum search depth is reached. If a match between a node and w_2 exists (statement 7), the definition of w_1 that is an

Algorithm 1 Word Sense Disambiguation

```
    Let $w_1$ be the term to be disambiguated,
    let $w_2$ the term used for disambiguation,
    let $S(x)$ be the set of polysemous variants of term $x$,
    let $G(y)$ be the set of terms in the gloss of the variant $y$,
    let $sent(y)$ be the function that returns the sentiment score
    for variant $y$,
    let $anc(x)$ be the ancestor of $x$ which is child of $w_1$,
    let $Q$ be an empty queue of terms.
 1: function WSD($w_1$, $w_2$, $depth$)
 2:     $Q.push(w_1)$
 3:     while Q not empty && depth > =1 do
 4:         $t \leftarrow Q.pop()$
 5:         for each x in S(t) do
 6:             for each j in G(x) do
 7:                 if $w_2 == j$ then
 8:                     return $sent(anc(j))$
 9:                 $Q.push(j)$
10:             $depth \leftarrow (depth - 1)$
11:     return null
```

ancestor of the current node is selected as the most fitting semantic variant through the *anc(x)* function. Its sentiment score is assigned to w_1 and the algorithm successfully terminates; otherwise a *null* value is returned (statement 11).

As an illustrative example, we consider the sentence "He played in this competition". After standard text pre-processing operations (i.e., tokenization, lemmatization and POS filtering), we obtain the bigram "play competition". The word "play" is polysemous: there are 52 occurrences of the term in SentiWordNet. The related sentiment scores range from −0.5 to 0.25: the choice of using the mean value of the scores as the sentiment score does not guarantee acceptable accuracy, because of the large variability of scores in the different semantic variants of the term. In order to disambiguate the word and determine the most fitting sentiment score, we apply the WSD algorithm: the results are shown in Fig. 2 (due to the high branching factor of the resultant search tree, some irrelevant definitions are not displayed).

In the Figure, boxes represent terms in SentiWordNet with related glosses; dotted boxes contain semantic variants of the same term. The first step is the generation of the first level of the search tree, whose nodes contain the definitions of the term "play". The information on the nodes is represented in the form (word part-of-speech#variant-number#sentiment-score); here the attribute part-of-speech assumes values a(ttribute), v(erb) and n(oun). The algorithm searches for the word "competition" in every definition of "play" but does not find it. Therefore, the second level of the search tree is generated, adding a new child for every semantic variant of each word of the definitions. Now the algorithm succeeds, finding a match between the term "competition" and the definition of "sport" (the highlighted node). Hence, the score of the ancestor of "sport" is assigned to the word "play".

The disambiguation of a term is a computationally intensive task, as it requires to search in a n-ary imbalanced tree. The computational complexity of the algorithm is O($\beta^{\delta}\tau^{\delta-1}$), where β is the average number of semantic variants of a term in SentiWordNet, τ is the average number of words in the gloss of a semantic

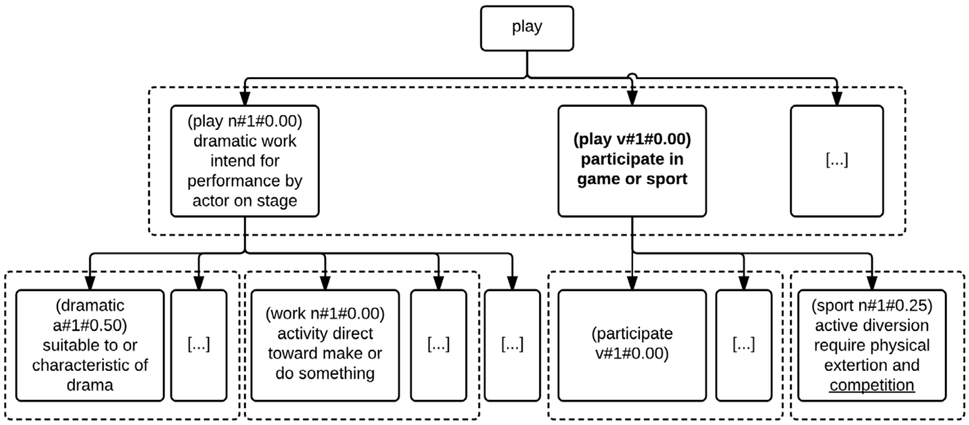

play
(play n#1#0.00) dramatic work intend for performance by actor on stage
(play v#1#0.00) participate in game or sport
[...]
(dramatic a#1#0.50) suitable to or characteristic of drama
[...]
(work n#1#0.00) activity direct toward make or do something
[...]
[...]
(participate v#1#0.00)
[...]
(sport n#1#0.25) active diversion require physical extertion and competition

Fig. 2. Example of the WSD algorithm.

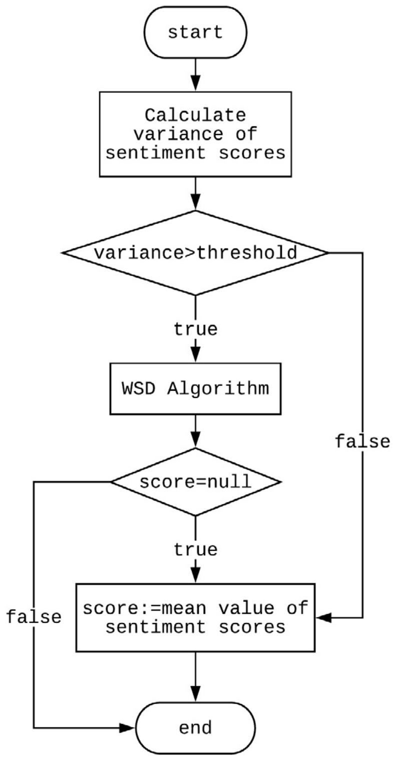

start
Calculate variance of sentiment scores
variance>threshold
true
false
WSD Algorithm
score=null
true
false
score:=mean value of sentiment scores
end

Fig. 3. Workflow of *word sense disambiguation*.

variant, and δ is the threshold on the search depth. In real world applications it could potentially lead to execution times that are unacceptable for near real-time analysis. A possible solution is to only disambiguate polysemous terms having a great difference in sentiment scores among their semantic variants, while computing the sentiment scores of the other terms as the mean value of their semantic variants. The idea is depicted in Fig. 3, where the variability of the sentiment scores of a polysemous term is measured in terms of variance and γ represents a chosen threshold.

4.3. Negation handling

In sentiment analysis, the correct evaluation of negative sentences is a challenging task, since there are no fixed rules to determine the scope of negation. A negation word (e.g., "not", "no") is defined as a word that alters the semantics of a sentence by inverting the polarity of a certain number of following words. A simple approach for negation handling consists in inverting the polarity of the first term following the negation word, but this technique is unsuitable in many cases, for instance in presence of intensifiers (e.g., we are not very happy). A more robust approach must rely on the detection of the scope of negation, that is the number of terms that are affected by negation. In general, the negation window cannot be fixed, as it depends on the particular structure of each sentence: for instance, complex sentences can have several dependent clauses, connected by many conjunctions, and only a subset of them may be altered by negation. The proposed algorithm parses the sentence using a statistical dependency parser [45], in order to analyze its grammatical structure and separate clauses; after that it builds the dependency-based parse tree and searches for negation words through a depth-first search (DFS) strategy. In our approach we make the assumption that a negation word only affects terms belonging to the same clause. Therefore, if a negation word is found in a tree node, the algorithm inverts the polarity of its following sibling nodes (exploring their subtrees, if necessary), as they all belong to the same clause. Furthermore, the sentiment score of the negation word is set equal to 0, since a negation does not have a positive/negative meaning by itself. The pseudo-code of the negation handling algorithm is shown in Algorithm 2.

Algorithm 2 Negation Handling

```
    Let children(n) be the function that returns the children nodes
    of node n,
    let isLeaf(node) be the function that returns true if node
    is a leaf
    let isNegation(t) be the function that returns true if t is
    a negation word,
    let score(t) be the sentiment score of term t
    let rLeaves(n) be the leaves of the right sibling nodes of n.
1:  procedure NH(node)
2:      if isLeaf(node) then
3:          if isNegation(node) then
4:              $score(node) \leftarrow 0$
5:              for each n in rLeaves(node) do
6:                  $score(n) \leftarrow score(n)(-1)$
7:      else
8:          for each t in children(node) do
9:              NH(t)
```

To better illustrate the algorithm, we consider the sentence "Anna did not love the concert since the songs were horrible". The sentence is composed of two clauses and the negation (i.e., the word "not") has the effect of inverting the polarity of its following

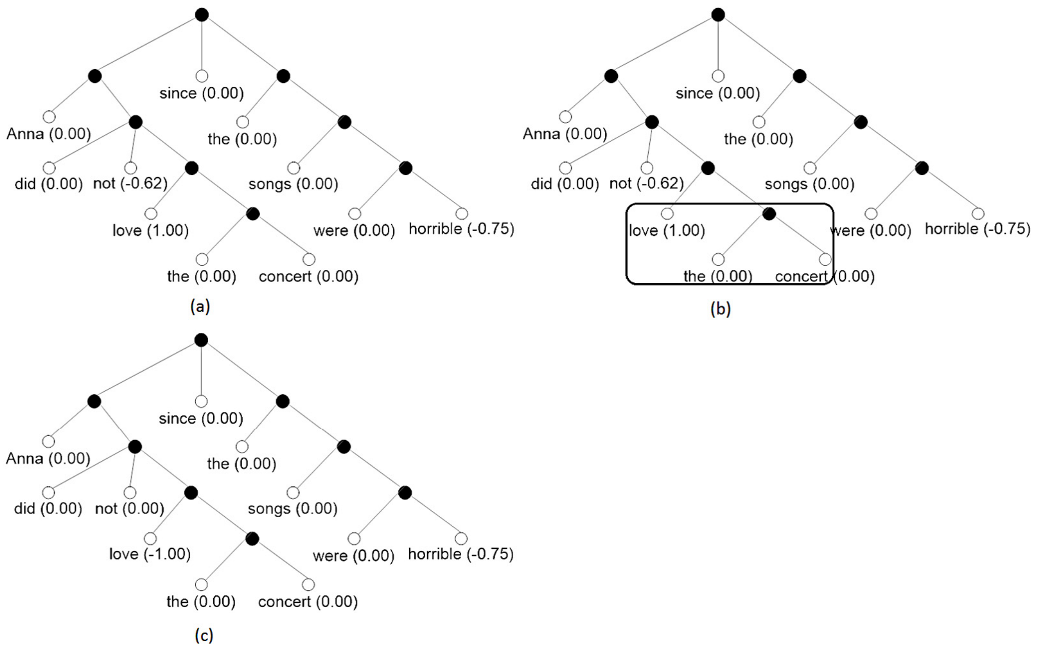
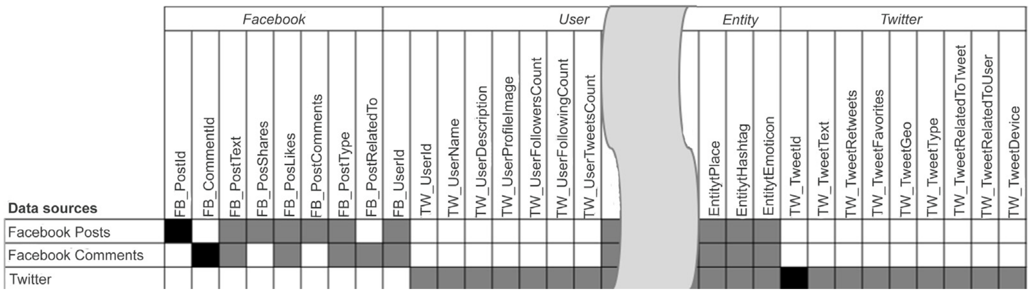


Fig. 4. Example of the *negation handling* technique: (a) the algorithm builds the parse tree and (b) searches for words affected by negation, then (c) their sentiment scores are inverted and the sentiment score of the negation word is set to 0.



	Facebook								User									Entity			Twitter								
Data sources	FB_PostId	FB_CommentId	FB_PostText	FB_PosShares	FB_PosLikes	FB_PostComments	FB_PostType	FB_PostRelatedTo	FB_UserId	TW_UserId	TW_UserName	TW_UserDescription	TW_UserProfileImage	TW_UserFollowersCount	TW_UserFollowingCount	TW_UserTweetsCount		EntitytPlace	EntitytHashtag	EntitytEmoticon	TW_TweetId	TW_TweetText	TW_TweetRetweets	TW_TweetFavorites	TW_TweetGeo	TW_TweetType	TW_TweetRelatedToTweet	TW_TweetRelatedToUser	TW_TweetDevice
Facebook Posts	■		▨	▨	▨	▨	▨		▨								▨	▨	▨	▨									
Facebook Comments		■	▨		▨		▨	▨	▨								▨	▨	▨	▨									
Twitter										▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	■	▨	▨	▨	▨	▨	▨	▨	▨

Fig. 5. An excerpt of our data model.

words in the first clause. The corresponding dependency-based parse tree (Fig. 4(a)) is built by the dependency parser through the analysis of the grammatical relations in the sentence. In the parse tree, each leaf node contains a term and its sentiment score. The algorithm searches for negation words using a *depth-first search* (DFS) strategy and finds the negation word "not", so it explores the subtree of its right sibling node (Fig. 4(b)). The subtree contains three words, namely {"love", "the", "concert"}; in particular, the word "love" has a positive score (i.e., +1). For the effect of negation, the sentiment score of these words is inverted and, finally, the sentiment score of the negation word "not" is set to 0 (Fig. 4(c)). The second clause, instead, is correctly left unchanged and the negative score of the word "horrible" is preserved.

5. Implementation

5.1. Data definition

The first step is the definition and analysis of data sources. We have selected Facebook and Twitter and we have defined what types of data from these platforms are relevant for our system. In particular, for Facebook we are interested in retrieving users' and pages' posts, in particular those that match with a set of given keywords, as well as information about users. To this purpose, we have used the Facebook Graph API[3] for extracting these contents and related information (e.g., timestamp, number of likes, shares, comments). We also retrieved users' details, which are not given with the post, by making a Facebook Query Language (FQL) request. The obtained information about users are: name, surname, gender and local information (in the form `<language>_<region>`) . We have gathered all comments related to a post through FQL requests, in order to monitor the reactions of other people to thoughts and opinions expressed in the post. For Twitter, we have used Twitter API[4] to search for tweets containing a set of given keywords. Contrary to Facebook API, the Twitter API response already contains all information about the tweets and the users who created them.

We have chosen the *data lake* as data model, since it offers high versatility. In fact, it allows the collection of all data in a single store, regardless of the variety of data structures of the analyzed sources. An excerpt of our data model is shown in Fig. 5: each row represents a data source (e.g., Facebook posts, Facebook comments and tweets) and each column is an attribute. If the color of a cell is black it means that the attribute is a primary key for that source, while a gray cell means that it has been extracted from that source; white cell is an attribute that is not related to the source. Columns without white cells represent global attributes. Therefore, to add

3 https://developers.facebook.com/docs/graph-api.
4 https://dev.twitter.com/overview/api.

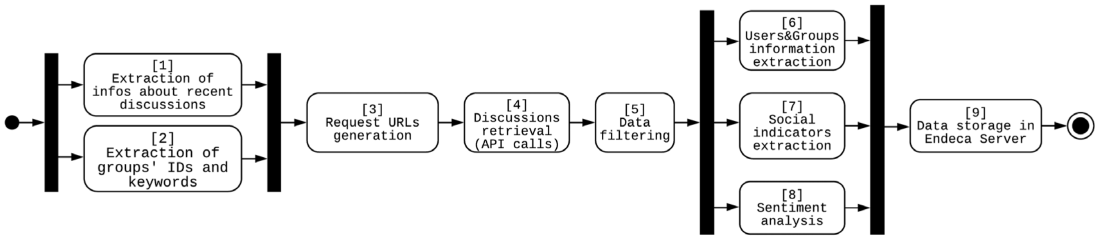

[1] Extraction of infos about recent discussions
[2] Extraction of groups' IDs and keywords
[3] Request URLs generation
[4] Discussions retrieval (API calls)
[5] Data filtering
[6] Users&Groups information extraction
[7] Social indicators extraction
[8] Sentiment analysis
[9] Data storage in Endeca Server

Fig. 6. ETL process for LinkedIn discussions.

a new data source we only need to reconcile heterogeneities. To this purpose, we followed a bottom-up and incremental approach: first, we considered the attributes extracted from a single query and added to our data model. Then we added attributes from another query and so on. In this way, at each step we only have heterogeneity between two schemas: the analyzed query and the partial data structure resulting from the previous integration. The final schema is composed by local and global attributes. Local attributes are specific for a single data source (e.g., the number of retweets for a tweet), while global attributes are common to every data source (e.g., the text of a post/tweet). This approach guarantees an easy integration of new data models, thus allowing for a fast addition of new social networks to the SID system. As an example, in [44] a master student added the entire ETL process for analyzing LinkedIn discussions to the Social Information Discovery system in a few days.

In order to ensure an efficient representation of extracted data, we have chosen Oracle Endeca Information Discovery,[5] because it is based on a column-oriented DBMS, which offers a better performance when querying for aggregate values, and uses in-memory computing to further reduce the response time.

5.2. ETL design

The ETL processes are developed using Oracle EID Integrator Designer, which is the ETL tool in the Endeca Information Discovery suite, and consists of the retrieval, transformation and integration of social data. As an example, let us consider the ETL process to collect LinkedIn discussions (see Fig. 6). The process starts by querying the database to obtain the timestamps of the last loaded discussion for every keyword of interest. The timestamp is used to build a request to LinkedIn Application Programming Interface (API), in order to request discussions that have been created after that timestamp. Since LinkedIn API sometimes returns spam posts or posts that are unrelated to the searched keyword, the next step consists of applying a filter on the results.

At this point we launch three parallel tasks: the retrieval of user's geographical information, the extraction of social indicators and the text analysis. In order to extract geographical information, for each discussion we make a request to Google Maps API for retrieving geospatial coordinates of the user's city/country. With this information, the system is able to show on a map the most active regions in terms of number of published contents and the geographical distribution of authors. Finally, the ETL process merges information and loads them into the database. Given the very dynamic nature of social media, the information associated with a discussion usually change over time. Therefore, we implemented another ETL process to check for updates, to extract recently changed information and subsequently update the system database.

[5] http://www.oracle.com/technetwork/middleware/endeca/overview/index.html.

This modular approach allows ETL designers to easily integrate new data sources: in fact, the only steps that need to be added are those related to data extraction. Therefore, the integration of a new social network simply reduces to querying its API to obtain the desired information and add new mappings between these data and the attributes in the data model.

The execution time of the ETL process is a critical issue, as it influences the data freshness. We worked on the efficiency of the ETL processes to guarantee near-real-time data loading: at the moment, the system is able to extract and analyze data with a maximum delay of 5 min from the creation of the content. At that speed, it can collect and process up to 13 million tweets per month and 0.2 million Facebook contents (posts and comments) per month.

5.3. Noise detection

Since a considerable amount of extracted data is unrelated to the domain of interest (e.g., spam posts) or is not directly analyzable (e.g., tweets only containing links), there is the need to filter out noisy content in order to enhance the quality of the analysis. To this purpose, we introduce a noise detection technique that takes into account both graph-based and text-based features. Due to the definition of some Twitter-specific features, the scope of application of the technique is limited to tweets. However, this is not a significant limitation, since Twitter is the data source that offers the largest amount of available data through API calls (one order of magnitude more than Facebook), so it represents the most suitable social network for large scale analysis.

The technique uses machine learning algorithms to perform tweet classification and it is based on four features. Two of the four considered features are also defined in [40], that is:

- *reputation* = $\frac{following}{following+followers}$, where *following* represents the number of users followed by the tweet's author and *followers* is the number of users that are currently following him/her.
- *duplicate tweets*, that is tweets that have been created by the same user and have the same textual content, except for eventual mentions, which are usually varied by spammers in order to avoid being detected by Twitter's spam detection algorithm.

In addition to them, we introduced two novel features:

- *the amount of common words*. Since spam posts usually have a small quantity of common words, as they contain links and products/brands names, this feature is defined as the ratio between the number of tweet's words that are also in a chosen dictionary (e.g., Cambridge Dictionary, Oxford Dictionary, WordNet) and the total number of words.
- *the presence of verbs*. Verbs cover a key role in the sentence and hence their absence is a potential spam signal. For this reason, we perform a preliminary *part-of-speech* (POS) tagging to identify verbs in the tweet and we use this binary (0-1) feature to represent the absence/presence of verbs.

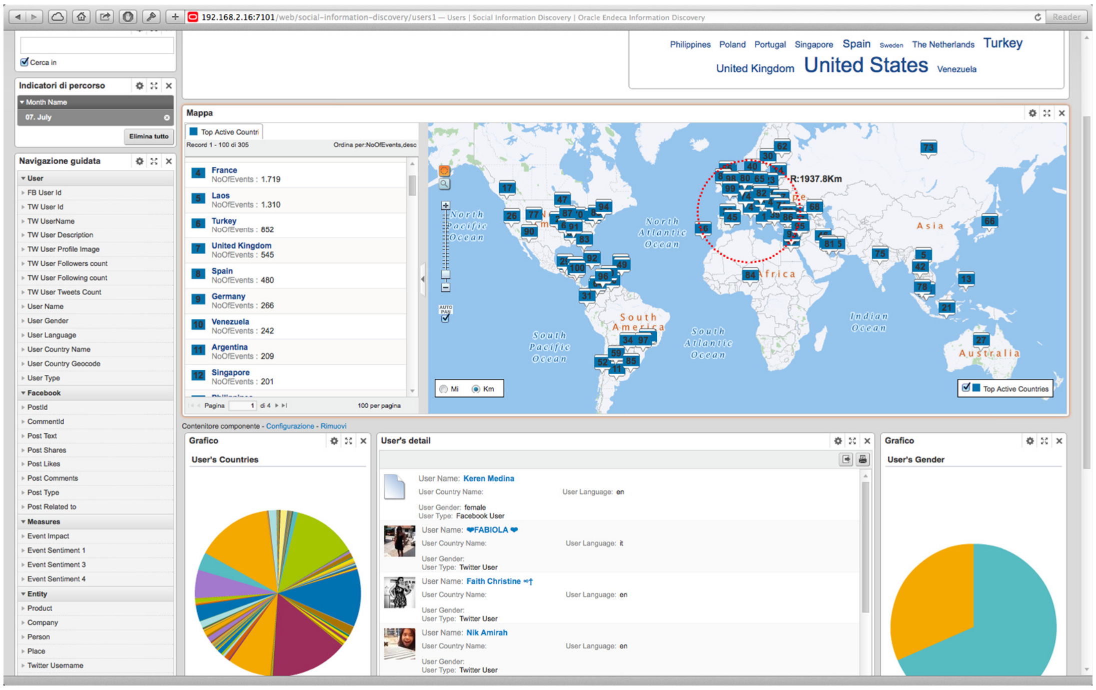


Fig. 7. An example of Users perspective: users can be analyzed with respect to their geographical distribution and language. Users with the highest social influence (influencers) are reported on the right. A set of filters (left sidebar) allow for a dynamic and highly customizable analysis.

5.4. User interface

In order to build a highly interactive and versatile user interface that provides advanced analytical tools, we used Oracle EID Studio, which is the component of Endeca Information Discovery used for the creation of the user interface. We have built the interface following the idea of giving users different data perspectives. In particular we have designed four different point-of-views, which show all the available data: the PostsTweets perspective, the Users perspective, the TextAnalysis perspective and the Competitors perspective. The first three perspectives focus respectively on a quantitative analysis of contents, characteristics of users and sentiment analysis. The last perspective, on the other hand, provides a comparative analysis with a set of user-defined competitors (e.g., other brands selling a similar product) over the above metrics.

Each perspective is available as a panel in the user interface, and the user can move from one perspective to another. Queries are formulated by setting different filters, like presence or absence of keywords, popularity of the content, characteristics of authors, geographical origin, and so forth. Active and available filters are shown in the left sidebar; results are returned in various forms (mainly graphical) in the rest of the panel. When adding or deleting filters, the interface is automatically updated in every perspective. A filter can be set by clicking on a term or a graphical object (e.g., a bar of a histogram, a portion of a pie chart, an element of a map) shown in a perspective.

In order to show the functionalities offered by the user interface, we use the Users perspective (Fig. 7) as an illustrative example. This perspective shows information about social network users with regard to their language and geographical distribution and allows to restrict the analysis to a specific region or country, setting a geospatial filter by simply defining a center and a radius on the map. This tool supports analysts in understanding how a product is seen by the different populations. Another interesting feature is the list of influencers, that is the users that have the highest social influence – measured in terms of likes, share, retweets and so forth – on the analyzed topic: it helps analysts in identifying users whose opinions impact on large crowds and eventually contact them to propose collaborations. On the left, the box "Selected Refinements" shows currently active filters, while "Available Refinements" contains the list of all the available filters that can be set to customize the analysis.

Another interesting feature, especially in the perspective of using the system as a support tool for collaborative development and enhancement of product's characteristics, is represented by word clouds (Fig. 8). On the top of the figure, two boxes contain the most frequently used positive and negative words: these two word clouds provide valuable information about the most appreciated feature and the critical problems of a product/brand. For example, the words in Fig. 8 are the result of searching for the keyword Ducati and they allow to discover that the grip and the brackets are appreciated characteristics of Ducati bikes, while the tank is often associated with negative comments. The presence of the words *peak* and *pike* in the word cloud of positive words is justified by the existence of the model *Ducati Multistrada 1200 Pikes Peak*. When coupled with geospatial filtering, these word clouds allows to rapidly detect which are the most appreciated/hated characteristics of a brand (or a product) for each country. On the bottom of the figure, the word cloud of hashtags shows the most frequent hashtag, thus providing information about the trending topics that are associated with the analyzed keyword, while the word cloud of mentions contains the usernames of the users that are more frequently mentioned in tweets.

We would like to note that our system is not only a decision support system for marketing, but it is also a valuable tool in crowd-sourcing and, hence, open innovation activities. For example, the information given by word clouds, namely the entities (e.g., product features) that are considered positive/negative by users, are valuable for catching the way in which people "talk" about a topic and hence can promote a collaborative development of new features. Furthermore, by varying the set of search keywords, the analyst can move the analysis from a specific product to an entire domain (e.g., motorbikes). Hence, the analysis can be aimed at studying a new need, an innovative idea for a new product, an improvement of existing products and so forth. For what concerns software usability, we have not yet performed a

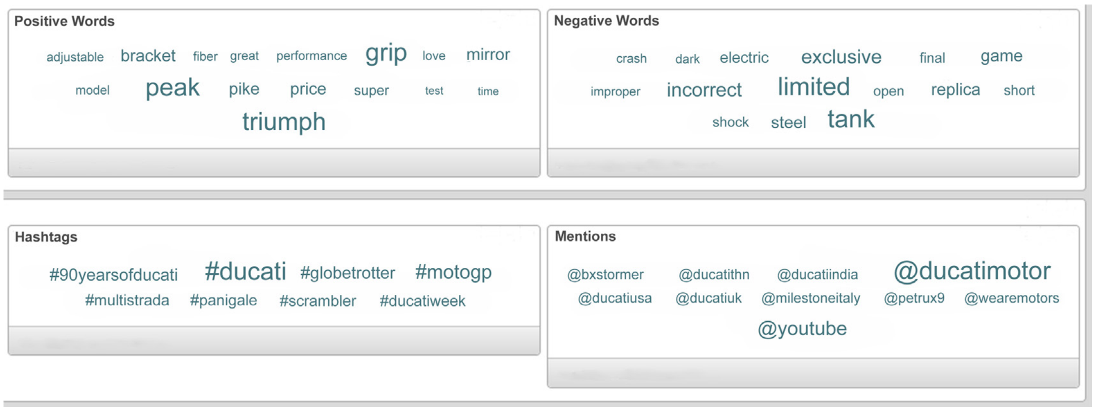

Positive Words
adjustable bracket fiber great performance grip love mirror model peak pike price super test time triumph
Negative Words
crash dark electric exclusive final game improper incorrect limited open replica short shock steel tank
Hashtags
#90yearsofducati #ducati #globetrotter #motogp #multistrada #panigale #scrambler #ducatiweek
Mentions
@bxstormer @ducatithn @ducatiindia @ducatimotor @ducatiusa @ducatiuk @milestoneitaly @petrux9 @wearemotors @youtube

Fig. 8. An example of word clouds. From top left: most frequent terms appearing in positive and negative tweets; most frequent hashtags; most frequent mentions.

usability test but we plan to do it as future work. In the last months we have shown our platform to several students and businessmen that were interested in using the software and they were all able to use it quickly, without a learning phase.

6. Evaluation

This section presents some experimental results aiming at analyzing the performance of several modules of the social information discovery system on real-world data. The experiments have been divided into two parts: in the first part, we focus on the evaluation of the classification accuracy of the noise detection technique and we compare the results obtained by our approach with a state-of-the-art algorithm. In the second part, we evaluate the sentiment analysis process and we measure the accuracy of the system using different metrics. We analyze both the performance of sentiment score calculation and sentiment classification on four datasets consisting of manually collected, cleaned and annotated tweets. The purpose of the analysis is to investigate the effects of coupling the word sense disambiguation and negation handling algorithms, and hence show that the use of these techniques leads to significant improvements in sentiment analysis in comparison to traditional lexicon-based classification.

6.1. Noise detection

In order to evaluate the effectiveness of the proposed features for the noise detection task, we trained on our features several classic classification algorithms, namely Multilayer Perceptron (MLP), Support Vector Machines (SVM), Naïve Bayes (NB) and Decision Trees (DT). We used a dataset composed of 400 tweets, along with information about the number of following and followers of each author. The dataset has been manually annotated by a human annotator and is balanced, i.e. 200 spam/noisy and 200 non-spam tweets. We validated our models through a k-fold cross-validation (k=10). In contrast with [41], we noticed that the Naïve Bayes classifier outperforms all other methods, reaching a 91.2% classification accuracy. In order to compare our approach with the state-of-the-art algorithm proposed by Wang [40], we trained and tested the abovementioned classifiers on our dataset, using both the features used in [40] and our features. For what concerns the MLP classifier, we set two hidden layers and we tested both hyperbolic tangent and rectified linear unit (ReLU) activation functions. SVM has been tested with both linear and radial basis kernel functions. The results of the experiments are shown in Fig. 9. When using our features we reached higher accuracy than [40] on every machine learning algorithm that we considered, with a remarkable +2.6% improvement on classification accuracy when using the NB

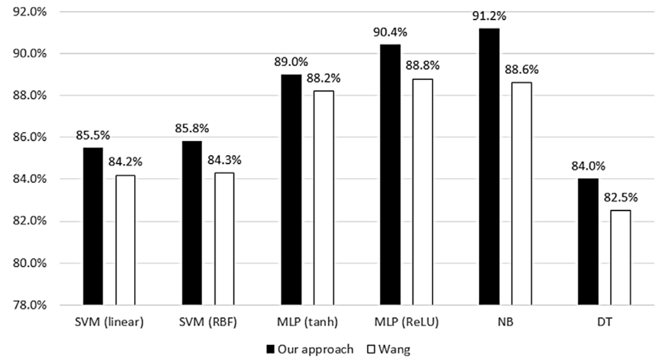


Fig. 9. Comparison between our approach and the spam detection algorithm proposed by Wang. Experiments are conducted on several machine learning algorithms. The classification accuracy is reported in the *y*-axis.

classifier. Therefore, we can conclude that the proposed features are able to improve the accuracy of the state-of-the-art algorithms for noise detection in Twitter.

6.2. Sentiment analysis

6.2.1. Results

In order to show the effectiveness of the WSD and NH algorithms, we performed some experiments using five different datasets of manually annotated tweets. The goal of these experiments are: (i) to measure the improvements introduced by the algorithms with respect to the traditional lexicon-based approach, in which a tweet's sentiment score is computed as the simple mean value of each word's sentiment score, and (ii) to compare our approach to other lexicon-based and learning-based approaches.

The first dataset is a corpus of 100 tweets about the movie American Sniper, that have been manually cleaned (removing spam, links, emoticons and retweets) and labeled by five human annotators; given that the opinion of a single annotator can be questionable, each tweet has been evaluated by three different people. Given the nature of the topics covered in the movie, which raise moral and political issues, this dataset is particularly challenging for sentiment classifiers because of the wide range of opinions it contains, sometimes even conflicting in the same sentence.

The second dataset is a corpus of 497 manually annotated tweets about several different topics, ranging from known brands (e.g., Apple) to politicians (e.g., Obama). Testing a sentiment classifier on a dataset with such a variety of topics can be valuable, as people generally use different writing styles in different contexts.

Table 1
Number of negative (Neg), neutral/objective (Neu) and positive (Pos) sentences in each dataset.

Dataset	Neg	Neu	Pos	Total
AmericanSniper	33	32	35	**100**
Multiset	177	139	181	**497**
SST	912	387	911	**2210**
SemEval2016	326	635	703	**1664**
TSAD	2657	=	2343	**5000**

Moreover, this dataset is useful to test the system as a general purpose sentiment classifier. In the rest of the paper we refer to this corpus as Multiset.

The third dataset is part of the Stanford Sentiment Treebank (SST) and is described in [24]. It consists of 2210 sentences that have been annotated with respect to five classes (i.e., very negative, negative, neutral, positive, very positive). Since we are interested in 3-class sentiment analysis, we considered both *very negative* and *negative* sentences as negative, as well as *positive* and *very positive* sentences as positive.

The fourth dataset is a corpus of 2000 tweets that has been used as a test set in SemEval-2016[6] (Task 4). We were unable to download the entire dataset because some tweets were deleted (or not available, due to modified authorization status) and hence our final dataset consists of only 1664 tweets.

The last dataset[7] consists of 5000 tweets (2657 negative and 2343 positive tweets) that have been randomly selected from the "Twitter Sentiment Analysis Dataset"[8] (TSAD), that is composed of 1.578.627 tweets that have been classified by analyzing emoticons in the text.

More information about the class distribution in each dataset are reported in Table 1.

We performed six experiments on each dataset: first, we used a traditional lexicon-based approach (LBA), which means that we simply calculated sentiment scores of a sentence as the mean value of the scores of its words. Then we separately used the word-sense disambiguation (WSD) and negation handling (NH) algorithms and we tested the complete sentiment analysis process (i.e., we coupled the WSD and NH algorithms). For what concerns the parameter tuning of the WSD algorithm, in our experiment we set the maximum search depth to 2. In general, adding algorithms (WSD, NH or both) to our sentiment analysis pipeline has the effect of improving classification accuracy, while decreasing the speed of analysis. Furthermore, we evaluated the performance of Subjectivity Lexicon,[9] which is another lexical resource for sentiment analysis. Subjectivity Lexicon provides information about the subjectivity (i.e., weak/strong subjectivity) and the polarity (i.e., positive/negative polarity) of each term. We assigned a -1/+1 score to negative/positive terms with strong subjectivity and a -0.5/+0.5 score to negative/positive terms with weak subjectivity.

Finally, we compared our sentiment analysis pipeline with a learning-based approach, namely Stanford CoreNLP [24]. The comparison has also been useful to evaluate the generalizability of learning-based approaches, since we trained Stanford CoreNLP on the SST dataset (as in [24]) and we analyzed how it performs on the other datasets. The results are shown in Table 2.

Adding the WSD and NH techniques respectively results in an average 2.6% and 4.6% improvement of classification accuracy, if compared to the traditional lexicon-based approach (LBA). It is also remarkable that combining both techniques results in an average +6.7% improvement of classification accuracy, confirming our expectations in terms of their compositional effects. The highest accuracy (67%, +8% with respect to LBA) is reached on the American Sniper dataset. In general, we observe that the NH algorithm leads to improvements greater than the WSD algorithm. This phenomenon can be explained by the fact that negations have clause-level effects and hence their correct analysis can have positive effects on many words. Note that, as expected, our full sentiment analysis pipeline performs better than traditional lexicon-based algorithms (LBA) but its average classification accuracy is lower than state-of-the-art deep learning approaches. Nevertheless, it is important to note that, even if deep learning approaches show higher accuracy in the domains they are trained on (i.e., SST dataset), they offer no guarantees in terms of classification accuracy in presence of cross-domain datasets. As a matter of fact, reusing the trained net on sentences from a different domain, the accuracy of the network drops and is lower than the one of our approach (see Table 2). Moreover, the performance of our classifier are partly affected by the use of SentiWordNet, which has been semi-automatically annotated and hence has a quite low quality of annotation. However, using SentiWordNet (LBA) we reached higher accuracies than Subjectivity Lexicon on 3 out of 5 datasets and using the full sentiment pipeline we reached the highest accuracies on all the considered datasets. The creation of stable and accurate lexical resources for sentiment analysis is an active research field we are currently working in. Another consideration is that the goal of our work is to design and develop a platform for near-real-time analysis of social contents through Exploratory Data Analysis (EDA), with the purpose of providing analysts with immediate insights on fresh data, hence giving priority to speed over accuracy. In fact, in this context it is more important to have a "fast" (rather than "accurate") tool for sentiment analysis. For what concerns the generalizability of analysis, a lexical resource needs to be refined due to evolution of the language, but this evolution is usually slow and the update is needed only rarely. On the contrary, deep learning methods need to be retrained for each specific application domain. In theory, with the right amount of data, a deep learning approach could generalize much more than lexicon-based approach, but there are two main issues. First, using data from different domains could lead to have terms (or sentences) that are polarized in different ways and this is a source of noise for the network. Second, due to the high complexity of natural language, there is the need to manually annotate a very large dataset (sometimes even millions of sentences), which is an expensive and time-consuming activity. For all these reasons, lexicon-based approaches can be considered a valid option for near-real-time analysis of social contents.

6.2.2. *Performance optimization*

The performance of the sentiment analysis process largely depends on the execution time of the WSD phase, which represents the system's bottleneck. For this reason, we focused on the optimization of SentiWordNet, in order to speed up the search of a word in another word's glosses. Since stop words, such as conjunctions or articles, do not give information about the contextual meaning of surrounding (polysemous) words, we filtered them out both from the analyzed tweets/posts and words' glosses in SentiWordNet. Moreover, we created a pre-lemmatized version of SentiWordNet, so as to not lemmatize each term of every gloss whenever a polysemous word needs to be disambiguated.

As depicted in Fig. 10 the execution time of the WSD phase has been drastically reduced. The effects of the optimizations are more visible when using lower thresholds on the variance of the semantic variants' sentiment scores. In particular, when the WSD algorithm is applied to every polysemous word (i.e., threshold=0), such optimizations result in a 75.8% reduction of the execution

6 http://alt.qcri.org/semeval2016/task4/.
7 http://kdmg.dii.univpm.it/?q=content/CTS16_SI_dataset.
8 http://thinknook.com/wp-content/uploads/2012/09/Sentiment-Analysis-Dataset.zip.
9 http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/.

Table 2
Classification accuracy with: a traditional lexicon-based algorithm based on SentiWordNet (LBA), the word sense disambiguation algorithm (WSD), the negation handling algorithm (NH), their combination (WSD+NH), Subjectivity Lexicon (SL) and Stanford CoreNLP (SC)

Dataset	LBA	WSD	NH	WSD+NH	SL	SC
AmericanSniper	59.0%	64.0%	65.0%	67.0%	58.0%	72.0%
Multiset	56.4%	58.0%	61.1%	62.6%	51.4%	62.8%
SST	54.2%	53.9%	57.2%	60.9%	55.3%	80.8%
SemEval2016	46.1%	48.7%	51.6%	52.3%	44.8%	60.7%
TSAD	60.2%	64.4%	64.2%	66.5%	61.4%	50.3%
Average	**55.2%**	**57.8%**	**59.8%**	**61.9%**	**54.2%**	**65.3%**

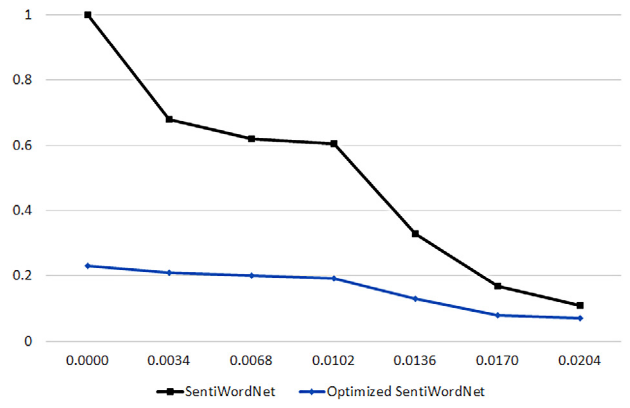


Fig. 10. Effects of the optimization of the WSD algorithm on the total execution time of the sentiment analysis process. On the *x*-axis different values for the variance threshold are reported, while on the *y*-axis the execution times of the entire sentiment analysis process, normalized with respect to the execution time of the experiment with threshold=0 and standard SentiWordNet, are reported.

time. For instance, the time needed to analyze a test set composed of 100 tweets reduced from 1475 to 357 s, which means an average execution time (for the entire sentiment analysis pipeline) of 3.57 s per tweet. Without using the word-sense disambiguation the average execution time is 1.12 s per tweet, so applying the word-sense disambiguation to every term (i.e., threshold=0) results in a 3.2x increase in execution time. The optimizations have also minimized the difference in execution time among different threshold values: for instance, the difference between the execution time with threshold=0 and threshold=0.02 has reduced from a 10x to a 2.5x factor.

7. Conclusion

The goal of this work is the development of an integrated system for information discovery from multiple social networks, which allows for the analysis of users' opinions and characteristics and is based on exploratory data analysis techniques. Furthermore, the paper introduces a novel set of features that are demonstrated to improve the classification accuracy of state-of-the-art noise detection algorithms for Twitter. In the system, the traditional lexicon-based sentiment analysis is enhanced by two algorithms, which are for, respectively, the disambiguation of polysemous words and the correct handling of negated sentences. The former algorithm detects the most suitable semantic variant of a polysemous word with respect of its context, by searching for the shortest path in a lexical resource from the polysemous word to its nearby words. The latter, on the other hand, detects the right scope of negation through the analysis of their parse trees. Experiments performed on four datasets show that coupling these algorithms results in a +6.7% improvement of classification accuracy in 3-class sentiment analysis.

In future work, we plan to extend the system by taking into account more social networks, including those where social content consists predominantly of images or videos, such as Youtube and Instagram. To this purpose, a future direction of research is the integration of sentiment analysis with image analysis, also in the perspective of better understanding what users' opinions are about: for instance, if a user makes a positive comment about Audi and adds a link to a picture of a specific car model, it may be possible to associate its opinion with that particular model. Multilingualism is a well-known limit in the analysis of social content, so we intend to extend our sentiment analysis technique to other languages through the use of multilingual lexical resources, such as MultiWordNet [46]. Another open issue is the sentiment analysis of idiomatic sentences, that are informal sentences that have a meaning different from the standard meaning of the words in the expression (e.g., "to cry over a spilt milk" means "to complain about a loss"). These sentences are very common in social networks, especially in Twitter, since the limits on message length force users to find shorter ways to express a certain content. To address such problem we plan to use available online dictionaries, such as Urban Dictionary,[10] to recognize idiomatic forms and replace them in the sentence with their explicit meaning. We also plan to perform a comparison between our scope modeling algorithm and others that are based on syntactic parsing, in order to determine the most accurate technique to detect the scope of negation.

References

[1] C. Diamantini, A. Mircoli, D. Potena, A negation handling technique for sentiment analysis, in: 2016 International Conference on Collaboration Technologies and Systems, CTS, 2016, pp. 188–195.
[2] C. Diamantini, D. Potena, A. Sabelli, S. Scattolini, An integrated system for social information discovery, in: 2014 International Conference on Collaboration Technologies and Systems, CTS, 2014, pp. 353–360.
[3] J. Tukey, Exploratory data analysis, reading, Addison-Wesley, MA, 1977.
[4] J.T. Behrens, C. Yu, Exploratory data analysis, Handb. Psychol. 1 (2003) 33–64.
[5] P.F. Velleman, D. Hoaglin, Exploratory data analysis, APA Handb. Res. Methods Psychol. 3 (2012) 51–70.
[6] C. Shearer, The crisp-dm model: the new blueprint for data mining, Data Wareh. 5 (2000) 13–22.
[7] A. Perer, B. Shneiderman, Integrating statistics and visualization: Case studies of gaining clarity during exploratory data analysis, in: SIGCHI Conference on Human Factors in Computing Systems, 2008, pp. 265–274.
[8] P.A. Gloor, Y. Zhao, Analyzing actors and their discussion topics by semantic social network analysis, in: Tenth International Conference on Information Visualization, 2006, pp. 130–135.
[9] B. Suh, L. Hong, P. Pirolli, E.H. Chi, Want to be retweeted? Large scale analytics on factors impacting retweet in twitter network, in: Second International Conference on Social Computing, 2010, pp. 177–184.
[10] F. Colace, L. Casaburi, M.D. Santo, L. Greco, Sentiment detection in social networks and in collaborative learning environments, Comput. Hum. Behav. 51 (2016) 1061–1067.
[11] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, R. Passonneau, Sentiment analysis of twitter data, in: ACL 2011 Workshop on Languages in Social Media, 2011, pp. 30–38.
[12] L. Oneto, F. Bisio, E. Cambria, D. Anguita, Statistical learning theory and elm for big social data analysis, IEEE Comput. Intell. Mag. 11 (2016) 45–55.

[10] http://www.urbandictionary.com/.

[13] A. Tripathy, A. Anand, S.K. Rath, Document-level sentiment classification using hybrid machine learning approach, Knowl. Inform. Syst. 53 (3) (2017) 805–831.
[14] O. Appel, F. Chiclana, J. Carter, H. Fujita, A hybrid approach to the sentiment analysis problem at the sentence level, Knowl.-Based Syst. 108 (2016) 110–124.
[15] M. Shams, A. Baraani-Dastjerdi, Enriched lda (elda): combination of latent dirichlet allocation with word co-occurrence analysis for aspect extraction, Expert Syst. Appl. 80 (2017) 136–146.
[16] S. Baccianella, A. Esuli, F. Sebastiani, Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining, LREC, 2010, pp. 2200–2204.
[17] B. Heerschop, F. Goossen, A. Hogenboom, F. Frasincar, U. Kaymak, F. de Jong, Polarity analysis of texts using discourse structure, in: 20th ACM Conference on Information and Knowledge Management, CIKM11), 2011, pp. 1061–1070.
[18] S.M. Mohammad, From once upon a time to happily ever after: Tracking emotions in mail and books, Decis. Support Syst. 53 (2012) 730–741.
[19] R. Moraes, J.F. Valiati, W. Gaviã.Neto, Document-level sentiment classification: An empirical comparison between svm and ann, Expert Syst. Appl. 40 (2013) 621–633.
[20] A. Sharma, S. Dey, Using Self-Organizing Maps for Sentiment Analysis, Cornell University Library, 2013.
[21] B. Pang, L. Lee, Opinion mining and sentiment analysis, Found. Trends Inform. Retr. 2 (1) (2008) 1–135.
[22] S. Poria, E. Cambria, A. Gelbukh, Aspect extraction for opinion mining with a deep convolutional neural network, Knowl.-Based Syst. 108 (2016) 42–49.
[23] E. Cambria, Affective computing and sentiment analysis, IEEE Intell. Syst. 31 (2016) 102–107.
[24] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, in: Conference on Empirical Methods in Natural Language Processing, EMNLP, 2013.
[25] A. Go, L. Huang, R. Bhayani, Twitter sentiment analysis, in: Final Projects from CS224N for Spring 2008/2009, The Stanford Natural Language Processing Group.
[26] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, S. Lehmann, Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm, in: 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2017, pp. 1616–1626.
[27] A. Mircoli, A. Cucchiarelli, C. Diamantini, D. Potena, Automatic emotional text annotation using facial expression analysis, in: CEUR Workshop Proceedings, vol. 1848, 2017, pp. 188–196.
[28] A. Moreo, M. Romero, J.L. Castro, J.M. Zurita, Lexicon-based comments-oriented news sentiment analyzer system, Expert Syst. Appl. 39 (2012) 9166–9180.
[29] A. Neviarouskaya, H. Prendinger, M. Ishizuka, Recognition of affect, judgment, and appreciation in text, in: 23rd International Conference on Computational Linguistic (2010), 2010, pp. 806–814.
[30] M. Taboada, J. Brooke, M. Tofiloski, K.D. Voll, M. Stede, Lexicon-based methods for sentiment analysis, Comput. Linguist. 37 (2011) 267–307.
[31] M. Lesk, Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone, in: SIGDOC 1986: Proceedings of the 5th Annual International Conference on Systems Documentation, 1986, pp. 24–26.
[32] M. Wiegand, A. Balahur, B. Roth, D. Klakow, A. Montoyo, A survey on the role of negation in sentiment analysis, in: Workshop on Negation and Speculation in Natural Language Processing, pp. 60–68.
[33] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up? Sentiment classification using machine learning techniques, in: EMNLP, 2002.
[34] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, in: HLT/EMNLP, 2005.
[35] Y. Choi, C. Cardie, Learning with compositional semantics as structural inference for subsentential sentiment analysis, in: Conference on Empirical Methods in Natural Language Processing, 2008, pp. 793–801.
[36] A. Kennedy, D. Inkpen, Sentiment classification of movie reviews using contextual valence shifters, in: Computational Intelligence, vol. 22, 2016.
[37] L. Jia, C. Yu, W. Meng, The effect of negation on sentiment analysis and retrieval effectiveness, in: CIKM, 2009, pp. 1827–1830.
[38] M.A.M. Shaikh, H. Prendinger, M. Ishizuka, Assessing sentiment of text by semantic dependency and contextual valence analysis, in: ACII, 2007, pp. 191–202.
[39] A.H. Wang, Detecting spam bots in online social networking websites: A machine learning approach, in: 24th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, 2010.
[40] A.H. Wang, Don't follow me: Spam detection in twitter, in: Proceedings of the 2010 International Conference on Security and Cryptography, SECRYPT, 2010, pp. 1–10.
[41] M. Mccord, M. Chuah, Spam detection on twitter using traditional classifiers, in: International Conference on Autonomic and Trusted Computing, Springer, 2011, pp. 175–186.
[42] E.F. Codd, A relational model of data for large shared data banks, Commun. ACM 26 (1) (1983) 64–69.
[43] M. Golfarelli, D. Maio, S. Rizzi, The dimensional fact model: A conceptual model for data warehouses, Int. J. Coop. Inf. Syst. 7 (2–3) (1998) 215–247.
[44] C. Diamantini, A. Mircoli, D. Potena, E. Storti, Semantic disambiguation in a social information discovery system, in: Collaboration Technologies and Systems (CTS), 2015, pp. 326–333.
[45] M.A. Covington, A fundamental algorithm for dependency parsing, in: 39th Annual ACM Southeast Conference, 2001, pp. 95–102.
[46] E. Pianta, L. Bentivogli, C. Girardi, Multiwordnet: Developing an aligned multilingual database, in: First International Conference on Global WordNet, 2002, pp. 293–302.

Claudia Diamantini is associate professor at the Università Politecnica delle Marche, Dipartimento di Ingegneria dell'Informazione. Previously she has been lecturer at the same University for modules in the areas of Operating Systems and Information Systems and Databases.

Her research interests include: supervised learning algorithms, data mining and knowledge discovery, data semantics and interoperability. She has been working on these topics within national and international research projects and collaborations.

Alex Mircoli received the M.Sc. degree in Computer Science Engineering from the Università Politecnica delle Marche in 2015. At present, he is a Ph.D. candidate at the Dipartimento di Ingegneria dell'Informazione, at the same university.

His research interests include knowledge discovery in databases, data mining, data warehousing, sentiment analysis and social network analysis.

Domenico Potena received the Ph.D. in Information Systems Engineering from the Università Politecnica delle Marche, Italy, in 2004. From June 2005 to October 2008, he was post-doctoral fellow at the Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione — Università Politecnica delle Marche.

At present, he is an assistant professor at the Università Politecnica delle Marche, Dipartimento di Ingegneria dell'Informazione. His research interests include knowledge discovery in databases, data mining, data warehousing, information systems and service oriented architectures.

Emanuele Storti received the Ph.D. degree in Computer Engineering from the Università Politecnica delle Marche, Italy, in 2012, and is currently a post-doctoral fellow at the Dipartimento di Ingegneria dell'Informazione, at the same university.

His research interests include Semantic Technologies, Knowledge Management, Open Data, Business Intelligence. knowledge as a support for interoperability in distributed and collaborative eco-systems of organizations.