Expert Systems With Applications 61 (2016) 282–289

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Using information retrieval for sentiment polarity prediction

Anderson Uilian Kauer, Viviane P. Moreira*

Institute of Informatics, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brazil

ARTICLE INFO

Article history:
Received 18 January 2016
Revised 18 May 2016
Accepted 25 May 2016
Available online 30 May 2016

Keywords:
Sentiment analysis
Opinion mining
Information retrieval
Polarity classification
Twitter

ABSTRACT

Social networks such as Twitter are used by millions of people who express their opinions on a variety of topics. Consequently, these media are constantly being examined by sentiment analysis systems which aim at classifying the posts as positive or negative. Given the variety of topics discussed and the short length of the posts, the standard approach of using the words as features for machine learning algorithms results in sparse vectors. In this work, we propose using features derived from the ranking generated by an Information Retrieval System in response to a query consisting of the post that needs to be classified. Our system can be fully automatic, has only 24 features, and does not depend on expensive resources. Experiments on real datasets have shown that a classifier that relies solely on these features outperforms established baselines and can reach accuracies comparable to the state-of-the-art approaches which are more costly.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

With over 500 million posts a day[1], Twitter[2] has consolidated itself as a major forum for expressing personal opinions on a variety of topics. Because of its popularity, this microblogging service has been the target of numerous studies from a broad range of research areas including Psychology, Sociology, Marketing, and Computer Science. For example, in Mostafa (2013), the analysis of tweets is used to determine the sentiment towards sixteen global brands.

Sentiment analysis, also called Opinion Mining, is dedicated to the computational study of opinions and sentiments expressed in text (Pang & Lee, 2008). This topic has been attracting increasing attention from the research community. Out of the different aspects of opinions that can be studied, the *polarity* of sentiments is the most well investigated. It consists in predicting whether the opinion expressed in the text is *positive* or *negative*.

While most of the research focuses on product reviews, recently, a number of studies on Twitter posts (or simply *tweets*) have emerged. Sentiment Analysis on Twitter can be done at three different levels: (*i*) entity, (*ii*) tweet, or (*iii*) expression. Entity-level analysis deals with discovering the overall opinion about an entity or topic, tweet-level analysis identifies the polarity of individual tweets, and expression level analysis deals with specific phrases within a tweet. Our focus is on the second – tweet-level analysis. The added challenge of analysing tweets (compared to product reviews) is their shorter length – at most 140 characters – which results in very sparse vector representations. In addition, the variety of topics, and the informal vocabulary, characterised by slangs, abbreviations, and misspellings, pose added difficulties to its computational treatment.

Successful approaches for polarity classification on tweets use one or more of the following: resources such as lexicons (which are sometimes manually created) (Fersini, Messina, & Pozzi, 2016; Speriosu, Sudan, Upadhyay, & Baldridge, 2011; Zhang, Ghosh, Dekhil, Hsu, & Liu, 2011), costly preprocessing such as part-of-speech tagging (Fersini et al., 2016; Go, Bhayani, & Huang, 2009; Hu, Tang, Tang, & Liu, 2013; Saif, He, & Alani, 2012), numerous features (Fersini et al., 2016; Go et al., 2009; Saif et al., 2012; Speriosu et al., 2011; Zhang et al., 2011) large amounts of training data (Bakliwal et al., 2012), and elaborated machine learning methods such as classifier ensembles (Coletta, da Silva, Hruschka, & Hruschka, 2014; Martìn-Valdivia, Martìnez-Cámara, Perea-Ortega, & Ureña López, 2013; da Silva, Hruschka, & Hruschka, 2014). In this work, we propose a method called Sentiment Analysis Based on Information Retrieval (SABIR) which uses none of the above. We show that classification accuracy comparable to the state-of-the-art can be achieved with a single classification algorithm using only 24 features. Unlike existing approaches, we do not use the words of the tweets as features. Our features are derived from the ranking generated by an Information Retrieval System in response to a query q which consists of the tweet that we wish to classify. The

* Corresponding author.
E-mail addresses: aukauer@inf.ufrgs.br (A.U. Kauer), viviane@inf.ufrgs.br (V.P. Moreira).

[1] http://www.internetlivestats.com/twitter-statistics/
[2] http://www.twitter.com

http://dx.doi.org/10.1016/j.eswa.2016.05.038
0957-4174/© 2016 Elsevier Ltd. All rights reserved.

ranking has the n most similar tweets for which we already know the class in decreasing order of similarity to the unlabelled tweet q. The rationale is to leverage information of the class of the similar posts to classify q.

We have carried out experiments with four datasets of tweets which have been used in similar studies. Since the training data for the classification system can be generated without manual annotation (Barbosa & Feng, 2010; Go et al., 2009), SABIR can be fully automatic. Our results have shown that there is no significant difference between SABIR and the best baseline classifier we implemented using over one thousand features.

2. Related work

The literature on sentiment analysis abounds on methods for classifying the polarity of opinionated texts, such as product reviews (Pang & Lee, 2008). In recent years, in interest on treating tweets has grown and several approaches were proposed. Martínez-Cámara, Martín-Valdivia, Urena-López, and Montejo-Ráez (2014) present a survey devoted exclusively to this topic. The task of identifying the polarity of a tweet is typically modelled as a classification problem. Its solution relies on machine learning algorithms and sentiment lexicons (*i.e.,* a list of opinion words and their polarities).

Building a supervised classifier requires labelled training examples. Since hand-labelled data is very costly to obtain, alternative approaches have relied on automatic but noisy labels. Go et al. (2009) use emoticons to assign the label and show that various machine learning algorithms have accuracies of over 80% when trained over this noisily labelled data. With the same goal, Barbosa and Feng (2010) have used existing websites on sentiment analysis of tweets as a source of training data.

Most of the related work use a bag-of-words (BoW) approach in which the words in the tweets are used as features (*i.e., n*-g, usually unigrams and bigrams) (Davidov, Tsur, & Rappoport, 2010; Fersini et al., 2016; Go et al., 2009; Saif et al., 2012; Vosoughi, Zhou, & roy, 2015), combined to sentiment lexicons (Fersini et al., 2016; Speriosu et al., 2011; Zhang et al., 2011). Part-of-speech (POS) tags have also been widely used (Barbosa & Feng, 2010; Fersini et al., 2016; Go et al., 2009; Hu et al., 2013; Saif et al., 2012). Emoticons and Twitter specific features (such as retweets, hashtags, and follower graph) have also been exploited.

Ghiassi, Skinner, and Zimbra (2013) examined traditional feature selection strategies and proposed a semi-automatic method to identify useful features. They generated a lexicon with 187 features derived for a dataset of tweets on Justin Bieber. The authors mention that whether the features could be used to classify posts about other entities is still to be established.

Carvalho, Prado, and Plastino (2014) applied a genetic algorithm to select the paradigm words that will help determine the polarity of other words. They found an improvement compared to approaches in which the paradigm words are fixed.

More recently, da Silva et al. (2014), following Lin and Kolcz (2012), have analysed the use of feature hashing to solve the problem of the sparsity of the vectors created from tweets when the words are used as features, *i.e.,* in a BoW approach. In this approach, the features are hash integers rather than strings. The authors report that feature hashing was outperformed by the BoW approach in all but one dataset (HCR).

Vosoughi et al. (2015) enriched a standard bigram model with features representing contextual information about the tweet. The added features were: the US state, the hour of the day, the day of the week, the month and the author of the tweet. Their experiments identified a gain of 10% in accuracy with the added features. The limitation of this approach is that such contextual information is not available in the standard datasets used in the literature.

Fersini et al. (2016) applied feature expansion by adding features which explore the presence of adjectives, emoticons, emphatic and onomatopoeic expressions and also expressive lengthening of words to a BoW model. The authors found that adjectives were the most discriminative of those features bringing gains ranging from 0.49 to 4% in accuracy depending on the dataset. In order to derive these added features, a POS-tagger and a sentiment lexicon are necessary.

To avoid the need of labelled training data, a sentiment lexicon with words and their assigned polarities can be used for polarity prediction. While such lexicons are costly to generate, there are a few of them readily available. The limitation with regards to their application on Tweeter data is that tweets tend to have slangs, misspellings, and colloquial expressions which will not typically be included in a lexicon. To overcome this problem, Saif, He, Fernandez, and Alani (2016) proposed an approach to capture the semantic information inferred from co-occurrence patterns to generate a dynamic representation of the words. Their experiments have shown gains in two out of three datasets compared to lexicon-based baselines.

While the most popular machine learning algorithms for sentiment analysis are Naïve Bayes, Multinomial Naïve Bayes, Support Vector Machines, and Maximum Entropy, recent studies (Coletta et al., 2014; Fersini et al., 2016; da Silva et al., 2014) have explored the combination of classifiers. Classifier and cluster ensembles (Coletta et al., 2014; Fersini et al., 2016; da Silva et al., 2014) improved classification quality but bring extra computational costs.

3. Classifying the polarity of tweets

A Twitter post, or *tweet*, expresses the opinions or sentiments of its author about an entity. As mentioned in Section 2, the traditional approach for classifying the polarity of a tweet is to implement a classifier using unigrams as features (*i.e.,* BoW). However, this tends to result in a very sparse set of features because of the large diversity of vocabulary in the tweets. Our hypothesis is that twitter posts that are similar tend to belong to the same class. Thus, information about the class of the n most similar posts may help classify the polarity of a given unlabelled tweet. This notion is analogous to the one in Weren et al. (2014) which was applied to predicting gender and age of social media posts.

Unlike approaches which work by selecting the most representative terms to be used as features, both for sentiment analysis in general (Deng, Luo, & Yu, 2014; Nicholls & Song, 2010; O'Keefe & Koprinska, 2009) and specifically for tweets (Carvalho et al., 2014; Ghiassi et al., 2013; da Silva et al., 2014), we do not employ the words in the tweets as features. Thus, SABIR is not a term selection approach.

SABIR is composed of two steps. The first step consists in obtaining the n most similar posts in relation to the tweet we wish to classify. In the second step, we use information about these n posts as features (or attributes) to train a supervised classifier. An overview of the process is shown in Fig. 1.

More formally, our goal is: given a set of tweets $T = \{t_1, t_2, ..., t_m\}$ for which the class $c_i \in \{positive(+), negative(-)\}$ is known and a set of unlabelled tweets $Q = \{q_1, q_2, ..., q_p\}$ (*i.e.,* for which the class is unknown), we use information about the similarity of each element $q_i \in Q$ in relation to the elements $t_j \in T$ to predict the class of q_i. Information on the similarity of the tweets is taken from an Information Retrieval System as described in Section 3.1.

Our approach can work with one or more sets of labelled instances, as those are required for indexing and for training. In our experiments, we have explored two settings: (*i*) using three sets of data: a large dataset of noisily labelled tweets for indexing; a manually labelled dataset for training, and an unlabelled dataset

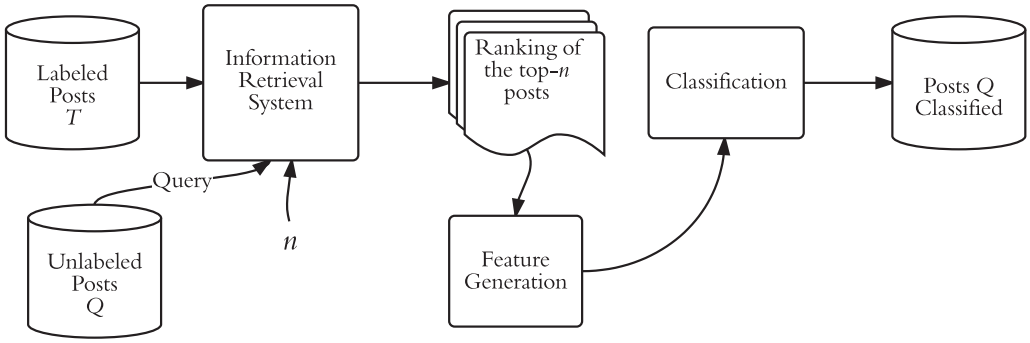


Fig. 1. Overview of SABIR.

for testing, and (*ii*) using a single dataset for indexing, training and testing (in this case, the post that needs to be classified is discarded from the results of the query so as to avoid an unfair advantage).

While Information Retrieval Systems can quickly index millions of documents, learning in text classification is usually applied to at most a couple thousand instances. Indexing has a linear time complexity (Baeza-Yates & Ribeiro-Neto, 2011), whereas widely used classifiers such as SVM has a complexity of $O(m^2{*}n)$ for n features and m instances (Chapelle, 2007). Thus, using fewer features could potentially help training on larger datasets. A runtime experiment is provided in Section 4.5.

3.1. Information retrieval system

Information Retrieval Systems (IRS) apply similarity functions to rank a set of items (usually textual documents) in response to a user query. In this work, an IRS is used to index the labelled tweets in T, and rank them in relation to each unlabelled tweet in Q. The ranked list with the n highest scoring items for each q_i serves as a source of features which will later be used by a classifier. If a post q_i is closest to positive posts, then it is likely to be positive too. The idea is analogous to the principle behind a k-NN classifier in which a test instance is classified as belonging to the majority class of its k-closest neighbours. An extremely simple but naïve approach would be taking the prevalent class in the ranking as the class of the post. However, this evidence alone is not enough. Thus, our method extracts different sources of evidence from the ranking of the n most similar posts in relation to q. SABIR considers a variety of attributes ranging from simple counts to more elaborate metrics that combine the rank and the score of the retrieved results.

The first step is to index the labelled posts (T) using an IRS. Then, each unlabelled post $q_i \in Q$ is used as query and the n most similar posts are retrieved and ranked in decreasing order of similarity. In this work, we employ the widely used Okapi BM25 algorithm to generate the ranking. Although other metrics could have been used, we opted for BM25 due to its good results in IR experiments (Manning, Raghavan, & Schütze, 2008). Table 1 shows an example of such a ranking for n =25.

3.2. Feature generation

A total of 24 features are extracted from the ranking (12 for each class). Notice that only one ranking is necessary to produce the 24 features. Aggregation functions such as average, max, min, sum, and count are used for each class (positive and negative). Table 2 shows the values for the features extracted from the ranking in Table 1. In this example, there are 12 negative items, so $count_{-} = 12$. The average score for the negative posts is 21.723 and that is the value for the feature avg_{-}.

Table 1
Example of ranking for a post q.

Rank	Score	Class
1	25.897	–
2	25.541	+
3	25.158	–
4	22.729	–
5	22.204	–
6	21.576	–
7	21.468	–
8	21.426	+
9	21.011	+
10	20.985	+
11	20.666	–
12	20.666	–
13	20.546	+
14	20.546	+
15	20.435	–
16	20.070	–
17	20.032	–
18	20.025	+
19	19.974	+
20	19.962	+
21	19.934	+
22	19.902	+
23	19.889	+
24	19.779	–
25	19.771	+

In addition to these aggregation functions, we derived another feature, called ϕ, that takes into consideration the absolute and the relative (*i.e.,* in relation to the class) ranks of each item. It is given as:

$$\phi_c = \sum_{r=1}^{n_c} \frac{rank_{rel}}{rank_{abs}} \tag{1}$$

where c is the class; n_c is the number of retrieved posts for that class, $rank_{rel}$ is the relative position of the post among the posts of that class, and $rank_{abs}$ is the absolute position of the post in the overall ranking.

The remaining attributes are a combination of the metric ϕ and the aggregation functions. They are: $\phi_{avg_c} = \frac{\phi_c}{avg_c}$, $\phi_{max_c} = \frac{\phi_c}{max_c}$, $\phi_{min_c} = \frac{\phi_c}{min_c}$, $\phi_{sum_c} = \frac{\phi_c}{sum_c}$, and $\phi_{count_c} = \frac{\phi_c}{count_c}$. The aggregation functions avg_c, max_c, min_c, and sum_c compute the average, maximum, and minimum scores for class c, respectively; and $count_c$ is the number of posts from class c in the ranking.

The final feature, $\phi_{c_{positional}}$, combines the scores of the retrieved posts and their positions in the ranking. It is given as:

$$\phi_{positional_c} = \sum_{r=1}^{n} \left(\frac{rank_{rel}}{rank_{abs}} \right) \times S(q_i) \tag{2}$$

where $S(q_i)$ is the score for post q_i.

Table 2
Features derived from the ranking.

Feature	Value
avg_{-}	21.723
max_{-}	25.897
min_{-}	19.779
sum_{-}	260.68
$count_{-}$	12
ϕ_{-}	8.5822
$\phi_{positional_{-}}$	188.44
$\phi_{avg_{-}}$	0.3950
$\phi_{max_{-}}$	0.3313
$\phi_{min_{-}}$	0.4338
$\phi_{sum_{-}}$	0.0329
$\phi_{count_{-}}$	0.7151
avg_{+}	20.732
max_{+}	25.541
min_{+}	19.771
sum_{+}	269.51
$count_{+}$	13
ϕ_{+}	5.5743
$\phi_{positional_{+}}$	115.51
$\phi_{avg_{+}}$	0.2688
$\phi_{max_{+}}$	0.2182
$\phi_{min_{+}}$	0.2819
$\phi_{sum_{+}}$	0.0206
$\phi_{count_{+}}$	0.4287
class	–

An important aspect of creating a classification system using IR-based features is that the dimensionality is greatly reduced and, at the same time, classification quality is improved (as shown by our experiments). Rather than using the words themselves as features, which would lead to several thousand features even in small datasets, we are able to classify with high accuracy using only 24 attributes. Although our features have no linguistic meaning, the intention is that they should capture latent discourse and semantic properties of the text while Okapi BM25 captures the lexical properties.

3.3. Classification

The second stage in SABIR involves using a supervised machine learning algorithm. Based on labelled training instances, a learning model is generated. The goal is to obtain a model that is able to generalise and classify unseen data. Since it is well-known that there is no single machine that is better than all the others on all tasks, a number of classification algorithms were tested to identify the ones that are best suited to our attributes. Classifiers from different families such as functions (Logistic Regression, Simple Logistic, and Support Vector Machines), trees (J48 and REPTree), Bayesian (Naïve Bayes and Naïve Bayes Multinomial), and rule-based (Conjunctive Rule) algorithms were tested. Their results are presented in Section 4.

4. Experiments

In order to test our proposed approach, we ran experiments using four datasets of real Web data. We have also compared our results to a number of baselines, including state-of-the-art approaches. In the next Section, we describe the experimental setup and our results. In addition, as our method has the number of documents retrieved by the query as an input parameter (n), we show how this number affects our results.

Table 3
Details of the datasets.

Dataset	# tweets	# positive	# negative	# unigrams
STD-train	1.6M	800K	800K	480785
STD-test	359	182	177	1683
HCR-train	614	213	401	2869
HCR-test	658	154	504	2927
OMD	1488	541	947	3411
STS-Gold	2034	632	1402	5139

4.1. Experimental setup

Datasets. Four datasets composed of real tweets, which have also been applied in other related words (Bakliwal et al., 2012; Carvalho et al., 2014; Coletta et al., 2014; Go et al., 2009; Saif, Fernández, He, & Alani, 2013; Saif et al., 2012; Saif et al., 2016; da Silva et al., 2014; Speriosu et al., 2011), were used in our experiments. Their details are given in Table 3.

- **Stanford-Twitter sentiment corpus (STD)**[3] which is divided into two subsets *STD-train* and *STD-test*. STD-train, is composed of 1.6 million automatically labelled tweets. The automatic labelling method is known as *noisy-labelling* (Go et al., 2009), and it basically consists in assigning a sentiment to the tweet based on the sentiments associated to the emoticon present in the tweet. For example: tweets containing `:)`, `:-)`, `: )`, `:D`, or `=)` were classified as positive and tweets containing `:'(` `:(`, or `:-(` were considered negative. This method eliminates the manual effort in labelling, but the resulting dataset contains noise. Tweets with irony and sarcasm, for example, could be misclassified. STD-test, on the other hand, has 359 tweets which were *manually* labelled. The datasets were obtained using the Twitter API by posing queries on different topics containing names of products, companies, and people.
- The **health care reform (HCR)**[4] dataset was assembled in 2010. It contains tweets with the hashtag "#hcr" which concern the US health care reform and was a controversial issue. The tweets were manually annotated as positive, negative, or neutral. Similar to what was done in related work and in our baselines, neutral tweets were not considered in our experiments.
- The **Obama-McCain debate (OMD)**[5] dataset is composed of 3238 tweets crawled in 2008 during the first U.S. presidential debate between the candidates. The tweets were annotated with the aid of Amazon Mechanical Turk[6] as *positive, negative, mixed*, or *other*. For our experiment, we only kept the tweets that were tagged *positive* or *negative* for which at least two thirds of the annotators have agreed.
- The **stanford sentiment gold standard (STS-Gold)**[7] dataset was assembled in Saif et al. (2013). It contains tweets for which three annotators have agreed on whether the sentiment label was *positive* or *negative*.

Preprocessing. The raw text of the tweets was modified as follows: all characters were converted to lowercase, non alphabetic characters (including numbers and punctuation) were removed, and repeated characters were removed (*e.g., sooo* is modified to *so*). All tweets from both datasets (including training and testing) were preprocessed.

Information retrieval system. There is a number of freely available IRS which could have been used. We chose

[3] Available at http://help.sentiment140.com
[4] Available at https://bitbucket.org/speriosu/updown
[5] Available at https://bitbucket.org/speriosu/updown
[6] http://www.mturk.com
[7] Available from http://tweenator.com/

Zettair (Billerbeck et al., 2004) for its simplicity and effectiveness in indexing. Okapi BM25 was used to generate the rankings. The similarity score between tweets q_i and t_j. It is calculated as:

$$BM25(q_i, t_j) = \sum_{w \in q_i} \log\left(\frac{m - f_w + 0.5}{f_w + 0.5}\right) \times \frac{(k_1+1) f_{w,t}}{K + f_{w,t}} \quad (3)$$

where w is a word in the tweet, m is the number on indexed tweets, f_w is the number of occurrences of word w in the dataset, and $f_{w,t}$ is the number of occurrences of w in t_j. K is $k_1((1-b) + b \times L_t/avgL)$ where L_t is the length of t_j in bytes and $avgL$ is the average length of the indexed tweets. The default values for the constants k_1 and b were used (1.2 and 0.75, respectively).

We implemented two versions of SABIR. In the first version, which we call SABIR-noisy, indexing was applied only to the *STD-train* dataset (*i.e.,* the automatically labelled dataset). The index generated by Zettair for had 1,599,997 documents (or tweets), a total of 21,972,420 tokens, of which 480,785 were distinct. Notice that while indexing took 8 only seconds (on an i5 3.2 GHz processor with 8Gb of main memory), training a classifier with all these instances would take many hours or even days (depending on the algorithm). This index was queried using posts from the other datasets to allow for the calculation of features. Motivated by the guidelines used in evaluation campaigns such as TREC and CLEF, for each query, the top one thousand most similar tweets were retrieved (*i.e.,* $n = 1000$). This issue is further discussed in Section 4.4. In the second version of SABIR, we indexed the dataset that needed to be classified. In these cases, the time taken during the indexing phase was at most 0.6 s. Then, each query would retrieve its corresponding post at the top of the ranking. To avoid the unfair advantage that this would bring to our method, we discarded this post from the ranking. Thus, the features derived to classify the unseen post, relied solely on evidence coming from other labelled instances.

Classification tool. We used the classifiers implemented in Weka (Hall et al., 2009) according to their default parameters. Our results have shown that the best algorithms differed across datasets. The best results have been obtained by Naïve Bayes Multinomial (on STD), Maximum Entropy (on OMD and STS-Gold), and SVM (on HCR). We then chose Maximum Entropy (MaxEnt), and used it in all versions of SABIR in all datasets. In order to be comparable to the literature, 10-fold cross validation was applied to all datasets except from the HCR dataset which has separate training (HCR-train) and test sets (HCR-test).

Evaluation metrics. The standard evaluation metrics applied to classification systems were used. They are: accuracy (Acc), precision (Pr), recall (Rec), and F-measure with equal weights to Pr and Rec (F1), calculated as $Acc = \frac{TP+TN}{TP+TN+FP+FN}$, $Pr = \frac{TP}{TP+FP}$, $Rec = \frac{TP}{TP+FN}$, $and F1 = 2 \times \frac{Pr \times Rec}{Pr+Rec}$ where TP, TN, FP, and FN stand for true positive, true negative, false positive, and false negative, respectively. Pr, Rec and F1 are micro-averaged, which means instances are given equal weights.

Baselines. We have implemented four baselines in which the words in the tweets (unigrams) are used as features (*i.e.,* a BoW approach). The following supervised learning algorithms were used: Naïve Bayes (NB), Multinomial Naïve Bayes (MNB), Support Vector Machines (SVM), and Maximum Entropy (MaxEnt). The same instances used for training/testing were applied to all methods (including ours). We also include results reported in the literature on the same datasets.

4.2. Feature analysis

In order to analyse the discriminative power of our features, we calculated their Information Gain (IG) (Mitchell, 1997). IG measures the expected reduction in entropy (*i.e.,* uncertainty) considering just the feature and the class. The results for all datasets are shown in Table 4. The values for IG are noticeably lower on the HCR dataset. Looking at the classification metrics on Table 5, one can see that the HCR dataset had worst scores on the vast majority of the methods. This implies that this dataset is harder to classify and may explain the lowest scores for IG. In addition, the best features are different from one dataset to another. Still it is worth noticing that the best features in both cases include a various types of attributes from simple counts to more elaborate metrics that take the rank and the score of the retrieved results into consideration. This shows that, for generality sake, keeping the variety is important.

Table 4
Best ten features in terms of information gain.

STD		HCR		OMD		STS-Gold	
IG	Feature	IG	Feature	IG	Feature	IG	Feature
0.382	ϕ_{avg_+}	0.061	avg_+	0.2051	$\phi_{positional_+}$	0.3905	ϕ_{count_-}
0.361	$count_-$	0.040	avg_-	0.2037	ϕ_{count_+}	0.3881	ϕ_{count_+}
0.361	$count_+$	0.034	ϕ_{min_-}	0.2015	ϕ_{count_-}	0.2708	ϕ_+
0.360	ϕ_{count_-}	0.034	sum_+	0.1665	sum_+	0.2669	$\phi_{positional_+}$
0.343	ϕ_{min_+}	0.033	ϕ_{sum_-}	0.1642	ϕ_-	0.2275	ϕ_-
0.339	ϕ_{count_+}	0.033	ϕ_{sum_+}	0.161	avg_+	0.1674	sum_+
0.332	ϕ_+	0.031	ϕ_{avg_-}	0.1555	ϕ_+	0.1589	$\phi_{positional_-}$
0.329	ϕ_-	0.031	$\phi_{positional_+}$	0.1547	ϕ_{sum_+}	0.1512	$count_-$
0.310	$\phi_{positional_-}$	0.030	min_-	0.1538	ϕ_{avg_+}	0.1508	$count_+$
0.302	ϕ_{max_+}	0.028	min_+	0.1099	ϕ_{min_+}	0.0981	avg_+

Table 5
Classification results comparing the two versions of SABIR against the baseline classifiers. Whenever the statistical test has shown that the baseline was better, worse or no different from a version of SABIR, a $^{\triangle}$, $^{\triangledown}$, or a $^{\diamond}$, respectively, is shown. The first symbol refers to a comparison against SABIR-Noisy, while the second is for the comparison against SABIR. Best results in bold.

		NB	MNB	SVM	MaxEnt	SABIR-noisy	SABIR
STD	**Acc**	75.8$^{\diamond\triangledown}$	80.8$^{\diamond\diamond}$	74.7$^{\diamond\triangledown}$	78.6$^{\diamond\diamond}$	78.3$^{\diamond}$	**81.9**
	Rec	75.8	80.8	74.7	78.6	78.3	**81.9**
	Pr	75.8	80.8	74.7	78.6	78.3	**82.0**
	F1	75.7	80.8	74.6	78.6	78.3	**81.9**
HCR	**Acc**	73.2$^{\diamond\diamond}$	**79.1**$^{\diamond\triangle}$	75.0$^{\diamond\diamond}$	68.0$^{\triangledown\triangledown}$	77.4$^{\diamond}$	73.4
	Rec	73.2	**79.1**	75.8	68.0	77.4	73.4
	Pr	74.7	76.5	**77.1**	76.9	75.7	73.8
	F1	73.9	75.9	75.9	**76.3**	74.6	73.6
OMD	**Acc**	74.1 $^{\triangle\diamond}$	76.5$^{\triangle\diamond}$	77.0 $^{\triangle\diamond}$	74.1$^{\triangle\diamond}$	68.4$^{\triangledown}$	**77.5**
	Rec	74.1	76.5	77.0	74.1	68.4	**77.5**
	Pr	73.7	76.2	**77.4**	75.5	67.5	77.1
	F1	73.8	76.3	**77.1**	74.4	64.5	77.0
STS-Gold	**Acc**	79.7$^{\triangledown\triangledown}$	**84.8**$^{\diamond\diamond}$	84.7$^{\diamond\diamond}$	76.5$^{\triangledown\triangledown}$	83.1$^{\diamond}$	84.5
	Rec	79.7	**84.8**	84.7	76.5	83.1	84.5
	Pr	80.1	**84.6**	**84.6**	78.8	82.8	84.2
	F1	79.9	84.2	**84.6**	77.2	82.9	84.3

4.3. Results

We compared our results against the baselines we implemented ourselves and also against the best results published in the literature.

Comparison against baselines. Table 5 reports Accuracy, Precision, Recall, and F-measure for the two versions of SABIR and the baselines we implemented. The results for the baselines indicate how they compare to SABIR-Noisy and SABIR, respectively. A Wilcoxon Signed-Rank test[8] was applied on the accuracy results for each instance under the different methods. Whenever a base-

[8] Since our data was not normally distributed, parametric tests such as the T-Test, could not be employed.

line is statistically outperformed by SABIR-Noisy or SABIR, a $^{\triangledown}$ is shown. In contrast, when the baseline significantly outperforms either SABIR-Noisy or SABIR, a $^{\triangle}$ is shown. Cases in which no significant difference was found are denoted by a $^{\diamond}$.

In all datasets, at least one version of SABIR was not statistically different from the best baseline. This was achieved using only 24 features, while the baselines use far more features (between 1683 and 5139, depending on the dataset, as shown Table 3). In the STD dataset, SABIR outperforms all baselines, this difference was considered significant in relation to NB (p-value = 0.0226) and SVM (p-value = 0.0078) – as denoted by the symbols $^{\diamond\triangledown}$.

Comparing SABIR-Noisy and SABIR, we found that in most cases, SABIR has the best results. However, the difference was not considered statistically significant in three out of four cases. The exception was in the OMD dataset, in which SABIR-Noisy was significantly worse than SABIR and all baselines. A visual inspection of the distribution of the instances with respect to the class showed that the attributes of the noisy version were not able to provide a clear separation between the classes. This possibly happened because evidence coming from another dataset (*i.e.,* STD-train) was not as accurate as the evidence coming from the same dataset. On the other hand, for the HCR dataset, SABIR was outperformed by SABIR-Noisy. Using SVM with SABIR in this dataset made the accuracy results of SABIR increase to 79.8% (surpassing the best baseline), while SABIR-Noisy decreases slightly (76.9%). This could be explained by the fact that in machine learning no single algorithm yields the best results in all datasets.

Comparing against published results. We now discuss the accuracy of SABIR compared to results published in the literature on the same datasets in nine studies, namely: Bakliwal et al. (2012), Carvalho et al. (2014), Coletta et al. (2014), Go et al. (2009), Hu et al. (2013), Saif et al. (2012, 2016), da Silva et al. (2014), Speriosu et al. (2011). Most of those works use unigrams as features, amounting to thousands of attributes. Fig. 2 shows the accuracies for the state-of-the-art methods and the two versions of SABIR.

On the STD dataset, SABIR was outperformed by the best accuracy reported so far (Bakliwal et al., 2012) by about five percentage points. However, that result was obtained with 1.6 million instances of training data and expensive resources to derive the features such as spelling correction, part-of-speech tagging, and sentiment lexicons. Speriosu et al. (2011) also had better results but with features like Twitter follower graph. Go et al. (2009) had a slightly better result than SABIR, but 364K features were used. SABIR came in fourth place, achieving higher accuracies than methods that apply feature selection through feature hashing (FH) (da Silva et al., 2014) or genetic algorithms (Carvalho et al., 2014).

In the OMD and STS-Gold datasets, SABIR achieved the second best results. In these cases, the winning approach by Saif et al. (2013) implements a BoW model.

The HCR dataset behaved differently from the other three as SABIR-Noisy achieved a better score than SABIR and also we were outperformed by a number of approaches from the state-of-the-art. The information gain results on HCR (see Table 4), were also lower than in the others, indicating that the correlations between the features and the class were smaller.

Error analysis: In order to identify patterns in the classification errors, we analysed cases in which our assumption that the classes of similar posts would be the same class of the post we wish to classify did not hold. Looking into those cases, we could establish three main reasons: (*i*) posts with interrogations that have sentiment words but in which these words do not express a sentiment; (*ii*) presence of irony; and (*iii*) posts in which there is a negation that inverts the polarity of the sentiment. In all these cases, the problems seem to stem from the fact that annotations were done automatically with no human intervention. Since we did not use any NLP tools to treat these cases, they may affect the results of the classifiers.

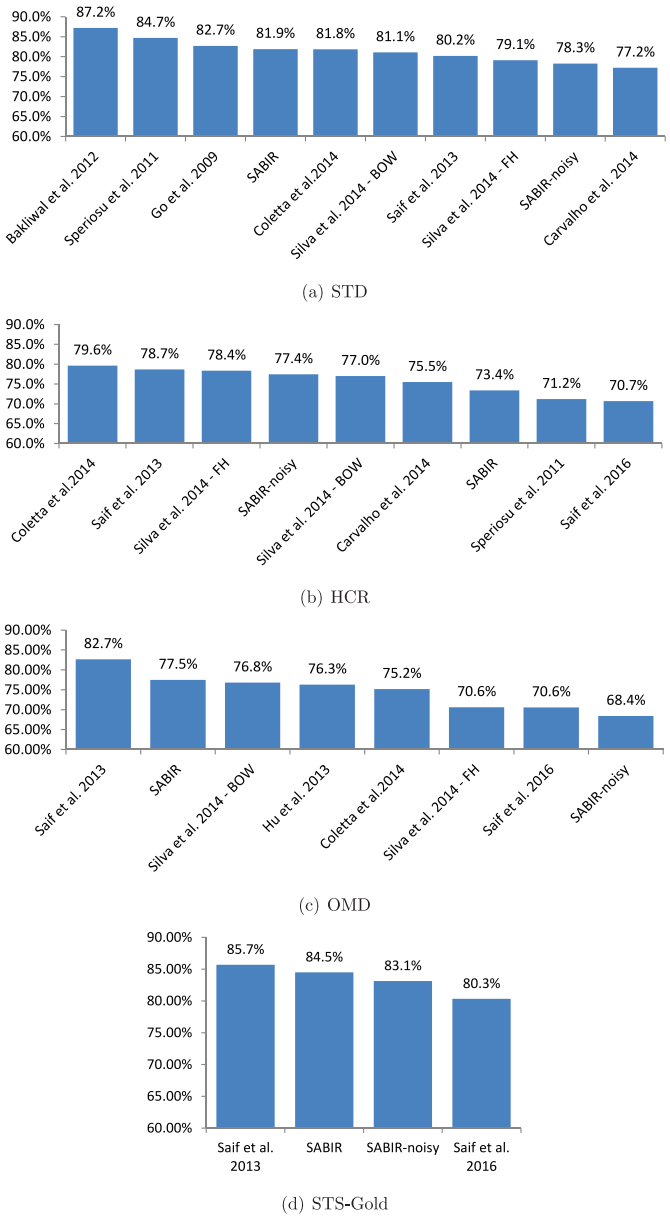


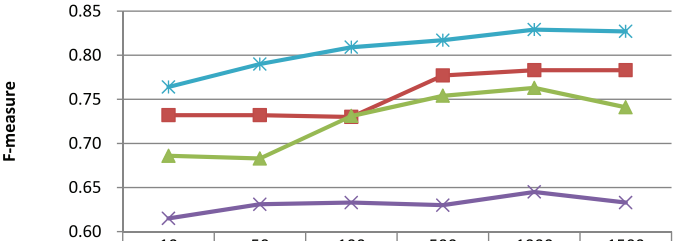
Fig. 2. Accuracy results for the state-of-the-art and SABIR

4.4. Sensitivity of the F-1 measure as a function of n

SABIR receives the number of documents to be retrieved n as input parameter. Intuitively, n cannot be too small so that it does not provide enough evidence for the analysis. On the other hand, if n is too large, posts that are not so similar to the query will be retrieved and add noise to the features. We have tested several values of n, namely 10, 50, 100, 500, 1000, and 1500 in the SABIR-Noisy version. Since our index has about 1.6 million posts, the retrieved posts represent a very small fraction of the total ($\sim$ 0.006% of all documents). The results are shown in Fig. 3. The highest values of F-1 are achieved for $n = 1000$ in all datasets. Still, the widest variation considering all results was of 7 percentage points, which shows that SABIR is not too sensitive to this parameter.

Table 6
Runtime Experiment (time in seconds). The runs which took less time than the total time taken by SABIR are marked with a $^{\triangledown}$, while the ones that take longer are marked with a $^{\triangle}$.

Instances	Features					SABIR			
	100	500	1000	2000	5000	indexing	querying	training	total
1000	0.29$^{\triangledown}$	0.62$^{\triangledown}$	1.31$^{\triangledown}$	2.52$^{\triangle}$	–	0.29	1.69	0.26	2.24
2000	0.57$^{\triangledown}$	2.12$^{\triangledown}$	7.36$^{\triangledown}$	18.00$^{\triangle}$	40.90$^{\triangle}$	0.60	6.98	1.81	9.39
5000	10.23$^{\triangledown}$	21.77$^{\triangledown}$	24.23$^{\triangledown}$	44.58$^{\triangle}$	66.61$^{\triangle}$	0.93	24.39	3.32	28.64
10000	44.87$^{\triangledown}$	90.94$^{\triangle}$	127.97$^{\triangle}$	175.47$^{\triangle}$	281.27$^{\triangle}$	1.29	62.52	5.25	69.06
20000	198.34$^{\triangle}$	359.43$^{\triangle}$	495.43$^{\triangle}$	737.82$^{\triangle}$	1121.07$^{\triangle}$	1.70	179.17	7.67	188.54



	10	50	100	500	1000	1500
STD	0.73	0.73	0.73	0.78	0.78	0.78
HCR	0.69	0.68	0.73	0.75	0.76	0.74
OMD	0.62	0.63	0.63	0.63	0.65	0.63
STS-Gold	0.76	0.79	0.81	0.82	0.83	0.83

Fig. 3. F1 as a function of n.

4.5. Runtime analysis

In order to compare the runtime of SABIR and the baselines', we have run them on a series of data samples varying the number of features and the number of instances. The samples were randomly taken from the STD-train dataset. We used SVM as the baseline algorithm as it is arguably the most widely used classifier for text. The times reported for SVM refer to the number of seconds taken for training the BoW classifier, while the times reported for SABIR include the time spent in indexing, querying, and training a MaxEnt classifier using our 24 features. These tests were run on a regular PC with i5 3.2 GHz processor and 8Gb of main memory. Table 6 shows the results. The most time-consuming part of SABIR is the querying phase. Our queries are composed of all words in the tweets, which require fetching the list of all documents in which each of these words occur to calculate the similarity score (BM25). Still, it took only 9 ms per query in the largest index. SABIR is more efficient than BoW approaches whenever over 1 K features are used or over 10 K instances were used, showing the potential to work with large datasets. All datasets used in our experiments had over 1 K unigrams.

5. Conclusion

We have proposed SABIR a method for polarity classification of tweets based on an Information Retrieval system. Our goal was to leverage the information on the class of (labelled) similar tweets to classify new unlabelled posts. We have proposed and tested 24 features that are based solely on the ranking provided by a search engine in response to queries consisting of the tweets we wish to classify. Since the labels for the indexed posts can be generated without human intervention, our method can be completely automatic.

Comparative experiments on a number of baselines have shown promising results. The accuracy of SABIR is not statistically different from the accuracy of our best baseline. In comparison to the state-of-the-art results, we reach similar scores relying solely on the 24 proposed features.

The scope of this article was limited, and therefore number of alternatives are still open for exploration.

- The number of results to retrieve (n) is a parameter of SABIR. Our preliminary experiments have shown it impacts classification quality, thus, a deeper study and possibly the proposal of a method for estimating suitable values of n with respect to the characteristics of the dataset.
- In this work, we used BM25 as ranking function, however other alternatives (such as cosine and DFR) could be explored and perhaps combined.
- SABIR could also be explored in other sentiment classification tasks other than tweets. This could be achieved by changing the type of data analysed (reviews instead of tweets, for example) or by changing the type of classification (stance instead of polarity, for example).
- Finally, SABIR-Noisy used automatically labelled tweets, and in nearly all cases was outperformed by the version in which human labelled training data was used. One line of investigation we could pursue could be applying methods to filter misclassified instances hoping to increase classification accuracy.

Acknowledgements

This work was partially supported by CNPq-Brazil (Project No 305141/2015-5). A. U. Kauer received a scholarship by CNPq.

References

Baeza-Yates, R., & Ribeiro-Neto, B. (2011). *Modern information retrieval: The concepts and technology behind Search* (2nd). Addison-Wesley Professional. ACM Press Books)

Bakliwal, A., Arora, P., Madhappan, S., Kapre, N., Singh, M., & Varma, V. (2012). Mining sentiments from tweets. In *Proceedings of the 3rd workshop in computational approaches to subjectivity and sentiment analysis. WASSA '12* (pp. 11–18).

Barbosa, L., & Feng, J. (2010). Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd international conference on computational linguistics: Posters. COLING '10* (pp. 36–44).

Billerbeck, B., Cannane, A., Chattaraj, A., Lester, N., Webber, W., Williams, H. E., et al. (2004). RMIT university at TREC 2004. In *Proceedings text retrieval conference (TREC)*.

Carvalho, J., Prado, A., & Plastino, A. (2014). A statistical and evolutionary approach to sentiment analysis. In *International joint conferences on web intelligence (WI) and intelligent agent technologies (IAT) - volume 02. WI-IAT '14* (pp. 110–117).

Chapelle, O. (2007). Training a support vector machine in the primal. *Neural Computation, 19*(5, May), 1155–1178.

Coletta, L., da Silva, N., Hruschka, E., & Hruschka, E. (2014). Combining classification and clustering for tweet sentiment analysis. In *Intelligent systems (BRACIS), 2014 Brazilian conference on, Oct* (pp. 210–215).

Davidov, D., Tsur, O., & Rappoport, A. (2010). Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd international conference on computational linguistics: Posters. COLING '10* (pp. 241–249).

Deng, Z.-H., Luo, K.-H., & Yu, H.-L. (2014). A study of supervised term weighting scheme for sentiment analysis. *Expert Systems with Applications, 41*(7), 3506–3513.

Fersini, E., Messina, E., & Pozzi, F. (2016). Expressive signals in social media languages to improve polarity detection. *Information Processing & Management, 52*(1), 20–35.Emotion and Sentiment in Social and Expressive Media

Ghiassi, M., Skinner, J., & Zimbra, D. (2013). Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert Systems with Applications, 40*(16), 6266–6282.

Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224n project report*.Stanford 1, 12

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD explorations newsletter, 11*(1), 10–18.

Hu, X., Tang, L., Tang, J., & Liu, H. (2013). Exploiting social relations for sentiment analysis in microblogging. In *Proceedings of the sixth ACM international conference on web search and data mining. WSDM '13* (pp. 537–546).

Lin, J., & Kolcz, A. (2012). Large-scale machine learning at twitter. In *Proceedings of the 2012 ACM SIGMOD international conference on management of data. SIGMOD '12* (pp. 793–804).

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press.

Martìn-Valdivia, M. T., Martìnez-Cámara, E., Perea-Ortega, J.-M., & Ureña López, L. A. (2013). Sentiment polarity detection in spanish reviews combining supervised and unsupervised approaches. *Expert Systems with Applications, 40*(10), 3934–3942.

Martínez-Cámara, E., Martín-Valdivia, M. T., Urena-López, L. A., & Montejo-Ráez, A. R. (2014). Sentiment analysis in twitter. *Natural Language Engineering, 20*, 1–28.

Mitchell, T. M. (1997). Machine learning. *McGraw Hill series in computer science*. McGraw-Hill.

Mostafa, M. M. (2013). More than words: Social networks' text mining for consumer brand sentiments. *Expert Systems with Applications, 40*(10), 4241–4251.

Nicholls, C., & Song, F. (2010). Advances in artificial intelligence: 23rd canadian conference on artificial intelligence, canadian AI 2010. In *Comparison of feature selection methods for sentiment analysis* (pp. 286–289).

O'Keefe, T., & Koprinska, I. (2009). Feature selection and weighting methods in sentiment analysis. In *Proceedings of 14th australasian document computing symposium, Dec.*.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2*(1-2, Jan.), 1–135.

Saif, H., Fernández, M., He, Y., & Alani, H. (2013). Evaluation datasets for twitter sentiment analysis: A survey and a new dataset, the STS-gold. In *Proceedings of the first international workshop on emotion and sentiment in social and expressive media: approaches and perspectives from AI (ESSEM 2013) A workshop of the XIII international conference of the italian association for artificial intelligence (AI*IA 2013)* (pp. 9–21).

Saif, H., He, Y., & Alani, H. (2012). Semantic sentiment analysis of twitter. In *The semantic web–ISWC 2012* (pp. 508–524).

Saif, H., He, Y., Fernandez, M., & Alani, H. (2016). Contextual semantics for sentiment analysis of twitter. *Information Processing & Management, 52*(1), 5–19.Emotion and Sentiment in Social and Expressive Media.

da Silva, N. F., Hruschka, E. R., & Hruschka, E. R., Jr. (2014). Tweet sentiment analysis with classifier ensembles. *Decision Support Systems, 66*, 170–179.

Speriosu, M., Sudan, N., Upadhyay, S., & Baldridge, J. (2011). Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the first workshop on unsupervised learning in NLP. EMNLP '11* (pp. 53–63).

Vosoughi, S., Zhou, H., & roy, d. (September 2015). Enhanced twitter sentiment classification using contextual information. In *Proceedings of the 6th workshop on computational approaches to subjectivity, sentiment and social media analysis* (pp. 16–24).

Weren, E. R., Kauer, A. U., Mizusaki, L., Moreira, V. P., de Oliveira, J. P. M., & Wives, L. K. (2014). Examining multiple features for author profiling. *Journal of Information and Data Management, 5*(3), 266.

Zhang, L., Ghosh, R., Dekhil, M., Hsu, M., & Liu, B. (2011). Combining lexicon-based and learning-based methods for twitter sentiment analysis. *Technical report*.HP, http://www.hpl.hp.com/techreports/2011/HPL-2011-89.html