,

UNIVERSITÀ POLITECNICA DELLE MARCHE

DOCTORAL SCHOOL ON INFORMATION ENGINEERING
CURRICULUM "INGEGNERIA INFORMATICA, GESTIONALE E DELL'AUTOMAZIONE"

Lexicon- and learning-based techniques for emotion recognition in social contents

---

Author:
Alex MIRCOLI

Supervisors:
Dr. Domenico POTENA
Prof.ssa Claudia DIAMANTINI

"The more original a discovery, the more obvious it seems afterwards."

Arthur Koestler

Abstract

In recent years, the massive diffusion of social networks has made available large amounts of user-generated content, which often contains authentic information about people's emotions and thoughts. The analysis of such content through emotion recognition provides valuable insights into people's feeling about products, services and events, and allows to extend traditional processes of Business Intelligence. To this purpose, in the present work we propose novel techniques for lexicon- and learning-based emotion recognition, in particular for the analysis of social content. For what concerns lexicon-based approaches, the present work extends traditional techniques by introducing two algorithms for the disambiguation of polysemous words and the correct analysis 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 detects the right scope of negation through the analysis of parse trees.

Moreover, the paper describes the design and implementation of an application of the lexicon-based approach, that is a full-fledged platform for information discovery from multiple social networks, which allows for the analysis of users' opinions and characteristics and is based on Exploratory Data Analysis.

For what concerns learning-based approaches, a methodology has been defined for the automatic creation of annotated corpora through the analysis of facial expressions in subtitled videos. The methodology is composed of several video preprocessing techniques, with the purpose of filtering out irrelevant frames, and a facial expression classifier, which can be implemented using two different

approaches.
The proposed techniques have been experimentally evaluated using several real-world datasets and the results are promising.

Abstract

In tempi recenti, la diffusione massiva dei social network ha reso disponibili grandi quantità di contenuti generati dagli utenti, i quali spesso contengono informazioni autentiche in merito alle emozioni ed ai pensieri delle persone. L'analisi di tali contenuti attraverso tecniche di emotion recognition offre informazioni preziose in merito alla percezione di prodotti, servizi ed eventi, permettendo di estendere i tradizionali processi di Business Intelligence. A tal fine, nella presente tesi sono proposte tecniche innovative, basate sia sull'uso di risorse lessicali (lexicon-based) che di algoritmi di machine learning (learning-based), per l'emotion recognition, in particolare con applicazioni a contenuti social. Per quanto riguarda gli approcci lexicon-based, vengono estese le tecniche classiche introducendo due algoritmi, rispettivamente per la disambiguazione delle parole polisemiche e l'analisi delle frasi contenenti negazioni. Il primo algoritmo individua la variante semantica di una parola polisemica più adatta al contesto cercando il percorso più breve, all'interno di una risorsa lessicale, fra la parola polisemica e le parole vicine. Il secondo, invece, individua lo scope della negazione mediante analisi dell'albero sintattico.

La tesi presenta inoltre la progettazione e l'implementazione di una piattaforma basata su approcci lexicon-based per l'analisi delle opinioni espresse dagli utenti in vari social network. Per quanto concerne gli approcci learning-based, è stata definita una metodologia per la creazione autoamtica di corpora annotati attraverso l'analisi delle espressioni facciali in video sottotitolati. La metodologia propone l'utilizzo di numerose tecniche di video preprocessing, per il filtraggio dei frame non rilevanti, e di un classificatore di espressioni facciali, implementabile mediante due approcci differenti.

Le tecniche proposte sono state valutate sperimentalmente attraverso numerosi dataset e i risultati sono promettenti.

Contents

Chapter 1

Introduction

1.1 Overview

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. Such information is considered authentic, as in social networks people usually feel free to express their thoughts. Therefore, the analysis of this user-generated content provides valuable information on how a certain topic or product is perceived by users, allowing firms to address typical marketing problems as, for instance, the evaluation of customer satisfaction or the measurement of the appreciation of a new marketing campaign. Moreover, the analysis of customers' opinions and feelings about a certain product helps business owners to find out possible issues and may suggest new interesting features, thus representing a valid tool for open innovation.

However, due to the speed at which social content is created and modified, in order to be valuable this information needs to be processed within a short time after its creation. Given the large amount of available content, there is the need for algorithms for the analysis of user's thoughts.
For this reason, in the last years many researchers [1] focused on techniques for the automatic analysis of writer's opinions and emotions, generally referred to as techniques for affective computing. Opinions and emotions expressed in a text are highly correlated (as we will show in Chapter 5) and they can automatically evaluated through, respectively, sentiment analysis and emotion recognition algorithms. Due to the above-mentioned correlation, sentiment analysis can be informally considered as a "coarse-grained" version of emotion recognition.
More rigorously, *sentiment analysis* is the automatic analysis of a writer's attitude with respect to some topic [2]; 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.
Emotion recognition is defined as the computational treatment of the emotional aspects of text [3]. Its goal is the classification of the typology and intensity of emotions expressed by the writer.
Due to the intrinsic complexity of the human language, these tasks offer 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 content, especially in microblogging platforms, since the constraint on message length

forces users to express themselves in a more creative (and less intelligible) way. For this reason, in the last years a multitude of different algorithms has been proposed in literature for the affective analysis of social content. These algorithms can be usually reconducted to two main approaches, which respectively rely on machine learning (learning-based techniques) [4] and annotated lexical resources (lexicon-based techniques) [5]. Learning-based techniques usually reach high accuracy but they need considerable amounts of annotated training data. Moreover, they are very domain-specific, so the creation of a new annotated dataset is required whenever the model needs to be retrained for a different domain. On the other hand, the lexicon-based approach requires the availability of large corpora.
At the moment, few well-established open-access corpora exist and they are mainly available for sentiment analysis (e.g., SentiWordNet [6]). In fact, emotions are considered by few corpora (e.g., [7]) and typically for small sets of words. For this reason, we will focus on lexicon-based algorithms for sentiment analysis. Nevertheless, the lack of lexical resources represents a major limitation for the research in the field of emotion recognition. Therefore, there is a need for techniques to automatically (and objectively) annotate emotions in text, in order to dynamically create language- and domain-specific corpora for emotion recognition.

1.2 Contribution of this work

The present work aims at proposing novel algorithms and methodologies for sentiment analysis and emotion recognition in social content, with the purpose of overcoming well-known limitations in these fields. More in detail, our main contributions are:

- a context-aware methodology for lexicon-based sentiment analysis, which may increase the accuracy of traditional approaches

based on lexical resources by taking into account information about the contextual meaning of words and the effects of negations. In particular, we propose two algorithms for, respectively, word-sense disambiguation and negation handling.

- the definition of a design methodology for Social Information Discovery systems that is enriched by lexicon-based techniques for text analytics, in order to allow for semantic analysis of social content.

- a novel approach for the automatic building of corpora for emotion recognition (that we have already presented in [8]) that relies on the analysis of subtitled videos and, in particular, is based on: (i) the analysis of the emotional content of the video through facial expression analysis algorithms and (ii) the subsequent text annotation through correlation between facial expressions and subtitles.

The rest of the thesis is organized as follows: in Chapter 2 we present some related work on both lexicon- and learning-based sentiment analysis and emotion recognition, as well as Exploratory Data Analysis and noise detection in social networks. Chapter 3 describes the proposed context-aware technique for lexicon-based affective computing and provides a thorough experimental evaluation of the technique. In Chapter 4 we present the design and implementation of a Social Information Discovery platform extended with text analysis capabilities, as an application of lexicon-based techniques to the analysis of social content. Chapter 5 is devoted to the description of the methodology for the automatic creation of emotionally-annotated corpora on the basis of the analysis of facial expressions in subtitled videos. Finally, Chapter 6 draws conclusions and discusses future work, while Appendix A outlines further Ph.D. activities related to the physical database optimization in presence of high and highly-variable volume of data.

Chapter 2

Related Work

This Section outlines some of the most relevant contributions related to the present work that can be found in recent literature. They range from sentiment analysis and emotion recognition to noise detection and flexible data analytics. The main focus is on works that are related to the analysis of social content.

2.1 Sentiment analysis

In recent literature, much work has been written on sentiment analysis of user-generated content [9], including reviews and social discussions ([10], [11]). Sentiment analysis can be performed at different levels: at document level [12], at sentence level [13] and at aspect level [14]. In our work we only focus on the analysis of sentiment at sentence level, since social contents usually consist of a single sentence, especially in case of microblogging platforms like Twitter. Unlike reviews, which are usually long and can be used to discuss several aspects of a topic (e.g., in case of a smartphone: price, design, performance, battery duration and so on), a social content is generally in the form of 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 ([6], [15], [16]) and machine learning ([17], [18])

approaches. The former involves the use of lexical resources (e.g., SentiWordNet [6]), while the latter use statistical models trained on human annotated datasets. An extensive survey on the existing sentiment analysis techniques can be found in [19]. 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 [20], [21]). In [22] 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 [23], 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. [24] 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 [8], 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. Hence, several authors (e.g., [25], [26]) proposed to build domain-independent sentiment classifiers using lexicon-based approaches. A lexicon-based classifier for sentiment analysis is proposed in [27], 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. Our approach differs from this work in that we also consider polysemous words (i.e., words whose meaning varies according to context) and hence we introduce a context-based word-sense disambiguation algorithm (see Sect. 3.2). Our disambiguation algorithm has some similarities with [28], in that they both disambiguate a word on the basis of the glosses of nearby words. Nevertheless, our approach differs from [28] in the way the gloss (i.e., a short sentence describing the sense of the term) is used for the disambiguation: while in [28] it is proposed to evaluate the overlaps between the glosses, our algorithm searches the contextual words in the polysemous word's glosses.

Traditional supervised techniques are based on the bag-of-words (BOW) approach, that is a vector representation of words' frequency in the sentence that does not take into account any information about 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 [29]. In [4] 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. [30] 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 [4], as it also considers other polarity shifters, such as intensifiers (e.g., very)

and diminishers (e.g., barely). Choi and Cardie [31] 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 dynamically detect the scope of negation. A first attempt to dynamically model the effects of negation, which has been performed by taking into account grammatical dependencies among words, can be found in [32], where the authors use a parser 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. Moreover, they model the impact of negation words in a different way from us. Jia et al. [33] determine the scope of negation by considering static (e.g., because) and dynamic (e.g., for) clause delimiters and heuristic rules, while [34] 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.

2.2 Emotion Recognition

The theory of archetypal emotions proposed by the psychologist Paul Ekman [35] is one of the first attempts to systematically describe and analyze facial expressions. It defines a framework of six universal basic emotions, from which every emotion is derived through linear composition: anger, disgust, fear, happiness, sadness and surprise. Such theory is based on the Facial Action Coding System (FACS), which is an anatomical system for describing any observable facial expression. As stated in [8], the use of FACS offers many advantages, since they are language- and domain-independent and hence facial expression analysis can be carried out for every language and scope of application. Ekman's theory has been widely

used in image-based facial expression recognition (e.g., [36], [37]), even if many works only consider a subset of universal emotions [38]. A comprehensive survey on facial expression recognition can be found in [39].

Only few authors have focused on the creation of annotated datasets for facial expressions recognition. Among them, Sun et al. [40] used an hidden camera to build a database of authentic facial expressions where the test subjects are showing the natural facial expressions based upon their emotional state. They also use such database to evaluate the performance of several machine learning techniques in facial expression recognition. However, each video in their dataset has a unique emotion by design and hence their approach is not suitable for real-world videos, where people express a large variety of emotions; moreover, the released dataset is relatively small. An attempt to solve the problem of scarcity of annotated data is presented in [41], where the authors use a computer graphics program to synthetize more than 100,000 faces expressing emotions.

Mo et al. [42] propose a set of features for emotion recognition in videos that are based on the Hilbert-Huang Transform (HHT), which is able to capture the time-varying characteristics of visual signals. A different approach is used in [43], where, instead of manually engineering facial features, the authors combine Linear Regression Classification model (LRC) and Principal Component Analysis (PCA) in order to automatically select features.
Cohen et al. [44] propose Hidden Markov Models (HMMs) for automatically segmenting and recognizing facial expressions from video sequences. Our video preprocessing phase is similar to [45] in the use of face detection but we also consider some other aspects, such as face orientation. Poria et al. [38] propose to perform facial expression recognition through Open-face and SVM classifiers; however, there are many differences between their work and ours, including

that we use both Action Units and point distances as features and we evaluate the technique on a real-world video dataset.

2.3 Exploratory Data Analysis

The Exploratory Data Analysis (EDA) has been introduced in [46] 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 ([47], [48]). 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 [49]. In recent years, EDA has evolved in order to face the challenges posed by Big Data sources, such as social networks (e.g. [50], [51]). In [50] advanced methods and tools for graph visualization are introduced with the purpose of studying interactions among the networks' users. An approach similar to the one presented in this Thesis is adopted in [52], where EDA techniques are used to analyze tweets.

2.4 Noise detection in social networks

A general definition of noise in social networks is not limited to spam, but it also includes all the contents that would not be useful for the analysis, such as Facebook posts or tweets that only contain 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 [53] a machine learning approach is proposed to detect spam bots in social networks. Wang [54] 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, but we define other textual features that are suitable for noise detection. An approach similar to [54] is used in [55] but experimental results are different, since in [55] the Naïve Bayes classifier shows poor performance in comparison to other classical machine-learning algorithms, such as Support Vector Machines and Random Forests.

Chapter 3

Lexicon-based techniques

In this Chapter we present a methodology for affective computing that is based on lexical resources[56]. Since classical lexicon-based techniques usually have the limitation of not taking into account contextual information, we extend traditional approaches by also introducing two algorithms for word-sense disambiguation and negation handling. Furthermore, we evaluate the impact of both algorithms (both in single and coupled configuration) on classification accuracy. As already discussed in Chapter 1, due to the lack of lexical resources for emotion recognition, lexicon-based algorithms have been developed only for sentiment analysis.

3.1 The Sentiment Analysis Process

Lexicon-based approaches involve calculating sentence polarity by means of annotated dictionaries or corpora, generally referred to as *lexical resources*, in which there is a correspondence between sentence elements and sentiment scores (or sentiment classes). A sentence element is usually an n-gram, that is a sequence of n contiguous words. The majority of available lexical resources for sentiment analysis provides annotations for 1-grams (i.e., single words), even if there have been some attempts to build corpora for 2-grams and 3-grams. In lexicon-based approaches each n-gram is associated with

the relative sentiment score/class and the overall sentence polarity is calculated, on the basis of these scores/classes, through an aggregate function (typically the average() function). As a consequence, such approaches have two major limitations:

- their accuracy strictly depends on the size (i.e., number of words) of the lexical resource and the quality of their annotation.
- the effects of semantic compositionality are ignored. The principle of compositionality, also known as Frege's principle, states that the meaning of a sentence is determined not only by the meanings of its words, but also by the rules used to combine them. Traditional lexicon-based approaches do not consider such rule and hence are unable to capture variations of meaning that derive from composition. In other words, they are context-unaware, i.e., they do not take into account the context of words.

The purpose of this Chapter is to illustrate and evaluate two context-based algorithms with the goal of developing a context-aware algorithm for lexicon-based sentiment analysis. The lexical resource chosen in this work is SentiWordNet 3.0 [6], that is an extended version of the WordNet ontology, where words are annotated with a sentiment score between -1 (max negativity) and 1 (max positivity). 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.

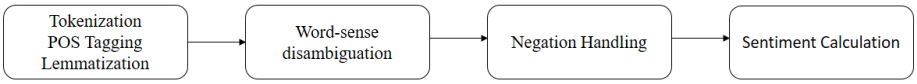

FIGURE 3.1: The sentiment analysis process

The whole sentiment analysis process includes several steps, as depicted in Figure 3.1.

Tokenization is the first phase of the process: input text, extracted from social networks, is split into words. After that, each word is tagged using a part-of-speech (POS) tagger: this operation allows to identify their lexical category. Through POS tagging we are able to filter out words that are not relevant for sentiment analysis, as they are not sentiment-bearing, i.e. do not express opinions on the analyzed subject. In particular, we only maintain the four lexical categories that are more relevant for sentiment analysis: nouns, verbs, adverbs and adjectives.

POS tagging is available as a predefined function in most NLP libraries, but different tools often adopt different conventions for what concerns the definition of lexical categories. For instance, we performed POS tagging through the open source library *Stanford Log-linear Part-Of-Speech Tagger*, described in [57]. In such library, tags are expressed in Penn standard[1], therefore we needed to convert them to SentiWordNet tagging system: Table 3.1 shows the adopted mapping. Some Penn tags are omitted, since they are not related to any SentiWordNet tag.

The next step is the lemmatization, that is the process of determining the canonical form (lemma) of a given inflected word (e.g. the lemma for “plays”, “played”, “playing” is “play”). This step

[1]https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

TABLE 3.1: Mapping between Penn and SentiWordNet POS tags.

POS	PENN tags	SentiWordNet tags
adjective	JJR, JJS	a
noun	NN, NNS, NNP, NNPS	n
adverb	RB, RBR, RBS	r
verb	VB, VBD, VBG, VBN, VBP, VBZ	v

is necessary since terms appear in lexical resources in their canonical forms. The order between part-of-speech tagging and lemmatization is important: as a matter of facts, suffixes provide valuable information for POS classification. For instance, the suffix -ed suggests the presence of a verb. Since lemmatization removes suffixes, by putting POS tagging before lemmatization it is possible to increase the performance of the POS tagger.

In addition to the above-mentioned preprocessing tasks, we introduce a word sense disambiguation algorithm for polysemous words (Section 3.2) and a negation handling algorithm for negative sentences (Section 3.3). Finally, we calculate the sentiment score of the post/tweet as the mean value of its words' sentiment scores.

3.2 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). 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

([58]) that is based on the analysis of nearby words. 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 $< g_1, g_2, \ldots, g_n >$ such that g_j (with $j > i$) is the gloss of $w_j \in g_i$, g_1 is one of w_a's glosses 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 w1 by searching, in the w1's glosses, for an occurrence of the word w2 directly before or after w1. Both the previous and following words give information about the contextual meaning of w1. We first select the word following w1 as w2; if w2 is not useful to disambiguate w1, i.e., w2 is not in SentiWordNet or w2 is not in the glosses of w1, then we assign to w2 the word preceding w1. If both previous and following words are not useful to disambiguate w1, then the sentiment score of w1 is calculated as the mean value of the sentiment scores of its semantic variants. In order to increase the chance for the selected w2 to be useful for the disambiguation of w1, we added a preprocessing phase of POS filtering with the purpose of deleting from the sentence all words that are not names, pronouns, adjectives or verbs. If a matching between w1 and the chosen w2 exists, the algorithm stops and selects the semantic variant having the matching gloss as the most suitable meaning for w1. Otherwise, the search is extended to every definition of each word included in w1's glosses, then to each definition of each word of those definitions and so forth, up to a maximum search depth. Setting a threshold on maximum search depth is motivated by the fact that words are correlated to w1 only

up to a certain depth.
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 w1. 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.

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 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(w1, w2, depth)
2:      Q.push(w1)
3:      while Q not empty && depth > =1 do
4:          t ← Q.pop()
5:          for each x in S(t) do
6:              for each j in G(x) do
7:                  if w2 == j then
8:                      return sent(anc(j))
9:                  Q.push(j)
10:             depth ← (depth − 1)
11:     return null
```

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 it 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 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 Figure 3.2 (due to the high branching factor of the resultant search tree, some irrelevant definitions are not displayed). In the Figure, boxes represent terms

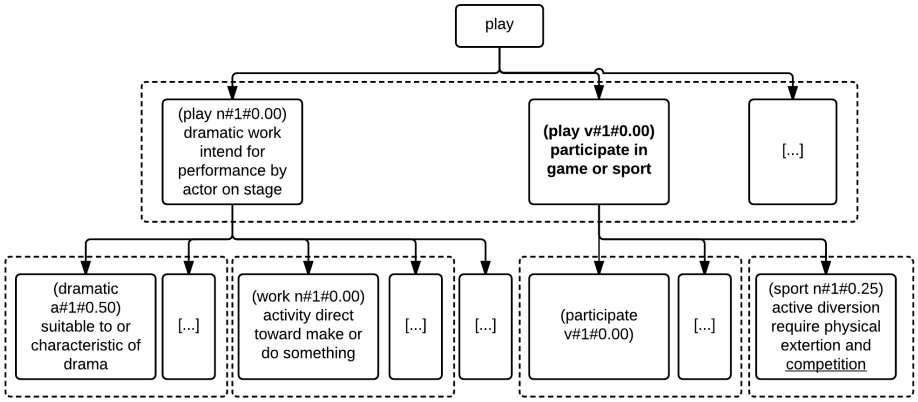

FIGURE 3.2: Example of the WSD algorithm

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 variant, and δ is the threshold on the search depth. In real world applications it could potentially lead to execution times that are unacceptable

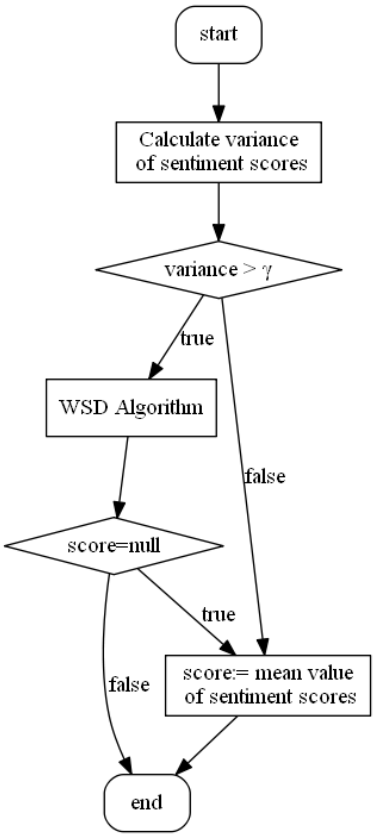

FIGURE 3.3: Workflow of the *word sense disambiguation algorithm*

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 Figure 3.3, where the variability of the sentiment scores of a polysemous term is measured in terms of variance and γ represents a chosen threshold. A potential issue of the algorithm stops when it finds the first occurrence of w_1 in a gloss of w_2. However, we can hypothesize situations in which, at the same tree depth, there is more than a gloss containing w_1: in such situations, all the semantic variants, except for the first in terms of

appearance, are ignored. Therefore, as a future work, we plan to evaluate the number (and the frequency) of overlaps among different glosses of the same term in SentiWordNet. If the frequency will result significant, we will define a strategy to handle this particular situations.

3.3 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 [59] parses the sentence using a statistical dependency parser [60], 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 $children(n)$ return the children nodes of node n,
let $isLeaf(node)$ return true if *node* is a leaf
let $isNegation(t)$ return 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) ← 0
5:             for each n in rLeaves(node) do
6:                 score(n) ← 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 words in the first clause. The corresponding dependency-based parse tree (Figure 3.4(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 DFS strategy and finds the negation word "not", so it explores the subtree of its right sibling node (Figure 3.4(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

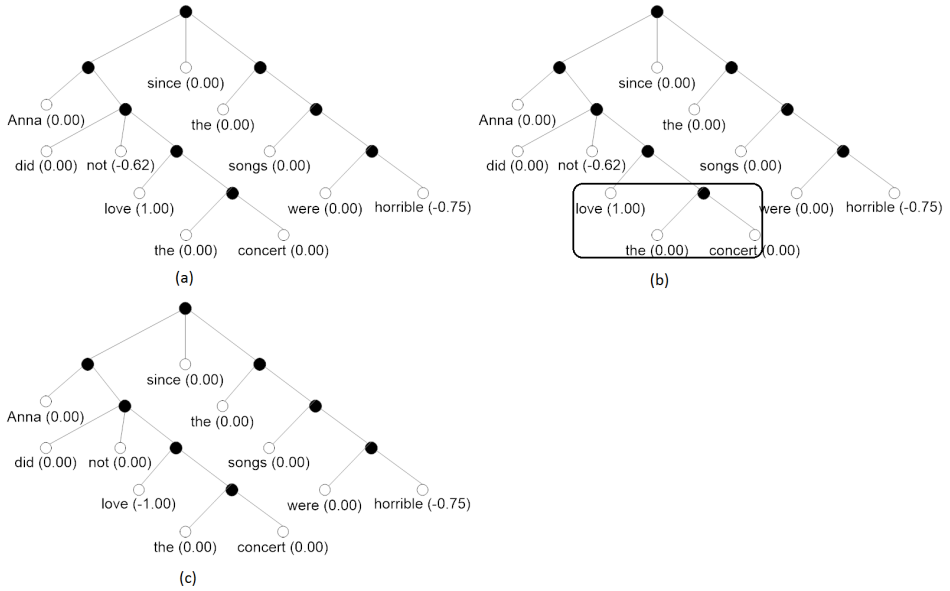

FIGURE 3.4: Example of the *negation handling* technique: (a) the algorithm builds the parse tree and (b) searches for words affected by negation, then (c) their sentiment scores are inverted and the sentiment score of the negation word is set to 0.

negation word "not" is set to 0 (Figure 3.4(c)). The second clause, instead, is correctly left unchanged and the negative score of the word "horrible" is preserved.

3.4 Evaluation

3.4.1 Results

In order to show the effectiveness of the WSD and NH algorithms, we performed some experiments using five different datasets. Since our goal is to test the performance of the sentiment analysis algorithm in analyzing social content, we used only datasets composed 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 labelled 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. Moreover, this dataset is useful to test the system as a general purpose sentiment classifier. In the rest of the thesis we refer to this corpus as Multiset.
The third dataset is part of the Stanford Sentiment Treebank (SST) and is described in [22]. 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[2] (Task 4). We were unable to download the entire dataset because some tweets were deleted (or not available,

[2]http://alt.qcri.org/semeval2016/task4/

due to modified authorization status) and hence our final dataset consists of only 1664 tweets.
The last dataset[3] consists of 5000 tweets (2657 negative and 2343 positive tweets) that have been randomly selected from the "Twitter Sentiment Analysis Dataset"[4] (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 3.2.

TABLE 3.2: 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**

We performed six experiments on each dataset: first, we used a traditional lexicon-based approach (LBA), i.e. we simply calculated the sentiment score of a sentence as the mean value of the sentiment scores of its words. In case of polysemous words, we computed their sentiment score as the mean value of the scores of their semantic variants. Then we separately used the word-sense disambiguation (WSD) and negation handling (NH) algorithms, and, finally, 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

[3] http://kdmg.dii.univpm.it/?q=content/CTS16_SI_dataset
[4] http://thinknook.com/wp-content/uploads/2012/09/Sentiment-Analysis-Dataset.zip

maximum search depth to 2. In general, adding algorithms (WSD, NH or both) to the sentiment analysis pipeline has the effect of improving classification accuracy, while decreasing the speed of analysis.

A further experiment has been performed, where Subjectivity Lexicon[5] has been used instead of SentiWordNet. 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. Then, we calcuted the sentiment score of a sentence as the mean value of the sentiment scores of its words and we converted these scores into classes by setting proper thresholds (i.e., $score < -0.1 \rightarrow negative$; $-0.1 \leq score \leq 0.1 \rightarrow neutral$; $score > 0.1 \rightarrow positive$).

Finally, we compared our sentiment analysis pipeline with a learning-based approach, namely Stanford CoreNLP [22]. 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 [22]) and we analyzed how it performs on the other datasets. The results are shown in Table 3.3.

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

[5]http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

TABLE 3.3: 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%**

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 3.3). 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 define an algorithm suitable for near-real-time analysis of social contents, 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.

3.4.2 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.

We performed a set of experiments on a laptop with Intel i7 3632QM (2.20 GHz) and 8 GB RAM and we measured the execution time of the WSD phase. As depicted in Figure 3.5 the execution time has been drastically reduced. In the Figure, on the x-axis are reported different values for the variance threshold, while on the y-axis are reported 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. The effects of the optimization 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 optimization results in a 75.8% reduction of the execution time. For instance, the time needed to analyze a test set composed of 100 tweets reduced from 1475 to 357 seconds, which means an average execution time (for the entire sentiment analysis pipeline) of 3.57 seconds per tweet. Without using the word-sense disambiguation the average execution time is 1.12 seconds 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 optimization has 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.

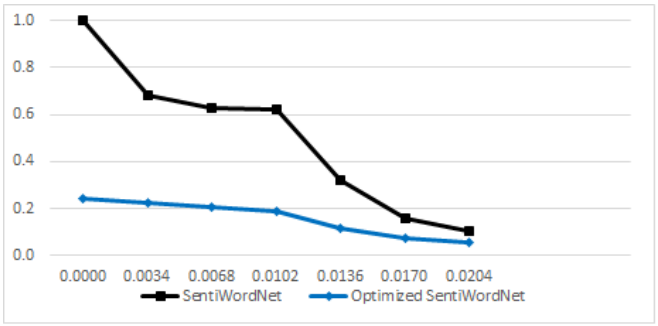

FIGURE 3.5: Effects of the optimization of the WSD algorithm on the total execution time of the sentiment analysis process.

Chapter 4

Social Information Discovery

This Chapter is devoted to presenting an application of lexicon-based techniques to the analysis of text extracted from multiple social networks, namely a Social Information Discovery system.
Lexicon-based techniques are considered more suitable for the analysis of social content than learning-based approaches, since they usually generalize better and hence they can deal with the variety of topics discussed in social networks.
The purpose of this Chapter is to show how the analytic power of the system can be extended by implementing such techniques, which enable for the analysis of users' thoughts. We introduce the design methodology, also discussing the major challenges that have to be faced when analyzing social networks, and we present a prototype of the system that has been developed using the Oracle Endeca Information Discovery framework. The choice of focusing on sentiment analysis is imposed by to the lack of lexical resources for emotion recognition.

4.1 Overview

A Social Information Discovery system is a platform that enables the simultaneous analysis of multiple social networks in an integrated scenario. Such 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. To overcome these limitations and also exploit the great amount of text data available on social networks, a social information discovery system should implement techniques for versatile data analytics and offer tools for the evaluation of the semantic aspect of texts. To this purpose, in our system we 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.
As already discussed in Section 3, 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 these reasons, 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.

4.2 Design Methodology

Social network data adhere to the 5V model (i.e., Volume, Velocity, Variety, Veracity and Value) of Big Data, since they are characterized by big volume, high rate of growth and a variety of (mainly

unstructured) types, and their analysis usually gives valuable insights about the domain of interest. As a consequence, an effective and efficient Social Information Discovery project must be based on a fast and scalable system that allows for quick and flexible 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 defined a 4-step methodology that focuses on the critical aspects 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 is described in Section 4.3.

4.2.1 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 enough to expand as soon as new demands of analysis are made. This means to adopt a 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, add 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 ([61]), 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 [62], 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 (e.g., data from a new social network) 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 analysing 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.

[1] https://en.wikipedia.org/wiki/ACID

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.

4.2.2 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 volume and high variability (i.e., large amounts of data are created and/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 not to 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:

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

[2]https://en.wikipedia.org/wiki/NoSQL

- 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 processes, degree of parallelism of ETL activities, and so forth). Note that poor system performance may result in a loss of trust by analysts.

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 [54], 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 product names or brands, 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.

4.2.3 Text Analysis

Textual data are the main source of data in a large part of social networks. Although users also share images and videos, text messages are considered to be more content-rich and, hence, more useful for the analysis of what happens in the network. Messages often contain opinions and feelings about a specific topic; that information is considered authentic, since in social networks people usually feel free to express their thoughts, and hence its analysis is valuable. In our system we identify two main text analysis tasks:

- *entity extraction*, that is the extraction of people, brands, products, places, hashtags and all other pieces of information that may be relevant for the analysis

- *sentiment analysis*, that is the determination of the user's opinion about a particular topic

Multilingualism is an important aspect to take into account when analyzing social text data, 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 the majority of academic and commercial tools is 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. Moreover, it should be noted that 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.

4.2.4 User Interface Design

The user interface should allow for a fast and flexible analysis, it has to be highly interactive and it must enable users to explore data and to ask questions to the system. For these motivations, we design the user interface on the basis of the principles the Exploratory Data Analysis (EDA) paradigm. Our 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.

4.3 Implementation

4.3.1 Data definition

The first step is the definition and analysis of data sources. We have selected three of the most widespread social networks, i.e. Facebook, LinkedIn and Twitter, and we have defined what set 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] to extract 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.

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

For what concerns LinkedIn, we used the LinkedIn API[5] to extract discussions from different groups. Unlike other social networks, as Facebook and Twitter, these API have several limitations in the quantity and typology of available information:

- the only available information are the discussions of groups where the developer account is member. Therefore, additional work is required to select and join relevant groups for the application domain.
- API does not allow performing a keyword-based search. Hence, the full set of discussions have to be extracted. API only offers the possibility to extract discussions since a defined date or to retrieve the most recent/popular content. This requires the presence of a filter after the extraction process.

These limitations does not make LinkedIn the best platform for social information discovery processes, for instance if compared to Twitter. In particular, the first constraint forces the developer to make wide a priori analysis of the existing LinkedIn groups. Moreover, the latter constraint could affect the accuracy of the analysis, as all discussions are collected although they are not about the topic under analysis. Nevertheless, the analysis of LinkedIn discussions is still interesting, since focusing only on a restricted set of groups allows us to a-priori reduce noise sources and to filter out the typical noise of Big Data.

For what concerns the data model, we have chosen the *data lake* because 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 Figure 4.1: each row represents a data source (e.g., Facebook posts, Facebook comments and tweets) and each column is an attribute.

[5]https://developer.linkedin.com/docs/rest-api

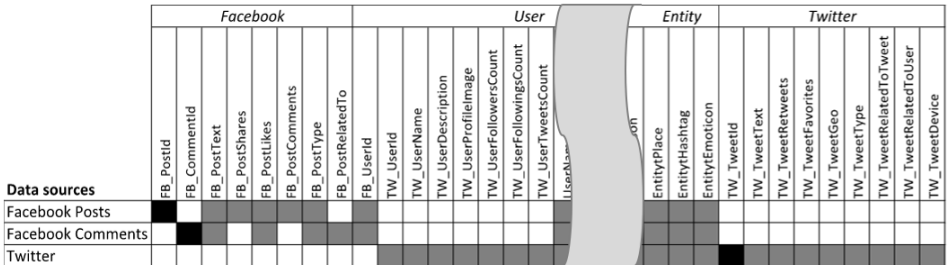

FIGURE 4.1: An excerpt of our data model.

If the color of a cell is black it means that the attribute is a primary key for that source, while a grey 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 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 them to our data model. Then we added attributes from another query and so on. In this way, at each step we only have to reconcile 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. In order to ensure an efficient representation of extracted data, we have chosen Oracle Endeca Information Discovery[6], because it is based on a column-oriented DBMS, which offers better performance when querying for aggregate values, and uses in-memory computing to further reduce the response time.

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

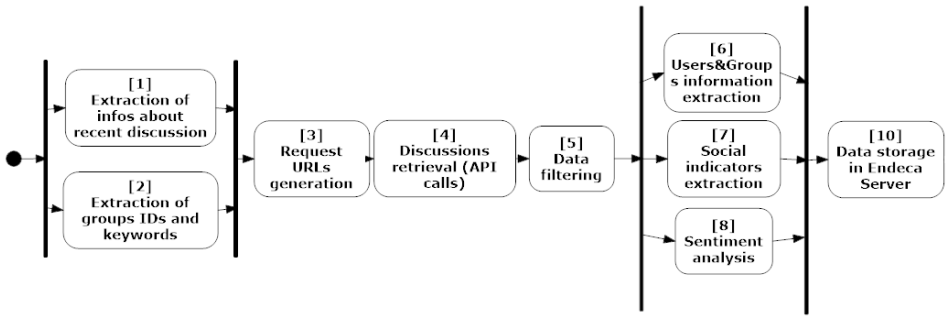

FIGURE 4.2: ETL process for LinkedIn discussions.

4.3.2 ETL design

The ETL processes are developed using Oracle EID Integrator Designer, which is the ETL tool in the Endeca Information Discovery suite. These processes allow for the automatic retrieval, transformation and integration of social data. As an example, let us consider the ETL process to collect LinkedIn discussions (see Figure 4.2). 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.
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 minutes 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.

Evaluation of Noise Detection

In order to evaluate the effectiveness of the proposed technique for noise detection, we trained 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 constrast with [55], 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

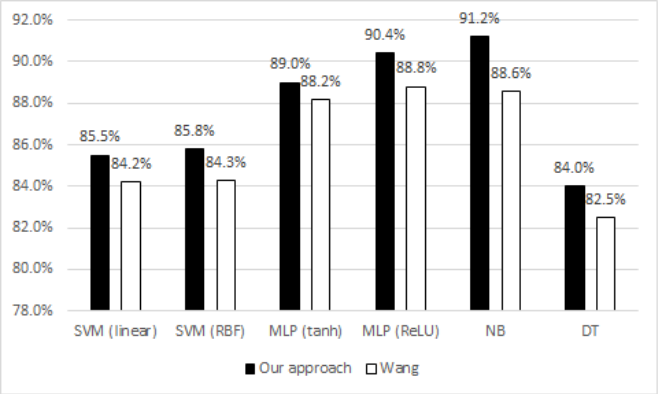

FIGURE 4.3: Comparison between our approach and the spam detection algorithm proposed by Wang. Experiments are conducted on several machine learning algorithms. The classification accuracy is reported in the y-axis.

Wang [54], we trained and tested the abovementioned classifiers on our dataset, using both the features used in [54] 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 Figure 4.3. When using our features we reached higher accuracy than [54] on every machine learning algorithm that we considered, with a remarkable +2.6% improvement on classification accuracy when using the NB 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.

4.3.3 Text Analysis

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 speed and generalizability of analysis. In particular, we implemented the sentiment analysis workflow that has been described in Chapter 3. At the moment the focus is on English messages, due to the scarcity of well-established lexical resources for other languages.

4.3.4 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 (Figure 4.4) 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 (Figure 4.5). 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 Figure 4.5 are the result of searching for the keyword

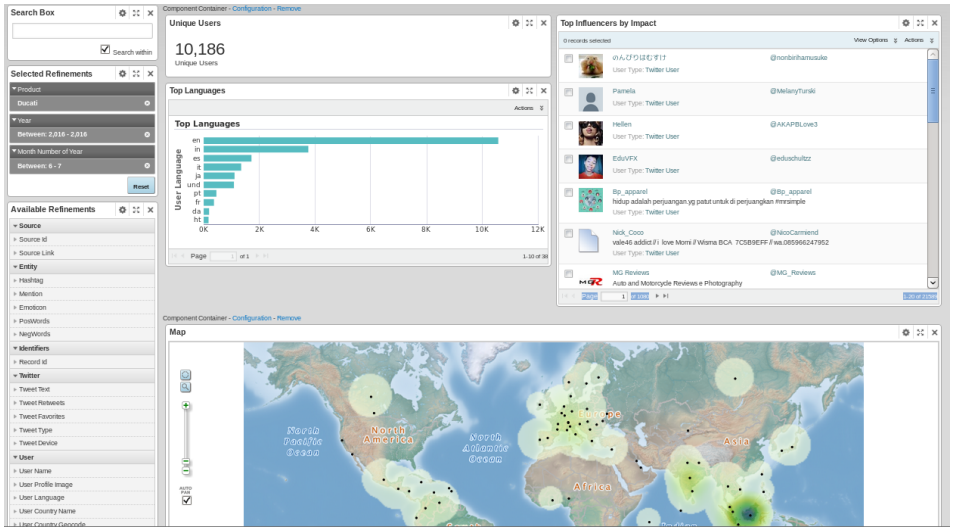

FIGURE 4.4: An example of Users perspective: users can be analyzed with respect to their geographical distribution and language. Users with the highest social influence (influencers) are reported on the right. A set of filters (left sidebar) allow for a dynamic and highly customizable analysis.

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.

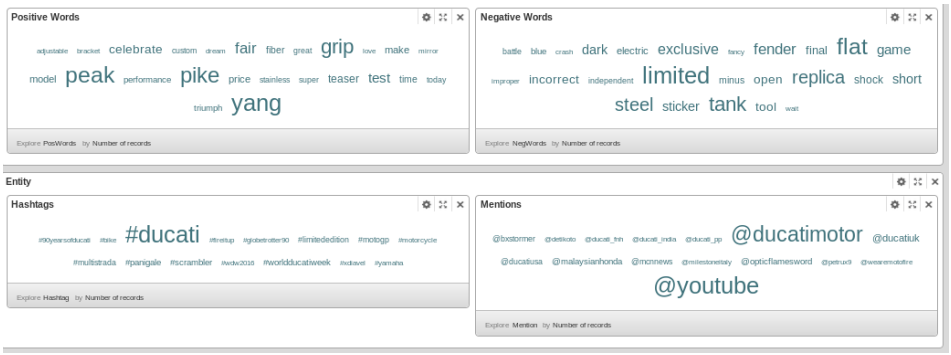

FIGURE 4.5: An example of word clouds. From top left: most frequent terms appearing in positive and negative tweets; most frequent hashtags; most frequent mentions.

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 crowdsourcing 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 usability test but we plan to do it as future work. However, 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.

Chapter 5

Learning-based techniques for emotion recognition

In this Chapter we define a methodology for the automatic annotation of subtitles on the basis of the analysis of facial expressions in videos. The analysis of facial expressions is performed through machine learning algorithms, on the basis of a set of pre-engineered facial features that are extracted from video frames. The purpose of the proposed technique is the creation of annotated corpora and datasets, that may enhance the performance of both lexicon- and learning-based algorithms for emotion recognition.

5.1 Overview

Regardless of the preferred approach, the accuracy of algorithms for emotion recognition in texts strongly relies on the existence of large datasets with high-quality annotations. At the moment, the lack of textual resources in literature represents a major limitation and, hence, the use of algorithms for emotion recognition in texts is limited to those specific applications where it is possible to create a small corpus/training set for a restricted domain. For this reason, there is the need for techniques that may allow for the automatic

annotation of texts, on the basis of clues about their emotional content that could be extracted, for instance, from multimedia content. To this purpose, starting from the assumption that facial expressions and speech are usually coherent, as demonstrated by various experiments [63], we propose to exploit the information given by speaker's facial expressions in videos to annotate subtitles. Even if emotion recognition in videos can be performed in several different ways (e.g., by analyzing speech, voice intensity and modulation and so on), the approach proposed in this work is based on facial expression analysis because of empirical evidences that show that the greatest part of the emotional content in a conversation is represented by nonverbal and paraverbal communication. Hence, it seems legitimate to design a technique that takes into account the positions and movements of facial muscles in order to extract useful information on the expressed emotions. To this purpose, we propose a novel methodology for the automatic creation of emotionally annotated corpora through the above-mentioned analysis of speakers' facial expressions in subtitled videos. These corpora will enable the analysis of the emotional aspects of user-generated content through both lexicon- and learning-based approaches, respectively allowing for the automatic creation of lexical resources and training sets. Due to the popularity of video-sharing platforms, such as YouTube, a multitude of subtitled videos has been made publicly available: as a consequence, an automatic annotation technique will allow for a massive annotation of text data.

In the present work, we model human emotions according to Ekman's theory [35], that states the existence of six archetypal emotions: anger, disgust, fear, happiness, sadness, surprise. Text annotation by means of the analysis of these basic facial expressions offers several advantages:

- every emotion can be viewed as a combination of the six basic emotions and hence annotating with respect to basic emotions

allows for the representation of every human emotion.

- the technique is language- and domain-independent, so emotional annotations can be carried out for every language and scope of application.
- facial expressions are demonstrated to be universal [64], that is unrelated to speaker's personal characteristics (e.g., sex, ethnicity).

As a consequence, the proposed methodology can be used for the annotation of text in any language, starting from the analysis of videos containing any kind of human subject.

5.2 Methodology

In this Section we describe the methodology for the automatic emotional annotation of text on the basis of the analysis of speaker's facial expressions in videos. The proposed methodology consists of three phases, as depicted in Figure 5.1: first, a data source is selected, with particular attention to some issues that could preclude the feasibility and/or the accuracy of emotion detection (phase one). Afterwards, both subtitles and frames are analyzed in order to filter out non-relevant scenes (phase two). Finally, facial expressions of people appearing in the remaining frames are analyzed and the resulting emotions are assigned to the corresponding subtitle chunk (phase three). A detailed description of each methodology step is presented in the following subsections, along with the discussion of the main issues of each phase.

5.2.1 Source Selection

The first phase involves the selection of the data source. This is a crucial phase, as the quality of the final annotated corpus strongly

FIGURE 5.1: The methodology for the emotional annotation of text

depends on the selection of a suitable set of input videos. Even if emotions are expressed through universally shared patterns (e.g., the tightening of the lip corner in anger expressions), there are several factors that can impact on speaker's spontaneity and expressiveness and that must be considered when selecting the video categories to be analyzed. To this purpose, it is important to note that "posed facial behavior are mediated by separate motor pathways than spontaneous facial behaviors" [40] and hence their analysis could be misleading.

We performed an intense activity of video scouting, consisted in the manual analysis of several YouTube videos, in order to find the most suitable video categories. A list of issues that we faced in this activity, and can potentially impact on the quality of text annotation, includes:

- lack of expressiveness: in some video typologies, such as news reports or product reviews, speakers are required to maintain an expressionless face, in order to give objectivity to their speech. As a consequence, the analysis of their facial expressions can be misleading, as emotion-bearing words could be annotated as neutral.

- interpretation: in case of movies or theatrical monologues, some actors are required to play characters having a specific personality. In such circumstances, facial expressions are altered by acting: a criminal, for instance, might have a scary face even when talking about happy things.

- reported speech: when people report what another person has said, facial expressions reflect their personal feelings and hence detected emotions can be in contrast with the original meaning of the sentences.
- external factors: external factors impacting on speaker's mood can affect the correlation between speech and facial expressions. For instance, an eyewitness interviewed immediately after a plane crash would probably show fear expressions, regardless of the specific words he is pronouncing.
- subtitles quality: video and subtitles might not be in synchronization, thus facial expressions could not correspond to subtitle text, or subtitles may not be accurate, as they have been automatically generated.

The above-listed issues are noise factors that cannot be totally avoided but the selection of a proper data source can effectively impact on their presence in analyzed videos. Furthermore, some of these problems can be automatically detected in videos: for example, interviews and news reports can often be identified through the analysis of the video title.

5.2.2 Video Preprocessing

Sentiment Analysis of subtitles

In presence of large amount of videos, their analysis can be resource-intensive and time-consuming. Usually only a limited percentage of collected videos contains significant emotions and it is important to recognize them, so as to avoid analyzing the remaining, less significant videos. For this reason we propose to initially perform 3-class sentiment analysis of subtitles, with the purpose of detecting neutral sentences, which generally do not have any emotional valence,

and discard videos whose subtitles are mostly neutral. Such technique is based on the assumption of a strong correlation between the opinion and the emotions that are expressed in a text; for an empirical evaluation of such correlation, refer to Section 5.3.1. An important advantage is that it is possible to initially download only subtitles, which are far smaller than videos, and filter out videos not having any emotional content without having to download and analyze video streams. As a consequence, times and resources needed for the analysis are drastically reduced.

However, the exploitation of subtitles for the automatic selection of videos that are relevant from an emotional perspective raises some issues: first, the feasibility of the analysis is subject to the availability of subtitles in a language supported by sentiment analysis tools. Second, the quality of subtitles is relevant for sentiment analysis: in particular, an accurate transcription of speech and a correct synchronization between text and speech are required. These aspect are also crucial for the following methodology steps. For these reasons, it is preferable to only consider manually created subtitles. Video sharing platforms, such as YouTube, usually provide information about the nature of subtitles (i.e., manually created or automatically generated using speech recognition) through appropriate API calls.

Video Splitting

After selecting the most relevant videos through sentiment analysis, videos are splitted into chunks (i.e., small portions of the video). This operation allows for a better evaluation of emotions, since in such case the latter are more likely to be constant. Generally speaking, short chunks (e.g., 2-3 seconds) have higher probability of containing a single, constant emotion. Therefore, a simple strategy could consist in the use of temporal windows of fixed length. The

approach is fast, as it does not require video and/or text preprocessing, but it does not offer any guarantee about the presence of a unique emotion in the chunk. Moreover, if a subtitle is associated to more than one chunk, each giving different information about the emotional content of the scene, it is difficult to determine the right emotion to assign to the subtitle. Consequently, it is preferable to split videos on the basis of the temporal information given by subtitles[1]: each video chunk begins at the start of a subtitle chunk and ends at its end. From a semantic point of view, this approach is reasonable, since different sentences are usually about different topics and hence may convey different emotions: if we split videos on the basis of sentences we have higher probability of obtaining video chunk with uniform emotions. Obviously, there must be a correspondence between sentences and subtitles, which is often verified in case of manually created subtitles. The approach can be further refined by taking into account syntactical aspects: for instance, the presence of adversative conjunctions (e.g., "The battery is excellent but the user interface is really confusing") in a subtitle may indicate a variation in the emotional content of the sentence and, probably, in the facial expression of the speaker.

Frame Analysis

After splitting videos into chunks, each chunk is subject to a preprocessing phase, whose output is the set of F frames to be analyzed. From a computational perspective, a desirable property for F is that $|F| \ll |C|$, where C is the total number of frames in the chunk, since the analysis of facial expressions is a computationally-intensive task. Many frames may be discarded without significant loss of information, both by choosing a proper sampling rate and

[1]Subtitles are usually divided into multiple chunks, for each of which are indicated the start and the end time

by filtering out irrelevant frames.

The choice of the sampling rate *sr*, that is the number of frames per second (FPS) to be extracted, has implications on both speed and accuracy. A high value for the parameter *sr* implies a higher number of frames to be analyzed, with a consequent increase in the execution time of the facial expressions analysis. Since a speaker is expected to exhibit almost identical expressions in a block of consecutive frames, the analysis of the entire block is redundant. On the other hand, by choosing a small value for *sr* (e.g., $sr<1$) there is the risk to extract many irrelevant frames, such as those where there are transitions from an expression to another. Furthermore, facial expressions are somewhat dependent on the concomitant phonatory movements. For instance, the pronunciation of the vowel [a] requires speakers to widely open their mouth, that could be interpreted as a surprise expression. As a consequence, the analysis of a too small amount of frames is error-prone, especially in presence of speakers with a great articulation of open vowels. We performed some preliminary experiments in order to find a value for *sr* that would balance speed and accuracy. We found that $2 \leq sr \leq 4$ offers a classification accuracy comparable to the analysis of every frame, while reducing the whole execution time of approximately one order of magnitude.

Some sampled frames should be discarded as they do not contain useful information or they are not suitable for the following phase of facial expression analysis. A non-exhaustive list of factors that may affect facial expression recognition includes:

1. lateral position of face and consequent non-recognizability of an eye.

2. presence of many people in a frame, since it requires the use of

more sophisticated face recognition techniques to track emotional variations of each person in the chunk.

3. phonatory movements that may alter mouth opening (e.g., the pronunciation of phonems like /ee/ or /ow/).

4. bearded men or people with glasses/hats that can mask facial points.

5. old white people, since white eyebrows are similar to skin color and are hardly detected by face detection algorithms.

In particular, in this work we focus on the first two factors and hence we analyze the presence (and number) of faces, their orientation and the recognizability of eyes and mouth in each frame. To this purpose, we use the Viola-Jones face detector [65] implemented in OpenCV 3.0, which is based on Haar features. Such classifier is often inaccurate in detecting mouth: in fact, in preliminary experiments we found that it detects several mouths in the upper part of face. Hence, we improved the technique by defining boundaries for the upper and lower part of face (see Fig.5.2) and by discarding detected mouths and eyes whose coordinates were out of respective areas. In particular, we divided the face rectangle into two smaller rectangles with width equal to original width and height respectively equal to 55% (top rectangle) and 45% (bottom rectangle) of original height.

On the basis of the above-mentioned analysis, we assign a score between 0 and 10 to each frame (see Table 5.1) and we filter out frames with score below a certain threshold γ. Faces where less than two eyes are detected are strongly penalized, since the area around eyes contains relevant markers for the following phase of facial expression recognition. In preliminary experiments we found that $\gamma \geq 8$ offers accurate facial expression recognition.

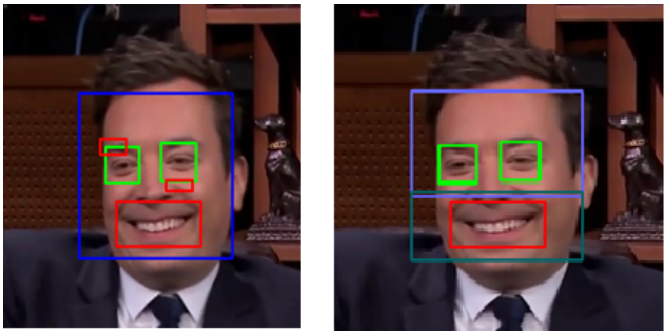

FIGURE 5.2: Example of mouth detection before (a) and after (b) the definition of boundaries for the upper and lower part of face. Detected mouth and eyes are discarded if their coordinates are out of respective areas.

TABLE 5.1: Classification of frame quality on the basis of recognizable faces, eyes and mouths

Score	10	9	8	7	6	5	4	3	2	1	0
Faces	1	1	1	1	1	1	1	1	2	>2	0
Eyes	2	2	2	2	1	1	0	0	0	0	0
Mouths	1	2	0	≥2	1	0,>1	1	0,>1	0	0	0

5.2.3 Emotion Recognition

The output of the previous phase is a set of chunks, one for each portion of subtitles, sampled at a certain sampling rate *sr* and composed of frames containing a single, frontal face. The last phase consists in the analysis of the emotions expressed in these chunks and the association of the corresponding subtitles with such emotions. The analysis of chunks can be conducted in two different ways: by analyzing the facial expressions contained in each frame, and then averaging the results, or by averaging the facial expressions (i.e., the coordinates of the facial points that identify the facial expression) contained in the chunk. Both approaches have pros and cons: for instance, by averaging the coordinates of facial points we

lose the possibility of capturing the variations of the emotions inside the chunk but we mitigate the noise introduced by phonatory movements.

Frame-level analysis

In this phase, for each frame f_i of each chunk C_j we analyze the speaker's facial expression with respect to Ekman's theory of six basic emotions (i.e., anger, disgust, fear, happiness, sadness, surprise) and we obtain the emotion vector e_i, defined as $e_i = [e_i^{(an)} e_i^{(di)} e_i^{(fe)} e_i^{(ha)} e_i^{(sa)} e_i^{(su)}]^T$, where $e_i^{(\alpha)}$ represents the value of the α emotion in f_i. At the end of the analysis, each subtitle chunk C_j is associated with the emotion matrix E_{C_j}. At the moment, we perform frame-level facial expression analysis through the free version of Microsoft Emotion API. Unfortunately, no information about its classification accuracy is provided by Microsoft. The facial expression analysis is performed with respect to eight classes: in addition to Ekman's basic expressions, the software evaluates the contempt and the neutral expressions. The value of the latter is calculated as the 1's complement of the sum of the other vector components, each of which has a value in [0,1]. In early experiments we found that the analysis of facial expressions of people while they are speaking is a challenging task, since the degree of mouth opening (that is considered a key feature by the facial expression analyzer) of a speaker strongly depends on the articulation of the speech. Although we limited our experiments to Microsoft Emotion API, it is plausible that other tools may have the same issue, as a large number of facial expression analysis techniques in literature relies on this feature (e.g., [66]). We hypothesized that mouths could be removed from images without compromising the analysis, as the software may still rely on the position of eyes, forehead and eyebrows as emotion markers. We

tested our hypothesis on a small test set and we noticed a 20% increase in classification accuracy when mouths were manually hidden before performing the facial expression analysis.

The intermediate output of this phase is, for each chunk C_j, an emotion matrix E_{C_j} and an annotation in the form $s_j \Leftrightarrow E_{C_j}$, where s_j represents the subtitle related to the chunk C_j. This level of annotation provides information about the distribution of emotions in each frame, while we are more interested in a chunk-level annotation. Therefore, we define an aggregate function $f : E_j \rightarrow \tilde{e}_j$ that applies an aggregation operator (e.g., max, avg) to each row of E_j, outputting a vector $\tilde{e}_j$ having a single aggregate value for each emotion. To bind each subtitle chunk to a class label l_j, the transform function $g : \tilde{e}_j \rightarrow l_j$ is applied in order to evaluate the dominant emotion and assign the related class label to s_j. The choice of g is non-trivial, as in some situations there is no correspondence between the maximum value of $\tilde{e}_j$ and the actual dominant emotion. For instance, in case of noisy scenes, the facial expression analyzer would probably assign lower values to emotions, as it is not able to detect enough emotion markers in speaker's face. In this scenario, the function g should discard the highest value in $\tilde{e}_j$, whenever it is related to the neutral emotion and there is another emotion with a sufficiently high value. At the moment, in our implementation we set $f = avg()$ and $g = max()$ but other solutions are under investigation: for example, we are interested in performing experiments using $f = max()$.

An example of this approach is depicted in Figure 5.3, where ten frames belonging to the subtitle chunk "I am anguished and tormented" (Figure 5.3(a)) are processed by a facial expression analyzer, whose output is a set of emotion vectors, that are plotted in Figure 5.3(b). In the line chart, the sadness expression (blue line, circular marker) dominates the central part of the graph (frames 3-7), while disgust expressions (green line, x-shaped marker) appear on

the final frames. By applying $f = avg()$ to frame-level emotion vectors we obtain a single chunk-level vector, in which the component with the highest value (i.e., sadness) is selected through $g = max()$ as the representative emotion of the chunk.

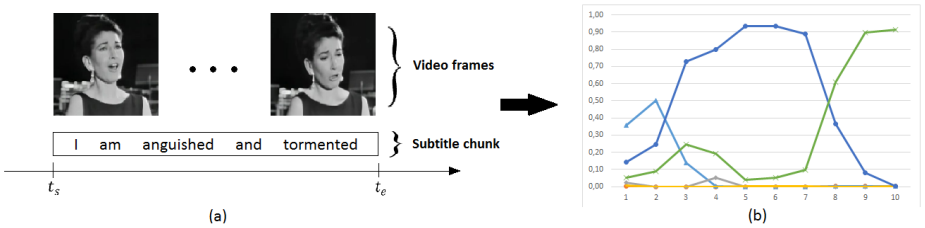

FIGURE 5.3: An example of frame-level facial expression analysis.

In preliminary experiments we found that the annotation accuracy of the system is quite variable and depends on the quality of input videos (e.g., expressive speakers, synchronized subtitles). For some videos we reached a remarkable 66% classification accuracy in 8-class emotional annotation but we plan to perform a wider experimentation in order to determine the most suitable aggregation operators, since their choice highly impacts on the overall accuracy of the annotation.

Chunk-level analysis

The goal of this part is to show how to extract information about emotion through a chunk-level analysis [67]. To this end, relevant features are extracted from each frame and then aggregated to be fed to a facial expression classifier. The entire process is depicted in Fig.5.4.

FIGURE 5.4: The workflow of the chunk-level facial expression analysis.

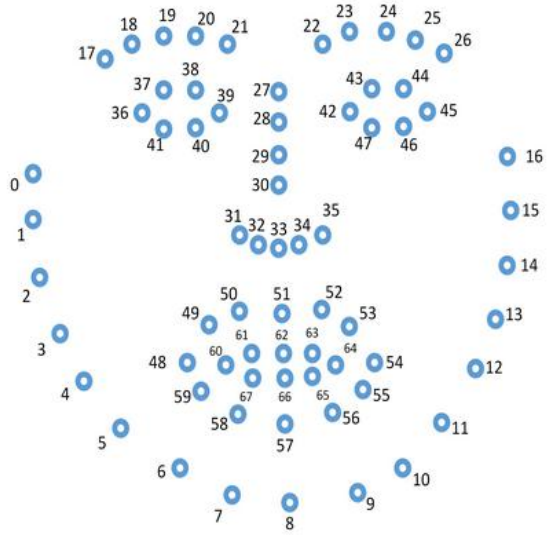

FIGURE 5.5: The 2-D landmarks point distribution .

To extract facial features we use the Open-Face library[2], a tool intended for facial landmarks detection and Action Unit (AU) recognition. The Open-face software allows for the extraction of 68 characteristic facial points from each frame. The 2-D landmarks point distribution is shown in Fig.5.5. Facial points are used to construct facial features for each frame. We manually define a set of 19 features, calculated as distances in pixels between some characteristic points. Note that Open-face provides both 2-D and 3-D location estimations of the 68 landmarks, but we only consider the 2-D coordinates in pixels, since the 3-D representation is intrinsically noisier because it is based on an estimation of the camera position in world space. The chosen distances between points are:

[2]https://cmusatyalab.github.io/openface/

{(17,33), (18,33), (19,33), (20,33), (21,33), (22,33), (23,33), (24,33), (25,33), (26,33), (38,40), (37,41), (44,46), (43,47), (51,57), (48,54), (50,58), (52,56)}

These distances are then normalized through division by the distance (29,30); such distance is expected to remain constant over frames as it is not affected by any facial muscle contraction. Along with the above distances we also consider AUs, since they are related to the expression of basic emotions. Open-face detects 17 AUs and assigns them a level of intensity between 0 and 5. Moreover, the presence of the "Orbicularis Oris" action unit, commonly known as "lip suck" gesture, is reported as a 0/1 value. We therefore construct a vector of 37 facial features (18 action units and 19 distances) for each frame. Since we have one feature vector for each frame, from the analysis of each chunk we obtain a feature matrix.

The subsequent step is feature aggregation among frames belonging to the same chunk: this is performed by averaging the values of each feature over the frames in the chunk. This operation has the effect of reducing the distortion introduced by phonatory movements, such as the natural mouth opening for the pronunciation of the phoneme /ee/, that could be interpreted as happiness. A problem in averaging the feature vectors related to a chunk is that, in case of video chunks containing people expressing two or more emotions that are in contrast, the aggregated feature vector will be likely classified as neutral. This issue outlines the importance of creating video chunks with homogeneous emotions.

The final step are the analysis of facial expressions contained in each video chunk and, consequently, the annotation of the related subtitles. To this purpose, the aggregated feature vector of each chunk is classified through supervised learning techniques. At the moment, this represents the critical point of the chunk-level analysis, since it requires the creation of a manually annotated training

set. For an empirical comparison of the performance of several algorithms on a real-world video dataset refer to Section 5.3.3.

5.3 Experiments

We evaluated three critical aspects of the proposed methodology: the correlation between opinions and emotions, the accuracy of the Video Preprocessing phase in filtering out non-relevant chunks and the performance of the facial expression classifier. For the latter, we focused on the technique for chunk-level analysis of facial expressions, since it is the only that does not depends on external software for the classification of expressions. As a future work, we plan to evaluate the overall accuracy of the entire methodology with further experiments.

5.3.1 Correlation between opinions and emotions

As seen in Section 5.2.2, the proposed methodology is based on the hypothesis that, for each sentence s, an association rule in the form $s_p \rightarrow s_e$ exists between the presence of non-neutral sentiment polarity s_p and emotions s_e. For this reason, we empirically evaluated the confidence of such rule by considering the test data of the SemEval-2007 Task #14 dataset[3]. The analyzed dataset is composed of 1000 headlines annotated by human operators with respect to both emotions and polarity. Since annotations were in the form of numerical values (from 0 to 100), coherently with [58] we defined a threshold $\gamma = 20$ to discriminate between neutral and sentiment-/emotion-bearing sentences. As a result, two labels were assigned to each sentence: neutral/non-neutral polarity (s_{np}/s_p) and neutral/non-neutral emotion(s_{ne}/s_e). The resulting label distribution is shown in Table 5.2.

[3]http://web.eecs.umich.edu/~mihalcea/affectivetext/

TABLE 5.2: Empirical evaluation of the correspondence between sentence polarity and emotion

		Sentiment	
		Positive/Negative	Neutral
Emotion	Yes	753	136
	No	67	21

The confidence of the association rule $s_p \rightarrow s_e$ is defined in Eq.5.1, where $supp(X)$ represents the support of X, that is the number of occurrencies of X in the considered dataset divided by the total number of occurrencies in the dataset.

$$conf(s_p \rightarrow s_e) = \frac{supp(s_p \wedge s_e)}{supp(s_p)} = \frac{\frac{753}{1000}}{(\frac{753+67}{1000})} = 0.92 \qquad (5.1)$$

The high confidence of the rule demonstrates that a sentence with non-neutral polarity usually conveys an emotional content, which supports our preliminary hypothesis. Other association rules that may be derived from Table 5.2 have lower confidence, such as $s_e \rightarrow s_p$ (confidence: 0.85) or $s_{np} \rightarrow s_{ne}$ (confidence: 0.13), which suggests that a perfect one-to-one correspondence between sentiment polarity and emotions does not exists. Anyway, the lower confidence of those rules does not affect the soundness of our approach, since they are irrelevant for the proposed methodology. It can be observed that 136 emotion-bearing sentences (i.e., 15.3% of the total) are filtered out, i.e. those sentences that express emotions while having neutral polarity. Anyway, it does not represent a limitation because the methodology is meant to be used for the analysis of Big Data sources, where the loss of a small percentage of data is negligible due to the high volume of available data.

5.3.2 Chunk Selection

In order to evaluate the accuracy of the Video Preprocessing phase (Section 5.2.2) in extracting significant chunks from videos, we carried out some experiments on a dataset composed of 50 YouTube videos which were retrieved using the keyword "monologue". Such keyword was chosen after a preliminary scouting phase with the purpose of maximizing the probability of retrieving videos containing a single person expressing emotions, so as to have a dataset that would allow to accurately evaluate the analysis of facial expressions. Videos have been split into chunks on the basis of the temporal information provided by subtitles; consecutive non-subtitled frames have been considered as a single chunk. We obtained a total of 1456 chunks, of which 995 from subtitled frames and 461 from non-subtitled frames. Chunks have been manually annotated by three volunteers with respect to the presence of visible faces and/or emotions. Subtitles have been hidden so as not to influence the annotators. We made the following assumptions:

- the "visible_face" label is assigned to a chunk only if a single, frontal, clearly detectable face appears in the scene for at least 75% of the total chunk length; otherwise, the "other" label is assigned to the chunk. The restriction to a single face is motivated by the intrinsic noise generated by the presence of multiple faces in a frame: for instance, the spectators of a show often exhibit different facial expressions and hence it is difficult to correctly annotate the frames in which they are filmed.

- the "emotion" label is assigned to a chunk only in presence of a clearly distinguishable, non-neutral emotion for at least 75% of the total chunk length; otherwise, the "no_emotion" label is assigned to the chunk.

- chunks labelled as "other" are also automatically labelled as "no_emotion", coherently with the previous considerations on the recognizability of facial expressions in frames containing more than one person.

The label distribution is shown in Table 5.3.

TABLE 5.3: Label distribution in the experimental dataset

	emotion	no_emotion
visible_face	982	224
other	-	250

The 982 chunks with labels "emotion" and "visible_face" were marked as "Accepted"; the remaining 474 chunks were marked as "Refused". Then, we applied our Video Preprocessing phase to such dataset. The result is shown in Table 5.4, where the presence of emotions is evaluated through sentiment analysis (on the basis of the correlation discussed in Chapter 5.3.1).

TABLE 5.4: Confusion matrix for the Video Preprocessing phase

	Actual Accepted	Actual Refused
Predicted Accepted	825	43
Predicted Refused	157	441

The technique has an overall 86.9% accuracy, while precision=0.84 and recall=0.87 (F_1-score=$2 \cdot \frac{precision \cdot recall}{precision + recall}$=0.85). For what concerns the "Accepted" class, which represents the most significant class for the Video Preprocessing phase, we notice a remarkable precision (i.e., 0.95) but a lower recall (i.e., 0.74). However, as already discussed, we point out that precision is much more relevant than recall for our analysis, since we are considering Big Data sources like YouTube and the goal is to define corpora having a high quality

of annotation. Therefore we can conclude that the proposed Video Preprocessing phase is suitable for the selection of relevant chunks.

5.3.3 Emotion Recognition

Experimental setup

In order to evaluate the performance of the facial expression classifier based on chunk-level analysis (Section 5.2.3), we built a dataset of real-world annotated videos. At first, we collected a large set of videos from YouTube by searching for specific keywords (e.g., monologues, tv shows), with the aim of obtaining results with clear emotional content. After automatically dividing each video in chunks on the basis of subtitles, each chunk was labelled with regard to its emotional content by three human evaluators. We considered valid only those annotations in which at least 2 of the 3 evaluators agreed on the presence of a specific emotion. Similarly to [38], we limited our analysis to the following emotions: anger, happiness, neutral and sadness. Among the annotated video chunks, we randomly selected 200 chunks in order to have a balanced dataset. The class distribution is shown in Table 5.5: In addition to the classifi-

TABLE 5.5: Class distribution for the considered dataset

Emotion	Number of occurrences
Anger	50
Happiness	50
Neutral	50
Sadness	50

cation accuracy, we also considered the execution time of the entire emotion recognition phase (i.e., feature extraction, aggregation and classification of facial expressions) as a key metric of the system, in

order to evaluate if the proposed technique is suitable for the analysis of large amounts of videos.

In order to improve the quality of emotion recognition, we chose to extract features only from those frames where the confidence level of the Open-face software in detecting human faces was higher than a given threshold. We empirically verified that considering only frames with confidence above 0.95 ensures a high level of accuracy in landmark detection.

We trained several traditional classification algorithms on our features, namely Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Tree (DT) and Random Forests (RF). In order to thoroughly evaluate the performance of the selected classifiers, we performed a 10-fold cross validation. For what concerns the SVM classifier, the internal parameters were estimated through a preliminary phase of parameter optimization. We chose a C-SVC Support Vector Machine with a radial basis function (rbf) kernel and the following values for the internal parameters: $\gamma = 0.166$, C=0.979 and $\epsilon = 0.049$. The SVM was trained with a 1-vs-all configuration in order to perform multiclass classification.

Results

The total execution time of the emotion recognition phase for our dataset of 200 videos (2.86GB) is about 12 minutes on an laptop with an Intel i7-2670QM CPU and 8GB RAM, which means that the system processes videos at 3.89 MB/s. Such performance allows for the analysis of large amounts of videos in reasonable times. It is also important to notice that execution time can be further reduced by considering higher sampling rates.

The experimental results for the selected classifiers are summarized in Table 5.6. The table clearly shows that the SVM classifier

TABLE 5.6: Accuracy of the considered algorithms estimated with 10-fold cross validation.

Algorithm	**Accuracy**
SVM	72%
k-NN	58.5%
DT	48%
RF	50%

outperforms the other models, with a remarkable +13.5% improvement in accuracy with respect to k-NN. The experiments suggest that data are not linearly separable, since using a non-linear classifier (i.e., SVM with rbf kernel) results in higher classification accuracy.
The confusion matrix for the SVM classifier and the results in terms of precision and recall are reported in Table 5.7.

TABLE 5.7: Confusion matrix for the SVM classifier

	Actual Neutral	**Actual Happiness**	**Actual Anger**	**Actual Sadness**
Predicted Neutral	36	1	5	4
Predicted Happiness	8	44	5	11
Predicted Anger	2	3	35	6
Predicted Sadness	4	2	5	29
Recall	**72.00%**	**88.00%**	**70.00%**	**58.00%**
Precision	**78.26%**	**64.71%**	**76.09%**	**72.50%**

The model has recall=0.72 and precision=0.73, while F1-score=0.725. The confusion matrix shows that the model has high recall for happiness, anger and neutral emotions; neutral and anger classes also have the highest precision. The last column suggests that the model is less accurate in the classification of sadness, since there are many misclassifications: it indicates that our model would benefit from the introduction of more sadness-related features, i.e. point distances whose variations are strongly related to sadness.

Chapter 6

Conclusion

The goal of this thesis is the definition of novel techniques for lexicon- and learning-based emotion recognition, in particular for the analysis of social content. For what concerns lexicon-based approaches, the present work extends the traditional techniques by introducing two algorithms for the disambiguation of polysemous words and the correct analysis 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 detects the right scope of negation through the analysis of 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 with regard to traditional approaches.

Moreover, the thesis describes the design and implementation of an application of the lexicon-based approach, that is a full-fledged platform for information discovery from multiple social networks, which allows for the analysis of users' opinions and characteristics and is based on Exploratory Data Analysis.

For what concerns learning-based approaches, a methodology has been proposed for the automatic creation of annotated corpora through the analysis of facial expressions in subtitled videos. The methodology is composed of several video preprocessing techniques,

with the purpose of filtering out irrelevant frames, and a facial expression classifier, which can be implemented using two different approaches. Experiments on datasets of YouTube videos show that the proposed video preprocessing phase is suitable for the selection of relevant video chunks and that the proposed features for facial expression analysis lead to a 72% accuracy on 4-class emotion recognition.

The future directions of research are mainly related to learning-based techniques for the automatic creation of annotated corpora, since the latter are fundamental for every application of emotion recognition. We plan to extend our methodology by adding speech recognition, with the purpose of delimiting frames related to each word and hence implement a word-level annotation, instead of a chunk-level annotation. Word-level annotation could extend the fields of application of the generated corpora and also improve the quality of annotation, especially in those cases where a video chunk contains more than one sentence or sentences with multiple emotions. Another future work is represented by the development of statistical techniques for the estimation, on the basis of a collection of annotated subtitles, of the annotation for relevant n-grams. Moreover, it would be useful to define a format for annotations that may contemplate different emotional valences for a single word, in function of its contextual usage. Finally, an important direction of research regards the determination of the best aggregation function for the emotion vectors in frame-level analysis of facial expressions. A possible alternative to the avg() function is represented by the max() function, which allows to capture the most intense emotion in the chunk. To this purpose, we plan to extend our experiments by collecting and analyzing a large dataset of YouTube videos.

Appendix A

Further Ph.D. Activities

In this Chapter we briefly outline some side activities carried on during the Ph.D., in particular on the physical optimization of large databases with time-varying workloads. In the era of cloud computing, characterized by elastic resources and a pay-per-use model, an optimized data access is important as it translates into the opportunity of reducing system resources and related costs. Database optimization may be performed in several ways: we focused on physical optimization and, in particular, on the index selection problem. Index selection, that is the selection of an appropriate set of indexes for a given database and workload, is challenging for many reasons: first, since database schemas of real applications are usually large (as they consist of several tables, each with many columns) and indexes can be defined on one or more columns, the space of indexes that are relevant for a given workload is usually very large. Secondly, even the estimation of the workload can be inaccurate, especially in case of complex applications, since it depends on user behavior, that may be unpredictable and may also change over time. In case of production database, it is possible to enable database logging in order to capture data that, when combined with information about the size of database tables, may allow for an estimation of the actual workload. However, since database logs usually contain a huge number of queries, a manual analysis of such logs is often infeasible: for this reason we proposed a methodology for index

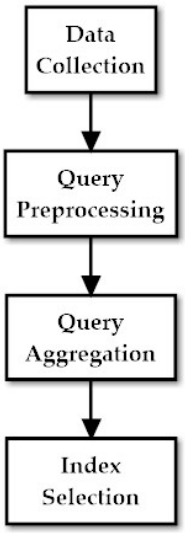

FIGURE A.1: The proposed methodology for physical database optimization

selection that relies on the automatic analysis of database logs.

The methodology consists of four phases, as depicted in Figure A.1: in the first step, queries executed on the database are collected over a fixed temporal window and hence preprocessed (step two) in order to dereference aliases and remove punctual information. In the third phase, queries are aggregated on the basis of their characteristics (e.g., typology, attributes) and, finally, a list of recommended indexes is provided (step four) on the basis of the analysis of both the attribute frequency and the cardinality (i.e., the number of records) of the tables involved in queries.

In order to demonstrate the effectiveness of the proposed solution, we also performed an experimental evaluation on the case study Nuvola, a multitenant SaaS software for school. Nuvola supports all the typical school activities and it is currently used by more than 1,000 Italian schools, serving a total of about 1.5 million users. Several billion queries are executed daily, thus making Nuvola a data-intensive application. We enabled database logging for

a single mid-size school in a temporal window of 24 hours, collecting a total of 23,087,671 SQL queries, and we used such log both for index selection and experiments. Before the optimization, the database contained a set of indexes that had been manually selected by database administrators on the basis of the expected average workload for all the schools and hence did not fit well to the specific workload of the examined school. By applying our index selection algorithm, we found 11 promising indexes that we added to the database. Then, we evaluated execution time and hardware usage before and after the optimization by considering different amounts of RAM. The results demonstrated that adding indexes selected by our methodology results in a consistent reduction of execution time, as shown in Table A.1.

TABLE A.1: Total (TQET) and average (AQET) query execution time

RAM	**16 GB**	**8 GB**	**4 GB**	**2 GB**
TQET (without indexes)	9h 20m	9h 26m	9h 31m	9h 34m
TQET (with indexes)	4h 28m	5h 10m	5h 34m	5h 39m
AQET (without indexes)	1.46 ms	1.47 ms	1.48 ms	1.49 ms
AQET (with indexes)	0.70 ms	0.81 ms	0.87 ms	0.91 ms

A more detailed discussion of this work can be found in [68].

Bibliography

[1] C. C. Chien and T. You-De, "Quality evaluation of product reviews using an information quality frame-work", *Decision Support Systems*, vol. 50, pp. 755–765, 2011.

[2] K. Wojcik and J. Tuchowski, "Feature based sentiment analysis", in *3rd International Scientific Conference on Contemporary Issues in Economics, Business and Management*, 2014, pp. 484–490.

[3] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey", *Ain Shams Engineering*, vol. 5, no. 4, pp. 1093–1113, 2014.

[4] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? sentiment classification using machine learning techniques", in *2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[5] G. Qiu, X. He, F. Zhang, Y. Shi, J. Bu, and C. Chen, "Dasa: Dissatisfaction-oriented advertising based on sentiment analysis", *Expert Systems with Applications*, vol. 37, pp. 6182–6191, 2010.

[6] S. Baccianella, A. Esuli, and F. Sebastiani, "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining", in *International Conference on Language Resources and Evaluation (LREC)*, 2010, pp. 2200–2204.

[7] S. Mohammad and P. Turney, "Crowdsourcing a word-emotion association lexicon.", *Computational Intelligence*, vol. 29, pp. 436–465, 2013.

[8] A. Mircoli, A. Cucchiarelli, C. Diamantini, and D. Potena, "Automatic emotional text annotation using facial expression analysis", in *International Conference on Advanced Information Systems Engineering (CAiSE)*, vol. 1848, 2017, pp. 188–196.

[9] F. Colace, L. Casaburi, M. De Santo, and L. Greco, "Sentiment detection in social networks and in collaborative learning environments", *Computers in Human Behavior*, vol. 51, pp. 1061–1067, 2016.

[10] A. Agarwal, B. Xie., I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of twitter data", in *ACL 2011 Workshop on Languages in Social Media*, 2011, pp. 30–38.

[11] L. Oneto, F. Bisio, E. Cambria, and D. Anguita, "Statistical learning theory and elm for big social data analysis", *IEEE Computational Intelligence Magazine*, vol. 11, pp. 45–55, 2016.

[12] A. Tripathy, A. Anand, and S. K. Rath, "Document-level sentiment classification using hybrid machine learning approach", *Knowledge and Information Systems*, vol. 53, no. 3, pp. 805–831, 2017.

[13] O. Appel, F. Chiclana, J. Carter, and H. Fujita, "A hybrid approach to the sentiment analysis problem at the sentence level", *Knowledge-Based Systems*, vol. 108, pp. 110–124, 2016.

[14] M. Shams and A. Baraani-Dastjerdi, "Enriched lda (elda): Combination of latent dirichlet allocation with word co-occurrence analysis for aspect extraction", *Expert Systems with Applications*, vol. 80, pp. 136–146, 2017.

[15] B. Heerschop, F. Goossen, A. Hogenboom, F. Frasincar, U. Kaymak, and F. de Jong, "Polarity analysis of texts using discourse structure", in *20th Association for Computing Machinery (ACM) Conference on Information and Knowledge Management (CIKM'11)*, 2011, pp. 1061–1070.

[16] S. M. Mohammad, "From once upon a time to happily ever after: Tracking emotions in mail and books", *Decision Support Systems*, vol. 53, pp. 730–741, 2012.

[17] R. Moraes, J. F. Valiati, and W. Gavião Neto, "Document-level sentiment classification: An empirical comparison between svm and ann", *Expert Systems with Applications*, vol. 40, pp. 621–633, 2013.

[18] A. Sharma and S. Dey, "Using self-organizing maps for sentiment analysis.", in *Cornell University Library*, 2013.

[19] B. Pang and L. Lee, "Opinion mining and sentiment analysis.", *Foundations and trends in information retrieval*, vol. 2, no. 1, pp. 1–135, 2008.

[20] S. Poria, E. Cambria, and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network", *Knowledge-Based Systems*, vol. 108, pp. 42–49, 2016.

[21] E. Cambria, "Affective computing and sentiment analysis", *IEEE Intelligent Systems*, vol. 31, pp. 102–107, 2016.

[22] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank.", in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[23] A. Go, L. Huang, and R. Bhayani, "Twitter sentiment analysis.", in *Final Projects from CS224N for Spring 2008/2009, The Stanford Natural Language Processing Group*.

[24] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and 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.

[25] A. Moreo, M. Romero, J. L. Castro, and J. M. Zurita, "Lexicon-based comments-oriented news sentiment analyzer system", *Expert Systems with Applications*, vol. 39, pp. 9166–9180, 2012.

[26] A. Neviarouskaya, H. Prendinger, and M. Ishizuka, "Recognition of affect, judgment, and appreciation in text.", in *23rd international conference on computational linguistic (2010)*, 2010, pp. 806–814.

[27] M. Taboada, J. Brooke, M. Tofiloski, K. D. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis", *Computational Linguistics*, vol. 37, pp. 267–307, 2011.

[28] 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.

[29] M. Wiegand, A. Balahur, B. Roth, D. Klakow, and 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.

[30] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis.", in *2005 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2005.

[31] Y. Choi and 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.

[32] A. Kennedy and D. Inkpen, "Sentiment classification of movie reviews using contextual valence shifters", in *Computational Intelligence*, vol. 22, 2016.

[33] L. Jia, C. Yu, and W. Meng, "The effect of negation on sentiment analysis and retrieval effectiveness", in *2008 International Conference on Information and Knowledge Management (CIKM)*, 2009, pp. 1827–1830.

[34] M. A. M. Shaikh, H. Prendinger, and M. Ishizuka, "Assessing sentiment of text by semantic dependency and contextual valence analysis", in *2007 International Conference on. Affective Computing Intelligent Interaction (ACII)*, 2007, pp. 191–202.

[35] P. Ekman, "An argument for basic emotions", *Cognition and emotion*, vol. 6, pp. 169–200, 1992.

[36] C. Pramerdofer and M. Kampel, "Facial expression recognition using convolutional neural networks: State of the art", *ArXiv preprint arXiv:1612.02903*, 2016.

[37] G. Benitez-Garcia, T. Nakamura, and M. Kaneko, "Multicultural facial expression recognition based on differences of western-caucasian and east-asian facial expressions of emotions", *IEEE Transactions on Information and Systems*, vol. 5, pp. 1317–1324, 2018.

[38] S. Poria, H. Peng, A. Hussain, N. Howard, and E. Cambria, "Ensemble application of convolutional neural networks and multiple kernel learning for multimodal sentiment analysis", *Neurocomputing*, vol. 261, pp. 217–230, 2017.

[39] E. Sariyanidi, H. Gunes, and A. Cavallaro, "Automatic analysis of facial affect: A survey of registration, representation and recognition", *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, pp. 1113–1133, 2015.

[40] Y. Sun, N. Sebe, M. Lew, and T. Gevers, "Authentic emotion detection in real-time video", in *Proceedings of the International Workshop on Computer Vision in Human-Computer Interaction (CVHCI)*, 2004, pp. 94–104.

[41] W. Sun, H. Zhao, and Z. Jin, "A complementary facial representation extracting method based on deep learning", *Neurocomputing*, vol. 306, pp. 246–259, 2018.

[42] S. Mo, J. Niu, Y. Su, and S. K. Das, "A novel feature set for video emotion recognition", *Neurocomputing*, vol. 291, pp. 11–20, 2018.

[43] J. Huang and C. Yuan, "Weighted-pcanet for face recognition", in *Proceedings of the 2015 International Conference on Neural Information Processing (ICONIP)*, 2015, pp. 246–254.

[44] I. Cohen, N. Sebe, A. Garg, L. S. Chen, and T. S. Huang, "Facial expression recognition from video sequences: Temporal and static modeling", *Computer Vision and Image Understanding*, vol. 91, pp. 160–187, 2003.

[45] Z. Yu and C. Zhang, "Image based static facial expression recognition with multiple deep network learning", in *Proceedings of the 2015 International Conference on Multimodal Interaction (ICMI)*, 2015, pp. 435–442.

[46] J. Tukey, "Exploratory data analysis. reading", in *MA: Addison-Wesley*, 1977.

[47] J. T. Behrens and C. Yu, "Exploratory data analysis", *Handbook of Psychology*, vol. 1, pp. 33–64, 2003.

[48] P. F. Velleman and D. Hoaglin, "Exploratory data analysis", *APA Handbook of Research Methods in Psychology*, vol. 3, pp. 51–70, 2012.

[49] C. Shearer, "The crisp-dm model: The new blueprint for data mining", *Data Warehousing*, vol. 5, pp. 13–22, 2000.

[50] A. Perer and B. Shneiderman, "Integrating statistics and visualization: Case studies of gaining clarity during exploratory data analysis", in *2008 Conference on Human Factors in Computing Systems*, 2008, pp. 265–274.

[51] P. A. Gloor and Y. Zhao, "Analyzing actors and their discussion topics by semantic social network analysis", in *Tenth International Conference on Information Visualization*, 2006, pp. 130–135.

[52] B. Suh, L. Hong, P. Pirolli, and 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.

[53] 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.

[54] ——, "Don't follow me: Spam detection in twitter", in *Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT)*, 2010, pp. 1–10.

[55] M. Mccord and M Chuah, "Spam detection on twitter using traditional classifiers", in *International Conference on Autonomic and Trusted Computing*, Springer, 2011, pp. 175–186.

[56] C. Diamantini, A. Mircoli, D. Potena, and E. Storti, "Social information discovery enhanced by sentiment analysis techniques", *Future Generation Computing Systems (In press)*, 2018.

[57] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network", in *2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003, pp. 173–180.

[58] C. Diamantini, A. Mircoli, D. Potena, and E. Storti, "Semantic disambiguation in a social information discovery system", in *Proceedings of the 2015 International Conference on Collaboration Technologies and Systems (CTS)*, 2015, pp. 326–333.

[59] C. Diamantini, A. Mircoli, and D. Potena, "A negation handling technique for sentiment analysis", in *Proceedings of the 2016 International Conference on Collaboration Technologies and Systems (CTS)*, 2016, pp. 188–195.

[60] M. A. Covington, "A fundamental algorithm for dependency parsing", in *39th annual Association for Computing Machinery (ACM) southeast conference*, 2001, pp. 95–102.

[61] E. F. Codd, "A relational model of data for large shared data banks", *Communications of the Association for Computing Machinery (ACM)*, vol. 26, no. 1, pp. 64–69, 1983.

[62] M. Golfarelli, D. Maio, and S. Rizzi, "The dimensional fact model: A conceptual model for data warehouses", *International Journal of Cooperative Information Systems*, vol. 7, no. 2-3, pp. 215–247, 1998.

[63] S. Livingstone, W. Thompson, M. Wanderley, and C. Palmer, "Common cues to emotion in the dynamic facial expressions of speech and song", *The Quarterly Journal of Experimental Psychology*, vol. 68, pp. 952–970, 2015.

[64] P. Ekman, R. Sorenson, and W. Friesen, "Pan-cultural elements in facial displays of emotions", *Science*, vol. 164, pp. 86–88, 1969.

[65] P. Viola and M. Jones, "Robust real-time object detection", *International Journal of Computer Vision*, pp. 137–154, 2001.

[66] Y. Tian, T. Kanade, and J. Cohn, "Recognizing lower face action units for facial expression analysis", in *The 4th IEEE International Conference on Automatic Face and Gesture Recognition (FG'00)*, 2000, pp. 484–490.

[67] M. Mircoli and G. Cimini, "Automatic extraction of affective metadata from videos through emotion recognition algorithms", in *2018 International Workshop on Metadata Usage and Management (M2U 2018)*, 2018, pp. 191–202.

[68] C. Diamantini, M. Mircoli, M. Moretti, D. Potena, and V. Tempera, "Workload-driven database optimization for cloud applications", in *2017 International Conference on High Performance Computing Simulation (HPCS 2017)*, 2017, pp. 595–602.