UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Time-aware online reputation analysis

Peetz, M.-H.

Publication date
2015
Document Version
Final published version

Link to publication

Citation for published version (APA):
Peetz, M.-H. (2015). *Time-aware online reputation analysis*. [Thesis, fully internal, Universiteit van Amsterdam].

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:14 Nov 2026

5

Estimating Reputation Polarity on Microblog Posts

Social media monitoring and analysis has become an integral part of the marketing strategy of businesses all over the world [154]. Companies can no longer afford to ignore what is happening online and what people are saying about their brands, their products and their customer service. With growing volumes of online data it is infeasible to manually process everything written online about a company. Twitter is one of the largest and most important sources of social media data [117]. Tweets can go viral, i.e., get retweeted by many other Twitter users, reaching many thousands of people within a few hours. It is vital, therefore, to automatically identify tweets that can damage the reputation of a company from the possibly large stream of tweets mentioning the company.

Tasks often considered in the context of online reputation management are *monitoring* an incoming stream of social media messages and *profiling* social media messages according to their impact on a brand or company's reputation. We focus on the latter task. In particular, we focus on the problem of determining the *reputation polarity* of a tweet, where we consider three possible outcomes: positive, negative, or neutral. Knowing the reputation polarity of a single tweet, one can either aggregate this knowledge to understand the overall reputation of a company or zoom in on tweets that are dangerous for the reputation of a company. Those tweets need counteraction [254].

The reputation polarity task is a classification task that is similar to, but different in interesting ways, from *sentiment analysis*. For example, a post may have a neutral sentiment but may be negative for reputation polarity. Consider, for instance, the statement *The room wifi doesn't work.*, which is a factual statement that may negatively impact the reputation of a hotel.

There are two standard benchmarking datasets for reputation polarity, the RepLab 2012 dataset [6] and the RepLab 2013 dataset [7], made available as part of RepLab, a community-based benchmarking activity for reputation analysis. In view of the distinction that we have just made between sentiment analysis and reputation polarity, it is interesting to observe that the best performing reputation polarity classifiers at RepLab are sentiment-based. The main research question we address in this chapter is:

RQ2.1 For the task of estimating reputation polarity, can we improve the effectiveness of baseline sentiment classifiers by adding additional information based on the sender, message, and receiver communication model?

The RepLab 2012 and 2013 datasets have different training and testing scenarios: the 2012 dataset uses a training and testing setup that is independent of individual brands or companies ("entities"), while this dependence is introduced in the 2013 dataset. We ask:

RQ2.2 For the task of estimating reputation polarity, how do different groups of features perform when trained on entity-(in)dependent or domain-dependent training sets?

Our last research question is exploratory in nature. Having introduced new features and interesting groups of features, we ask:

RQ2.3 What is the added value of features in terms of effectiveness in the task of estimating reputation polarity?

Without further refinements, **RQ2.3** is a very general research question. One of the contributions of this chapter, however, is the way in which we model the task of determining the reputation polarity of a tweet as a classification problem: we build on communication theory to propose three groups of features, based on the *sender* of the tweet, on the *message* (i.e., the tweet itself), and on the *reception* of the message, that is, how the tweet is being perceived.

While we use and compare some features that are known from the literature [171], a second contribution that we make in this chapter consists of new features to capture the reception of messages—this is where the difference between reputation polarity and sentiment analysis really shows.

Furthermore, as we will see below, reputation polarity class labels are highly skewed and data for some features is missing; our third contribution below consists of an analysis of sampling methods to alleviate the problem of skewness.

Another important contribution that we make concerns the way in which we operationalize the reputation management task. Social media analysts use company-specific knowledge to determine the reputation polarity [57]. In line with this, we discover that sets of tweets pertaining to different entities may be very different in the sense that different features are effective for modeling the reputation polarity. We therefore provide an operationalization of the reputation polarity task using the RepLab 2012 dataset in which we train and test on company-dependent datasets instead of using a generic training set. We find that we can avoid overtraining and that training on far fewer data points (94.4% less) per entity gives up to 37% higher scores. The observation transfers to the RepLab 2013 dataset which is operationalized in precisely that way.

Finally, this chapter adds a new point of view for the business analysis perspective: here our biggest contribution is the difference in performance of features when trained on entity or domain dependent or independent data. Features pertaining to the author of the message seem to be generalizable while others do not.

We proceed with a definition of the reputation polarity task in Section 5.1. Section 5.2 introduces our features and reputation polarity model. We detail our experiments, results and analysis in Section 5.3 and 5.4, respectively. We conclude in Section 5.5.

5.1 Task Definition

The current practice in the communication consultancy industry is that social media analysts manually perform labeling and classification of the content being analysed [6]. Two of the most labour intensive tasks for reputation analysts are *monitoring* and *profiling* of media for a given company, product, celebrity or brand ("entity"). The monitoring task is the (continuous) task of observing and tracking the social media space of an entity for different topics and their importance for the reputation of the entity. Here, the retrieval and aggregation of information concerning the entity is most important. Technically, the monitoring task can be understood as consisting of two steps as follows:

(Cluster) cluster the most recent social media posts about an entity thematically, and

(Rank) assign relative priorities to the clusters.

In this chapter we focus on the profiling task, which is the (periodic) task of reporting on the status of an entity's reputation as reflected in social media. To perform this task, social media analysts need to assess the relevance of a social media post for an entity and the likely implications on the entity's reputation that the post has. Specifically, when working on Twitter data as we do in this chapter, the profiling task consists of two subtasks, i.e., to assess for a given tweet

(Relevance) whether the tweet is relevant to the given entity, and

(Polarity) whether the tweet has positive, negative, or no implications for the entity's reputation.

The relevance assessment subtask is very similar to WePS3 [5] and to the retrieval task assessed at the TREC Microblog 2011 and 2012 tracks [177, 228]. The polarity subtask is new, however, and so far, it has received little attention from the research community. It is a three-class classification task: a tweet can have a *negative*, *positive*, or *no* implication at all (i.e., it is neutral) for the reputation of an entity. This class label is what we call the *reputation polarity* of a tweet.

5.1.1 Reputation Polarity vs. Sentiment

We claimed in the introduction that reputation polarity and sentiment are not the same. We will now substantiate that claim in two ways. One is to see whether there is a correspondence between sentiment classes and reputation polarity classes. In fact, some research suggests that negative online sentiment influences the reputation of a company [182].

From the RepLab 2012 dataset (see Section 3.4.1) we randomly selected 104 tweets from the training set: for all six entities in the training set, we selected up to 6 tweets per reputation polarity class.[1] A group of 13 annotators was then asked to, independently

[1] As we will see below, some classes are underrepresented. We therefore do not always have 6 tweets per class.

from each other, annotate each tweet, indicating whether the sentiment towards the mentioned entity was positive, neutral, or negative.[2] For cases where the tweet is unclear, we added an undefined class.

Now, to get a sense of whether sentiment classes correspond to reputation classes we begin by taking a look at example annotations. As an example, one of the tweets that all annotators agree is neutral for sentiment but negative for reputation polarity is:

> #Freedomwaves - latest report, Irish activists removed from a Lufthansa plane within the past hour. (5.1)

This is a factual, neutral statement, hence the sentiment is neutral. However, the mentioning of an airline together with potential terrorism makes the airline seem unsafe. The reputation polarity for the entity Lufthansa is therefore negative.

Here is a second example to show that reputation class labels and sentiment class labels do not correspond:

> The look at Emporio Armani was inspired by a "Dickens, romantic and punk style" hybrid on Japanese teenagers... (5.2)

There is lots of disagreement concerning the sentiment label of this tweet (6 negative, 1 neutral, 6 positive), while the reputation polarity label is neutral. The sentiment in the tweet is really not clear and, according to our annotators, depends on whether the interpretation of the style is positive of negative. The reputation polarity is considered neutral because it is an objective fact.

Next, we look more formally at the levels of inter-annotator agreement for sentiment and for reputation polarity. According to [6, 57] the annotation of reputation polarity can only be done by experts: they need to know the entities and the current developments in the entities' sectors. We have one expert annotator, separate from the 13 annotators used for sentiment annotations; this single expert annotator annotated for reputation polarity. For sentiment annotation, non-experts can do the annotations at high levels of reliability. As we have a non-fixed number of annotators (see above), we cannot use Cohen's or Fleiss's κ to measure the inter-annotator agreement. We can, however, use Krippendorff's α: we compare the agreement of all 13 sentiment annotators with the average agreement of each of the annotators with the annotator of the reputation polarity. Here we find that the Krippendorff's α score for sentiment annotation is moderate ($\alpha = 0.503$), while the average Krippendorff's α score for reputation polarity is only fair ($\alpha = 0.2869$), thus indicating that we are dealing with two different annotation tasks.

After having defined and motivated the reputation polarity task, we now turn to modeling the task.

5.2 Modeling Reputation Polarity

In this section we provide our model for estimating reputation polarity. For the remainder of the paper we are working with Twitter data; details of our experimental setup are provided in Section 5.3.

[2] http://ilps.science.uva.nl/sites/ilps.science.uva.nl/files/replab2012-sentiment.txt

Table 5.1: Features and types of feature used in the chapter. The acronyms are explained in Sections 5.2.1, 5.2.2, 5.2.3 and 5.2.4.

	Sender	Message	Reception
Baselines			WWL SS
Additional	time zone location user language (ulang) #followers list count verified account age geo enabled username	metadata #punctuation marks (#punct) tweet language (tlang) llr (5) llr (10) textual #hashtags #usernames (#user) #links favourited	I-WWL I-SS I-WWL-RP I-SS-RP

We treat the reputation polarity task as a three-class classification problem. We introduce baseline features based on the literature, i.e., mainly using sentiment classifiers, in Section 5.2.1. We go beyond the baseline features by introducing different types of feature, that we group together in a manner inspired by the transmission model from communication theory [224]. Independently, a similar grouping of features has been used by [19] to manually distinguish opinion and sentiment in news. They analyse annotation procedures and find that three different views need to be addressed. In each communication act, we have a *sender* who sends a *message* and a *receiver* who receives this. So, we have three types of feature:

(Sender) features based on the sender of the tweet that we are trying to classify,

(Message) features based on the (content of the) tweet itself, and

(Reception) features based on the reception of a tweet.

In Section 5.2.2 and 5.2.3 we introduce the sender and message features, respectively. We explain different means to compute reception features in Section 5.2.4. In Section 5.2.5 we explain how we combine the features in a classification paradigm. Table 5.1 provides an overview of our features and their types.

5.2.1 Baseline: Sentiment features

We use two approaches to estimate the sentiment score of a tweet. We start with a simple, but effective, way of estimating the sentiment of short texts that is based on manually created sentiment word lists [147]. After that we consider a more sophisticated approach, based on SentiStrength, a state of the art sentiment analysis tool for social media [240].

We begin by introducing our notation. We use p to denote negative (-1), neutral

(0), or positive (1) reputation polarity of a given tweet.[3] We write W to denote the vocabulary of all words; w stands for an element of W. A tweet T is contained in the set of all tweets $\mathcal{T}$. We also consider the subset $\widehat{\mathcal{T}} \subseteq \mathcal{T}$. This is the subset of tweets for which the reputation polarity needs to be estimated. We write *react*(T) to denote the set of reactions (replies or retweets) available for tweet T. Impact features are learnt with a learning rate δ_i. Specifically, we use a simple linear decay function for our learning rate so that $\delta_i = \delta_0 \cdot \frac{1}{i}$. Finally, we use a polarity filter that returns an item x only if it has the same sign as polarity p:

$$PF(x,p) = \begin{cases} x & \text{if } \mathit{sign}(x) = \mathit{sign}(p) \\ 0 & \text{otherwise,} \end{cases} \tag{5.3}$$

where *sign*(x) is the sign of x.

We write *sent*(T, R) to denote the sentiment of a tweet; superscripts indicate different scoring functions introduced in the following sections: *sent*$^{\text{WWL}}(T, R)$ and *sent*$^{\text{SS}}(T, R)$ use weighted word lists and SentiStrength, respectively. R denotes the term scoring function.

Weighted word lists (WWL) Let *sent_word*(w, p) be the sentiment score of a term w based on sentiment wordlists for different polarities p. This can be the basis of an overall sentiment score, by summing the sentiment of terms:

$$\mathit{sent}^{\text{WWL}}(T, \mathit{sent_word}(\cdot,\cdot)) = \sum_{w \in T} \sum_{p \in \{-1,0,1\}} \mathit{sent_word}(w,p). \tag{5.4}$$

In specific cases in our discussions below, we formalize the association between words and sentiment using a scoring function $R : W \times \{-1, 1\} \to [0, 1]$ that maps a word w and polarity p to *sent_word*(w, p):

$$\mathit{sent}^{\text{WWL}}(T, R) = \sum_{w \in T} \sum_{p \in \{-1,0,1\}} R(w,p). \tag{5.5}$$

Below, we consider different scoring functions R_i, where $R_0(w, p) = \mathit{sent_word}(w, p)$.

SentiStrength (SS) SentiStrength [240] is a word list-based sentiment scoring system. It generates sentiment scores based on predefined lists of words and punctuation with associated positive or negative term weights. Word lists are included for words bearing sentiment, negations, words boosting sentiment, question words, slang and emoticons. The standard setting of SentiStrength has been optimized for classifying short social web texts by training on manually annotated MySpace data. Thelwall et al. [240] provide extensive details of the features used and of the training methodology.

We used the standard out-of-the-box setting of SentiStrength. We write *sent*$^{\text{SS}}(T, R_i)$ to denote usage of SentiStrength with the term weights R_i. The score of a single term w is denoted *sent_word*$^{\text{SS}}(w, \cdot)$.

[3] In sentiment analysis researchers usually only score for negative and positive, assuming that the negative and positive will cancel another out and create a score for neutral [180]. We do the same. The classifier still classifies as -1, 0, or 1.

5.2.2 Sender features

According to social media analysts, the sender is a key factor in determining the impact of a message [57]. How do we capture the sender? Information about the sender can be provided by the sender herself, providing nominal features such as the *time zone* and *location* she is in, and the *language* she speaks. The intuition behind those features is that if a sender located in Europe talks about a brand only distributed in the US in German, this does not impact the reputation of a company as much. It can also be an artefact of her standing in the Twitter community, such as the number of *followers* or the number of *lists* the sender has been added to, both of which are numerical features. Other sender features we use are directly associated with the creation and validation of the account: whether the account has been *verified* (nominal), the *age of the account* (numerical), and whether the automatic transmission of the *geographical location* (nominal) has been enabled. In particular the verification and account age are important to identify spammers: young, unverified accounts are probably more likely to be spam accounts than verified accounts. Verified accounts are never accounts from the general public [249]. The location of the sender the moment the tweet was sent may indicate that she is in the vicinity of the brand, or as mentioned above, in a non-relevant are. All features are encoded in the JSON-formatted data obtained through the Twitter API. The account age is the number of days the account existed prior to the last tweet in the collection.

5.2.3 Message features

Moving on to the message features, we use several metadata message features. We use numerical features derived from tweets such as the number of *links*, *usernames*, and *hashtags*. Those features are extracted from the tweet: usernames begin with an `@`, hashtags with a `#`, and we used regular expressions to extract the number of urls and punctuation marks. The intuition behind the features stems from the idea of monitoring the quality of tweets [266] or the potential of being retweeted [171]. Tweets with many hashtags often hijack trending topics, and are spam-like. Intuitively, they should not have a large impact on the reputation of a company. Similarly, tweets that are of a very colloquial nature do not necessarily have a large impact on the reputation. However, tweets with a question are engaging [171]. The tweet can be *favourited* (a nominal feature) by other users. The number of times a tweet was favourited is a lower bound of the number of times a tweet was actually read. This indicates the reach of a tweet. This information is provided in the JSON formatted data downloaded from Twitter.

We further use textual message features, such as the identified language, the number of punctuation marks, and discriminative terms. We use language identification [46] to identify the *language* (a nominal feature) of the tweet, which may be different to the language set as standard by the user. As our final textual message feature we select discriminative terms. We either use five or ten terms with the highest log likelihood ratio (llr (5), or llr (10)) of the two models built on the texts of messages in the positive and negative classes, respectively, in the training set [155].

5.2.4 Reception features

We move on to our reception features, the third column in Table 5.1. Reception features are meant to estimate how a tweet is being received. An initial operationalization of this idea is simply to determine the sentiment of a tweet. But we do not stop there. In communication theory [24], the reception of a message is said to depend on the responses that it generates, and in particular on the sentiment in these responses (and not just in the originating message). Below, we present an algorithm that aims to capture the iterative nature of this perspective by taking into account the sentiment in the reactions to a message. Here, a reaction to a tweet is either a direct reply or a retweet to the tweet.

Our reception features, then, come in two groups: a group of baseline features that provide initial estimates of the reception of a tweet by computing its sentiment score in different ways (see Section 5.2.1), and a second group that iteratively re-estimates the reception based on the initial estimations provided by the features in the first group. Below, we refer to the first group as *baseline* or *sentiment features* (WWL, SS) and the second group as *impact features* (I-WWL, I-SS, I-WWL-RP, I-SS-RP).

As pointed out above, we assume that a tweet's perceived impact on the reputation of an entity is reflected in the sentiment of the replies that it generates. This estimation, in turn, can be used to update the word lists used for sentiment analysis with the terms in the tweet, assigning the added words to the classes predicted for the tweets in which they are contained. The updated word lists can then be used to re-estimate the impact in the replies of the same tweet, but also other tweets. We assume that the overall, combined reputation polarity of reactions to a tweet is the same as the reputation polarity of the tweet itself.

Essentially, this approach assumes that there is an entity-specific latent word list that denotes different terms for reputation. This list is updated iteratively, so that estimating the impact of a tweet is an process that can be computed using a variant of the Expectation Maximization algorithm described below. Here, the latent parameters are the entity and polarity specific scoring function R based on a word list. The goal of the algorithm is to maximize the *impact*(T) of a tweet T.

Algorithm 1 provides a schematic overview of the process. There are three key phases, initialization, expectation and maximization, which we explain below.

Initialization Recall that we record sentiment scores in a scoring function R; this is the latent scoring function at iteration 0 for the polarity p:

$$R_0(w, p) = PF(\textit{sent}(w), p). \tag{5.6}$$

Maximization The maximization step is the estimation of the impact as solicited in the reactions *react*(T) to a tweet T. To estimate the impact, we estimate the average sentiment of the replies based on iteratively altered word lists. For iterations $i > 0$,

$$\textit{impact}_i(T) = \frac{1}{|\textit{react}(T)|} \sum_{T_r \in \textit{react}(T)} \textit{sent}(T_r, R_{i-1}). \tag{5.7}$$

For the sentiment estimation at every round i, ($\textit{sent}_{i-1}$) we can use the approaches listed in Section 5.2.1. The maximization step can be performed by the sentiment classifier by

Algorithm 1: Impact features, computed using the EM algorithm.

Input: $\mathcal{T}$, the set of all tweets
Input: $\widehat{\mathcal{T}}$, the set of all tweets for which the reputation polarity needs to be estimated
Input: *react*(T), the set of all reactions to tweet T
Input: $\delta_0 < 0$, the learning rate
Input: N, the number of EM-iterations of the algorithm
Input: $P(x, p)$, see Eq. 5.3
Input: C, the scoring system, either WWL or SS
Input: *sent_word*$(\cdot, \cdot)$, the sentiment of a word given a pre-defined word list

Output: *sent*(T, R_N) for all $T \in \widehat{\mathcal{T}}$

// Initialization
$i = 1$
$R_0(w, 1) = PF(\textit{sent}^C(w), 1)$
$R_0(w, -1) = PF(\textit{sent}^C(w), -1)$
$\textit{impact}_0(T) = \frac{1}{|\textit{react}(T)|} \sum_{T_r \in \textit{react}(T)} \textit{sent}^C(T_r, R_0)$ // (Eq. 5.7)
while $i < N$ **do**
    // Expectation
    forall the $p \in \{-1, 0, 1\}$ **do**
        foreach $T \in \mathcal{T}$ **do**
            foreach $w \in T$ **do**
                $\delta_i = \delta_0 \frac{1}{i}$
                $\widehat{R}_i(w, p) = R_{i-1} + \delta_i \frac{1}{|\widehat{\mathcal{T}}|} \sum_{T \in \widehat{\mathcal{T}}} PF(\textit{impact}_i(T), p)$
                // (Eq. 5.8)
                $R_i(w, p) = \frac{\widehat{R}_i(w, p)}{\sum_{w_i \in W} \widehat{R}_i(w_i, p)}$ // (Eq. 5.9)
            end
        end
    end
    // Maximization
    foreach $T \in \mathcal{T}$ **do**
        $\textit{impact}_i(T) = \frac{1}{|\textit{react}(T)|} \sum_{T_r \in \textit{react}(T)} \textit{sent}^C(T_r, R_i)$ // (Eq. 5.7)
    end
end

retraining based on the current word list. We do not retrain; instead, we treat the word lists as the algorithms' own positive and negative word lists.

Estimation The estimation of the latent variable $R_i(w,p)$ for term w and polarity p is done by interpolating the variable $R_{i-1}(w,p)$ with the average polarity of the tweets in which the term occurs. Formally,

$$\widehat{R}_i(w,p) = R_{i-1} + \delta_i \frac{1}{|\widehat{\mathcal{T}}|} \sum_{T \in \widehat{\mathcal{T}}} PF(\textit{impact}_i(T), p), \tag{5.8}$$

where δ_i is the interpolation factor and $\delta_i \leq \delta_{i-1}$. We normalize the scoring function R_i such that

$$R_i(w,p) = \frac{\widehat{R}_i(w,p)}{\sum_{w_i \in W} \widehat{R}_i(w_i,p)}. \tag{5.9}$$

The impact polarity of a tweet T is therefore estimated as $\textit{sent}(T, R_N)$, where N is the number of iterations of Algorithm 1. Using $\textit{sent}(T, R_0)$ is equivalent to simply estimating the sentiment, as explained in Section 5.2.1.

We write I-WWL and I-SS to refer to the impact features as computed by Algorithm 1, where the sentiment of a tweet T has been estimated using the WeightedWorldList ($\textit{sent}^{\text{WWL}}(T,N)$) and SentiStrength ($\textit{sent}^{\text{SS}}(T,N)$), respectively. Similarly, the impact features I-WWL-RP and I-SS-RP use only the replies to tweets in the computation of Algorithm 1.

We detail our parameter settings in Section 5.3.

5.2.5 Classification

As pointed out above, we model the task of estimating the reputation polarity of a tweet as a three-class classification problem. We use decision trees to combine and learn the features. Decision trees are known to perform well when faced with nominal and missing data [211].[4] They are essential to our setting because they are human and non-expert understandable. This characteristic is vital for social media analysts who need to explain successes and failures of their algorithms to their customers.

5.3 Experimental Setup

To answer our research questions as formulated in the introduction, we run a number of experiments. We use two datasets. The first, RepLab 2012, was introduced at CLEF 2012 [6]. Based on lessons learnt, the second dataset, RepLab 2013, was introduced at CLEF 2013 [7]. A detailed description of the datasets can be found in 3.4.1 and 3.4.2. We detail the preprocessing of our data in Section 5.3.1. We then describe our approaches to sampling to address the strong class imbalance in our datasets (Section 5.3.2). Section 5.3.3 outlines the operationalization of the task and the different training procedures. Based on this, our experiments, parameter settings, and evaluation procedures are explained in Sections 5.3.4, 5.3.5, and 5.3.6.

[4] In preliminary experiments on the RepLab 2012 and 2013 datasets, we examined the performance of support vector machines and random forests. Both performed much lower than decision trees, due to a large number of missing features.

5.3.1 Preprocessing

We separate punctuation characters from word characters (considering them as valuable tokens) and keep mentions, hashtags, and smilies intact. Language identification is done using the method described in [46]. We use publicly available sentiment word lexicons in English [109, 148] and Spanish [193] to estimate the weighted word list baselines (WWL).

5.3.2 Sampling

As we will see below, the data that we work with displays a strong imbalance between classes. In particular, far more tweets are labeled with positive than negative reputation in the RepLab 2012 dataset. To deal with this, we consider two strategies, both relying on the following notation. Let S_c be the sample size for each polarity class ($p \in \{-1, 0, 1\}$), and let M denote the size of the largest polarity class. and m denote the size of the smallest polarity class. We *oversample* for each polarity class p by selecting each data point K_p times (where $K_p = \lfloor \frac{M}{S_p} \rfloor$), and pad this with k_p (where $k_p = M \bmod S_p$) randomly sampled data points from the polarity class p. As an alternative we also consider *undersampling* by randomly selecting $m \bmod S_p$ data points from the majority classes until we have at most the number of data points in the minority class [49].

5.3.3 Experiments

Table 5.2 describes different setups we have using the two datasets RepLab 2012 and RepLab 2013. In the following, we describe how the training scenarios and task conditions interplay.

Table 5.2: Different setups datasets, experimental conditions and training scenarios. If a training scenario is possible, this is marked with a ✓.

		Training scenario		
Dataset	Condition	entity-independent	entity-dependent	domain-dependent
RepLab 2012	standard (C2012-1)	✓	–	–
	alternative (C2012-2)	✓	✓	–
RepLab 2013	standard (C2013)	✓	✓	✓

We consider three alternative training scenarios (columns 3–5 in Table 5.2). In one scenario, which we call *entity-independent* (column 3), we follow the official setup provided by RepLab 2012 [6]. Here, training is done on tweets from different entities than the testing (see 3.4.1). There is a natural alternative training scenario. In addressing the polarity detection task, social media analysts tend to make use of different criteria and strategies for different companies; the criteria are often based on customer requests [57]. The *entity-dependent* training scenario (column 4) captures this idea. Here, every company has a separate training set and, in addition to that, an entity-independent training

set can be used for parameter tuning. Finally, in the *domain-dependent* training scenario (column 5), we group data for entities into different domains, i.e., *automotive*, *banking*, *universities*, and *music*. This follows the idea that some features and feature classes can be modelled in a cross-entity manner, but still depend on a specific domain.

Let us look how we can operationalise the training scenarios specifically on the different *datasets* and *task conditions*. For the RepLab 2012 dataset, the standard setting is to learn on a training set consisting of data for six entities that are *independent* from the entities to test on. We call this:

C2012-1 the training and testing condition published for RepLab 2012.

The standard condition for RepLab 2012 is defined as follows: the training set consists of the first 6 entities and testing is being done on the remaining entities.

The standard condition C2012-1 does not follow what has become the customary approach in human online reputation management, where one social media analyst is often responsible for no more than two companies that are followed over an extended period of time [57]. This custom gives rise to a second training and testing that is applicable to RepLab 2012:

C2012-2 an alternative time-based training and testing condition.

The alternative condition for RepLab 2012 is defined as follows. Following [25], we use an incremental time-based split of the testing and training data *per entity*. Here, we sort the tweets according to their time stamps and train the classifier on the first K tweets and evaluate on the next K tweets. We then train on the first $2K$ tweets and evaluate on the next K tweets, etc. The total F-score is the mean F-score over all splits. This also allows for entity-dependent training on the temporally first 50% tweets, without inappropriately "leaking" future tweets into the training set. Additionally, every tweet is only being used once for evaluation. We use $K = 25$. For this scenario we had to discard four more entities (12, 15, 27, 32) because at least one class contained no more than a single tweet. The alternative condition C2012-2 allows for entity-dependent and entity-independent training and testing.

Let us go back to Table 5.2. For the RepLab 2013 dataset, we follow the standard training and testing setup used at RepLab 2013 [7]:

C2013 the training and testing condition published for RepLab 2013.

Here, every entity is part of one of four domains: automotive, banking, universities, and music, see Chapter 3.4.2. Training is performed on data that was published three months before the beginning of the test set: there may therefore be a temporal gap between the training and test set. This training set allows for all three training scenarios: we can do entity-independent training on the full dataset per entity, entity-dependent training on the training data for that specific entity, and domain dependent, combining all training data from the entities of one domain. We do not do C2012-2, the incremental time-based splitting for RepLab2013. Recall that the motivation for C2012-2 was the lack of data for entity-dependent training. However, the dataset RepLab2013 has been carefully designed for the scenario to follow a specific entity over an extended period of time, and providing an entity-dependent training set. The training set was collected three months

before the test set, the dataset therefore features a natural temporal split between training and testing set. Using the original training and testing setting, we ensure comparability of the results.

5.3.4 Parameters

We use the J48 decision tree implementation in Weka [97]. For the impact features (I-WWL, I-WWL-RP, I-SS, I-SS-RP) we train our parameters using cross-validation (CV) using a fold per entity, as we found that leave-one-out CV over-estimates the training performance due to leaking information from tweets that share an entity. The parameters that were selected based on our training procedure are $N = 25$ iterations, a learning rate of $\delta_0 = 1.5$ for I-WWL-RP, I-WWL and $\delta_0 = 1$ for I-SS-RP, I-SS, respectively.

5.3.5 Runs

We test the classification performance (for reputation polarity) using the two baseline features as well as the 21 additional individual sender, message and reception features listed in Table 5.1. In our experiments we also test and compare the performance of the following combinations of features:

(S) all sender features;

(M) all message features;

(R) all reception features;

(S+M) all sender and message features combined;

(S+R) all sender and reception features combined;

(M+R) all message and reception features combined;

(S+M+R) all sender, message and reception features combined;

(FS) feature selection applied to the combination of sender, message and reception features.

For feature selection we generate an ordered list of features where we evaluate the contribution of a feature by measuring its information gain with respect to the class. To find the optimal number of features in a set, we used the decision tree classifier and cross-validation [97].

Our experiments start with a run that does not correct for the class imbalance present in our data. In our experiments we contrast the different ways of correcting for this, through oversampling and undersampling. We also contrast the outcomes of the alternative training scenarios listed in Table 5.2.

5.3.6 Evaluation and Significance Testing

We present evaluation scores on the overall output of our classification experiments (with English and Spanish results combined, as per the instructions at RepLab 2012 and RepLab 2013). Our main metric is the F-score. We use other metrics such as balanced accuracy (BAC) or reliability and sensitivity (R and S, respectively) where appropriate. We

use the Student's t-test to evaluate weak and strong significance ($p < 0.05$ and $p < 0.01$, respectively). We apply Bonferroni corrections to account for multiple comparisions.

As over- and undersampling introduce a certain level of randomness, results for approaches that use over- or undersampling are repeated 100 times. We report the average F-score over those 100 runs and include the standard deviation of the results.

5.4 Results and Analysis

We start with an initial experiment that motivates our use of sampling in Section 5.4.1 and analyse the overall performance with respect to the sentiment baselines and baselines from the literature 5.4.2. In Section 5.4.3 we analyse how the different training scenarios influence the results and Section 5.4.4 discusses single features in depth.

5.4.1 Preliminary Experiment: Sampling

Table 5.3 shows the F-scores for the entity-independent and dependent training scenario in the alternative training and testing condition on the RepLab 2012 dataset (C2012-2). We observe that when no sampling is being done on the training set, the negative and neutral polarity classes have an F-score of 0. Table 5.4 shows the F-scores for the entity-independent and dependent, as well as for the domain-dependent training scenario in the standard training and testing condition on the RepLab 2013 dataset (C2013). Here as well, we observe that when no sampling is being done on the training set, the negative and neutral polarity classes have an F-score of 0.

Table 5.3: Classification results for reputation polarity, entity-independent and entity-dependent training scenarios, in the *alternative training and testing condition (C2012-2)*, using all features (S+M+R), and performing no sampling, oversampling and undersampling on the training sets for RepLab 2012. Total F-score and broken up for different reputation classes (-1, 0, 1). The column labeled "#ent w/0" shows the average number of entities where at least one class has an F-score of 0.

	entity-independent					entity-dependent		
	-1	0	1	all	BAC	all	#ent w/0	BAC
no sampling	0.0000	0.0000	0.5091	0.2629	0.3517	0.6199	21.68	0.4474
oversampling	0.1767	0.3892	0.3403	0.3522	0.3421	0.5548	16.89	0.4653
undersampling	0.3434	0.0000	0.0000	0.1534	0.3040	0.4387	14.83	0.4264

Figure 3.1 shows that the class of negative reputation polarity is indeed underrepresented in the training set for RepLab 2012 (but overrepresented for RepLab 2013, see Figure 3.2). For the entity-dependent training condition we cannot measure this directly, as the class distributions differ over the different entities. Table 5.3 shows the number of entities where at least one class had an F-score of 0; this is the case for more than 70% of the entities. For the RepLab 2013 data, Table 5.4 shows that without sampling, all entities have at least one class with an F-score of 0. In Figure 3.1 and 3.2 we see that the class

Table 5.4: Classification results for reputation polarity, entity-independent, entity-dependent, and domain dependent training scenarios, in the *standard training and testing condition for RepLab 2013 (C2013)*, using all features (S+M+R), and performing no sampling (*ns*), oversampling (*os*) and undersampling (*us*) on the training sets for RepLab 2013. Total F-score and broken up for different reputation classes (-1, 0, 1). The column labeled "#ent w/0" shows the average number of entities where at least one class has an F-score of 0.

	entity-independent					entity-dependent			domain-dependent		
	-1	0	1	all	BAC	all	#ent w/0	BAC	all	#ent w/0	BAC
ns	0.0000	0.0000	0.7336	0.4251	0.4575	0.6190	61.00	0.6023	0.4886	61.00	0.5168
os	0.2567	0.2273	0.5539	0.4498	0.4380	0.4433	10.79	0.4412	0.4455	1.25	0.4453
us	0.2966	0.2794	0.4986	0.4089	0.4443	0.2928	35.08	0.3721	0.3719	26.09	0.4105

distributions of the entities are far from balanced. And indeed, the training and testing set have a lot of missing feature values (see Section 3.4.2 in Chapter 3). This motivates the use of sampling methods. Table 5.3 shows that oversampling distributes the performance better over different classes, while undersampling does not: in cases with too little and missing training data undersampling does not help but hurts. Based on these observations, we use oversampling for all further experiments. For the RepLab 2013 data, Table 5.4 shows that while undersampling helps ameliorating the effect of skewed class distributions on this dataset, oversampling results in lower entities with at least one F-score of 0 and has a higher F-score in general. We therefore also use oversampling for experiments on the RepLab 2013 data.

5.4.2 Overall Performance

With our preliminary experiment out of the way, we turn to our first research question, which we repeat here for convenience:

RQ2.1 Can we improve the effectiveness of baseline sentiment classifiers by adding additional information based on the sender, message, and receiver communication model?

We answer this research question based on the RepLab 2012 and RepLab 2013 datasets. The first column with numerical results in Table 5.5 shows the F-scores for different features and groups of features using an entity-independent training scenario based on RepLab 2012. It displays results based on the oversampling method. Testing has been done on the entire test set. Two of our runs outperform the highest score achieved at RepLab 2012 (0.495): feature selection on all features (FS, 0.4690) and the group of message features (M, 0.5562).[5] Table 5.7 gives an overview of significant differences between groups of features for C2012-1. We see that the group of message features

[5]It should be noted that the best performing system at RepLab 2012 is a closed source, knowledge intensive system for which details are not publicly available [6], so that further detailed comparisons are not possible.

(M) and feature selection based on all features (FS) perform significantly better than the strong SS baseline. All feature combinations outperform WWL. We see that the combination of feature groups works best: in particular the message group (M) and the reception group (R). Only using the sender features (S) decreases the results significantly. Feature selection including mostly I-WWL and #links performs significantly better than most other feature combinations, except for using just the message features (M).

The second column with numerical results in Table 5.5 displays the F-score using the entity-independent training scenario and evaluating on the same incremental time splits as in the entity-dependent setting. The third column with numerical results in Table 5.5 shows the F-scores for the entity-dependent training scenario, in the alternative training and testing condition (C2012-2), for different features and feature groups, using oversampling. The two columns labeled C2012-2 are not comparable to C2012-1. Table 5.8 gives an overview of significant differences between groups of features for C2012-2. For one, nearly every feature group performs significantly better than the baselines WWL and SS (only S performs worse than SS, and R does not perform significantly better). Secondly, in the entity-dependent training scenario, feature selection and most feature groups perform significantly better than the baseline features. Similar to the entity-independent runs in C2012-1 (see Table 5.7), we have significant improvements of the message features (M) and feature selection (FS) over the baseline features.

Next, we turn to the RepLab 2013 dataset. Table 5.6 shows the F-scores for the RepLab 2013 dataset following the entity-dependent, entity-independent, and domain-dependent training scenarios, for different features and feature groups, using oversampling. We find that, in the entity-dependent training scenario, our best feature group (M+R; F-score 0.5532) outperforms the best performing run at RepLab 2013 (SZTE NLP; F-score 0.38).[6] As to significant differences between feature groups, Table 5.10 shows that the feature group M+R performs significantly better than any other feature group and the baselines. In particular, the feature groups M and M+R, and applying feature selection (FS) perform significantly better than the baseline SS. Every feature group performs significantly better than the WWL baseline.

Additionally, we significantly outperform the baselines (see Table 5.9) with several feature groups in the entity-independent training scenario as well.

To conclude our discussion of RQ1, our best runs always perform better, for both RepLab 2012 and 2013, than the best performing runs found in the literature. Compared to the sentiment baselines, most of our feature groups perform significantly better than just using sentiment in the entity-dependent case on both datasets.

5.4.3 Entity-Independent vs. Entity-Dependent vs. Domain-Dependent

We turn to our second research question, which we repeat for convenience:

RQ2.2 How do different groups of features perform when trained on entity-(in)dependent or domain-dependent training sets?

Figure 5.1 compares the F-scores for different entities for the different feature groups for the entity-independent training scenario in the C2012-1 training and testing condition. We see that different feature groups affect different entities differently. The message feature group (M) is very strong on nearly all entities, but not for all (e.g., entity RL2012E36, *CaixaBank*), while the other feature groups vary strongly across entities.

[6] As an aside, in terms of the other metrics used at RepLab 2013, reliability and sensitivity, M+R also outperforms the best performing run at RepLab 2013. For reliability, M+R achieves 0.57 vs. 0.48 for SZTE NLP, and for sensitivity M+R scores 0.41 vs. 0.34 for SZTE NLP.

Table 5.5: Classification results (as F-scores) for reputation polarity, using oversampling for RepLab 2012. Entity-dependent (only alternative condition) and Entity-independent formulation (standard and alternative condition), with the test set based on incremental time based splitting. The numbers between C2012-1 and C2012-2 are not comparable per row. For each column, the first (second) number is the mean (standard deviation) of the 100 runs. The baseline features are included in the group of reception features R.

		C2012-1	C2012-2	
		entity-indep.	entity-indep.	entity-dep.
Baseline features	Random	0.3344 ±0.0068	0.3344 ±0.0068	0.3340 ±0.0073
	WWL	0.1414 ±0.0016	0.1536 ±0.0012	0.3850 ±0.0101
	SS	0.4186 ±0.0223	0.2771 ±0.0542	0.3959 ±0.0807
Sender features	followers	0.4231 ±0.0188	0.2777 ±0.0537	0.3953 ±0.0801
	verified	0.1904 ±0.0538	0.2718 ±0.0491	0.3877 ±0.0732
	location	0.2899 ±0.0450	0.2766 ±0.0527	0.3890 ±0.0742
	time zone	0.3450 ±0.0394	0.2841 ±0.0554	0.3895 ±0.0753
	ulang	0.3185 ±0.0484	0.2855 ±0.0569	0.3923 ±0.0772
	geo en.	0.3196 ±0.0591	0.2867 ±0.0580	0.3947 ±0.0792
	list. cnt	0.4104 ±0.0344	0.2833 ±0.0551	0.3910 ±0.0760
	acc. age	0.4027 ±0.0340	0.2870 ±0.0583	0.3898 ±0.0755
	user	0.2326 ±0.0735	0.2269 ±0.0456	0.3923 ±0.0772
Message features (metadata)	#links	0.4244 ±0.0073	0.2883 ±0.0597	0.3976 ±0.0817
	#usernames	0.3738 ±0.0075	0.4048 ±0.0095	0.4325 ±0.0109
	#hashtags	0.3162 ±0.0135	0.2857 ±0.0570	0.3938 ±0.0783
	favourited	0.1409 ±0.0000	0.2834 ±0.0548	0.3899 ±0.0754
Message features (textual)	#punct	0.3924 ±0.0209	0.2880 ±0.0594	0.3984 ±0.0824
	tlang	0.3794 ±0.0087	0.2838 ±0.0551	0.3886 ±0.0745
	llr (5)	0.3081 ±0.2118	0.4395 ±0.0480	0.6032 ±0.0053
	llr (10)	0.3168 ±0.1842	**0.4463 ±0.0990**	0.5873 ±0.0112
Reception features	I-WWL	0.2630 ±0.0916	0.2516 ±0.0513	0.3797 ±0.0836
	I-SS	0.3160 ±0.0462	0.2635 ±0.0532	0.4768 ±0.0658
	I-WWL-RP	0.2828 ±0.0825	0.2869 ±0.0583	0.3962 ±0.0804
	I-SS-RP	0.3448 ±0.0009	0.2774 ±0.0535	0.3918 ±0.0767
Groups of features	S	0.3596 ±0.0387	0.2843 ±0.0556	0.3975 ±0.0816
	M	**0.5562 ±0.0489**	0.4290 ±0.0441	0.6000 ±0.0056
	R	0.3906 ±0.0192	0.2104 ±0.0570	0.3936 ±0.0781
Combinations of groups	S+M	0.2403 ±0.0630	0.3824 ±0.0675	0.5557 ±0.0088
	S+R	0.3355 ±0.0476	0.2887 ±0.0581	0.4737 ±0.0640
	M+R	0.4197 ±0.0291	0.4085 ±0.0509	0.5870 ±0.0067
All	S+M+R	0.3413 ±0.0465	0.3522 ±0.0337	0.5548 ±0.0088
	FS	0.4690 ±0.0752	0.4202 ±0.0743	**0.6495 ±0.0092**

Table 5.6: Classification results (as F-scores) for reputation polarity, using oversampling. Entity-dependent, Domain-dependent, and Entity-independent formulation on RepLab 2013, for the *standard training and testing condition for RepLab 2013 (C2013)*. The baseline features are included in the group of reception features R. For each column, the first (second) number is the mean (standard deviation) of the 100 runs.)

		Training scenario		
		entity-indep.	entity-dep.	domain-dep.
Baseline features	Random	0.3576 ±0.0018	0.3574 ±0.0018	0.3575 ±0.0018
	WWL	0.1539 ±0.0357	0.1324 ±0.0701	0.1542 ±0.0417
	SS	0.4591 ±0.0029	0.4344 ±0.0867	0.4778 ±0.0180
Sender features	followers	0.3848 ±0.0497	0.4951 ±0.0404	0.4063 ±0.0412
	verified	0.1672 ±0.0139	0.1272 ±0.0588	0.1285 ±0.0038
	location	0.3376 ±0.0886	0.2906 ±0.0966	0.3243 ±0.0884
	time zone	0.3710 ±0.0607	0.3266 ±0.0627	0.3682 ±0.0728
	ulang	0.4555 ±0.0044	0.2619 ±0.0859	0.2773 ±0.0713
	geo en.	0.3831 ±0.0038	0.2959 ±0.1275	0.3017 ±0.1037
	list. cnt	0.2809 ±0.0592	0.2662 ±0.0340	0.4190 ±0.0459
	acc. age	0.4406 ±0.0333	0.4951 ±0.0403	0.4394 ±0.0358
Message features (metadata)	#links	0.3632 ±0.0045	0.3368 ±0.0675	0.3476 ±0.0141
	#usernames	0.1522 ±0.0192	0.2778 ±0.1270	0.1928 ±0.0582
	user	0.0000 ±0.0000	0.1850 ±0.0782	0.2029 ±0.0758
	#hashtags	0.3635 ±0.0178	0.2761 ±0.0854	0.2878 ±0.0558
	favourited	0.0680 ±0.0000	0.0680 ±0.0000	0.0680 ±0.0000
Message features (textual)	#punct	0.3416 ±0.0149	0.4216 ±0.0565	0.3793 ±0.0334
	tlang	0.4649 ±0.0034	0.2798 ±0.0944	0.4053 ±0.0077
	llr (5)	0.3537 ±0.0053	0.3845 ±0.0257	0.3763 ±0.0109
	llr (10)	0.3690 ±0.0058	0.4023 ±0.0327	0.3782 ±0.0172
Reception features	I-WWL	0.1624 ±0.0435	0.1173 ±0.0377	0.1768 ±0.0941
	I-SS	0.3129 ±0.0140	0.3433 ±0.1201	0.3725 ±0.0794
	I-WWL-RP	0.1594 ±0.0067	0.1188 ±0.0380	0.1798 ±0.1141
	I-SS-RP	0.3461 ±0.0040	0.3303 ±0.1213	0.2698 ±0.0675
Groups of features	S	0.4260 ±0.0173	0.4021 ±0.0502	0.4327 ±0.0153
	M	0.4599 ±0.0391	0.5019 ±0.0291	0.4466 ±0.0286
	R	0.4163 ±0.0287	0.4908 ±0.0513	0.4082 ±0.0276
Combinations of groups	S+M	0.4421 ±0.0111	0.4202 ±0.0537	0.4365 ±0.0158
	S+R	0.4479 ±0.0144	0.4298 ±0.0571	0.4447 ±0.0207
	M+R	**0.4935 ±0.0137**	**0.5532 ±0.0190**	**0.5037 ±0.0165**
All	S+M+R	0.4498 ±0.0107	0.4433 ±0.0595	0.4455 ±0.0240
	FS	0.3780 ±0.1654	0.5224 ±0.0390	0.4292 ±0.0702

Table 5.7: Significant differences, entity-independent training scenario for RepLab 2012 in the standard training and testing condition (*C2012-1*). Row $<$ ($>$) Column means that Row is statistically significantly worse (better) than column. A * indicates weak significance ($p > 0.05$) and $\sim$ no significant differences.

	WWL	SS	S	M	R	S+M	S+R	M+R	S+M+R
SS	$>$								
S	$>$	$\sim$							
M	$>$	$>$	$>$						
R	$>$	$\sim$	$>$	$<$					
S+M	$>$	$<$	$<$	$<$	$<$*				
S+R	$>$	$\sim$	$>$	$<$	$\sim$	$>$			
M+R	$>$	$\sim$	$>$	$<$	$\sim$	$<$	$>$		
S+M+R	$>$	$\sim$	$>$	$<$	$\sim$	$<$	$\sim$	$<$	
FS	$>$	$>$	$>$	$<$	$\sim$	$>$	$>$*	$\sim$	$>$*

This suggests that the estimation of reputation polarity is indeed very entity-specific and better performance can be reached by training per entity.

Table 5.5 shows the F-scores for the entity-independent training scenario, in the alternative training and testing condition, for different features, feature groups and their combinations, using oversampling. It also displays the F-score using the entity-dependent training scenario on the same incremental time splits as in the entity-independent setting. On average, the size of the training sets for the entity-dependent training is 5.6% (88.46 tweets) of the size of the training set in the entity-independent 2012 training. In general, entity-dependent training leads to better F-scores on the RepLab 2012 dataset: for all but one feature (*#usernames*) the F-score is significantly higher in the entity-dependent training setting than in the entity-independent setting. The average increase in F-score in the entity-dependent training scenario is 31.07%, with the best runs increasing by 35.30% (FS) and 36.51% (S+M+R) over the entity-independent training scenario.

The different columns in Table 5.6 compare the runs based on the entity-independent, entity-dependent, and domain-dependent training scenarios for the RepLab 2013 dataset. Again, we find that the entity-dependent training scenario leads to better results than the domain-dependent and entity-independent training scenarios, even though the latter two training scenarios have more training data. This is especially true for relatively strongly performing features and feature groups, such as the message group (M) and the reception group (R). We see that for relatively weak features (with an F-score below 0.4), the domain-dependent and entity-independent training scenarios lead to better results in 80% of all cases.

The sender group (S) itself, and all combinations that extend the sender group (S, S+M, S+R, and S+M+R) are weaker in the entity-dependent training scenario, but stronger for the entity-independent and domain-dependent training scenario. This suggests

Table 5.8: Significance, for the entity dependent training scenario, using oversampling for RepLab 2012, in the alternative training and testing condition *C2012-2*. Row $<$ ($>$) Column means that Row is statistically significantly smaller (larger) than column. A * indicates weak significance ($p > 0.05$) and $\sim$ no significant differences.

	WWL	SS	S	M	R	S+M	S+R	M+R	S+M+R
SS	$\sim$								
S	$\sim$	$\sim$							
M	$>$	$>$	$>$						
R	$\sim$	$\sim$	$>$	$<$					
S+M	$>$	$>$	$>$	$<$	$>$				
S+R	$>$	$>$	$>$	$<$	$>$	$<$			
M+R	$>$	$>$	$>$	$\sim$	$>$	$>$*	$>$		
S+M+R	$>$	$>$	$>$	$<$	$>$	$\sim$	$>$	$\sim$	
FS	$>$	$>$	$>$	$>$	$>$	$>$	$>$	$>$	$>$

that a more general model can be built to model the sender, possibly to support the entity-dependent training scenario. In the entity-dependent training scenario, the reception features are on par with message features and their combination leads to the strongest performing feature group. Figure 5.2 shows the performance broken down per domain for the entity-dependent training scenario. We see that for two domains the reception feature group (R) performs better than the message feature group, but the sender (S) and combined feature group (S+M+R) never outperform the other groups.

To conclude our discussion of **RQ2.2**, we have found that, in general, the entity-dependent training scenario yields higher effectiveness than the entity-independent or domain-dependent training scenarios, while using much less data on two datasets. For some features and feature groups like the sender group, the domain-dependent training scenario leads to better performance, which suggests that the sender aspect of reputation polarity is entity-independent.

5.4.4 Feature Analysis

We turn to our final research question, **RQ2.3**, which we repeat for convenience:

RQ2.3 What is the added value of features in terms of effectiveness in the task of estimating reputation polarity?

We start from the observation that the sentiment feature itself (SS) already outperforms the best run at RepLab 2013 [7]. We analyse the contribution of our other features and feature groups in the entity-independent and entity-dependent training scenarios for the RepLab 2012 dataset in Section 5.4.4 and Section 5.4.4, respectively, and for the RepLab 2013 dataset in Section 5.4.4.

Table 5.9: Significance, for the entity independent training scenario, using oversampling for RepLab 2013. Row $<$ ($>$) Column means that Row is statistically significantly smaller (larger) than column. A * indicates weak significance ($p > 0.05$) and $\sim$ no significant differences.

	WWL	SS	S	M	R	S+M	S+R	M+R	S+M+R
SS	$>$								
S	$>$	$\sim$							
M	$>$	$\sim$	$>$						
R	$>$	$>$	$\sim$	$\sim$					
S+M	$>$	$\sim$	$>$*	$<$	$\sim$				
S+R	$>$	$\sim$	$>$	$\sim$	$\sim$	$\sim$			
M+R	$>$	$>$	$>$	$>$	$>$	$>$	$>$		
S+M+R	$>$	$\sim$	$>$	$\sim$	$\sim$	$>$	$\sim$	$<$	
FS	$>$	$<$	$>$	$<$	$\sim$	$<$	$<$	$<$	$<$

Entity-independent training on RepLab 2012

As we see in the first result column in Table 5.5, for the standard setting C2012-1 on the RepLab 2012 dataset, the strongest single features are the number of links in a tweet (*#links*), the number of followers (*followers*) and the baseline feature *SS*. We see that every feature group has at least one strong feature.

Figure 5.3 shows a snapshot of an example decision tree that is based on a randomly selected oversampled training set in the entity-independent training scenario. The first decision made is whether the sentiment classification is positive: if so, the tweet has neutral reputation polarity. If the impact (I-WWL-RP) is positive, the reputation polarity is negative (but this affects only very few examples and can be considered spurious). The last decision is based on the occurrence of the term `http` in the tweet: If this occurs, it has positive reputation, otherwise it is negative.

Table 5.5 shows that for the entity-independent training scenario the performance of the impact features (I-WWL, I-WWL-RP) using a weak sentiment classifier (WWL) increase performance, and significantly so. With a strong underlying sentiment classifier (SS) the performance of the impact features (I-SS, I-SS-RP) decreases. For the entity-dependent training scenario, however, compared to the sentiment baselines, the performance increases significantly for I-SS and I-WWL-RP, but does not change significantly for the other impact features. The strongest single features are the number of followers a user has (*followers*), whether the user was added to a list (*listed count*), the age of the account (*account age*), and the number of links in a tweet (*#links*). All features relate to the authority of a user or the authority of a tweet. A tweet with exactly one link tends to contain more information and often links to news [183]. Additionally, we can see that the combination of LLR terms (i.e., llr (5) or llr (10)) does not perform well individually.

Table 5.10: Significance, for the entity dependent training scenario, using oversampling for RepLab 2013. Row $<$ ($>$) Column means that Row is statistically significantly smaller (larger) than column. A * indicates weak significance ($p > 0.05$) and $\sim$ no significant differences.

	WWL	SS	S	M	R	S+M	S+R	M+R	S+M+R
SS	$>$								
S	$>$	$<$*							
M	$>$	$>$	$>$						
R	$>$	$<$	$>$	$\sim$					
S+M	$>$	$\sim$	$\sim$	$<$	$<$				
S+R	$>$	$\sim$	$>$	$<$	$<$	$\sim$			
M+R	$>$	$>$	$>$	$>$	$>$	$>$	$>$		
S+M+R	$>$	$\sim$	$>$	$<$	$>$	$\sim$	$>$*	$<$	
FS	$>$	$>$	$>$	$\sim$	$\sim$	$>$	$>$	$<$	$>$

We can also see that combining different feature groups decreases performance as compared to the single feature group and as we can see in Table 5.7, significantly so. The best combination of features is the message feature group (M) alone and neither feature selection (FS) nor combining all features can improve on this.

Let us try to understand why feature selection does not perform that well based on the features it selects. Table 5.11 shows features ranked by information gain for a random oversampled training set. The most important features are mainly textual in nature: reception features and the number of links in a tweet. In nearly all cases, the feature selection selected the impact feature based on the weighted word list sentiment classifier (I-WWL, see Eq. 5.5 and 5.7) plus the number of links (*#links*) in a tweet and some LLR terms. Surprisingly, the impact features I-WWL and I-WWL-RP are the most important features: as shown in Table 5.5, they are not among the strong features. This shows that the training and test set are very different and models learnt on the training set are not optimal for the test set: using clear textual features like the message features, we are not prone to overfitting on a training set that does not quite fit the test set.

Let us now illustrate the effect of our impact features. Consider the following tweet:

> #spain's repsol threatens companies investing in seized ypf with legal actions.

Based on the original sentiment word list, this tweet would be classified as neutral for reputation polarity. However, there is a different tweet in the collection:

> repsol threatens to *sue* firms that help argentina.

Due to the term *sue*, the sentiment of this tweet is negative. This tweet was retweeted and as retweets are a reply to the tweet, the term *argentina* gets a negative connotation.

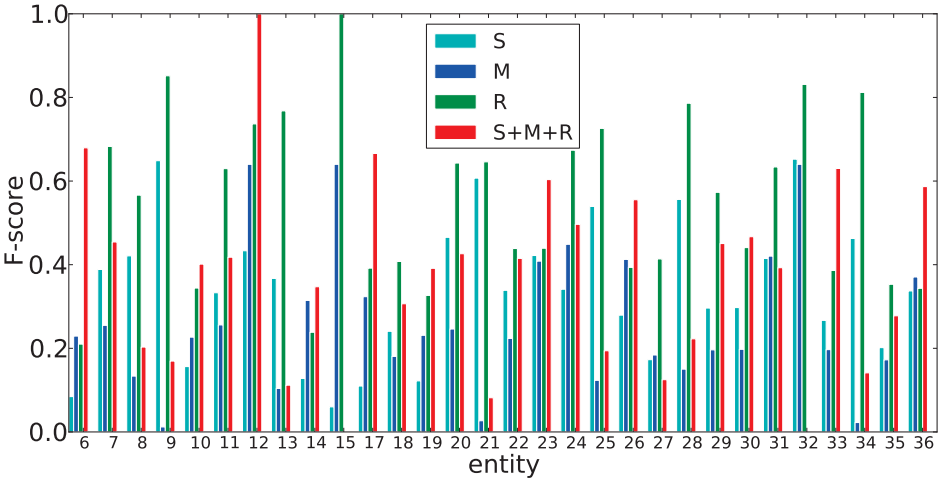

Figure 5.1: F-scores for different entities in the test set and different feature groups S, M, R, and S+M+R, in cyan, blue, green, and red, respectively. This is based on the *standard training and testing condition (C2012-1)*, with oversampling on the training set, for RepLab 2012.

The term *legal* often co-occurs with the term *argentina*. After 5 iterations of the Expectation Maximization (EM) algorithm the terms *legal* and *YPF* have strongly negative connotations. And indeed, this topic has a very negative reputation score: it is about Repsol loosing a company it acquired (YPF) due to expropriation by the Argentinian government. For a second example, consider the example tweet 5.1. The sentiment annotation by SS and WWL is neutral. However, after several iterations of the EM, I-WWL learnt the term *report* to be negative. Similarly, typical positive terms for an airline entity turn out to be *profit, truelove,* or *overjoyed*. Figure 5.4 shows how this effects the classifications by comparing the classifications of SS (sentiment) and I-SS-RP (impact). Impact "dampens" sentiment and measures something different: not all tweets with a polarized sentiment are classified with a polarized impact. In fact, we can see that the impact actually has very few negatively polarized tweets.

Figure 5.5 shows the development of the F-scores over the number of iterations of the EM algorithm (Algorithm 1) on the test data, when using different sets of reactions: all reactions (I-WWL, I-SS), only replies (I-WWL-RP, I-SS-RP), and only retweets (I-WWL-RT, I-SS-RT). Using the retweets for estimation (in the complete set and on its own), it takes much longer until convergence of the F-score: after 5 iterations of the EM, I-WWL-RP and I-SS-RP have nearly reached a plateau, while the other two estimators reach a plateau only after around 10 iterations. In general, the performance drops after 25 (WWL) and 30 (SS) iterations of the EM: it drops earlier for WWL because we use a higher value of δ_0 ($\delta_0 = 1.5$ vs. $\delta_0 = 1$) to discount the influence of what has been learnt in the previous iterations.

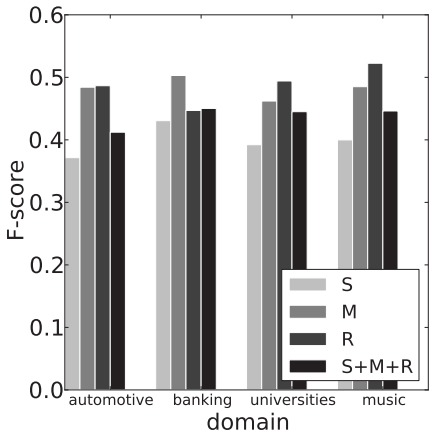

Figure 5.2: F-scores for different domains in the test set (in groups) and different feature groups S, M, R, and S+M+R, in white, light grey, and black, respectively. The results are based on the entity-dependent training scenario, oversampling on the training set, for RepLab 2013, in the *C2013* training and testing condition.

Entity-dependent training on RepLab 2012

We now switch to the entity-dependent training scenario and detail the performance of the features in the alternative training and testing condition, C2012-2. A closer look at the third numerical column in Table 5.5 shows that some single features perform exceptionally well. The first are the simple textual log-likelihood features (llr (5) and llr (10)), where using the top ten terms performs better than the top five. This feature also performs very well in the entity-independent training scenario. As it does not perform that well in the standard evaluation condition C2012-1, it may very well be an artefact of the test set. The second is the impact feature I-SS. This feature performs significantly better than every single feature, except for #usernames and the llr features. Finally, the feature #usernames performs very well too. Additionally, Table 5.8 shows that combining the weak sender feature group (S) with the reception group (R) improves the results.

Feature selection helps and we now examine which features were selected by the feature selection algorithm. We say that a feature is *frequent* for an entity if it was selected in more than 50% of all splits and runs. When we examine which features were selected by the feature selection algorithm for each individual entity, we see very varied feature sets: the mean number of features used is 11.18 $\pm$6.41. The most frequent features are the impact features I-WWL and I-WWL-RP, and the number of punctuation which are all frequent for 83.3% of the entities. In over 50% of all runs the sentiment features SS and WWL, as well as the number of followers, and the username were used. The two most commonly learnt feature sets were I-WWL together with at least one message (M) feature.

Let us provide an example. While feature selection selects the number of followers, punctuation, and hashtags, as well as I-SS for the entity RL2012E24 (*Bank of America*

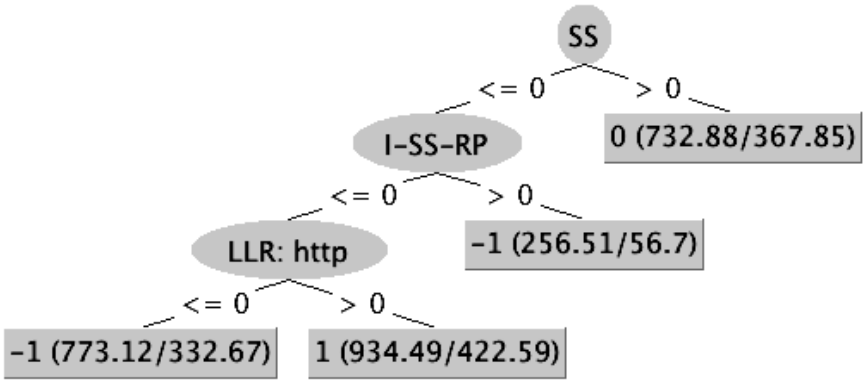

Figure 5.3: An example decision tree created by a randomly selected oversampled training set in the entity-independent training scenario for RepLab 2012, in the C2012-1 condition. The oval nodes represent features being tested, the edges possible decisions, and the rectangular nodes the assignment (with proportion of correctly classified tweets). Note how this is easy to interpret by social media analysts.

Corporation), it selects the username as most important feature for entity RL2012E35 (*Microsoft Corporation*). Indeed, for Microsoft Corporation, 93% of the usernames in the test set already appeared in the training set and 7 out of 10 of the LLR terms. For some companies, it therefore seems more important *who* says something than *what* they say. To analyse the reputation for the Bank of America Corporation, this is different: there is not much overlap in the user base, and it seems more important *what* they say and whether they support it with authority (*#links*, *#followers*). In other words, the reputation of different entities may depend on different factors—this reflects the practice described by Corujo [57]: reputation analysts treat each entity differently.

The improved performance under the entity-dependent training scenario over the entity-independent scenario does not mean that we cannot generalize across multiple entities: we can generalize to other entities within the same domain. In the RepLab 2012 test set, we identified three market domains amongst the entities: banking, technology, and cars (see Appendix 3.B). For the banking domain, the frequent features overlap: apart from the reception features, in particular impact (I-WWL-RP and I-WWL), the number of followers feature is frequent for all entities in this domain. Again, what people say and their authority is important. In the technology domain the textual message features are important, in particular punctuation. In the car domain, we see that the most common features are the impact features and textual features (such as *#punct*), but not the terms selected by llr.

To conclude, the impact feature and the textual features improve results in the entity-dependent training scenario and it is, in fact, best put to use in an entity-dependent man-

Table 5.11: The contribution of a feature based on the information gain with respect to a class, doing cross-validation on a randomly selected oversampled training set for the entity-independent training scenario, for English on RepLab 2012, in the condition *C2012-1*.

information gain	feature
1.287 ±0.007	username
0.701 ±0.026	I-WWL-RP
0.540 ±0.030	WWL
0.455 ±0.005	location
0.418 ±0.047	followers
0.231 ±0.008	WWL
0.204 ±0.053	account age
0.190 ±0.005	1st LLR term (`http`)
0.155 ±0.004	2nd LLR term (`co`)
0.142 ±0.004	# punctuation
0.141 ±0.004	3rd LLR term (`alcatel`)
0.130 ±0.004	# links
0.121 ±0.005	SS
0.102 ±0.002	time zone
0.051 ±0.002	ulang
0.044 ±0.002	10th LLR term (`patent`)
0.069 ±0.042	listed count
0.033 ±0.006	# hashtags
0.029 ±0.001	8th LLR term (`mobile`)
0.025 ±0.001	tlang
0.023 ±0.004	#usernames
0.020 ±0.001	9th LLR term (`app`)
0.015 ±0.001	5th LLR term (`lucent`)
0.015 ±0.001	I-SS
0.014 ±0.001	I-SS-RP
0.013 ±0.001	6th LLR term (`pingit`)
0.005 ±0.001	geo enabled
0.002 ±0.001	4th LLR term (`t`)
0.001 ±0.000	verified
0.000 ±0.000	7th LLR term (`microsoft`)
0.000 ±0.000	favourited

ner. We also find that reputation polarity and sentiment are different: often, the impact of the tweet and the authority of users matter more than its content.

Standard training condition on RepLab 2013

Finally, we turn to an analysis of the relative effectiveness of features on the RepLab 2013 dataset. The WWL feature performs very poorly. This is only a surprise if we look at the results for RepLab 2012. In general however, WWL is not a strong sentiment classifier [180]. We see that for WWL in RepLab 2013, the deviation in the classification

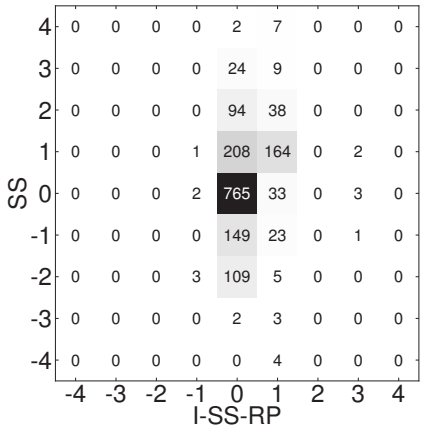

Figure 5.4: The comparision of sentiment values (SS) with impact value (I-SS-RP) on the training set of RepLab 2012.

is very low ($\pm$0.00002081), while for RepLab 2012 it is higher, namely $\pm$0.053, while for SS the deviation is $\pm$1.100 and $\pm$1.108 for RepLab 2012 and RepLab2013, respectively. This means that a lot of terms in the tweets were not part of the word lists and most tweets were classified neutral. Similar to the RepLab 2012 dataset, the impact features on the RepLab 2013 dataset are not as strong on their own. Having very few replies for the test dataset (as compared to RepLab 2012) harms the estimation of the impact.

Figure 5.6 shows the features selected by the feature selection algorithm under the entity-dependent training scenario for the RepLab 2013 dataset. The selected features vary between domains and between entities. The most frequent features selected vary strongly between the training methods. For the entity-dependent training scenario, # followers, account age, and the two impact features I-WWL and I-WWL-RP were selected in 89.6%, 88.9%, 74.8%, and 73.9% of all cases, respectively. The ten most common binary feature combinations are always with at least one of the two impact features, I-WWL and I-WWL-RP. For the domain-dependent training scenario this is different. Again, # followers and account age are very frequent (86.9% and 82.7%), however, the follow-up features are the location and some llr terms. For the entity-independent training scenario, feature selection is not really doing much: 82.8% of all features are selected in all, or all but one run.

In sum, the impact features are selected frequently. We also observe a strong difference in best performing features between the training scenarios.

To conclude this section, we have seen that our approach to polarity prediction performs better than the baselines for every dataset and under all training scenarios. We can clearly see that the more focussed the training set is, the better the training works: even when the amount of training material is much smaller. Finally, we have seen that different features stand out: while impact features alone do not perform well, they are very helpful in combination with features from the message group (M).

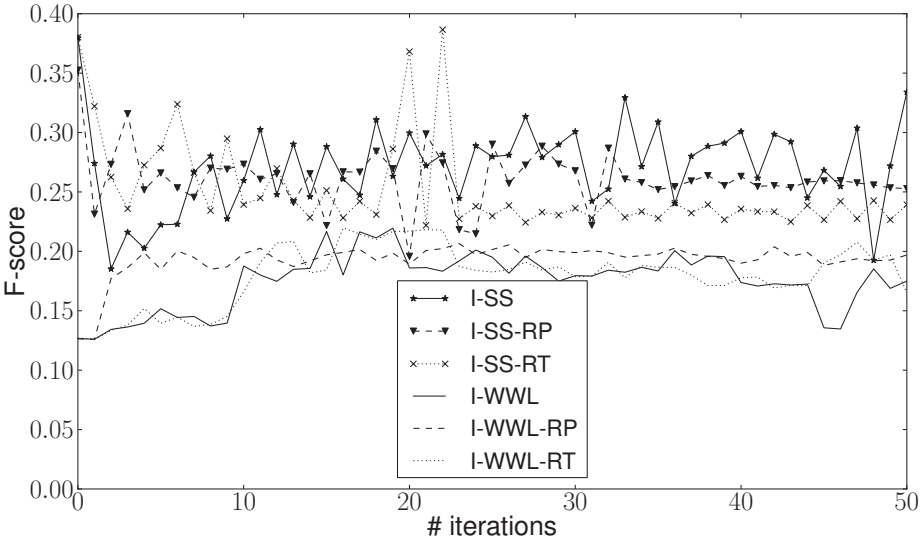

Figure 5.5: Development of F-scores for different iterations of the impact feature, with oversampling for RepLab 2012, in condition *C2012-1*. I-WWL-RT and I-SS-RT are the impact features using only retweets as reactions.

5.5 Conclusion

We have presented an effective approach to predicting reputation polarity from tweets. Starting from the observation that reputation polarity prediction is different from sentiment analysis, annotators agree moderately on sentiment, but their sentiment annotation only agrees fairly with the annotation of reputation polarity. Similarly, we find that sentiment classifiers are not enough to classify reputation polarity. We use three groups of features based on intuitions from communication theory and find that features based on authority and reactions from users perform best. We consider three training scenarios for the reputation polarity task, entity-independent, entity-dependent, where one trains the models in an entity-dependent manner, and domain-dependent, where training depends on the domain of the entity. While training on far less (94% less) data, entity-dependent training leads to an improvement of 25% over models trained in an entity-independent manner. We find that the selected features are diverse and differ between entities. From this, we conclude that predicting reputation polarity is best approached in an entity-dependent way. On the RepLab 2012 dataset, we find that training per domain instead of entity looks promising for some features. This is confirmed on the RepLab 2013 dataset, where we see that for sender features, training on domain specific datasets does help. We also find that to alleviate the imbalance of real-life data, oversampling is important. Finally, we find that our results transfer to a different and larger dataset, with consistent findings between the 2012 and 2013 editions of the RepLab datasets.

As to future work, we aim to look at sampling methods that take into account miss-

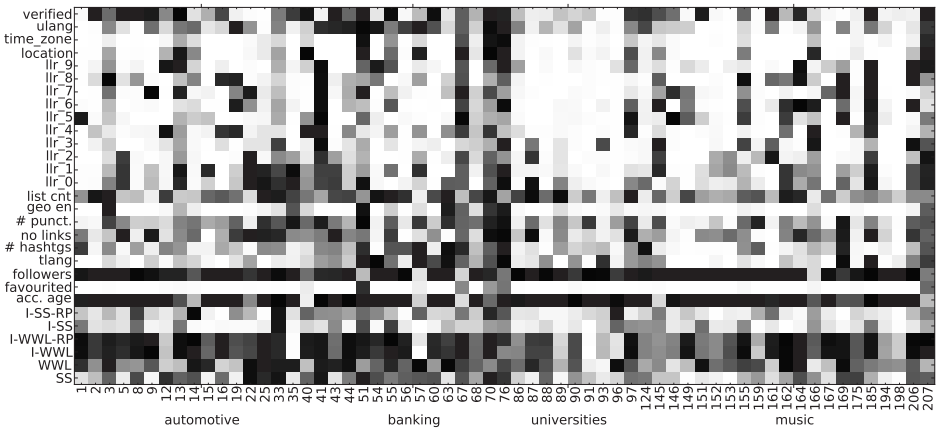

Figure 5.6: Selection of features per entity in the entity-dependent training scenario. Entities are plotted along the x-axis, features along the y-axis. The darkness indicates in how many of the 100 test runs a feature was selected per entity.

ing feature values, as this seems to be a problem for oversampling. With more data, language (thus culture) dependent analysis becomes more feasible. On the RepLab 2013 dataset, we can see a hint that reputation polarity with respect to the sender may be entity-independent. This hint and potential findings can be used for the RepLab2014/PAN task of author profiling and ranking [9]. In return, a successful author profiling approach can feed back at to the classification approach presented in this work. At RepLab2014 [9], a new dataset for the classification into different dimensions was introduced. We are curious in how far dimension and/or entity-dependent training combined with cascading algorithms may improve the results.

Future work will also concern the analysis of reactions to a tweet and how diverse they are with respect to sentiment. Additionally, active learning of reputation polarity may incorporate a constant influx of user feedback and may deal with changes of language over time. Finally, with respect to time, it would be interesting to perform longitudinal studies and see how our impact features can be adjusted for temporal changes.