Knowledge-Based Systems 84 (2015) 162–178

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

ConSent: Context-based sentiment analysis

Gilad Katz *, Nir Ofek *, Bracha Shapira

Department of Information Systems Engineering, Ben-Gurion University of the Negev, Beer Sheva, Israel

ARTICLE INFO

Article history:
Received 1 December 2014
Received in revised form 6 April 2015
Accepted 7 April 2015
Available online 12 April 2015

Keywords:
Sentiment analysis
Context
Machine learning
Noisy data

ABSTRACT

We present ConSent, a novel context-based approach for the task of sentiment analysis. Our approach builds on techniques from the field of information retrieval to identify key terms indicative of the existence of sentiment. We model these terms and the contexts in which they appear and use them to generate features for supervised learning. The two major strengths of the proposed model are its robustness against noise and the easy addition of features from multiple sources to the feature set. Empirical evaluation over multiple real-world domains demonstrates the merit of our approach, compared to state-of the art methods both in noiseless and noisy text.

1. Introduction

Sentiment analysis refers to the inference of people's views, positions and attitudes in their written or spoken texts. Before the coining of the term, the field was studied under names such as subjectivity [1], point of view [2] and opinion mining [3]. Nowadays, the field is rapidly evolving due to the rise of new platforms such as blogs, social media and user-generated reviews.

A large body of work exists on the analysis of latent sentiment in social media platforms such as Twitter. The goal of these studies is to extract timely and relevant information as well as to gauge widespread opinions and sentiment. Agarwal et al. [4], for example, used lexicons and machine learning methods to infer sentiment in tweets while Wang et al. [5] used graph and hashtag analysis to evaluate opinions regarding various topics. In industry, studies of this kind could be beneficial for corporations interested in engaging disgruntled customers and reducing customer turnover [6].

Another domain that has received increased attention in recent years is sentiment analysis of transcribed text, i.e., speech converted to text using automatic machine translation [7]. By analyzing transcribed call center conversations, for example, corporation could identify (and approach) dissatisfied customers, assess their attitudes towards services and products and identify problems at their early stages.

Sentiment analysis is considered a challenging natural language processing (NLP) problem [8], and particularly so for Twitter and transcribed text. Twitter is difficult to analyze due to the short length of the text as well as the non-standard abbreviations that are often used [9]. These two traits create a highly sparse representation of terms and difficulties in the identification of synonyms and other relations between terms. In transcribed text, the relatively high error rate of the transcription process [10] causes a decline in the performance of NLP techniques such as parsing. For example, parsing the transcribed utterance "*got we can't was may need to delete files*" yields a different structure from parsing the true utterance "*well we may need to delete some files*." The mistaken analysis is mainly due to the error at the root of the tree – *got* instead of *well*. Apart from errors in speech recognition, spontaneous speech is often grammatically erroneous and always without punctuation marks.

We present Context-based Sentiment analysis (*ConSent*), a novel approach for sentiment analysis which has proven effective both for regular texts (those that adhere to the rules of grammar and use existing words) and texts with a high degree of noise. Our proposed approach consists of two phases: we first apply techniques from the field of information retrieval to detect *key terms* in the text and analyze the *context* in which they appear. Then, we use the detected terms to generate features for supervised learning. ConSent has several traits that make it very suitable for sentiment analysis in general and noisy text in particular: first, it considers the context of terms, which is important when the text is noisy [11,12]; secondly, it does not rely on grammatical structures, which may not be reliable in noisy text; and finally, it enables an easy integration of information from additional sources, such as the metadata of the analyzed text (e.g., the length of the text, the time of creation, the use of emoticons, etc.).

* Corresponding authors. Tel.: +972 54 4229152; fax: +972 8 6479346.
E-mail address: katzgila@post.bgu.ac.il (G. Katz).

http://dx.doi.org/10.1016/j.knosys.2015.04.009
0950-7051/

We tested our approach on three benchmark datasets: hotel reviews from the TripAdvisor website,[1] movie reviews and Twitter. Additionally, we created two automatically transcribed datasets, from two real-world call-centers. While the reviews datasets are used to evaluate our approach in frequently used domains, the other datasets enable us to test ConSent in a more challenging setting: noisy and unstructured text. In our evaluation, ConSent fares better than the commonly used and highly effective baseline in the majority of cases.

The contribution of this paper is threefold: first, we present a novel method for sentiment analysis whose strength lies in its ability to effectively analyze context and handle noisy data. This stems from ConSent's ability to selectively analyze terms that are in close proximity to one another while ignoring terms it deems irrelevant. Secondly, we evaluate our method on two datasets of transcribed phone conversations, obtained from real-world telecommunication companies. These difficult-to-obtain datasets enable us to conduct an accurate evaluation of our algorithm as well as to report on lessons learned from this real-world data. Thirdly, we propose a novel set of meta-features – developed specifically for sentiment analysis in transcribed text. As shown by our experiments, these features can greatly enhance the performance of learning-based approaches.

The structure of the paper is as follows: in Section 2 we present related work; ConSent is presented in Section 3, and the evaluation process is described in detail in Section 4. Finally, we discuss our results and propose directions for future research in Section 5.

2. Related work

2.1. Sentiment analysis in text

The techniques utilized for sentiment analysis can be roughly divided to *lexical* and *learning-based* approaches. The lexical methods often use predefined dictionaries of terms annotated with "positive" or "negative" scores, indicating the strength (i.e. polarity) of the sentiment they represent. These dictionaries are then used to determine the sentiment of the analyzed document by examining the frequencies and other measures of the negative and positive words in the text. The lexical approach has been proven to be effective in multiple studies [13–17].

Using predefined lexicons has two main advantages: first, there is no need to tag texts for training, thus preventing the long and arduous task of creating a training set. Secondly, the fact that the lexicons are defined independently of the data prevents overfitting and enables their use on multiple datasets. This is a significant difference from learning-based approaches, where the methods are defined specifically for the analyzed dataset. In order to increase the effectiveness of this approach, researchers have sought to enrich the lexicons through the use of natural language processing (NLP) [18–20], parts-of-speech (POS) tagging [21–23] and terms co-occurrence analysis [24].

The main strength of the lexicon-based approaches is also their weakness: as the lexicons are predefined, they are unable to adapt to novel or domain specific forms of expressions [25]. In addition, lexicon-based approaches do not naturally produce the level of confidence in the classification, as is automatically provided by learning-based approaches. The level of confidence is important in many sentiment analysis related applications where ranking is desired or when only the "top x" items are the main concern. One common example of this is commercial applications.[2,3] Moreover, a lexicon-based approach cannot work well for some domains containing unstructured text, such as Twitter or transcribed conversations. The reasons for this are the non-standard abbreviations used in Twitter (for example, "whose" instead of "who is" [9]) and the special nature of spoken text, e.g., sentences that "end halfway," word repetitions, and appearances of "non-word" expressions such as "um" and "ah" in the text [26].

Learning-based approaches apply machine learning algorithms to analyze the sentiment of the text. Their main strength lies in their ability to analyze the text of any domain and produce classification models that are tailored to the problem at hand. In addition, they are language independent and can therefore be successfully applied to multiple languages [27]. Moreover, learning-based approaches are capable of incorporating additional sources of information in their decision process [28].

The features and models used for the task of sentiment analysis when applying the machine learning approach have evolved over the years. Initial work in this area proposed using the presence of hand-picked terms [13] as features for the training of a classifier. Later studies attempted to use techniques from the field of information retrieval to generate features by using term frequency (TF) vectors [29] as representations of the documents. Although it is not without merit, the bag-of-words (BOW) approach is incapable of representing the order and proximity of terms in the text. This poses significant limitations in some sentiment analysis scenarios, as explained in [30].

Studies proposing learning-based algorithms can be divided into two groups: the *features focused* algorithms propose new features, whereas the *models focused* algorithms propose new classification models. The former group concentrates on creating new features that enable new perspectives of the analyzed data. These features are then fed to commonly used classifiers (such as SVM [31] and Naïve Bayes [32]) in order to classify the sentiment of the text. The latter group is focused on proposing new algorithms and classification models.

The features-focused approaches presented in the literature are diverse, proposing both text-based features and the integration of information from multiple sources. Ye et al. [29] proposed the use of term frequency (TF) vectors, while Mullen and Collier [33] proposed augmenting it with additional information about specific phrases. Wilson et al. [11] proposed phrase-level features combined with a dictionary of pre-set terms; and Boiy and Moens [27] presented a combination of several types of features such as discourse features and features designed to model negation.

Other features-focused approaches attempt to bridge the gap between lexicon-based and learning-based approaches by assigning dynamic weights to sets of predefined terms. Works in this area include Choi and Cardie [34], who used machine learning together with voting; and the work by Balahur et al. [30], who put special emphasis on the text surrounding entities in the analyzed text.

The models-focused approaches propose new statistical models for the analysis of sentiments. Maas et al. [35] and Wang et al. [36] proposed a probabilistic model combined with regression, while Read and Carroll [37] used lexical association and text distribution analysis in order to infer sentiment. Other studies propose the use of additional learning techniques such as active learning [38] and co-training [39].

Our proposed approach is an attempt to combine features-focused and model-focused approaches. We first present a model for the discovery of specific terms and contexts which are indicative of sentiment. Then we generate a set of novel features, based upon the terms discovered by our model. These features can be fed to any standard classifier. In addition, our approach enable the easy addition of features from other sources of information, as we demonstrate in our evaluation in Section 4.

[1] www.tripadvisor.com.
[2] http://www.alchemyapi.com/.
[3] https://semantria.com/.

2.2. Sentiment analysis of call-center conversations

Call centers are very often the main channel through which corporations communicate with their customers. As customer satisfaction is strongly correlated with profitability [40,41], organizations strive to develop techniques and tools to help them identify issues that bother their customers. For example, a study by Reichheld [6] showed that a mild (5%) increase in customer loyalty may result in a profit increase of 25–95%.

Aside from improving customer satisfaction, the analysis of call-center conversations may assist corporations in identifying problematic products and services. For example, the early realization that the name of a product often appears alongside terms such as "broken" and "defective" may assist in avoiding expensive and damaging recalls. Studies focusing on detecting customer-related issues by analyzing transcripts of customer-agent conversations include those by Takeuchi and Yamaguchi [42] and Cailliau and Cavet [43].

Commonly, call-center conversations are transcribed using speech-to-text software. By converting speech to written text, techniques from the fields of information retrieval and natural language processing (NLP) can be used for its analysis. This approach has been used in academic papers, usually in conjunction with machine learning algorithms [44–46], as well as commercial products. Nonetheless, all speech-to-text solutions proposed to-date suffer from a relatively high error-rate. This is the case for multiple reasons – poor call quality, different accents, background noise and several people speaking at once, to name a few.

This high error rate, coupled with the fact that spoken language tends to be less structured than written language, makes the task of sentiment analysis in automatically transcribed text especially challenging. For this reason, many of the studies in this area focus on non-textual aspects of the call [47,48] such as rhythm, intonation and loudness. However, these features are not always available.

To the best of our knowledge, only the work by Park and Gates [26] has tackled the challenge of identifying sentiment in call-center conversations by using features extracted from the text itself. These text-based features are part of a multitude of textual and non-textual features used to analyze sentiment in conversations. The textual features included the existence of lexicon terms in the analyzed text as well as identification of competitor names. The non-textual features included the analysis of the pause lengths in the conversation, the talking speed—measured by the number of words divided by the speaking time—of the customer and the relative number of words spoken by each speaker (customer and agent) throughout the conversation.

ConSent differs from the work of Park and Gates' in several key aspects. First and foremost, while theirs is partly based on a lexicon, ours is based solely on machine learning. This prevents the need to pre-define lexicons, thus providing greater flexibility. Secondly, we do not analyze the text as a whole but instead focus on specific terms in specific contexts. Given the typical noisiness of transcribed text, this is a significant advantage as we focus on a small subset of indicative terms and are therefore less likely to be effected by transcription errors. Finally, some of Park and Gates' features are domain-specific—e.g., competitor and products names are used as a basis for some of the features—while ours are generic. It should be mentioned, though, that as one of the contributions of this work we propose a set of non-textual features that builds upon some of the features proposed by Park and Gates [26]. We use these features to improve the performance of our method in the domain of call-center conversations as well as to demonstrate how our model can easily utilize features from external sources.

2.3. Sentiment analysis in Twitter

Mining sentiment in Twitter has great potential in enabling researchers to better model and understand a large variety of human beliefs and behavior. The research areas involving Twitter include, among other issues, the detection of political opinion [49] and the measuring of quality of living [50]. However, despite being accessible to humans, Twitter is far less accessible to machine analysis, mainly due to its irregularities.

Textual irregularities in Twitter are numerous and diverse. For example, the letter 'c', which is included among the top 100 most frequently used terms according to Google's twitter frequency lexicon,[4] is often used instead of the word 'see'. Other irregularities include lack of punctuations and the use of informal text, slang, non-standard shortcuts and words concatenations. For example, the hashtag *#welcomehomefree* needs to be parsed and converted into "welcome home free".

The grammatical structure of Tweets is also different from common structure of longer texts (e.g. new articles). This is due to the maximal allowed length of Tweets (140 characters), which encourages the use of abbreviations. When combined with emoticons and hashtags, it is clear why typical NLP solutions such as full or shallow parsing are unlikely to work well [51,52]. Foster et al. [53], for example, report a drastic drop in performance in parsing, when moving from the Wall Street Journal (WSJ) domain to a Twitter dataset.

Sentiment analysis of twitter data has become increasingly popular in recent years. This trend led to the development of a publically available datasets and set of tasks, described in SemEval [54]. Some studies [17] propose utilizing existing lexicons for the generation of related features such as counting negative and positive terms, but these attempts proved largely insufficient due to high sparsity of the data [52]. Other studies presented a more elaborate approach: Evert et al. [55], for example, generated Twitter-specific features in addition to utilizing several lexical and knowledge base sources. Their features analyzed aspects such as the use of emoticons and upper case words. Recently studies presented at the Semeval 2014 shared task [54] mainly involve Twitter-specific feature harvesting and exploitation, in addition to a large usage of lexicons.

2.4. Comparison to the previous version of ConSent

The original rule-based version of ConSent, (CoBAn) presented in [56], was developed and evaluated in the field of data leakage detection. In order to determine whether a document contained confidential information, the algorithm used predefined formulae to determine the "confidentiality score" of the analyzed text. The model was also able to detect small amounts of *rephrased* confidential text hidden in larger non-confidential documents, a task which proved difficult both for fingerprinting algorithms [57,58] as well as BOW classifiers.

Despite its effectiveness in the security domain, the rule-based method needed to be modified in order to be successfully applied to the field of sentiment analysis. This was due to the following reasons:

(a) Sentiment analysis deals with highly nuanced text that requires consideration of multiple factors. This complexity is difficult to model using a rule-based approach.
(b) The rule-based approach needs to be adapted to every new domain. The weights assigned to each parameter in

[4] https://github.com/first20hours/google-10000-english/blob/master/google-10000-english.txt.

formulae need to be optimized for each dataset – a lengthy and difficult process.

(c) While metadata (non-textual features) can contribute greatly to the performance of sentiment analysis algorithms (as we later show in Section 4.4), the integration of these features into a rule-based system is difficult and was not applied in CoBAn. Judging the contribution of each feature is problematic, and combinations of features further complicate this task.

These reasons led us to the development of the method presented in this work: ConSent. Like its earlier version in [56], the first step of this method is the identification of indicative key terms and the relevant contexts in which they appear. However, instead of rules, we utilize the detected terms to generate a set of features that are far more capable of representing the various aspects of highly nuanced text.

3. The proposed approach

ConSent consists of two phases: *learning* and *detection*. During the learning phase we first generate a set of *key terms* and *context terms* from a training set. Then, we generate features based on these terms and train a classifier that will be used to analyze sentiment in the detection phase. The detection phase is designed as a classification task; we scan each document in search of the key and context terms, generate features based on the terms that were found, and use the classifier to identify sentiment in the document.

Notational conventions. T denotes the training set used to create the model. The labels of all documents in this set are known. The training set consists of two types of documents: *positive sentiment documents* and are denoted T_{pos} and *negative sentiment documents* and are denoted T_{neg}. A key term is denoted as t_{key} and a context term as $t_{context}$. The language model generated from a corpus of documents C is denoted as lm_c. The probability value of a term t in a language model is denoted as $lm(t)$.

3.1. The learning phase

The learning process consists of three steps. During the first step we identify terms whose presence in the text is indicative of the existence of sentiment, either positive or negative. These terms, which we refer to as *key terms*, serve as the first indicators of sentiment in the text. Then, during the second step, we identify a set of *context terms*for each key term. The context terms are used to model the text in the key term's vicinity. In the third step, these terms are used to create features for supervised learning. The products of this phase are sets of key and context terms, and the features generated from the training set documents.

The learning phase is presented in Fig. 1. The training requires a set of documents containing either positive or negative sentiment, denoted in Fig. 1 by blue and red circles, respectively. In our experiments, each dataset was trained on 80% of the original data and tested on the remaining 20% using the 5-fold cross validation procedure. To demonstrate the cross-corpus effect (presented in Section 4.4.3) we trained the model on one dataset and tested on another. Then, for either type of sentiment we generate the key terms which are later used to provide the initial indication for the existence of that type of sentiment. Next, for each key term we generate a set of context terms. These terms are used as validators to determine whether their "parent" key term truly appears in a context that expresses sentiment. Finally, we identify these terms in each document of the training set and generate a set of features for the said document. These sets of features are used to train a sentiment classifier.

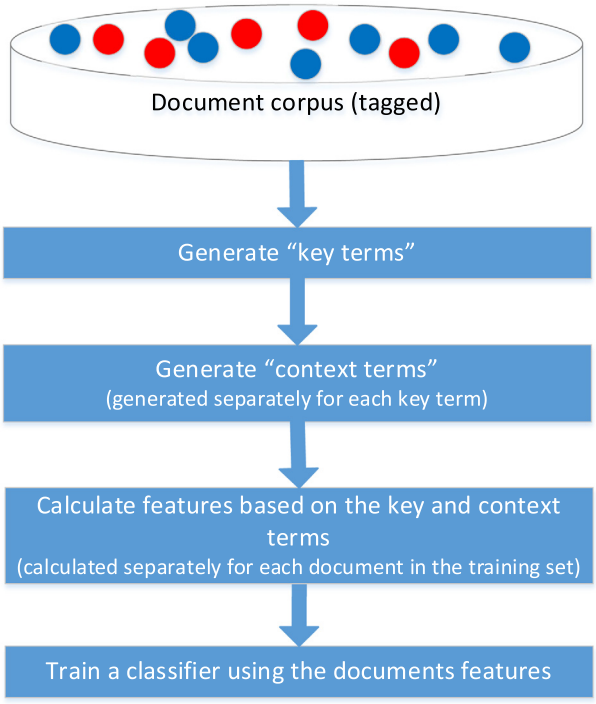

Fig. 1. The learning phase.

3.1.1. Key terms identification

The key terms identification process is based on *language modeling*, an approach mostly used in the field of information retrieval [59–61]. This technique represents a corpus of documents by the terms it contains; by denoting the value of each term in the corpus as the probability of selecting it when randomly sampling a term (or a sequence of terms) from the entire corpus, more frequent terms receive higher values in the language model. This approach can be represented as $lm(w_1, w_2, w_3 \ldots) = P(w_1, w_2, w_3 \ldots / C)$, where C denotes the number of terms in the documents corpus. Language modeling is often used for ranking documents in response to a query [62,63] and for related tasks such as query expansion [64,65] and query performance prediction [66,67]. In our experiments we have used the *tri-gram* model, which means a term can consist of either one, two or three words. While it has many similarities to the TF-IDF approach, this approach probability-based and is therefore considered to have a sounder theoretical base. An in-depth comparison of the two approaches can be found in [60].

The key term identification process is carried out *iteratively and separately* for the positive and negative sentiment. In the interest of clarity, the key terms identification process described below is only for *positive sentiment* even though the process is also repeated for negative sentiment. The key terms identification process is as follows:

1. Generate a language model for the set of positive-sentiment documents (T_{pos}). We denote this language model as lm_{pos}.
2. Generate a language model for the set of negative-sentiment documents (T_{neg}). We denote this language model as lm_{neg}.
3. For each term t in lm_{pos} we compute $score(t)$ which is the ratio of its frequency in positive-sentiment and negative-sentiment documents. This is done using Eq. 1.

$$score(t) = \frac{lm_{pos}(t)}{lm_{neg}(t)} \tag{1}$$

4. If $score(t)$ exceeds a predefined threshold, t is identified as a key term.

For each type of sentiment—positive and negative—the process returns a list of key terms and their scores. $Score(t)$ reflects the indicative power of a term t for the positive (or negative) sentiment of a document. All terms whose score is above a predefined threshold are identified as key terms. The algorithm describing this process is presented in Algorithm 1 (we present the process only for positive sentiment).

Algorithm 1. Identifying the key terms in the training set.

Generate_Key_Terms($T_{pos}, T_{neg}, key\_term\_val\_threshold$)
1. $key\_terms\_list \leftarrow \emptyset$
2. $lm_{pos} \leftarrow$ **Generate_Language_Model**(T_{pos})
3. $lm_{neg} \leftarrow$ **Generate_Language_Model**(T_{neg})
4. *for each* (t in lm_{pos})
   3.1. $term\_indicativeness = lm_{pos}(t)/lm_{neg}(t)$
   3.2. *if* ($term\_indicativeness \geqslant key\_term\_val\_threshold$)
      3.2.1 $key\_terms\_list.Add(t, term\_indicativeness)$
5. **return** key_terms_list

In order to correct for biases that may arise due to data sparseness, we apply the Dirichlet smoothing technique [68] when calculating the scores of the key terms. For example, we need to correct for bias in the case of terms that only appear in the positive set of documents, resulting in undefined values for Eq. 1. Smoothing is considered a key element in improving the performance of language models and its importance is evident by the large body of work dedicated to the subject [63,69,70].

This process can be best explained using an example. Let us consider a sentence that appears in a paragraph of a transcribed call that was tagged as negative: "*I'm always confused when I get my bill. I can't find... I have problems with things there*". In this example, our aim is to detect key terms for *negative* sentiment. The analysis process described in the previous paragraphs determined that several terms are likelier to appear when negative sentiment is expressed:

$$score(\text{“}confused\text{”}) = \frac{lm_{neg}(\text{“}confused\text{”})}{lm_{pos}(\text{“}confused\text{”})} = \frac{0.00065}{0.00041} = 1.585$$

$$score(\text{“}mybill\text{”}) = \frac{lm_{neg}(\text{“mybill”})}{lm_{pos}(\text{“mybill”})} = \frac{0.00113}{0.00048} = 2.354$$

$$score(\text{“}problems\text{”}) = \frac{lm_{neg}(\text{“problems”})}{lm_{pos}(\text{“problems”})} = \frac{0.00055}{0.00038} = 1.447$$

In this example, only the term "my bill" received a score that exceeds the threshold (2 in this example) and this is the only term that will be used as a key term. The other two terms, however, can be later considered as *context terms* for the key term. The identification of the context terms is presented in the following section.

3.1.2. Context terms identification

The context terms serve two purposes: (a) they are used to validate and assess the indicative power of the identified key terms and (b) they are used to model the relations between the identified key terms. We hypothesize that key terms that share context terms are more likely to indicate the existence of sentiment than key terms that share none. In this regard, the context terms are used for the task of Word Sense Disambiguation (WSD). The coupling of the context terms with specific key terms enables us to better determine the polarity of the term. In this sense we are in line with recent studies in this field [25,71].

To illustrate our point, consider the following example: let us assume that the term "damage" has been denoted as a key term and detected in a document during the detection phase. When the text around this term is analyzed, the presence of terms such as "big", "serious" or "lasting", all denoted as context terms during the learning phase, will lead us to assign greater weight to this term. If none of these terms appear, however, the key term is likely to be given less weight when determining the polarity of the document.

To identify context terms for each of the identified key terms, we generate a language model for each document excerpt that is associated with an identified key term in the positive and negative documents. We then compute the score of the terms that surrounds the key terms in each document excerpt in order to identify context terms, using Eq. 3 (below).

The process of identifying the context terms **for a single key term** (denoted as t_{key}) is as follows[5]:

1. Locate all instances of t_{key} both in T_{pos} and in T_{neg}. If a document contains several instances of the same key term, the following steps are applied for each instance separately.
2. For each instance found in T_{pos}, extract the terms located around t_{key} by using a sliding window of size X denoted as the *context span*. $\frac{x}{2}$ terms before and after the position of the key term are extracted. Every term in this *text excerpt*, aside from the key term itself, is considered a possible context term.
3. We denote each of the above text excerpts as a document excerpt d'. Excerpts from positive and negative documents will be denoted as d'_{pos} and d'_{neg}, accordingly. The set of all d'_{pos} will be denoted as D'_{pos} and D'_{neg} is similarly defined.
4. We generate language models for D'_{pos} and D'_{neg} and denote them $lm_{D'_{pos}}$ and $lm_{D'_{neg}}$ accordingly.
5. Next, for each "candidate" context term $t_{context}$ in $lm_{D'_{pos}}$ we calculate its score using Eq. 3:

$$score(t_{context}) = lm_{D'_{pos}}(t_{context}) - lm_{D'_{neg}}(t_{context}) \quad (3)$$

6. If the score of $t_{context}$ exceeds a predefined threshold, it will be defined as a context term for the key term t_{key}.

The algorithm for the identification of the context terms is presented in Algorithm 2. The main difference in the generation process of the context terms, in comparison to the key terms, is our use of subtraction instead of division when calculating the scores of the context terms (Eq. 3 vs. Eq. 1 that is used in Algorithm 1). We use subtraction in order to take into account the *absolute* probability of the context terms being associated with the key term, rather than considering only the *relative probability*. By doing so we are able to easily filter out infrequent terms in any context.

Recall the example presented in the previous section: consider the key term *my bill*, which has 100 occurrences in a negative context and 50 in positive. We consider two possible context terms for this key term – "*confused*" and "*problems*". *Confused*appears next to the key term in four negative contexts and one positive and *problems* appears in 40 negative, and 10 positive contexts. While the*probabilities ratios* of both terms are equal $\left(\frac{\frac{4}{100}}{\frac{1}{50}} = \frac{\frac{40}{100}}{\frac{10}{50}} = 2\right)$, it is clear that *confused* has a much weaker connection to the key term *my bill* than *problems*. By using subtraction we are able to model the fact that *problems* has a much stronger connection to

[5] Please note that we present the process only for positive sentiment, but that the process is repeated for negative sentiment as well.

the key term than *confused*: $score(confused) = \frac{4}{100} - \frac{1}{50} = 0.02$, $score(problems) = \frac{40}{100} - \frac{10}{50} = 0.2$.

Algorithm 2. Identifying the context terms in the training set.

Generate_Context_Terms(T_{pos}, T_{neg}, *key_terms_list*, *context_term_val_threshold*, *context_span*)
1. *context_terms_list* ← ∅
2. *foreach* (*kt* in key_terms_list)
   2.1. $text\_sections_{pos}$ ← **Generate_Text_Sections_Containing_KT**(*kt*, T_{pos}, *context_span*)
   2.2. $text\_sections_{neg}$ ← **Generate_Text_Sections_Containing_KT**(*kt*, T_{neg}, *context_span*)
   2.3. lm_{pos} ← **Generate_Language_Model**($text\_sections_{pos}$)
   2.4. lm_{neg} ← **Generate_Language_Model**($text\_sections_{neg}$)
   2.5. *foreach* (*t* in lm_{pos})
      2.5.1. $term_{indicativeness} = lm_{pos}(t) - lm_{neg}(t)$
      2.5.2. *if*(*term_indicativeness* ⩾ *context_term_val_threshold*)
         2.5.2.1. *context_terms_list.Add*(*t*, *term_indicativeness*)
3. **return** *context_terms_list*

The result of the two algorithms described above is a set of key terms, each affiliated with a set of context terms. The results can be visualized as a graph, as shown in Fig. 2. In the following section we describe the features that can be extracted from these terms.

3.1.3. Generating features for supervised learning

Following the completion of the two previous steps, we now possess a set of key terms and their context terms. We use these terms to generate features that are in turn utilized to train a classifier for the purpose of sentiment detection. Next we present and describe these features.

As the sentiment analysis is done at the document level, we need to generate our features for each document in the training set—both with positive and negative sentiment. This process, described in Algorithm 3, consists of two steps. First, we locate the key terms and their corresponding context terms in the document. Then, we generate our proposed features set based on the identified terms.

Algorithm 3. Locating key and context terms in each document in the training set and generating features.

Locate_Terms_And_Generate_Features(*D*, *key_terms_list*, *context_span*)
1. Foreach ($d \in D$) //for each document in the documents set
   1.1. Foreach ($kt \in key\_terms\_list$)
   1.1.1. If (!*d.Contains*(*kt*))
   //if the key term does not appear in the document
      1.1.1.1. **Continue**
   //we now update the properties of each *kt* object
   1.1.2. *kt.positions* ← **Locate_Key_Term_Positions_In_Text**(*d*,*kt*) //locate all the instances of the key term in the text and their positions
   1.1.3. *kt.context_terms* ← **Locate_Context_Terms_Positions_In_Text**(*d*, *kt.positions*,*kt.context_terms*, *context_span*)
   //locate the context terms of the analyzed key term that also appear in the text
   1.2. **Generate_Features_For_Document(D,d,key_terms_list)** //utilize the identified key and context terms to generate the features set for the document. Note that *key_terms_list* contains all the *kt* object, including those that were updated

Generating features for the analyzed document:

The features we generate for each document are of two types, based on the data they utilize: *key term-based features* and *context term-based features*. We now describe each of these types.

Key terms-based features:

This set of features is used to provide various aspects of the detected key terms: their presumed indicative value, their proximity to other key terms in the document, the number of times each of them appears in the text, etc.

- KT_Num – the number of key terms in the document.
- KTS_{max}, KTS_{avg} & KTS_{stdev} – the maximal, average and standard deviation of the scores of the key terms in the document.
- $KTNCT_{cnt}$ & $KTNCT_{perc}$ – the number of key terms without context terms, and their percentage of the total number of key terms in the document. We hypothesize that key terms that appear without relevant context provide a weaker indication of the presence of sentiment.
- KTF_{max}, KTF_{avg} & KTF_{stdev} – the max, average and standard deviation of the number of times each key term has appeared in the analyzed text. We hypothesize that multiple occurrences of the same key term provide a stronger indication of sentiment than a single appearance.
- $KTMD_{max}$, $KTMD_{avg}$ & $KTMD_{stdev}$ – for each key term, we calculate its minimal distance (in number of terms) from another key term. If the term appears more than once in the text, we check all instances and take the minimal value. Once these values have been obtained for all key terms, we extract the maximal value and calculate the average and the standard deviation over all values.

The abovementioned features are used to reflect our hypothesis that key terms that appear in close proximity to other key terms are more indicative of sentiment than those scattered throughout the text.

- $KTIW_{10}$, $KTIW_{20}$, $KTIW_{30}$ – the maximal number of key terms that can be found in the document within a sliding window of X terms. Three window sizes were used – 10, 20 and 30 words. This set of features also reflects the proximity of the key terms to one another.
- $KTSL_{max}$, $KTSL_{avg}$ & $KTSL_{stdev}$ – we denote a *sequence* as several consecutive occurrences of a key term in the document without any other key term appearing in between. For example, "[term1] …. [term1]" is a sequence of length 2, "[term1] [term1]…[term1]" is a sequence of length 3 and "[term1]…[term2].. [term1]" is not a sequence.

For each key term, we identify the length of its longest sequence and calculate the maximum, average and standard deviation of these values. These features are used to quantify a phenomenon we have observed, where a repeated use of a term indicates strong sentiment from the text's author's perspective.

- $KTLM_{avg}$ & $KTLM_{stdev}$ – the average and standard deviation of the language model values of the key terms in the document. These features assess how frequently our chosen key terms appear in the training set.
- $KTCO_{max}$, $KTCO_{avg}$ & $KTCO_{stdev}$ – for each pair of key terms in the document, we count the number of documents in the training set *T* in which they both appear (term co-occurrence). We then generate three features from these values: the maximum, average and standard deviation. Features of this kind are often used to measure the interconnection of key terms for query performance prediction [66,72].

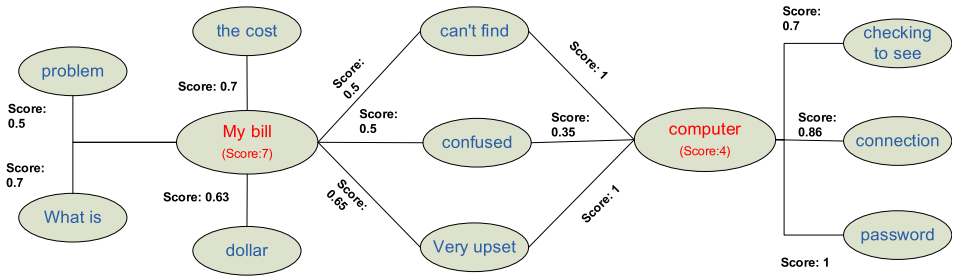

Fig. 2. A visualization of the key and context terms for negative sentiment in a transcribed call-center conversation. Red nodes indicate key terms and blue nodes indicate context terms. The scores of the key terms reflect their indicative value of the presence of negative sentiment. The scores on the edges connecting the key and context terms denote the likelihood of the context term appearing together with the key term. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- $NKTL_{max}$, $NKTL_{avg}$ & $NKTL_{stdev}$ – the maximum, average and standard deviation of the lengths of a portion of text (counted by terms), within the document where no key terms occur. These features quantify the areas in the text for which there was no indication of sentiment. Intuitively, if the analyzed text is long and the key terms appear only in a single section of the document than the likelihood of the text containing sentiment is low.
- $NKTLR_{max}$, $NKTLR_{avg}$ – these features are identical to the previous set of attributes (NKTL), but their values are normalized by the total length of the document.
- NCKT – the number of key terms which are contained within another key term. For example, the key term "know" is contained in the key term "don't know." We hypothesize that this overlap provides an indication of common meaning.
- POKT – the number of key terms in the analyzed document which have at least one shared word. For example, the key terms "don't know" and "know what" share the word "know" and are therefore overlapping.

Context terms-based features:

The context terms-based features aim to provide multiple perspectives of the context of key terms. Using this information, we are able to deal with cases where key terms appear in unrelated context or have a different meaning than those we consider indicative of sentiment. These features, to a large degree, differentiate our approach from rule-based and fingerprint-based [57,73,74] approaches which simply scan for key terms.

- CTN_{max}, CTN_{avg} & CTN_{stdev} – the maximal, average and standard deviation of the number of the context terms per key term in the analyzed document.
- CTP_{max}, CTP_{avg} & CTP_{stdev} – for each key term detected in the text, we compute the percentage of the context terms that were found in the analyzed text out of all those that were defined for it in earlier stages of the learning phase. For example, if a key term has five possible context terms and two were found in the text then the percentage of detected terms would be 40%.

These features enable us to differentiate between context terms that have multiple context terms and those that have few. If a key term has only three detected context terms but these terms are *all* its possible contexts, then this key term may be considered highly indicative.

- CTS_{max} CTS_{avg} &CTS_{stdev}– the maximal, average and standard deviation of the *average scores of the context terms* found in the document, per key term. These features are used to quantify the overall indicative value of the context.
- $CTSR_{max}$, $CTSR_{avg}$ & $CTSR_{stdev}$ – for each key term, we calculate the average score of its context terms that appear in the text. Next we divide it by the average score of all of its context terms (those that were identified for the key term in Section 3.1.2). We then extract the maximal value, the average and the standard deviation of these values.

These features attempt to measure, for each key term, whether its "stronger" or "weaker" contexts were detected in the analyzed text.

- CTF_{max}, CTF_{avg} & CTF_{stdev} – the max, average and standard deviation of the times each context term has appeared in the document (the values are first calculated on average for each key term). These features are used to determine whether the context of the key terms is diverse, i.e. multiple context terms appearing a small number of times or limited and repetitive.
- CKTD & CKTT – the number of key terms which shared *at least one* context term in the analyzed document and in the entire training set, respectively. We hypothesize that key terms that share context terms are more indicative than key terms that do not. These features enable us to quantify this measure.
- $SCTD_{max}$, $SCTD_{min}$, $SCTD_{avg}$ & $SCTD_{stdev}$ – for every pair of key terms, we count the number of their shared context terms detected *in the analyzed document*. We then extract the min and max values and calculate the average and standard deviation over all values. Our hypothesis, measured in these features, is that key terms that share multiple contexts are more strongly connected than key terms that share few terms or none at all.
- $SCTT_{max}$, $SCTT_{min}$, $SCTT_{avg}$ & $SCTT_{stdev}$ – for every pair of key terms found in the document, we count the number of their shared context terms out of all the possible context terms found for them in Section 3.1.2. We then extract the min and max values and calculate the average and standard deviation over all values.

These features are used to compare the interconnectedness of the key terms *in the analyzed document* (through their context terms) and in the entire documents corpus.

- $CTLM_{avg}$ & $CTLM_{stdev}$ – the average and standard deviation of the language model values of the context terms detected in the document.

In addition to the two features groups mentioned above we generate one meta-feature: TL – the overall length (in characters) of the analyzed document's text.

3.2. Enhancing the model with non-textual (meta) features

In this section we describe the reasons and advantages of incorporating additional features into the set of features presented

above. Then, we present a novel set of non-textual features designed specifically for the domain of transcribed text analysis.

3.2.1. Additional features and the curse of dimensionality

A major drawback of BOW-based solutions is the difficulty of integrating additional information into the model because of the large number of features it utilizes. The number of these features is measured in the thousands (one for each unique term in the text) and classifiers find it difficult to utilize them in an efficient manner and may also cause overfitting. This problem is known as the *curse of dimensionality* [75–77]. In addition, avoiding overfitting is also a well-known problem. The large number of features in BOW approaches also narrows the selection of possible classifiers, ruling out popular algorithms such as the C4.5 decision trees algorithm [78], which are incapable of efficiently dealing with a feature set of this magnitude.

Our model employs a much smaller set of advanced features, a fact that makes it less affected by the curse of dimensionality. For this reason, it is possible to add the following types of features to the model in order to improve its performance:

(a) **The incorporation of documents metadata** – usually, the metadata consists of a small set of useful non-textual features, providing additional information about the document. The curse of dimensionality can sometimes prevent BOW-based approached from using the full potential of this information (as shown in our experiments). Our proposed model, on the other hand, shows considerable improvement when these features are added.

(b) **Analyzing several text segments simultaneously** – as shown by our experiments on transcribed call-center conversations, calculating two sets of features—one for the text of the analyzed paragraph, and one for the text of the paragraph *combined with the text of its surroundings paragraphs*—and combining the two sets into a single features vector, boosted the performance of our model (please see section 4.4.1 for full experimental details). The baseline classifiers, however, did not benefit from the additional data as it requires adding thousands of additional features.

The ability to easily integrate additional features into our model, as well as the ability to employ a large number of classifiers (including those not usually applicable for text classification) is one of the major advantages of our method. This is important in domains where the text itself is merely one of several facets of the data, with transcribed text being one example.

3.2.2. Meta-features for transcribed text

In order to evaluate the possible benefits of incorporating additional features into our features set, we generated a set of features for our (noisy) dataset of transcribed phone calls and added them to the text-based features. The reason for choosing this dataset is that unlike written language, a great deal of sentiment is expressed by intonation, volume and speed of speech. These are all aspects that cannot be captured by text alone.

To the best of our knowledge, only Gates and Park [26] generated non-textual features for transcribed text. Their work focused on calculating features only at the *call level*. We build upon Park and Gates' work but generate a large set of non-textual features *both for the call and the utterance level*. We are interested in identifying the exact issues that trouble customers and therefore we need more fine-grained results that can be provided through the utterance level; but conversely, some utterances are very short and therefore difficult to classify. In those cases, information on the general course of the call can assist the classification process.

The next section presents the features used in our experiments, where features that were presented earlier by Gates and Park are marked with an asterisk.

Call-level features:

- *NW* – the number of words in the call.
- *NU* – the number of utterances in the call.
- *CD* – the time duration of the call.
- *AWC* – the number of words spoken by the agent throughout the call.
- *CWC*– the number of words spoken by the customer throughout the call.
- *AAWU* – the average number of words spoken by the agent in an utterance.
- *SAWU*– the standard deviation of the number of words spoken by the agent in an utterance.
- *ACWU* – the average number of words spoken by the customer in an utterance.
- *SCWU* – the standard deviation of the number of words spoken by the customer in an utterance.
- *AADU*– the average duration of an agent's utterance.
- *SADU* – the standard deviation of the duration of an agent's utterance.
- *ACDU* – the average duration of a customer's utterance.
- *SCDU* –the standard deviation of the duration of a customer's utterance.
- *WR** – the ratio between the number of words spoken by the customer and those spoken by the agent.
- *DR** – the ratio between the sum of the durations of the customer's utterances and that of the agent's. This feature was first proposed by Park and Gates [26].

These features are all designed to model the interaction between the customer and the agent. Park and Gates [26] suggested that dissatisfied customers tend to "dominate" the conversation. We have attempted to take this approach a step further by developing a set of features dedicated to this issue. For example, one of our features is the total duration of the conversation. We assume that a longer conversation means that the customer has not been able to resolve his problem quickly, something that could cause the appearance of negative sentiment.

Utterance-level features:

- *UP* – the position of the utterance in the conversation (how many utterances preceded it).
- *FWP* – the offset (number of words) from the beginning of the conversation to the first word of the utterance.
- *UNW* – the number of words in the utterance.
- *UD* – the duration of the utterance.
- *UWPS* – the number of words-per second in the utterance.
- *NWR* – the number of word repetitions in the utterance after the stemming of the text and the removal of stop words (i.e. very common words). We observed that agitated and upset users tend to repeat themselves during the same utterance. This feature is an attempt to model this phenomenon.
- *MNR* – the maximal number of times a word was repeated in the utterance.

All the features presented in this section attempt to generate different aspects of the data that may be indicative of negative sentiment, including long conversations, the customer restating the problem repeatedly, extremely long utterances which may indicate that the customer is relying the "history" of her problems, etc. As shown by our experiments in the following section, these features are very effective in detecting negative sentiment.

3.3. The detection phase

The purpose of the detection phase is to utilize a learning-based classifier (in our experiments the RotationForest [79] algorithm) to assign a score to new documents. This score indicates the classifier's level of certainty that the text of document contains the type of sentiment that we are interested in detecting.

The detection process takes place as follows: first, the text of the newly analyzed document is scanned for the key terms as presented in Section 3.1.1. Secondly, for each detected key term, we scan the text around it in search of its context terms, as described in Section 3.1.2. Finally, using the key and context term we generate the features presented in Section 3.1.3 and use a classifier to produce a score for the document. The detection process is presented in full in Algorithm 4.

Algorithm 4. Locating key and context terms in a document during the detection phase and calculating its score.

Calculate_Document_Score(d, *key_terms_list*, *context_terms_list*, *context_span*, *classifier*)
1. Foreach (*kt* ∈ *key_terms_list*)
   1.1. If (!*d.Contains*(*kt*)) //if the key term does not appear in the document
      1.1.1 **Continue**
   //we now update the properties of each *kt* object
   1.2. *kt.positions* ← **Locate_Key_Term_Positions_In_Text**(*d*, *kt*)
   1.3. *kt.context_terms* ← **Locate_Context_Terms_Positions_In_Text**(*d*, *kt.positions*, *kt.context_terms*, *context_span*)
2. *document_features* ← **Generate_Features_For_Document**(*d*, *key_terms_list*) //Note that *key_terms_list* contains all the *kt* object, including those that were updated
3. *document_score* ← **Classify_Document**(*classifier*, *document_features*) //here we call the machine learning-based classifier that utilizes the features generated for the analyzed document in step 2 in order to determine its score
4. *returndocument_score*

4. Evaluation

In order to conduct a comprehensive evaluation of our method, we used several types of datasets from different domains. To evaluate our method on text with little or no noise we use hotel reviews from TripAdvisor and movie reviews from the Internet Movie Database (IMDB). For evaluation on the more difficult task of noisy and short texts we use data from Twitter and automatically-transcribed phone calls made to the call centers of two large telecommunication companies.

It is also important to note that the evaluated datasets vary greatly in the length of the analyzed text. While the documents in the Twitter dataset are of course the shortest (~140 characters), the average length of a document in the Movie Reviews dataset (the one with the longest documents) is approximately 750 words – well over a standard page. This diversity enables us to evaluate the robustness of our proposed approach in multiple experimental settings. The average length of the documents are presented for each dataset in its dedicated section.

We compare the performance of our method with three well-known benchmarks: the first two – Support Vector Machines (SVM) [80] and Naïve Bayes [81] – are both applied on unigram term vectors extracted from the analyzed text. These algorithms are widely used for text classification and are reported to perform well in sentiment analysis tasks [8]. The third benchmark is the Delta-TFIDF approach [82], a popular term weighting approach that has been shown to outperform the standard TFIDF approach on some datasets. As described in [82], we used a bag of unigram and bigram terms for the modeling of the text.

We run these algorithms using Weka [83], the open source data mining software. Both algorithms were run using their default configurations. All our experiments were run using 5-fold cross validation. All experiments utilizing SVM were carried out using the LibSVM algorithm with a linear kernel.

In Section 4.1 we present the evaluation measures we chose for our experiments. In Sections 4.2 and 4.3 we show results from running our experiments on the TripAdvisor and movie reviews datasets, and in Sections 4.4 and 4.5 we evaluate ConSent on the more challenging datasets of transcribed call-center conversations and Twitter.

4.1. Evaluation measures

We use three measures in order to evaluate the performance of the algorithms used in this study:

(a) Accuracy – this measure presents the overall percentage of correctly classified instances, regardless of their type. This measure is the one used to evaluate the performance of sentiment analysis.
(b) Area under the curve (AUC) – this measure calculates the area under the Receiver Operating Characteristic (ROC) curve [84].
(c) Precision/recall curves – *precision* is defined as the percentage of relevant items out of the "top X" items retrieved by the algorithm, while *recall* is the percentage of retrieved relevant items of all the relevant items in the dataset [85].
(d) F-Measure – this frequently-used measure enables factoring both precision and recall into a single value, thus representing their harmonic mean [86]. The measure is calculated using the following formula: $= \frac{(1+\beta^2)\times recall\times precision}{(\beta^2\times precision)+recall}$, where β can be used to give greater weight in the score to either the precision or the recall. In our experiments we use $\beta = 1$, thus giving equal weight to the two measures.

Although accuracy is the most commonly-used measure for sentiment analysis tasks, we maintain that there is good reason to consider the other measures as well. The reason for this is the issue of dataset imbalance.

In many cases, sentiment analysis datasets are inflicted by imbalances in class distribution, where the number of instances belonging to one class significantly outnumber those that belong to another. For example, the TripAdvisor dataset used by Wang et al. [36] contains a significantly larger number of positively rated reviews (rated 4–5 stars out of 5) than negatively rated (rated 1–2 stars out of 5)—a ratio of approximately 5:1. In our transcribed calls datasets the imbalance ratio is even greater and reaches to approximately ten positive instances for every instance containing negative sentiment.

The Receiver Operating Characteristic (ROC) curve [84] is a graph produced by plotting the true-positive rate versus the false-positives rate over all possible acceptance thresholds. The area under this curve – the AUC – evaluates both classes simultaneously and at various decision points. Therefore, the classifier's performance is not overwhelmed by the majority class [87]. For this reason, we believe the AUC to be a much more suitable evaluation measure than accuracy for our experiments.

The same rationale applies for precision/recall curves and the F-Measure, as they enable to evaluate our performance on the top X

items, sorted by their probability of belonging to a certain class. This measure is often used in information retrieval when measuring the performance of a query; as the user is not likely to review thousands of documents, the top ranking documents are more important and thus receive greater weight. This measure enables us to analyze the performance of the evaluated algorithm from multiple perspective. This measure is particularly useful for real-world applications, where often only the top X items are used in the evaluation process.

4.2. TripAdvisor reviews

TripAdvisor is a travel services website which enables users to upload reviews of travel-related businesses (hotels, airlines, etc.). When uploading a review, it must be assigned a score ranging from one star (poor) to five stars (excellent). These reviews are available for download and their existing scores makes them very suitable for the task of sentiment analysis.

In our experiments we used 30,000 hotel reviews which were randomly sampled from the benchmark TripAdvisor dataset [36]. A sampled review with 1–2 stars was tagged as having negative sentiment, while reviews with 4–5 stars were tagged as having positive sentiment. Reviews with three stars were discarded.

The statistics on the dataset we created are presented in Table 1. It is clear that the positive sentiment reviews outnumber the negative ones with a ratio of approximately five-to-one (24,930 positive reviews vs. 5070 negative ones).

Table 1
Statistics on the TripAdvisor dataset.

	Number of reviews	Average review length (words)
Positive reviews	24,930	178
Negative reviews	5070	239

Table 2
Accuracy and AUC results for the TripAdvisor dataset. N,S and D mark a statistically significant difference from the performance of Naïve Bayes SVM and Delta-TFIDF, respectively, with ($p < 0.05$). Boldface marks the best result in the row.

	ConSent	SVM	Naïve Bayes	Delta-TFIDF
Accuracy	**0.9475**^{NSD}	0.94^{N}	0.798	0.904^{N}
AUC	**0.9677**^{NSD}	0.9587ND	0.8277	0.902^{N}
F-Measure	**0.8366**^{NSD}	0.8135ND	0.5612	0.6644^{N}

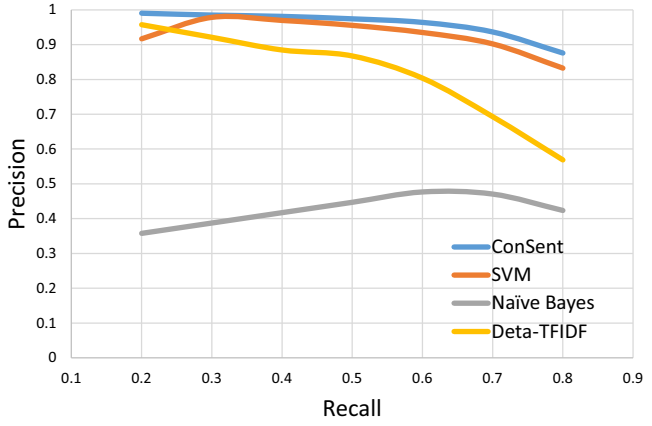

Fig. 3. The precision/recall curves for the negative sentiment in the TripAdvisor dataset.

Table 3
Statistics on the movie reviews dataset.

	Number of reviews	Average review length
Positive reviews	1000	789
Negative reviews	1000	707

Table 4
Accuracy and AUC results for the movie reviews dataset. N,S and D mark statistically significant difference from the performance of Naïve Bayes SVM and Delta-TFIDF, respectively, with ($p < 0.05$). Boldface marks the best result in the row.

	ConSent	SVM	Naïve Bayes	Delta-TFIDF
Accuracy	0.769^{D}	0.7805^{D}	**0.781**^{D}	0.73
AUC	**0.8577**^{SND}	0.7943	0.8477^{S}	.806
F-Measure	0.7795	0.7807	**0.7934**^{D}	0.7253

The results of our evaluation on this dataset are presented in Table 2 and in Fig. 3. The evaluation clearly shows that although all evaluated methods perform well, ConSent outperforms all other methods. Moreover, the improvement is statistically significant *both* in the accuracy and in the AUC measures ($p < 0.01$, using a paired t-test). The precision/recall curves, calculated for the minority class (negative sentiment) also show a significant improvement.

4.3. Movie reviews

We used the polarity dataset presented by Pang and Lee [88]. The dataset, which was extracted from the Internet Movie Database (IMDB), contains 1000 positive and 1000 negative movie reviews, and is available online[6] (polarity dataset v2.0). The statistics on this dataset are presented in Table 3. This dataset provides a very different experimental setting than the TripAdvisor dataset, as it is both small and balanced – a 1:1 ratio of positive and negative classes.

We once again compare the performance of our approach to those of the three baselines. The results for the accuracy and AUC measures are presented in Table 4 and the precision/recall curves are presented in Fig. 4. Although ConSent significantly outperforms the baseline methods when using the AUC measure ($p < 0.01$ using a paired t-test), there is no significant difference in performance when measuring accuracy. The reason for this apparent difference is the use of the "default" confidence threshold, which does not yield the optimal accuracy measure for our classifier (this issue was discussed at length in Section 4.1). This point is further illustrated by the precision/recall curves shown in Fig. 4, where our proposed method clearly outperforms the baseline methods.

4.4. Automatically transcribed phone conversations

We used two datasets of transcribed call-center conversations, obtained from large telecommunication corporations. The topics of the conversations cover a span of topics ranging from billing issues to technical issues to additional services. The data is divided into *utterances* which are uninterrupted sequences of words by the same speaker. In addition to the (transcribed) text of each utterance, our data also contains the identity of the speaker (customer or agent) and the duration of the utterance.

In order to create our training set, two human taggers were employed to read the conversations in the two datasets and identify utterances containing negative sentiment. Each conversation was reviewed by both taggers, and utterances in which there was disagreement were discarded. Overall, the taggers were in agreement regarding over 90% of the utterances in the two

[6] http://www.cs.cornell.edu/people/pabo/movie-review-data/.

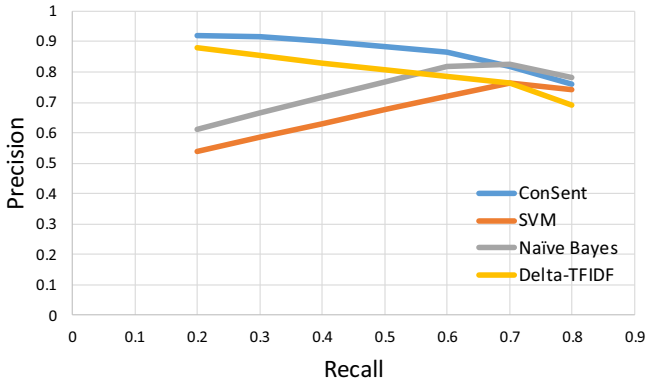

Fig. 4. The precision/recall curves for negative sentiment in the movie reviews dataset.

Table 5
Statistics on the two transcribed calls datasets.

	Tele1	Tele2
Number of calls	377	113
Number of utterances	20,338	15,182
Average number of utterances per call	53.94	134.35
Number of negative utterances	1178 (5.8%)	2336 (15.4%)
Average utterance length (words)	14.91	17.11

datasets. The statistics describing our data are presented in Table 5. The imbalance in this dataset is greater than in our TripAdvisor experiments – approximately 10 positive utterances for every negative one.

We performed three sets of runs: in the first set of experiments, we only used features that could be extracted from the text of the transcribed calls. In the second set of experiments, we incorporated the meta-features presented in Section 3.2.2 into the features sets of all algorithms and measured their improvement in performance. Finally, in the third set of experiments we evaluated the robustness of the different algorithms by training them on one transcribed calls dataset and evaluating them on the other.

4.4.1. Evaluation on transcribed text

In this set of experiments we only utilize features that can be extracted from the text of transcribed calls, without any use of the metadata presented in Section 3.2.2. We conduct two experiments in this set: one in which we only analyze the text of the utterance and another in which we *also analyze its surrounding utterances*, as we explained in Section 3.2.1.

Our rationale for extracting features from the surrounding utterances is as follows: as some utterances are short, we hypothesized that they would be difficult to analyze without taking into account additional context. For this reason we created two sets of features: one for the text of the analyzed utterance, and a second one for the text of the analyzed utterance and its surrounding utterances. The number of additional utterances was three in either direction—before and after the analyzed utterance. When the number of available utterances was smaller (for example, in the case of the first utterance in the conversation) we only used the available utterances.

Although using the surrounding utterances provides additional information for the classification process, it comes at the cost of doubling the number of features. We hypothesized that ConSent, which uses a small number of features and is therefore less susceptible to the curse of dimensionality, would be better positioned to utilize this additional information.

Table 6
Accuracy and AUC results for the Tele1 and Tele2 dataset. N, S and D mark statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively, while C marks statistically significant difference from ConSent. Boldface marks the best result in the row.

	ConSent	SVM	Naïve Bayes	Delta-TFIDF
Tele1				
Accuracy	**0.9321**^{N}	0.9273^{N}	0.8576	0.9285^{N}
AUC	**0.8789**^{D}	0.8679	0.8772	0.858
F-Measure	0.5208	**0.5529**ND	0.4834	0.4555
Tele2				
Accuracy	0.8974^{N}	**0.9136**^{CN}	0.8353	0.9065
AUC	0.9363^{N}	**0.9416**ND	0.8971	0.9142
F-Measure	0.7329	**0.7813**^{N}	0.6679	0.7264

The results of our evaluation on single utterances are presented in Table 6 and Figs. 5 and 6. The results of our evaluation when the surrounding utterances are also used are presented in Table 7 and Figs. 7 and 8. In Table 6 we see that on Tele1, ConSent outperforms the three baseline methods, although the improvement is not always statistically significant. On Tele2 we see that SVM fares best, sometimes to a statistically significant degree over the other three methods. We suspect that the reason for ConSent's relatively weaker performance stems from the smaller size of the dataset, as our proposed method requires a large enough set of documents in order to correctly infer the context.

The results, however, change considerably with the addition of the features extracted from the surrounding utterances. While the performance of ConSent improves as a result of the additional features, both SVM and Naïve Bayes show a significant decline in performance. The cause for this decline in performance is undoubtedly the curse of dimensionality—as the other methods utilize a TF vector containing thousands of terms, adding thousands of additional features makes the feature selection process far more difficult.

Finally, we compare the performance of ConSent *with and without* the additional features, to the performance of the SVM and Naïve Bayes baselines *without* the additional features. The results, which are presented in Table 8, show that the version of ConSent that utilizes both the features extracted from the utterance and its surrounding utterances fares better than all other methods on both datasets. Moreover, the improvement is statistically significant ($p < 0.05$) in the majority of measured. These results are also reflected in Figs. 9 and 10, which show the precision/recall curves for Tele1 and Tele2. It is clear that in Tele1 ConSent outperforms all other algorithms to a large degree. In Tele2, ConSent fares best in

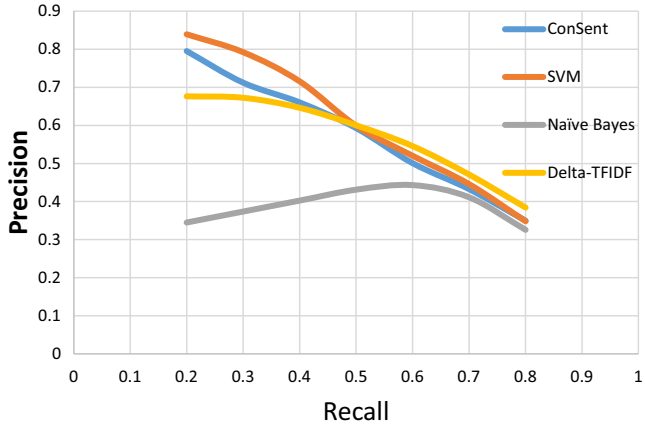

Fig. 5. The precision/recall curves for negative sentiment detection in the Tele1 dataset when using only text-based features extracted from the analyzed utterance.

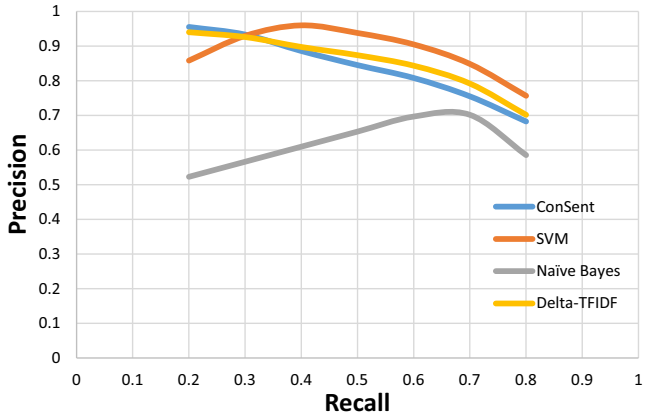

Fig. 6. The precision/recall curves for negative sentiment detection in the Tele2 dataset when using only text-based features extracted from the analyzed utterance.

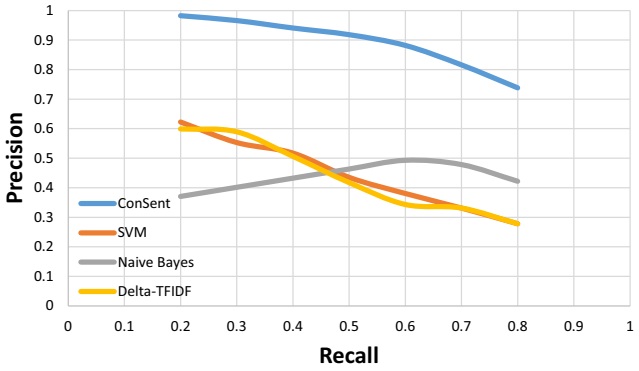

Fig. 8. The precision/recall curves for the negative sentiment class in the Tele2 dataset when using text-based features extracted *both* from the analyzed utterance and its surrounding utterances.

Table 7
The accuracy and AUC results for the transcribed calls dataset, when using *both* the text-based features of the utterance *and* its surrounding utterances. N, S and D mark statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively.

	ConSent	SVM	Naïve Bayes	Delta-TFIDF
Tele1				
Accuracy	**0.9395**^{SND}	0.8828^{N}	0.7925	0.8955^{N}
AUC	**0.9274**^{SND}	0.6891	0.7783^{SD}	0.7179^{S}
F-Measure	**0.5553**^{SND}	0.2897	0.3611^{SD}	0.2961
Tele2				
Accuracy	**0.9115**^{SND}	0.8068^{N}	0.7750	0.8047^{N}
AUC	**0.9492**^{SND}	0.7219	0.8005^{SD}	0.7335
F-Measure	**0.7639**^{SND}	0.4437	0.5701^{SD}	0.4888^{S}

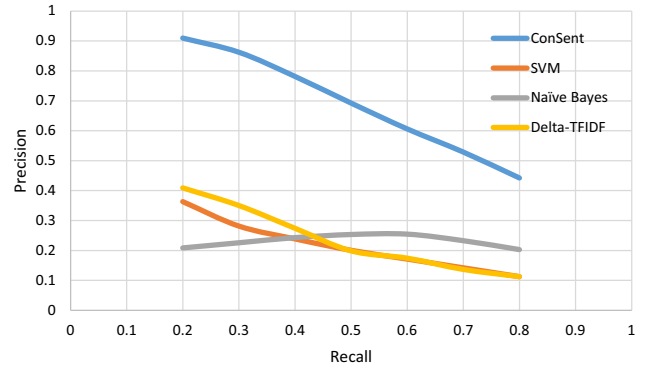

Fig. 7. The precision/recall curves for the negative sentiment class in the Tele1 dataset when using text-based features extracted *both* from the analyzed utterance and its surrounding utterances.

scenarios where high levels of precision are required but trails behind slightly in scenarios of higher recall.

4.4.2. Evaluation of transcribed text and metadata features

We now repeat the experiments of the previous section while also adding the meta-features described in Section 3.2.2 to the text-based features extracted from the utterance. Because using the surrounding utterances proved beneficial only for ConSent, we also present the results for adding both the meta-features and the features obtained from the surrounding utterances to the original features set.

The accuracy, AUC and F-Measure of all the algorithms are presented in Table 9 and the precision/recall curves are presented in Figs. 11 and 12. It is clear that although all algorithms benefit from the integration of the meta-features, ConSent fared best. In addition, we can see that ConSent is capable of effectively utilizing both the meta-features and the features extracted from the surrounding utterances.

Table 8
Accuracy and AUC results for the Tele1 and Tele2 dataset where both versions of ConSent (with and without features from surrounding utterances) are evaluates. N, S and D mark statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively, while C marks a statistically significant difference from ConSent without the surrounding utterances. Boldface marks the best result in the row.

	ConSent	ConSent and surrounding utterances	SVM	Naïve Bayes	Delta-TFIDF
Tele1					
Accuracy	0.9321^{N}	**0.9395**^{CSND}	0.9273^{N}	0.8576	0.9285^{N}
AUC	0.8789	**0.9274**^{CSND}	0.8679	0.8772	0.858
F-Measure	0.5208ND	**0.5553**^{CND}	0.5529ND	0.4834	0.4555
Tele2					
Accuracy	0.8924^{N}	0.9115^{CN}	**0.9136**^{CN}	0.8353	0.9065
AUC	0.9363ND	**0.9492**^{CSND}	0.9416^{N}	0.8971	0.9142^{N}
F-Measure	0.7329^{N}	0.7639^{CND}	**0.7813**^{CND}	0.6679	0.7264^{N}

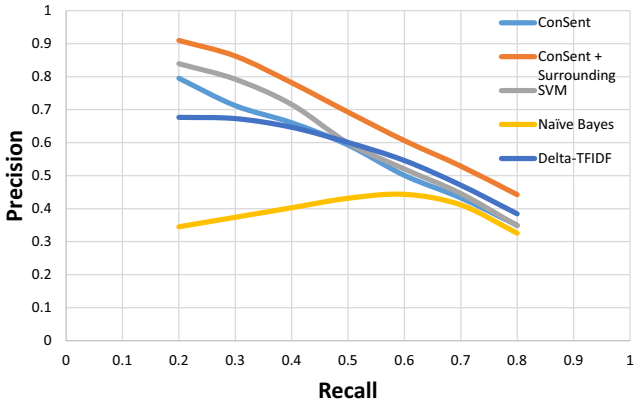

Fig. 9. The precision/recall curves for the negative sentiment class in the Tele1 dataset. The figure presents a comparison of the two versions of ConSent (with and without features from the surrounding utterances) and the three baseline approaches applies to features extracted only from the text of the utterance.

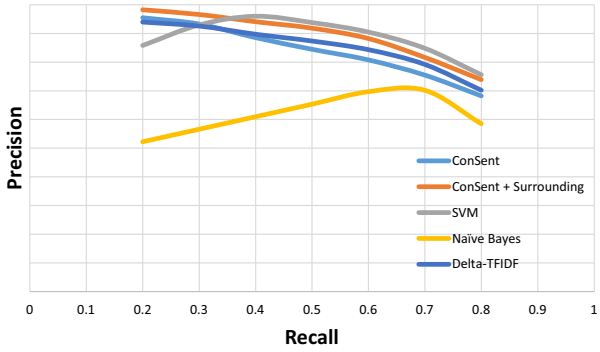

Fig. 10. The precision/recall curves for the negative sentiment class in the Tele2 dataset. The figure presents a comparison of the two versions of ConSent (with and without features from the surrounding utterances) and the three baseline approaches applies to features extracted only from the text of the utterance.

Table 9
Accuracy and AUC results for the Tele1 and Tele2 dataset where text-based features and meta-features are used in conjunction. N, S and D mark a statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively, while C marks a statistically significant difference from ConSent without the surrounding utterances. Boldface marks the best result in the row.

	ConSent	ConSent and surrounding utterances	SVM	Naïve Bayes	Delta-TFIDF
Tele1					
Accuracy	0.9362^{SN}	$\mathbf{0.9413^{CSND}}$	0.9276^{N}	0.8780	0.9282^{N}
AUC	0.9140^{SND}	$\mathbf{0.9273^{CSND}}$	0.8990^{N}	0.8178	0.8898^{N}
F-Measure	0.5208	0.5454	$\mathbf{0.5529^{CND}}$	0.4834	0.4555
Tele2					
Accuracy	0.8961^{N}	0.9063^{CN}	$\mathbf{0.9142^{CN}}$	0.8464	0.9282^{CN}
AUC	0.9376^{ND}	$\mathbf{0.9483^{CND}}$	0.9463^{CND}	0.8652	.8898
F-Measure	0.7385^{N}	0.7527^{CND}	$\mathbf{0.7835^{CND}}$	0.6651	0.7459^{N}

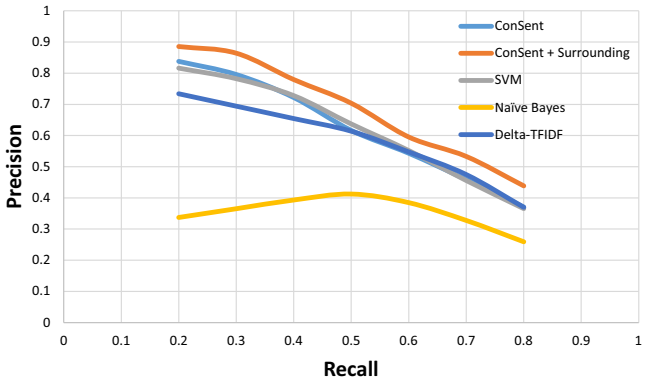

Fig. 11. The precision/recall curves for the negative sentiment class in the Tele1 dataset. The figure presents a comparison of the two versions of ConSent (with and without features from the surrounding utterances) and the two baseline approaches applied both on features extracted from the text of the utterance and the meta-features.

4.4.3. Cross-corpus evaluation

The creation of a tagged datasets of transcribed calls is a cumbersome task, as it requires a human tagger to listen to hundreds of conversations. For this reason, we were interested in evaluating the robustness of ConSent (and that of the baselines) by training a classifier on one dataset and using it to classify another. Therefore, we conducted two experiments: in the first we trained our classifier on Tele1 and evaluated the resulting classification model on Tele2. In the second experiment, the roles were reversed.

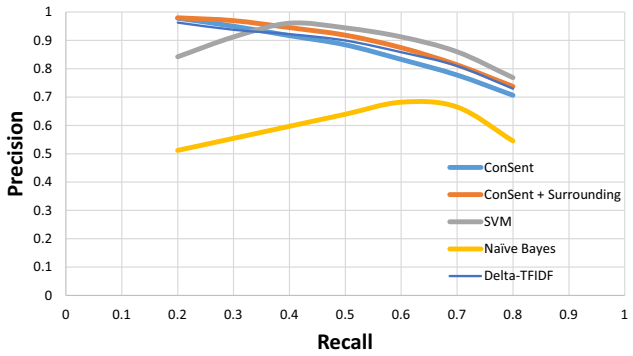

Fig. 12. The precision/recall curves for the negative sentiment class in the Tele2 dataset. The figure presents a comparison of the two versions of ConSent (with and without features from the surrounding utterances) and the two baseline approaches applied both on features extracted from the text of the utterance and the meta-features.

Both experiments were conducted as follows: the training set in its entirety was used to train a classifier which was then used to evaluate the test set. The test set was divided into five folds, in order to enable testing the results for statistical significance. The results of the evaluation with and without the mete-features are presented in Tables 10 and 11. As in the previous sections, we included the results for ConSent with and without the features extracted from the surrounding utterances. The precision/recall curves for this experiment are presented Figs. 13 and 14.

The results clearly show that the robustness of our proposed approach exceeds that of the three baseline approaches. We were able to reach a statistically significant improvement both in all measures.

4.5. Twitter

For our evaluation we used Sentiment140 dataset,[7] which contains 1,600,000 automatically tagged tweets (half positive and half negative) that serve as the training set and 359 manually tagged tweets that serve as the test set. Similarly to our evaluation on the transcribed calls datasets presented in Section 4.4, we present two sets of experiments: one set is run on features that are generated using the text of the tweet and the other combines the abovementioned features with meta-features specific to the Twitter domain.

The meta-features that were generated for our experiments were created based on those used in several studies that found such features to be useful [55,89,90]. The full list of features is presented below (**please note:** we define the end of a tweet as the last ten characters):

(1) *UC* – whether the tweet contains upper case letters.
(2) *RC* – whether the tweet contains a sequence of repeated characters. We define a sequence to be of a length of at least three characters (for example: "sooo good").
(3) *PE* – whether the tweet contains positive emoticons.
(4) *NE* – whether the tweet contains negative emoticons.
(5) *PEAE* – whether the tweet contains a positive emoticon at its end.
(6) *NEAE* – whether the tweet contains a negative emoticon at its end.
(7) *HTC* – the number of hashtags (#) in the tweet.
(8) *EMC* – the number of exclamation marks in the tweet.
(9) *EEM* – whether the tweet contains an exclamation mark at its end.

[7] http://help.sentiment140.com/home.

Table 10
Accuracy and AUC results for the Tele2 dataset where Tele1 is used as the training set. Both versions of ConSent (with and without features from surrounding utterances) are evaluates. *N*, *S* and *D* mark a statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively, while *C* marks a statistically significant difference from ConSent without the surrounding utterances. Boldface marks the best result in the row.

	ConSent	ConSent and surrounding utterances	SVM	Naïve Bayes	Delta-TFIDF
Accuracy					
Without meta-features	0.8624^{ND}	0.8771^{NCD}	$\mathbf{0.8782^{NCD}}$	0.8529	0.8535
With meta-features	0.8687^{N}	0.8771^{NC}	$\mathbf{0.8856^{NC}}$	0.7031	0.8633^{N}
AUC					
Without meta-features	0.8966^{SN}	$\mathbf{0.9219^{CSND}}$	0.8759	0.8804^{S}	0.8973^{S}
With meta-features	0.9023^{SN}	$\mathbf{0.9217^{CSND}}$	0.8844^{N}	0.7867	0.9004^{SN}
F-Measure					
Without meta-features	0.5268^{ND}	0.5914^{CND}	$\mathbf{0.6156^{CSND}}$	0.4987	0.4587
With meta-features	0.5763^{ND}	0.5966^{CND}	$\mathbf{0.6235^{CND}}$	0.5186^{D}	0.4877

Table 11
Accuracy and AUC results for the Tele1 dataset where Tele2 is used as the training set. Both versions of ConSent (with and without features from surrounding utterances) are evaluates. *N*, *S* and *D* mark statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes, SVM and Delta-TFIDF, respectively, while *C* marks a statistically significant difference from ConSent without the surrounding utterances. Boldface marks the best result in the row.

	ConSent	ConSent and surrounding utterances	SVM	Naïve Bayes	Delta-TFIDF
Accuracy					
Without meta-features	0.9021^{N}	$\mathbf{0.9295^{CNSD}}$	0.9041^{N}	0.8377	0.9061
With meta-features	0.9078^{N}	$\mathbf{0.9281^{CNSD}}$	0.9042^{N}	0.8469	0.907^{N}
AUC					
Without meta-features	0.9148^{N}	$\mathbf{0.9345^{CNSD}}$	0.9170^{N}	0.8983	0.9034
With meta-features	0.9148^{N}	$\mathbf{0.9328^{CNS}}$	0.9222^{N}	0.8555	0.9288^{N}
F-Measure					
Without meta-features	0.5455^{N}	$\mathbf{0.6027^{CSND}}$	0.5705	0.4672	0.54126^{N}
With meta-features	0.5548^{N}	$\mathbf{0.598^{CSND}}$	0.5745^{N}	0.4596	0.5672^{N}

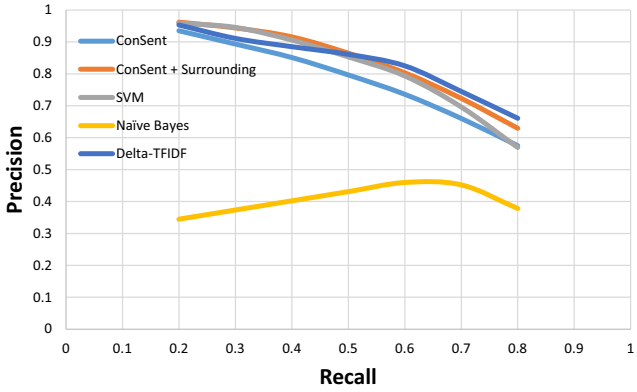

Fig. 13. The precision/recall curves for the negative sentiment class in the Tele2 dataset when the training is conducted on the Tele1 dataset. The set of features contains both the text-based features as well as the meta-features.

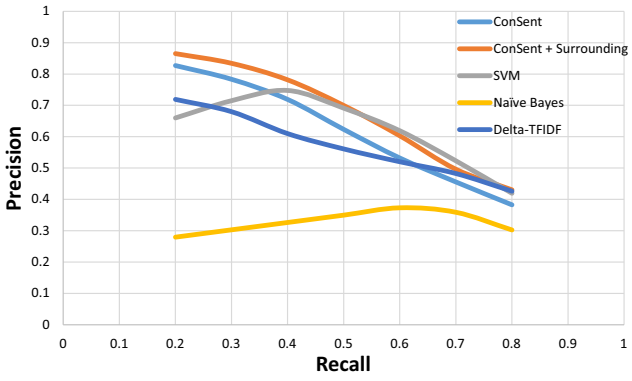

Fig. 14. The precision/recall curves for the negative sentiment class in the Tele1 dataset when the training is conducted on the Tele2 dataset. The set of features contains both the text-based features as well as the meta-features.

(10) *MC* – the number of exclamation points (!) and question marks (?) in the tweet.
(11) *CNW* – the number of negation words in the tweet. Negation words are terms and phrases such as "can't," "won't", and "never".
(12) *CQW* – the number of question words in the tweet. Question words are terms such as "who," "why" and "what."
(13) *TL* – whether the tweet contains a link.
(14) *WUC* – the number of words that start with an uppercase letter.
(15) *UCLC* – the number of uppercase letters in the tweet.
(16) *PET* – the number of terms indicating positive emotion found in the text. For this feature we used the lexicon presented in [17].
(17) *NET* – the number of terms indicating negative emotion found in the text.
(18) *PEH* – the number of hashtags indicating positive emotion that were found at the end of the tweet.
(19) *NEH* – the number of hashtags indicating negative emotion that were found at the end of the tweet.

The results of our evaluation are presented in Table 12. The results clearly indicate that all methods outperform Naïve Bayes to a statistically significant degree. In addition, the SVM and Delta-TFIDF baselines obtain the best results overall, but its improvement over ConSent is only statistically significant for the accuracy measure and not for AUC. Conversely, as shown by the precision/recall curves in Figs. 15 and 16, in certain recall points ConSent outperforms SVM by a wide margin.

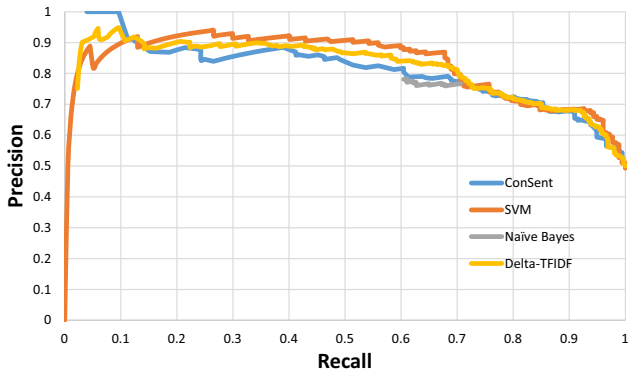

Fig. 15. The precision/recall curves for the Twitter dataset when using only text-based features.

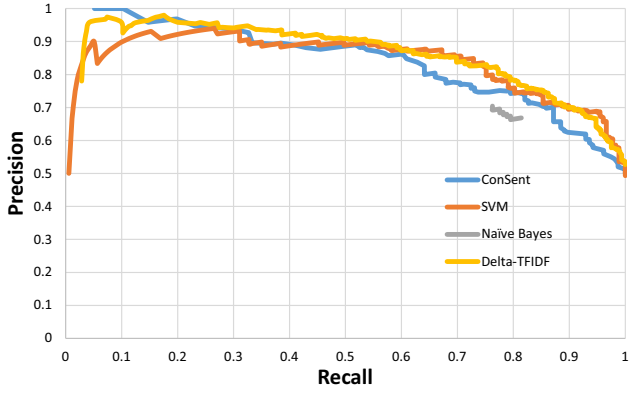

Fig. 16. The precision/recall curves for the Twitter dataset when using both text-based and meta-data features.

Table 12
The accuracy and AUC results for the Twitter dataset. *N* and *C* mark statistically significant difference ($p < 0.05$) from the performance of Naïve Bayes and ConSent, respectively. Boldface marks the best result in the row.

	ConSent	SVM	Naïve Bayes	Delta-TFIDF
Accuracy				
Without meta features	0.7493^{N}	$\mathbf{0.7604^{CN}}$	0.7214	0.7553^{N}
With meta features	0.7642^{N}	$\mathbf{0.7967^{CN}}$	0.7075	0.783^{NC}
AUC				
Without meta features	0.8335^{N}	0.8595^{N}	0.7536	$\mathbf{0.8668^{CN}}$
With meta features	0.8412^{N}	0.8755^{N}	0.7351	$\mathbf{0.8825^{NC}}$
F-Measure				
Without meta features	0.7428^{ND}	$\mathbf{0.744^{ND}}$	0.6913	0.7239^{N}
With meta features	0.7634^{N}	$\mathbf{0.7794^{ND}}$	0.7256	0.755^{N}

We hypothesize that the reason for ConSent's relative lower performance on Twitter compared to the other evaluated datasets, stems from the fact that on twitter there is very little context that can be analyzed. Despite this difficult setting, our approach manages to outperform the baselines on some scenarios, as shown by the precision/recall curves, and perform well overall. This attests to the robustness of our proposed approach.

4.6. Execution times comparison

We compare the execution times of ConSent to those of the SVM algorithm in order to provide the reader with a clear understanding of the algorithm's performance. The results are presented in Table 13. It is clear that although ConSent requires more running time than the "standard" SVM algorithm, the differences are great. Moreover, the *relative* increase in running time remains relatively consistent, indicating that ConSent is a viable option also for large datasets.

Table 13
The average running times (in seconds) of a single fold of the "standard" SVM algorithm and ConSent on each of the analyzed datasets. The (%) indicates the relative increase in running time for ConSent over the SVM algorithm.

Dataset	Movie reviews	TripAdvisor	Tele1	Tele2	Twitter
SVM	153.4	2766.5	546.2	505	24367.5
ConSent	172.3 (12.3%)	3263.3 (17.9%)	646.2 (18.3%)	622.6 (23.2%)	29210.2 (19.8%)

4.7. Conclusions and main findings

Based on the results of the experiments, we reached the following conclusions:

(a) ConSent shows statistically significant improvement over the baselines in the majority of cases. Moreover, the precision/recall curves indicate that our approach usually outperforms the baselines for the large majority of the possible recall values.
(b) Our evaluation on the noisy transcribed calls datasets clearly demonstrate that ConSent is more capable of utilizing meta-data than the BOW-based baselines. Moreover, ConSent was the only approach capable of utilizing information from surrounding text. This trait makes our approach particularly useful when analyzing text at the paragraph level. This demonstrates the robustness of our approach against noise.
(c) Analysis of the results of our experiments on the Twitter dataset seems to point that ConSent generally has decreased performance on extremely short texts. This is not surprising, as our approach places an emphasis on context – something Twitter greatly lacks. Despite this shortcoming, our approach fares well and even outperforms the baselines when low recall levels are required.
(d) Our evaluation shows a significant difference in the performance of ConSent over the two transcribed calls datasets. Our analysis of the data leads us to conclude that the difference stems from the size differences of the two datasets. Tele2 contains a significantly smaller number of documents than Tele1, and that trait seems to have prevented our model from creating key and context terms of the same quality as Tele1. From this we conclude that our approach requires a slightly larger training set than SVM or Naïve Bayes in order to maximize its performance. We believe this to be the case because of ConSent's two-step training process, which analyzes subsets of the training set for the selection of the context terms. As the nature of the approach is probabilistic, a very small training set may lead to non-optimal terms selection.

It should be noted that our evaluation has focused on learning-based approaches to sentiment analysis and not on lexical ones (please see Section 2). The reason for this is that although recent advances in both parsing [91] and parts-of-speech (POS) tagging [92] have enabled researchers to make great strides in the field of sentiment analysis, these approaches require domain-specific polarity lexicons [93,94] which are currently not in existence for transcribed text.

Overall, we can summarize that our approach performs well on multiple domains with varying degrees of noise. We believe that

the evaluation presented in this section demonstrates the merits of ConSent – high performance, robustness against noisy data, the ability to successfully utilize information on the structure of the text and the easy integration of features from external sources.

5. Summary and future work

In this study we present ConSent—a novel context-based method for sentiment analysis. Our approach builds on techniques from the field of information retrieval in order to identify key terms which are indicative of sentiment and the context in which they appear. Our experiments demonstrate that ConSent significantly outperforms leading baselines both in standard and "noisy" text.

The strength of our approach stems both from its focus on key terms and its heavy reliance on context, which has the following advantages: first, similarly to NLP approaches, ConSent takes into account the structure of the text but has the flexibility to cope with noise and missing parts of speech; and secondly, it enables the easy integration of information from additional sources, improving performance in the tested domains.

The small number of features utilized by our approach is another advantage of ConSent. In addition to avoiding the curse of dimensionality and the ability to integrate data from external sources, the small number of features enables us to model the structure of the text in ways that are not feasible for BOW classifiers. For example, we were able to model not only a single utterance in the transcribed conversations dataset, but also the surrounding paragraphs. Doing so in a "standard" BOW classifier would require doubling the thousands of features already in use, exacerbating the curse of dimensionality.

For future work, we plan to develop additional features for our method. As our key and context terms can be represented using a graph (as shown in Fig. 2), our aim is to use measures from graph theory such as betweenness and centrality [95–97] to enrich our feature set. Another interesting research prospect would be to adapt our method to multi-class problems (where there is a need to classify more than two item types simultaneously)—both in the field of sentiment analysis and in other classification tasks. Finally, we intend to utilize our key and context terms in order to identify exact sections in non-structured long text which contain positive and negative sentiment.

References

[1] R.W. Langacker, Observations and speculations on subjectivity, Iconicity Syntax 1 (985) (1985) 109.
[2] J. Scheibman, Point of View and Grammar: Structural Patterns of Subjectivity in American English Conversation, vol. 11, John Benjamins Publishing, 2002.
[3] B. Pang, L. Lee, Opinion mining and sentiment analysis, Found. Trends Inform. Ret. 2 (1–2) (2008) 1–135.
[4] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, R. Passonneau, Sentiment analysis of twitter data, in: Proceedings of the Workshop on Languages in Social Media, Association for Computational Linguistics, 2011.
[5] X. Wang, F. Wei, X. Liu, M. Zhou, M. Zhang, Topic sentiment analysis in twitter: a graph-based hashtag sentiment classification approach, in: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, ACM, 2011.
[6] F.F. Reichheld, Loyalty Rules!: How Today's Leaders Build Lasting Relationships, Harvard Business Press, 2001.
[7] B. Liem, H. Zhang, Y. Chen, An iterative dual pathway structure for speech-to-text transcription, in: Human Computation, 2011.
[8] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up?: Sentiment classification using machine learning techniques, Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, vol. 10, Association for Computational Linguistics, 2002, pp. 79–86.
[9] A. Pak, P. Paroubek, Twitter as a corpus for sentiment analysis and opinion mining, in: LREC, 2010.
[10] R. Zhang, E. Sumita, Boosting statistical machine translation by lemmatization and linear interpolation, in: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, Association for Computational Linguistics, 2007.
[11] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, in: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2005.
[12] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity: an exploration of features for phrase-level sentiment analysis, Comput. Linguist. 35 (3) (2009) 399–433.
[13] P.D. Turney, Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, 2002.
[14] V. Hatzivassiloglou, K.R. McKeown, Predicting the semantic orientation of adjectives, in: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, 1997.
[15] M. Taboada, C. Anthony, K. Voll, Methods for creating semantic orientation dictionaries, in: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC), Genova, Italy, 2006.
[16] H. Takamura, T. Inui, M. Okumura, Extracting semantic orientations of phrases from dictionary, in: HLT-NAACL, 2007.
[17] M. Hu, B. Liu, Mining and summarizing customer reviews, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2004.
[18] R.M. Tong, An operational system for detecting and tracking opinions in on-line discussion, in: Working Notes of the ACM SIGIR 2001 Workshop on Operational Text Classification, 2001.
[19] L. Polanyi, A. Zaenen, Contextual valence shifters, in: Computing Attitude and Affect in Text: Theory and Applications, Springer, 2006, pp. 1–10.
[20] S. Poria, A. Gelbukh, D. Das, S. Bandyopadhyay, Fuzzy clustering for semi-supervised learning–case study: construction of an emotion lexicon, in: Advances in Artificial Intelligence, Springer, 2013, pp. 73–86.
[21] V. Hatzivassiloglou, J.M. Wiebe, Effects of adjective orientation and gradability on sentence subjectivity, Proceedings of the 18th Conference on Computational Linguistics, vol. 1, Association for Computational Linguistics, 2000.
[22] J. Wiebe, Learning subjective adjectives from corpora, in: AAAI/IAAI, 2000.
[23] F. Benamara, C. Cesarano, A. Picariello, D.R. Recupero, V.S. Subrahmanian, Sentiment analysis: adjectives and adverbs are better than adjectives alone, in: ICWSM, 2007.
[24] S. Morinaga, K. Yamanishi, K. Tateishi, T. Fukushima, Mining product reputations on the web, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2002.
[25] E. Cambria, B. Schuller, Y. Xia, C. Havasi, New avenues in opinion mining and sentiment analysis, IEEE Intell. Syst. (2013) 1.
[26] Y. Park, S.C. Gates, Towards real-time measurement of customer satisfaction using automatically generated call transcripts, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, ACM, 2009.
[27] E. Boiy, M.-F. Moens, A machine learning approach to sentiment analysis in multilingual Web texts, Inform. Ret. 12 (5) (2009) 526–558.
[28] P. Melville, W. Gryc, R.D. Lawrence, Sentiment analysis of blogs by combining lexical knowledge with text classification, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009.
[29] Q. Ye, Z. Zhang, R. Law, Sentiment classification of online reviews to travel destinations by supervised machine learning approaches, Expert Syst. Appl. 36 (3) (2009) 6527–6535.
[30] A. Balahur, R. Steinberger, M. Kabadjov, V. Zavarella, E. Van Der Goot, M. Halkia, B. Pouliquen, J. Belyaeva, Sentiment Analysis in the News. arXiv preprint arXiv:1309.6202, 2013.
[31] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Mach. Learn. 46 (1–3) (2002) 131–159.
[32] H. Zhang, The optimality of naive Bayes, AA 1 (2) (2004) 3.
[33] T. Mullen, N. Collier, Sentiment analysis using support vector machines with diverse information sources, in: EMNLP, 2004.
[34] Y. Choi, C. Cardie, Learning with compositional semantics as structural inference for subsentential sentiment analysis, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2008.
[35] A.L. Maas, R.E. Daly, P.T. Pham, D. Huang, A.Y. Ng, C. Potts, Learning word vectors for sentiment analysis, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, vol. 1, Association for Computational Linguistics, 2011.
[36] H. Wang, Y. Lu, C. Zhai, Latent aspect rating analysis on review text data: a rating regression approach, in: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2010.
[37] J. Read, J. Carroll, Weakly supervised techniques for domain-independent sentiment classification, in: Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion, ACM, 2009.
[38] M. Potthast, Crowdsourcing a wikipedia vandalism corpus, in: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2010.

[39] X. Wan, Co-training for cross-lingual sentiment classification, Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, vol. 1, Association for Computational Linguistics, 2009.
[40] R. Hallowell, The relationships of customer satisfaction, customer loyalty, and profitability: an empirical study, Int J. Serv. Ind. Manage. 7 (4) (1996) 27–42.
[41] C. Homburg, A. Giering, Personal characteristics as moderators of the relationship between customer satisfaction and loyalty—an empirical analysis, Psychol. Market. 18 (1) (2001) 43–66.
[42] H. Takeuchi, T. Yamaguchi, Text mining of business-oriented conversations at a call center, in: Data Mining for Service, Springer, 2014, pp. 111–129.
[43] F. Cailliau, A. Cavet, Mining automatic speech transcripts for the retrieval of problematic calls, in: Computational Linguistics and Intelligent Text Processing, Springer, 2013, pp. 83–95.
[44] M. Garnier-Rizet, G. Adda, F. Cailliau, J.-L. Gauvain, S. Guillemin-Lanne, L. Lamel, S. Vanni, C. Waast-Richard, CallSurf: automatic transcription, indexing and structuration of call center conversational speech for knowledge extraction and query by content, in: LREC, 2008.
[45] J. Mamou, D. Carmel, R. Hoory, Spoken document retrieval from call-center conversations, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2006.
[46] G. Kurata, N. Itoh, M. Nishimura, A. Sethy, B. Ramabhadran, Leveraging word confusion networks for named entity modeling and detection from conversational telephone speech, Speech Commun. 54 (3) (2012) 491–502.
[47] R. Fernandez, A Computational Model for the Automatic Recognition of Affect in Speech, Massachusetts Institute of Technology, 2004.
[48] D. Ververidis, C. Kotropoulos, I. Pitas, Automatic emotional speech classification, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004, Proceedings (ICASSP'04), IEEE, 2004.
[49] J.E. Chung, E. Mustafaraj, Can collective sentiment expressed on twitter predict political elections? in: AAAI, 2011.
[50] L. Mitchell, M.R. Frank, K.D. Harris, P.S. Dodds, C.M. Danforth, The geography of happiness: connecting twitter sentiment and expression, demographics, and objective characteristics of place, PloS One 8 (5) (2013) e64417.
[51] N. Ofek, L. Rokach, P. Mitra, Methodology for connecting nouns to their modifying adjectives, in: Computational Linguistics and Intelligent Text Processing, Springer, 2014, pp. 271–284.
[52] D. Maynard, A. Funk, Automatic detection of political opinions in tweets, in: The Semantic Web: ESWC 2011 Workshops, Springer, 2012.
[53] J. Foster, Ö. Çetinoglu, J. Wagner, J. Le Roux, S. Hogan, J. Nivre, D. Hogan, J. Van Genabith, # hardtoparse: POS tagging and parsing the twitterverse, in: Proceedings of the Workshop On Analyzing Microtext (AAAI 2011), 2011.
[54] S. Rosenthal, P. Nakov, A. Ritter, V. Stoyanov, Semeval-2014 task 9: sentiment analysis in twitter, in: Proc. SemEval, 2014.
[55] S. Evert, T. Proisl, P. Greiner, B. Kabashi, P. für Korpuslinguistik, SentiKLUE: updating a polarity classifier in 48 hours, in: SemEval 2014, 2014, p. 551.
[56] G. Katz, Y. Elovici, B. Shapira, CoBAn: a context based model for data leakage prevention, Inform. Sci. 262 (0) (2014) 137–158.
[57] T.C. Hoad, J. Zobel, Methods for identifying versioned and plagiarized documents, J. Am. Soc. Inform. Sci. Technol. 54 (3) (2003) 203–215.
[58] M. Henzinger, Finding near-duplicate web pages: a large-scale evaluation of algorithms, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2006.
[59] V. Lavrenko, W.B. Croft, Relevance based language models, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New Orleans, Louisiana, United States, 2001, pp. 120–127.
[60] J.M. Ponte, W.B. Croft, A language modeling approach to information retrieval, in: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Melbourne, Australia, 1998, pp. 275–281.
[61] F. Song, W.B. Croft, A general language model for information retrieval, in: Proceedings of the Eighth International Conference on Information and Knowledge Management, ACM, Kansas City, Missouri, United States, 1999, pp. 316–321.
[62] Y. Lv, C. Zhai, Positional language models for information retrieval, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2009.
[63] C. Zhai, Statistical language models for information retrieval, Synth. Lect. Hum. Lang. Technol. 1 (1) (2008) 1–141.
[64] J. Arguello, J.L. Elsas, J. Callan, J.G. Carbonell, Document representation and query expansion models for blog recommendation, ICWSM 2008 (0) (2008) 1.
[65] T. Tao, X. Wang, Q. Mei, C. Zhai, Language model information retrieval with document expansion, in: Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Association for Computational Linguistics, 2006.
[66] G. Katz, A. Shtock, O. Kurland, B. Shapira, L. Rokach, Wikipedia-based query performance prediction, in: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM, 2014.
[67] A. Shtok, O. Kurland, D. Carmel, Predicting query performance by query-drift estimation, in: Advances in Information Retrieval Theory, Springer, 2009, pp. 305–312.
[68] C. Zhai, J. Lafferty, A study of smoothing methods for language models applied to ad hoc information retrieval, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2001.
[69] D. Hiemstra, Term-specific smoothing for the language modeling approach to information retrieval: the importance of a query term, in: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2002.
[70] J. Lafferty, C. Zhai, Document language models, query models, and risk minimization for information retrieval, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, New Orleans, Louisiana, United States, 2001, pp. 111–119.
[71] E. Cambria, D. Olsher, D. Rajagopal, SenticNet 3: a common and common-sense knowledge base for cognition-driven sentiment analysis, in: Twenty-eighth AAAI Conference on Artificial Intelligence, 2014.
[72] C. Hauff, L. Azzopardi, D. Hiemstra, F. De Jong, Query performance prediction: evaluation contrasted with effectiveness, in: Advances in Information Retrieval, Springer, 2010, pp. 204–216.
[73] D. Alassi, R. Alhajj, Effectiveness of template detection on noise reduction and websites summarization, Inform. Sci. 219 (2013) 41–72.
[74] S. Schleimer, D.S. Wilkerson, A. Aiken, Winnowing: local algorithms for document fingerprinting, in: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, ACM, San Diego, California, 2003, pp. 76–85.
[75] Y. Bengio, O. Delalleau, N. Le Roux, The Curse of Dimensionality for Local Kernel Machines, Techn. Rep, 2005, 1258.
[76] M. Verleysen, D. François, The curse of dimensionality in data mining and time series prediction, in: Computational Intelligence and Bioinspired Systems, Springer, 2005, pp. 758–770.
[77] P.F. Evangelista, M.J. Embrechts, B.K. Szymanski, Taming the curse of dimensionality in kernels and novelty detection, in: Applied Soft Computing Technologies: The Challenge of Complexity, Springer, 2006, pp. 425–438.
[78] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1) (1986) 81–106.
[79] J.J. Rodrıguez, L.I. Kuncheva, C.J. Alonso, Rotation forest: a new classifier ensemble method, IEEE Trans. Pattern Anal. Mach. Intell. 28 (10) (2006) 1619–1630.
[80] M.A. Hearst, S. Dumais, E. Osman, J. Platt, B. Scholkopf, Support vector machines, Intell. Syst. Appl. IEEE 13 (4) (1998) 18–28.
[81] D.D. Lewis, Naive (Bayes) at forty: the independence assumption in information retrieval, in: Machine Learning: ECML-98, Springer, 1998, pp. 4–15.
[82] J. Martineau, T. Finin, Delta TFIDF: an improved feature space for sentiment analysis, in: ICWSM, 2009.
[83] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The WEKA data mining software: an update, SIGKDD Explor. 11 (1) (2009).
[84] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recogn. 30 (7) (1997) 1145–1159.
[85] J. Davis, M. Goadrich, The relationship between precision–recall and ROC curves, in: Proceedings of the 23rd International Conference on Machine Learning, ACM, 2006.
[86] G. Hripcsak, A.S. Rothschild, Agreement, the f-measure, and reliability in information retrieval, J. Am. Med. Inform. Assoc. 12 (3) (2005) 296–298.
[87] T. Oommen, L.G. Baise, R.M. Vogel, Sampling bias and class imbalance in maximum-likelihood logistic regression, Math. Geosci. 43 (1) (2011) 99–120.
[88] B. Pang, L. Lee, A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts, in: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Barcelona, Spain, 2004, p. 271.
[89] S.M. Mohammad, S. Kiritchenko, X. Zhu, NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. arXiv preprint arXiv:1308.6242, 2013.
[90] E. Martınez-Cámara, S.M. Jiménez-Zafra, M.T. Martın-Valdivia, L.A. Urena-López, SINAI: voting system for twitter sentiment analysis, in: SemEval 2014, 2014, p. 572.
[91] R. Socher, J. Bauer, C.D. Manning, A.Y. Ng, Parsing with compositional vector grammars, in: Proceedings of the ACL Conference, Citeseer, 2013.
[92] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, N.A. Smith, Improved part-of-speech tagging for online conversational text with word clusters, in: HLT-NAACL, 2013.
[93] G. Qiu, B. Liu, J. Bu, C. Chen, Expanding domain sentiment lexicon through double propagation, in: IJCAI, 2009.
[94] Y. Zhao, B. Qin, T. Liu, Collocation polarity disambiguation using web-based pseudo contexts, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Association for Computational Linguistics, 2012.
[95] L.C. Freeman, Centrality in social networks conceptual clarification, Soc. Netw. 1 (3) (1978) 215–239.
[96] K.J. Keen, L. Etzkorn, Predicting students' grades in computer science courses based on complexity measures of teacher's lecture notes, J. Comput. Small Coll. 24 (5) (2009) 44–48.
[97] M. Fire, L. Tenenboim-Chekina, R. Puzis, O. Lesser, L. Rokach, Y. Elovici, Computationally efficient link prediction in a variety of social networks, ACM Trans. Intell. Syst. Technol. (TIST) 5 (1) (2013) 10.