UNED Online Reputation Monitoring Team at RepLab 2013*

Damiano Spina, Jorge Carrillo-de-Albornoz, Tamara Martín, Enrique Amigó, Julio Gonzalo, and Fernando Giner
{damiano,jcalbornoz,tmartin,enrique,julio}@lsi.uned.es, fginer3@alumno.uned.es

UNED NLP & IR Group
Juan del Rosal, 16
28040 Madrid, Spain
http://nlp.uned.es

Abstract. This paper describes the UNED's Online Reputation Monitoring Team participation at RepLab 2013 [3]. Several approaches were tested: first, an instance-based learning approach that uses Heterogeneity Based Ranking to combine seven different similarity measures was applied for all the subtasks. The filtering subtask was also tackled by automatically discovering filter keywords: those whose presence in a tweet reliably confirm (positive keywords) or discard (negative keywords) that the tweet refers to the company [16]. Different approaches have been submitted for the topic detection subtask: agglomerative clustering over wikified tweets, co-occurrence term clustering [10] and an LDA-based model that uses temporal information. Finally, the polarity subtask was tackled by following the approach presented in [14] to generate domain specific semantic graphs in order to automatically expand the general purpose lexicon SentiSense [9]. We next use the domain specific sub-lexicons to classify tweets according to their reputational polarity, following the emotional concept-based system for sentiment analysis presented in [8]. We corroborated that using entity-level training data improves the filtering step. Additionally, the proposed approaches to detect topics obtained the highest scores in the official evaluation, showing that they are promising directions to address the problem. In the reputational polarity task, our results suggest that a deeper analysis should be done in order to correctly identify the main differences between the Reputational Polarity task and traditional Sentiment Analysis tasks. A final remark is that the overall performance of a monitoring system in RepLab 2013 highly depends on the performance of the initial filtering step.

* This research was partially supported by the Spanish Ministry of Education (FPU grant nr AP2009-0507 and FPI grant nr BES-2011-044328), the Spanish Ministry of Science and Innovation (Holopedia Project, TIN2010-21128-C02), the Regional Government of Madrid and the ESF under MA2VICMR (S2009/TIC-1542) and the European Community's FP7 Programme under grant agreement nr 288024 (LiMoSINe).

1 Introduction

This paper describes the UNED Online Reputation Monitoring Team participation in RepLab 2013, which is focused on organizing and classifying tweet streams associated with an entity of interest, in order to facilitate to the analysts the labor of monitoring the reputation of an entity in Twitter. We have participated in four of the five subtasks proposed in the evaluation campaign: filtering unrelated tweets, classifying tweets according to its reputational polarity, detecting topics discussed in tweets and the full monitoring task.

The RepLab 2013 collection consists of 61 entities from four different domains: automotive, banking, university and music. Each of the entities has an associated set of around 750 manually annotated tweets for training and 1,500 tweets for testing. Tweets are written in English and Spanish, following the same (unbalanced) language distribution as in Twitter. Crawling was performed during the period from the 1st June 2012 till the 31st Dec 2012 using the entitys canonical name as query (e.g. "BMW"), and training/test datasets correspond to different time ranges. The corpus also comprises additional background tweets for each entity (up to 50,000, with a large variability across entities).

We have applied a range of different approaches to each of the tasks, plus a horizontal approach for all of them. The filtering task has been tackled with keyword recognition based techniques learned from each entity. Polarity has been addressed with semantic graphs for domain-specific affective lexicon adaptation. In the case of topic detection, a revisited version of our three algorithms presented in RepLab 2012 [10] have been tested again in this edition. The horizontal approach consists of an extended instance-based learning method over the training corpus in which the similarity is measured over multiple tweet extensions (author tweets, external link, etc.).

The paper is organized as follows. We describe the approaches and the results for the RepLab 2013 subtasks: filtering in Section 2, polarity in Section 3 and topic detection in Section 4. Then, we analyze the results for the full monitoring task in Section 5. We conclude in Section 6.

2 Filtering Subtask

The filtering task in RepLab 2013 is oriented to disambiguate tweets in order to discard those that do not refer to the company (i.e., related vs. unrelated). Here we describe the two different approaches we have tested to tackle this problem: instance-based learning over heterogeneity based ranking and filter keywords.

2.1 Instance-based Learning over Heterogeneity Based Ranking

The instance-based learning approach that we have tested is similar to the official RepLab 2013 baseline, where each tweet inherits the (manually annotated) tags from the most similar tweet in the training corpus of the same entity. The difference is that, instead of using Jaccard distance to compute tweet similarity as the baseline does, we have employed and combined multiple similarity measures. Our measures expand the tweets with different sources, and then apply cosine similarity. We have used the following sources to expand tweets: (i) the hashtags in the tweets, (ii) the content of the URLs linked in the tweets, (iii) the rest of tweets published by the same author and (iv) several parts of the wikipedia entries associated with words in the tweet (e.g. title, wikipedia category, etc). As one word can be associated with multiple Wikipedia entries, we have used *commonness probability* [11] for disambiguation (described in Section 4.2).

In order to combine all similarity measures, we have employed an unsupervised method called Heterogeneity Based Ranking (HBR) [5]. Basically, this method considers the *heterogeneity* (a notion of diversity or disimilarity) of the set of measures that corroborate that two texts are more similar to each other than other pair of texts. It consists of the following steps: first, for each tweet we consider the most similar tweets in the training corpus according to the Jaccard distance. Then, for each measure, we re-rank all these distances obtaining 100 similarity instances for each measure. Then, we apply the unsupervised ranking fusion algorithm HBR to combine all the rankings. Finally, we take the tag from the winner tweet. When we have no information to compute a given measure (e.g. a tweet with no external links), we assign the average similarity between the tweets in the corpus that do contain this information.

2.2 Filter Keywords Strategy

The filter keywords strategy has proved to be a competitive approach to company name disambiguation in Twitter [16]. A positive/negative filter keyword is an expression that, if present in a tweet, indicates a high probability that the tweet is related/unrelated to the company.

Here we explain the automatic classification approach, similar to [16], that we have tested on the RepLab 2013 Filtering subtask. At a glance, it consists of two steps: first, filter keywords are discovered by using machine learning algorithms (*keyword discovery*); second, tweets containing positive/negative filter keywords are used to feed a model that classifies the uncovered tweets (*tweet classification*).

Keyword Discovery. Given the tweet stream of an entity and its representative pages (i.e., the homepage and the English and Spanish Wikipedia pages), each term is represented by features that take into account the company's website, Wikipedia, Open Directory Project (ODP) and the RepLab 2013 collection itself[1].

Three families of features are defined:

- **Collection-based features:** This features are defined to capture differences between keywords and skip terms. In this approach, we added two additional specificity-based features, pseudo-document TF.IDF and KLD [15,10], to the existing set of collection-based features described in [16].

[1] See [16] for details about how the features are computed.

– **Web-based features:** This features should discriminate between positive and negative filter keywords. These features are: term frequency on the representative pages (homepage and the English and Spanish Wikipedia pages), as well as term occurrence in relevant search results in Wikipedia and in ODP.
– **Features expanded by co-occurrence:** We applied the co-occurrence expansion described in [16] to all the features enumerated above.

Then, terms in the training set are labeled as positive/negative/skip by considering the precision of the tweets covered by the term:

- If 85% of the tweets containing the term are RELATED, the term is considered as a positive filter keyword;
- if 85% of the tweets are UNRELATED then the term is labeled as a negative filter keyword;
- in other case, the term is labeled as a skip term.

Finally, the labeled instances described above are used to feed a positive-negative-skip classifier. We combine two classifiers: positive versus others and negative versus others, using the confidence thresholds learned by the classifiers (i.e., those used by default to decide the final label of each instance). Terms which are simultaneously under/over both thresholds are tagged as skip terms.

Tweet Classification. After classifying the terms, tweets containing at least one of those terms labeled as positive (negative) keywords are straightforwardly classified as related (unrelated), respectively As classified keywords are unlikely to cover all the tweets, we use a standard bootstrapping method to annotate the uncovered tweets. Tweets are represented as Bag-of-Words (produced after tokenization, lowercase and stop word removal) and term occurrence is used as weighting function; finally, a supervised machine learning algorithm is used to classify the tweets.

The tweet classification process has been carried out at the entity level, that is, for each entity we use the tweets retrieved by the keywords as seed, only using the training set of the entity, in order to classify automatically the remaining tweets of the entity.

2.3 Submitted Runs

Table 1 provides an overview of the filtering runs, showing the type of training data used in each of them. Run `UNED_ORM_filtering_1` is the instance-based learning over HBR, that works at the entity level as described in 2.1. Runs `UNED_ORM_filtering_3`, `UNED_ORM_filtering_4` and `UNED_ORM_filtering_5` follow the filter keyword approach described in Section 2.2, considering different training data to build the model used in the keyword classification step: `UNED_ORM_filtering_3` joins all the terms from the different entities in the training dataset to build a single model, while `UNED_ORM_filtering_4` and

UNED_ORM_filtering_5 build a specific model at the entity level. The difference between them is that, while UNED_ORM_filtering_5 only considers entity-specific data, UNED_ORM_filtering_4 uses exactly the complementary. The latter run simulates the semi-supervised scenario in which the system does not use any previously annotated data about the target entity (like in previous evaluation campaigns [2,4]). Since in this edition of RepLab, annotated data about the target entity is available, using this data instead of filter keywords to feed the tweet classification step will give us an idea of the performance that we can reach. For each entity, the run UNED_ORM_filtering_2 uses the tweets from the training data to learn the model that will directly classify all the tweets in the test data.

Table 1. Overview over the runs submitted for the filtering subtask.

	training data		
	entity (supervised)	all (supervised)	leave-entity-out (semi-supervised)
instance-based learning + HBR	UNED_ORM_filt_1		
filter keywords	UNED_ORM_filt_5	UNED_ORM_filt_3	UNED_ORM_filt_4
filter keywords (tweet classif. only)	UNED_ORM_filt_2		

In all the filter keywords' runs (from 2 to 5), tweets are tokenized using a Twitter-specific tokenizer [13], frequent words are removed using both English and Spanish stop word lists, and terms occurring less than 5 times in the collection are discarded. The machine learning algorithm used in all the experiments was Naïve Bayes, using the implementation provided by the Rapidminer toolkit [12].

2.4 Results

Table 2 reports the scores obtained for the evaluation metrics used in the filtering subtask: Accuracy, Reliability (R), Sensitivity (S) and $F_1(R, S)$. For each of the runs, the position on the official $F_1(R, S)$ RepLab rank is also shown.

Even if the results obtained in terms of accuracy are competitive, the scores according to Reliability and Sensitivity measures evidence the need for a better understanding of the problem. Comparing the results obtained with the baseline, only the run UNED_ORM_filtering_2 outperforms it in terms of accuracy and $F_1(R, S)$, which proves that there is still room for improvement.

As expected, runs that use previously annotated data from the entity are significantly better than semi-supervised approaches. A deeper analysis of the machine learning steps is needed to understand the actual limitations of the filter keywords approach.

Table 2. Results of the runs submitted for the filtering subtask.

Run	**Accuracy**	**Reliability** (R)	**Sensitivity** (S)	$F_1(R,S)$	**Rank**
`UNED_ORM_filtering_2`	0.8587	0.4254	**0.3840**	**0.3382**	19
baseline	**0.8714**	**0.4902**	0.3200	0.3255	21
`UNED_ORM_filtering_1`	0.8733	0.4732	0.3272	0.3019	27
`UNED_ORM_filtering_5`	0.8423	0.6742	0.2584	0.2539	42
`UNED_ORM_filtering_4`	0.5020	0.1696	0.2870	0.1429	61
`UNED_ORM_filtering_3`	0.5026	0.1713	0.2869	0.1428	62

3 Polarity Subtask

Most of the approaches that aim to detect polarity in texts make use of affective lexicons in order to identify opinions, sentiments or emotions. The main drawback of affective lexicons' development is the manual effort needed to generate good quality resources. Besides, these resources should be designed for a general purpose, not taking into account the peculiarities of a specific domain. The RepLab 2013 dataset is a perfect evaluation framework to study and analyze automatic methods for domain adaptation of affective lexicons.

Sentiment Analysis (SA) and Reputational Polarity (RP) are close but different tasks. While the first is mainly focused on identifying subjective content in product/services reviews, the second one aims to measure if a text has positive or negative implications for a company's reputation. This definition includes the well known, but less studied in SA, polar facts. Statements such as *"Report: HSBC allowed money laundering that likely funded terror, drugs, ..."* are clearly negative when analyzing the reputation of HSBC company, even though no opinion is expressed. The effect of polar facts can be reasonable biased using a domain-specific affective lexicon. In the example, a system using a lexicon that attaches to the words *money laundering*, *terror* or *drugs* a negative polarity or emotion will correctly classify this fact.

Within this premise our aim in this approach is to evaluate if the use of semantic graphs and word sense disambiguation [14] will help to generate domain-specific affective lexicons. Following this idea we generate domain specific semantic graphs in order to expand the existing lexicon SentiSense [9]. To classify tweets with reputational polarity we have used the emotional concept-based system for sentiment analysis presented in [8] and adapted it to work with English and Spanish texts simultaneously [7].

3.1 Semantic Graphs for Domain-Specific Affective Lexicon Adaptation

Our hypothesis is that using a semantic graph, where the nodes represent concepts and the edges represent semantic relations between concepts, the emotional meaning of previously labeled concepts can be spread to other concepts not labeled and that are strongly related in meaning. Even if this technique can be applied to a general purpose dataset, it seems that a domain specific dataset will produce better results due to the similarity in vocabulary of the different documents. To this end, we followed the approach presented in [14], where the main points are the use of a word sense disambiguation algorithm to properly obtain the correct sense of each word and the use of different measures of text similarity and WordNet relations between concepts to generate a semantic graph of a document. This graph is next used to identify the different topics that are dealt with in the document and to extract the most representative sentences in the document to generate an automatic summary. Instead of manually labeling the initial seed of concepts with emotional meaning, we use the SentiSense affective lexicon as a seed.

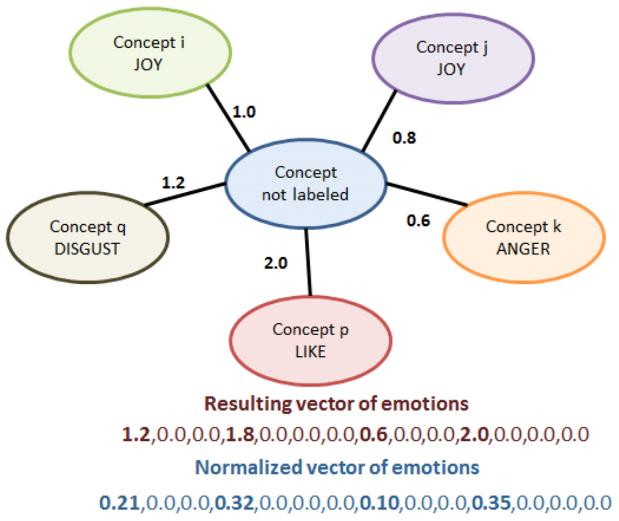
The method proposed for domain adaption of affective lexicons consists of three steps:

- **Concept Identification.** In order to determine the appropriate sense of each word in WordNet we use the UKB algorithm [1] that is available both for English and Spanish. According to this, in this step each tweet in the dataset is represented as WordNet concepts. In this step also all hypernyms of such concepts are retrieved in order to enrich the graph generation. Note that only nouns, verbs, adjectives and adverbs are used to generate the graph. Also, a list of stop words has been used to remove non-relevant words.
- **Graph generation.** We have analyzed different relations between concepts as studied in [14,7] in order to determine which are suitable to propagate the emotional meaning to closely related concepts. As the dataset consists of tweets and the number of words per tweet is 10 words average, using only semantic relations from WordNet produces very unconnected graph. For this reason, we have also included a co-occurrence relation that links concepts which co-occur between 3 to 10 times in the corpus. Our experiments with the training sets reveal that the co-occurrence relation and the WordNet relations of *hypernymy*, *antonymy* and *derived from* are the most appropriate for spreading the emotional meaning between concepts. We use similar weights as the proposed in [14,7] to ponder the different relations in the graph. To this end, we weight with 1.0 each pair of concepts related by the WordNet pointer *derived from* and with −1.0 for the antonymy one. Following the idea that a hypernym is a generalization of a concept, the weight assigned to these relations follow the equation 1. Finally, we weight with 1.0 each co-occurrence of two concepts.

$$weight(C_i; E_j) = 1/(depth(hyper_i) + 1) \quad (1)$$

- **Propagating emotions to new concepts.** Finally, in this step the semantic graph is used to extend the emotional categories to new concepts not labeled in SentiSense. To this end, a concept C_i not previously labeled in SentiSense is labeled as follows:
  - For all the incoming links that represent relations of *co-occurrence*, *hypernymy* and *derived from*, and that connect C_i with concepts of a same

Fig. 1. Example of propagating emotions to new concepts step



emotional category, the weights of all links are added. For the *antonymy* relations, the emotional categories are replaced by their antonyms, as given in the SentiSense lexicon. As a result, for the concept C_i we get a vector of 14 positions, each representing a SentiSense emotion, where the vector positions represent the weight of each emotion in the concept. It is important to note that these weights are domain-specific, since are derived from the domain semantic graph.

- The concept is finally labeled with multiple emotional categories, which weights are normalized at the concept level, so that the sum of the weights of all the emotional categories associated to the concept is 1.0. Figure 1 shows an example of an expanded concept.

We have tested different approximations to generate the domain-specific semantic graphs using the training set of RepLab 2013. Our first approximation used all entities of the same domain to generate the graph and to adapt the affective lexicon of SentiSense. We evaluated different approaches: using just related tweets of the training set, using all tweets of the test set, and using both related tweets of the training set and all tweets of the test set. We have also tested generating graphs at the entity level, that is to say, generating a graph for each entity.

3.2 Emotional concept-based system for Sentiment Analysis

The resulting domain-specific affective lexicons are used to identify emotions in the tweets of RepLab 2013 dataset using the emotional concept-based system for sentiment analysis presented in [8,7] and described here for clarification:

- **Pre-processing: POS Tagging and Concept Identification.** The objective of the first step is to translate each text to its conceptual representation in order to work at the concept level in the next steps and avoid word ambiguity. To this aim, the input text is split into sentences and the tokens are tagged with their POS. With this information, the system next maps each token to its appropriate WordNet concept using the UKB algorithm [1].
- **Emotion Identification.** Once the concepts are identified, the next step maps each WordNet synset to its corresponding emotional category in the SentiSense affective lexicon, if any. In this steps the different generated domain specific lexicons generated using the semantic graph approach are used.
- **Post-processing: Negation and Intensifiers.** In this step, the system has to detect and solve the effect of negations and intensifiers over the emotions discovered in the previous step. This process is important, since these linguistic modifiers can change the polarity and intensity of the emotional meaning of the text.To this end, our system first identifies the presence of modifiers using a list of common negation and intensification tokens. In such a list, each intensifier is assigned a value that represents its weight or strength. The scope of each modifier is determined using the syntax tree of the sentence in which the modifier arises. We assume as scope all descendant leaf nodes of the common ancestor between the modifier and the word immediately after it, and to the right of the modifier. However, this process may introduce errors in special cases, such as subordinate sentences or those containing punctuation marks. In order to avoid this, our method includes a set of rules to delimit the scope in such cases. These rules are based on specific tokens that usually mark the beginning of a different clause (e.g., *because*, *until*, *why*, *which*, etc.). Since some of these delimiters are ambiguous, their POS is used to disambiguate them. Once the modifiers and their scope are identified, the system solves their effect over the emotions that they affect in the text. The effect of negation is addressed by substituting the emotions assigned to the concepts by their antonyms. In the case of the intensifiers, the concepts that fall into the scope of an intensifier are tagged with the corresponding percentage weight in order to increase or diminish the intensity of the emotions assigned to the concepts.
- **Classification.** In the last step, all the information generated in the previous steps is used to translate each text into a Vector of Emotional Intensities (VEI), which will be the input to a machine learning algorithm. The VEI is a vector of 14 positions, each of them representing one of the emotional categories of the SentiSense affective lexicon. The values of the vector are generated as follows:
  - For each concept, C_i, labeled with an emotional category, E_j, the weight of the concept for that emotional category, $weight(C_i; E_j)$, is set to 1.0.

- If no emotional category was found for the concept, and it was assigned the category of its first labeled hypernym, $hyper_i$, then the weight of the concept is computed as:

$$weight(C_i; E_j) = 1/(depth(hyper_i) + 1) \tag{2}$$

- If the concept is affected by a negation and the antonym emotional category, $E_a nton_j$, was used to label the concept, then the weight of the concept is multiplied by $\alpha = 0.6$. This value has been empirically determined in previous studies. It is worth mentioning that the experiments have shown that α values below 0.5 decrease performance sharply, while it drops gradually for values above 0.6.
- If the concept is affected by an intensifier, then the weight of the concept is increased/decreased by the intensifier percentage, as shown in Equation 3.

$$weight(C_i; E_j) = weight(C_i; E_j) * (100 + intensifier\_percentage)/100 \tag{3}$$

- Finally, the position in the VEI of the emotional category assigned to the concept is incremented by the weight previously calculated.

3.3 Results

The polarity detection task in RepLab 2013 consists of classifying tweets as positive, negative or neutral. It is important to notice that the polarity is oriented to reputation. That is, an emotionally negative tweet is not necessarily negative from the reputation point of view (i.e. "I'm sad. Michael Jackson is dead"). Note that tweets that are unrelated according to human assessors are not considered in the evaluation.

For the individual evaluation of the reputational polarity task, performance is evaluated in terms of accuracy (% of correctly annotated cases) over related tweets. Reliability (R) and sensitivity (S) are also included for comparison purposes [6]. Overall scores (accuracy, R, S, F(R,S)) are computed as the average of individual scores per entity, assigning the same weight to all entities.

As the system for reputational polarity is a supervised method, we have tested different machine learning algorithms in order to determine the best classifier for the task. In our experiments the logistic regression model (*Logistic*) as implemented in Weka with default parameters has obtained the best results. As previously mentioned we tested different approaches for generating the domain-specific semantic graph, however we only could submit the runs using the graph generated using related tweets of the training set and at the domain level. So, the runs submitted are:

– `UNED_ORM_polarity_1`: This run uses the approach described in section 2.1, which is similar to the baseline but using different text similarity measures and the Heterogeneity Based Ranking approach to combine them.

– `UNED_ORM_polarity_2`: This run uses the system presented in [7] trained at the entity level.
– `UNED_ORM_polarity_3`: This run uses the system presented in [7] trained at the entity level, but using a balanced training instead of the whole training set.
– `UNED_ORM_polarity_4`: This run uses the system presented in [7] trained at the entity level but using the domain-specific SentiSense lexicon generated with the graph resulted of all tweets related of the training set of the same domain.
– `UNED_ORM_polarity_5`: This run uses the same system as `UNED_ORM_polarity_4`, but using a balanced training set instead of the whole training set.

Table 3. Results of the runs submitted for the polarity subtask.

Run	**Accuracy**	**Reliability** (R)	**Sensitivity** (S)	$F_1(R,S)$	**Rank**
`Best system`	0.6859	0.4765	0.3360	0.3770	1
`UNED_ORM_polarity_2`	**0.6164**	**0.3551**	0.1049	0.1472	21
`UNED_ORM_polarity_4`	0.6153	0.3325	0.1100	0.1445	22
`UNED_ORM_polarity_1`	0.5872	0.3164	**0.2906**	**0.2984**	26
`BASELINE`	0.5840	0.3151	0.2899	0.2973	28
`UNED_ORM_polarity_3`	0.5821	0.3388	0.1172	0.1618	31
`All positives`	0.5774	0	0	0	36
`UNED_ORM_polarity_5`	0.5746	0.3244	0.1423	0.1761	37

As shown in Table 3.3, the best performing configuration (`UNED_ORM_polarity_2`) is that which has been trained at the entity level and that uses the original SentiSense lexicon (without expanding it using semantic graphs). However, the difference with the (`UNED_ORM_polarity_4`) system is not significant. Both approaches perform better than the baseline in terms of accuracy, while their performance drops when evaluating with reliability and sensitivity measures. Even if a similar approach has been tested achieving promising results in [8], our results obtained in the RepLab 2013 dataset suggest that a deeper analysis should be done in order to correctly understand the task. Our intuition is that analyzing reputational polarity in Twitter slightly differs from analyzing reviews from films or news. First, the text in tweets contains multiple errors and misspellings, so the error introduced in the linguistic process (POS analysis, parsing, word sense disambiguation, etc.) is bigger than in well structured texts such as news or reviews. Second, we found that most of the vocabulary used contains positive emotional meaning, so this positivity is propagated to other concepts with this methods obtaining vectors of emotions with mostly positive emotions, which is translated in an over learning in the positive class. Finally, as in the filtering subtask, combining multiple similarity measures for instance-based learning slightly outperforms the baseline that only considering Jaccard similarity over the terms.

4 Topic Detection Subtask

This subtask consists of grouping entity related tweets according to topics. The organization does not provide any set of predefined topics. Given that the training corpus corresponds with a previous time range, new topics can appear in the test set. It is assumed that each tweet belong to one topic. In the following sections we describe the different approaches proposed for the topic detection subtask: LDA-based clustering, tweet wikified clustering and term clustering. Finally, we show the results obtained by the runs submitted.

4.1 LDA-based Clustering

We use an LDA-based model in order to obtain the topics of a collection of tweets. It is an unsupervised machine learning technique which uncovers information about latent topics across the corpora. The model is inspired by the TwitterLDA model [18] and TOT model [17]. TwitterLDA is an author-topic model that is based in the assumptions that there is a set of topics K in Twitter, each represented by a word distribution and each user has her topic interests modeled by a distribution over the topics. In this model a single topic is assigned for an entire tweet. In the TOT model, each topic is associated with a continuous distribution over timestamps, and for each generated document, the mixture distribution over topics is influenced by both word co-occurrences and the document's timestamp.

For the monitoring scenario there are two important characteristics that are present in TwitterLDA and TOT: tweets are about one single topic and each topic has a time distribution. We use this features in an LDA-based model with the following assumptions: (i) there is a set of topics K in the collection of tweets of an entity, each represented by a word distribution and a continuous distribution over time; and (ii) when a tweet is written, the author first chooses a topic based on a topic distribution for the entity.

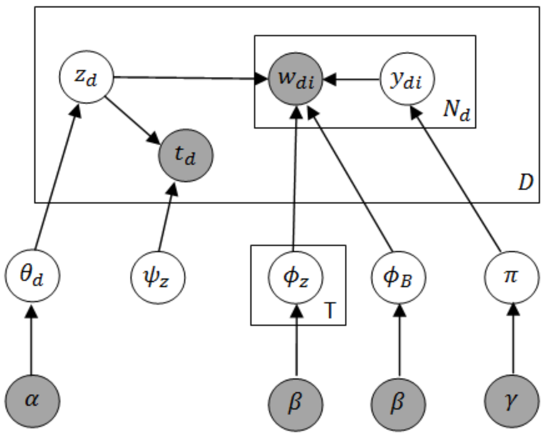
Then a bag of words is chosen one by one based on the topic. However, not all words in a tweet are closely related to the topic of that tweet; some are background words commonly used in tweets on different topics. Therefore, for each word in a tweet, the author first decides whether it is a background word or a topic word and then chooses the word from the respective word distribution of the entity.

The process is described as follows:

1. Draw $\theta_d \sim Dir(\alpha)$, $\phi_{\mathcal{B}} \sim Dir(\beta)$, $\pi \sim Dir(\gamma)$
2. For each topic $z = 1, ..., K$
   (a) draw $\phi^z \sim Dir(\beta)$
3. For each tweet $d = 1, ..., D$
   (a) draw a topic $z_d \sim Multi(\theta_d)$
   (b) draw a timestamp $t_d \sim Beta(\psi_{z_d})$
   (c) for each word $i = 1, ..., N_d$
      i. draw $y_{d_i} \sim Bernoulli(\pi)$
      ii. if $y_{d_i} = 0$: draw $w_{d_i} \sim Multi(\phi_{\mathcal{B}})$
         if $y_{d_i} = 1$: draw $w_{d_i} \sim Multi(\phi_{z_d})$

where: ϕ_{z_d} denotes the word distribution for topic z; ψ_{z_d} is the time distribution for the topic z; $\phi_{\mathcal{B}}$ the word distribution for background words and π denotes a Bernoulli distribution that governs the choice between background words and topic words. After applying the model, a topic is represented as a vector of probabilities over the space of words and a single topic is assigned for an entire tweet. We employ Gibss sampling to perform approximate inference. The graphical model is shown in Figure 2.

Fig. 2. Graphical model



The system also relies on transfer learning by contextualizing the target tweets with a large set of unlabeled "background" tweets that help improving the clustering. We include background tweets together with target tweets in the LDA model, and we set the total number of clusters. In practice, this means that the system can adapt to find the right number of clusters for the target data, overcoming one of the limitations of using LDA-based approaches (the need of establishing a priori the number of clusters).

We train the LDA-based model simultaneously over all the tweets in the target set and the background and fix the target number of clusters (as required by LDA) for the whole set. Note that the LDA-model labels each tweet with only one topic, so we will have a non-overlapping clustering. Once we have obtained the clustering, we extract all clusters that contain at least one target tweet, and remove non-target tweets from those clusters.

4.2 Wikified Tweet Clustering

This approach relies on the hypothesis that tweets sharing concepts or entities defined in a knowledge base –such as Wikipedia– are more likely to be talking about the same topic than tweets with none or less concepts in common.

Tweet Wikification. We use an entity linking approach to gather Wikipedia entries that are semantically related to a tweet. To this end, the COMMONNESS probability [11], based on the intra-Wikipedia hyperlinks, is used to select the most probable entity for each of the longest n-grams that were linked to Wikipedia articles. The approach computes the probability of a concept/entity c been the target of a link with anchor text q in Wikipedia by:

$$\text{Commonness}(c, q) = \frac{|L_{q,c}|}{\sum_{c'} |L_{q,c'}|}$$

where $L_{q,c}$ denotes the set of all links with anchor text q and target c.

We used both Spanish and English Wikipedia dumps. Spanish Wikipedia articles are then translated to the corresponding English Wikipedia article by following the inter-lingual links, using the Wikimedia API[2].

Tweet Clustering. After tweets are wikified, the Jaccard similarity between the sets of entities linked to the tweets is used to group them together: given two tweets d_1 and d_2 represented by the set of Wikipedia entities C_1 and C_2 respectively, if $\text{Jaccard}(C_1, C_2) > th$, then d_1 and d_2 are grouped together to the same cluster.

4.3 Term Clustering

Let us assume that each topic related to an entity can be represented with a set of keywords, that allow to the expert to understand what the topic is about. Considering this, we define a two-step algorithm that tries to (i) identify the terminology of each topic –by clustering the terms– and (ii) assigning tweets to the identified clusters.

We use Hierarchical Agglomerative Clustering (HAC) to build the term clustering. As similarity function, we use the confidence score returned by a classifier that, given a pair of co-occurrent terms, guesses whether both terms belong to the terminology of the same topic or not.

[2] http://www.mediawiki.org/wiki/API:Properties

Co-occurrence pair representation. We used different families of features to represent each of the co-occurring pairs:

- **Term features:** Features that describe each of the terms of the co-occurrence pair. These are: term occurrence, normalized frequency, pseudo-document TF.IDF and KL-Divergence [15]. These features were computed in two ways: (i) considering only tweets on the labeled corpus, and (ii) considering tweets in both the labeled and background corpus. Features based on the tweets meta-data where each term occurs are: Shannon's entropy of named users, URLs, hashtags and authors in the tweets where the term occurs.

- **Content-based pair term features:** Features that consider both terms of the co-occurrence pair, such as Levenshtein's distance between terms, normalized frequency of co-occurrences, Jaccard similarity between occurrences of each of the terms.
- **Meta-data-based pair term features:** Jaccard similarity and Shannon's entropy of named users, URLs, hashtags and authors between tweets where both terms co-occur.
- **Time-aware features:** Features based on the date of the creation of the tweets where the terms co-occurs. Features computed are median, minimum, maximum, mean, standard deviation, Shannon's entropy and Jaccard similarity. These features were computed considering four different time intervals: milliseconds, minutes, hours and days.

In our classification model each instance corresponds to a pair of co-occurrent terms $\langle t, t' \rangle$ in the entity stream of tweets. In order to learn the model, we extract training instances from the RepLab 2013 training dataset, considering the following labeling function:

$$label(\langle t, t' \rangle) = \begin{cases} \texttt{clean} & \text{if} \quad \max_j Precision(C_{t \cap t'}, L_j) > 0.9 \\ \texttt{noisy} & \text{in other case} \end{cases}$$

where $C_{t \cap t'}$ is the set of tweets where terms t and t' co-occurs and L is the set of topics in the goldstandard, and

$$Precision(C_i, L_j) = \frac{|C_i \cap L_j|}{|C_i|}$$

After this process, term pairs with 90% of the tweets belonging to the same cluster in the goldstandard (i.e., purity=0.9) are considered *clean* pairs. Otherwise, terms with less purity are labeled as *noisy* pairs.

Hierarchical Agglomerative Term Clustering. Using all the co-occurrence pair instances in the training set, we build a single binary classifier that will be applied to all the entities in the test set. Then, the confidence of a co-occurrence pair belonging to the *clean* class is used to build a similarity matrix between terms. A Hierarchical Agglomerative Clustering is then applied to cluster the terms, using the previously built similarity matrix. After building the agglomerative clustering, a cut-off threshold based on the number of possible merges is used to return the final term clustering solution.

Tweet clustering. The second step of this algorithm consists on assigning tweets to the identified term clusters. Each tweet is assigned to the cluster with highest Jaccard similarity.

4.4 Submitted Runs

A total of seven runs have been submitted for the topic detection subtask. Table 4 summarizes the approaches and the parameters used for each of the runs. `UNED_ORM_topic_det_1` consist of applying instance-based learning over HBR, analogously as described in Section 2.1. `UNED_ORM_topic_det_2` is the wikified tweet clustering approach, using the threshold for the Jaccard similarity of $th = 0.2$. This threshold has been empirically optimized using the training dataset[3]. `UNED_ORM_topic_det_3`,`UNED_ORM_topic_det_4` and `UNED_ORM_topic_det_5` use the term clustering approach using different machine learning algorithms to combine the features (Naïve Bayes and Logistic Regression) and different thresholds applied to the HAC. A threshold of 0.30 indicates that the final clustering considered is the one produced when the 30% of all possible merges are applied. In all the term clustering runs, the merges are carried out by the mean linkage criterion. Again, only terms after stopword removal and with occurrence greater than five are considered. Finally, `UNED_ORM_topic_det_6` and `UNED_ORM_topic_det_7` use the LDA-based clustering approach, where the transfer learning step is carried out considering background tweets from the entity with a max. of 10,000 tweets or from other randomly selected entity with 10,000 tweets, respectively.

4.5 Results

Table 5 shows the results obtained by the submitted runs in the topic detection subtask and compared to the baselines. The table reports scores for the metrics Reliability (R), Sensitivity (S) and F_1-measure of R and S, $F_1(R, S)$. It also reports the position of the runs in the ranking provided by the organizers. As a clustering task, Reliability (R) corresponds to BCubed Precision and Sensitivity (S) corresponds to BCubed Recall [6].

In general, all the approaches are in the top of the rank, performing significantly better than the baselines. The wikified tweet clustering gets the highest score in all the metrics. Term clustering approach seems to performs similarly when changing the HAC cut-off threshold or the machine learning algorithm. On the other hand, the LDA-based clustering approach performs better when the background collection used for transfer learning contain tweets from different

[3] Note that this is the only supervision used on this system.

Table 4. Overview over the runs submitted for the topic detection subtask.

Approach	**Parameters**	**Run**
instance-based learning + HBR		`UNED_ORM_topic_det_1`
wikified tweet clustering	Jaccard sim. threshold= 0.2	`UNED_ORM_topic_det_2`
term clustering	combination=N.Bayes, HAC threshold= 0.3	`UNED_ORM_topic_det_3`
term clustering	combination=N.Bayes, HAC threshold= 0.7	`UNED_ORM_topic_det_4`
term clustering	combination=Log.regression, HAC threshold= 0.7	`UNED_ORM_topic_det_5`
LDA-based clustering	background tweets only from the entity	`UNED_ORM_topic_det_6`
LDA-based clustering	background tweets also from other entities	`UNED_ORM_topic_det_7`

Table 5. Results of the runs submitted for the topic detection subtask.

Topic Detection	**Reliability** (R)	**Sensitivity** (S)	$F_1(R,S)$	**Rank**
`UNED_ORM_topic_det_2`	**0.4624**	**0.3246**	**0.3252**	1
`UNED_ORM_topic_det_7`	0.2976	0.2223	0.2407	5
`UNED_ORM_topic_det_3`	0.4168	0.2120	0.2311	7
`UNED_ORM_topic_det_4`	0.4162	0.2116	0.2299	8
`UNED_ORM_topic_det_5`	0.4162	0.2116	0.2299	9
`UNED_ORM_topic_det_6`	0.3415	0.1642	0.2099	16
`UNED_ORM_topic_det_1`	0.1536	0.2186	0.1745	21
baseline	0.1525	0.2173	0.1735	22
all-in-one	0.0678	1.0000	0.1210	33
one-in-one	1.0000	0.0386	0.0693	34

entities/test cases (in this run is ensures to have a collection of 10,000 tweets as background). Finally, as in the other subtasks, combining multiple similarity measures for instance-based learning slightly outperforms the baseline that only consider Jaccard similarity over the terms.

5 Full Monitoring Task

The full monitoring task combines the three subtasks that are typically addresses in a monitoring process: first, the tweets that are not related to the entity of interest are filtered out (filtering); then, the related tweets are clustered by topics (topic detection) and finally, those topics are ranked by priority (priority). Next we describe the submitted runs for the full monitoring task, as well as the obtained results.

5.1 Submitted Runs

For the full monitoring task, we submitted 8 different runs in total. The runs combine different approaches for each of the three subtasks, considering the subsystem as a pipeline (e.g., topic detection is carried upon the tweets labeled as related after the filtering step). Table 6 shows the different combinations submitted. For instance, the run `UNED_ORM_full_task_1` consists of applying instance-based learning over HBR for the three subtasks. Run `UNED_ORM_full_task_2` uses the straightforward tweet classification filtering system, the wikified tweet clustering techniques to detect topics over the related tweets and finally the baseline provided by the organizers for the priority subtask (instance-based learning over Jaccard distance).

Table 6. Overview of the runs submitted for the full monitoring task.

Run	Filtering	Topic Detection	Priority
`UNED_ORM_full_task_1`	`UNED_ORM_filt_1`	`UNED_ORM_topic_det_1`	`UNED_ORM_priority_1`
`UNED_ORM_full_task_2`	`UNED_ORM_filt_2`	`UNED_ORM_topic_det_2`	baseline
`UNED_ORM_full_task_3`	`UNED_ORM_filt_3`	`UNED_ORM_topic_det_2`	baseline
`UNED_ORM_full_task_4`	`UNED_ORM_filt_2`	`UNED_ORM_topic_det_3`	baseline
`UNED_ORM_full_task_5`	`UNED_ORM_filt_3`	`UNED_ORM_topic_det_3`	baseline
`UNED_ORM_full_task_6`	baseline	`UNED_ORM_topic_det_6`	baseline
`UNED_ORM_full_task_7`	baseline	`UNED_ORM_topic_det_7`	baseline
`UNED_ORM_full_task_8`	`UNED_ORM_filt_3`	`UNED_ORM_topic_det_2`	`UNED_ORM_priority_1`

5.2 Results

Table 7 shows the macro-averaged F_1 scores and the official rank of the eight runs submitted. The full monitoring task has been evaluated by means of a harmonic mean (F measure) over the six Reliability and Sensitivity measures (R and S for filtering, priority and topic detection).

In general, we can see that the overall F_1-measure significantly drops when the filtering subsystem is worse (`UNED_ORM_full_task_8`,`UNED_ORM_full_task_3`,`UNED_ORM_full_task_5`). The best results are obtained by using a simple filtering approach –an entity-specific tweet classification algorithm based on Bag-of-Words– with the wikified tweet clustering approach for detecting topics, and using the instance-based learning baseline for the priority subtask. Furthermore, considering any of our topic detection approaches perform equally in terms of the full task evaluation.

6 Conclusions

We have described and discussed here the systems submitted by UNED-ORM group to the RepLab 2013 evaluation campaign. We have participated in three

Table 7. Results of the runs submitted for the full monitoring task.

Run	$\mathbf{F_1}$	**Rank**
`UNED_ORM_full_task_2`	0.1923	1
`UNED_ORM_full_task_7`	0.1802	2
`UNED_ORM_full_task_4`	0.1736	3
`UNED_ORM_full_task_6`	0.1716	4
`UNED_ORM_full_task_1`	0.1577	13
`UNED_ORM_full_task_8`	0.1226	14
`UNED_ORM_full_task_3`	0.1128	15
`UNED_ORM_full_task_5`	0.1080	16

of the subtasks (filtering, topic detection and polarity), as well as in the full monitoring task.

The filtering subtask turned out to be crucial for the overall performance of a monitoring system. An in-depth analysis of the results is needed to understand the limitations of our filter keyword approach - which was initially developed for semi-supervised scenarios, not for a fully supervised scenario as in RepLab 2013.

In the reputational polarity task, an approach to generate semantic graphs for domain-specific affective lexicon adaptation has been tested. Even if a similar approach has been previously tested achieving promising performance in a traditional Sentiment Analysis task using news and reviews as input data, our results in the RepLab 2013 seem to be less competitive and deserve further analysis. In particular, more analysis on the differences between Reputational Polarity and traditional sentiment analysis is needed.

Three of our topic detection approaches (LDA-based clustering, wikified tweet clustering and term clustering) perform competitively with respect to other RepLab submissions. Still, the room for improvement is large, and it is not easy to assess how much of the problem can be solved automatically.

Finally, in terms of instance-based learning, the results suggest that extending tweets with associated contents (tags, external links, Wikipedia entries) does not provide useful signals to improve performance, at least in our current setting.

Future work will focus on the analysis of the results and the optimization of the different subsystems for the monitoring task.

References

1. Agirre, E., Soroa, A.: Personalizing pagerank for word sense disambiguation. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics. pp. 33–41. Association for Computational Linguistics (2009)
2. Amigó, E., Artiles, J., Gonzalo, J., Spina, D., Liu, B., Corujo, A.: WePS-3 Evaluation Campaign: Overview of the Online Reputation Management Task. In: CLEF 2010 Labs and Workshops Notebook Papers (2010)

3. Amigó, E., Carrillo-de-Albornoz, J., Chugur, I., Corujo, A., Gonzalo, J., Martín, T., Meij, E., de Rijke, M., Spina, D.: Overview of RepLab 2013: Evaluating Online Reputation Management Systems. In: CLEF 2013 Working Notes (Sep 2013)
4. Amigó, E., Corujo, A., Gonzalo, J., Meij, E., de Rijke, M.: Overview of RepLab 2012: Evaluating Online Reputation Management Systems. In: CLEF 2012 Labs and Workshop Notebook Papers (2012)
5. Amigó, E., Gimenez, J., Gonzalo, J., Verdejo, F.: Uned: Improving text similarity measures without human assessments. In: *SEM 2012: The First Joint Conference on Lexical and Computational Semantics
6. Amigó, E., Gonzalo, J., Verdejo, F.: A General Evaluation Measure for Document Organization Tasks. In: Proceedings of SIGIR 2013 (Jul.)
7. Carrillo-de-Albornoz, J., Chugur, I., Amigó, E.: Using an emotion-based model and sentiment analysis techniques to classify polarity for reputation. In: CLEF 2012 Labs and Workshop Notebook Papers (2012)
8. Carrillo-de-Albornoz, J., Plaza, L., Gervás, P.: A hybrid approach to emotional sentence polarity and intensity classification. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL'10). pp. 153–161 (2010)
9. Carrillo-de-Albornoz, J., Plaza, L., Gervas, P.: Sentisense: An easily scalable concept-based affective lexicon for sentiment analysis. In: Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12) (2012)
10. Martín-Wanton, T., Spina, D., Amigó, E., Gonzalo, J.: Uned at replab 2012: Monitoring task. In: CLEF (Online Working Notes/Labs/Workshop) (2012)
11. Meij, E., Weerkamp, W., de Rijke, M.: Adding semantics to microblog posts. In: Proceedings of the fifth ACM international conference on Web search and data mining (2012)
12. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: YALE: Rapid prototyping for complex data mining tasks. In: SIGKDD'06: Proceedings of the 12th International Conference on Knowledge Discovery and Data Mining (2006)
13. O'Connor, B., Krieger, M., Ahn, D.: Tweetmotif: Exploratory search and topic summarization for twitter. Proceedings of ICWSM pp. 2–3 (2010)
14. Plaza, L., Diaz, A.: Using semantic graphs and word sense disambiguation techniques to improve text summarization. Procesamiento de Lenguaje Natural 47, 97–105 (2011)
15. Spina, D., Meij, E., de Rijke, M., Oghina, A., Bui, M., Breuss, M.: Identifying entity aspects in microblog posts. In: SIGIR '12: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval (2012)
16. Spina, D., Gonzalo, J., Amigó, E.: Discovering filter keywords for company name disambiguation in twitter. Expert Systems with Applications 40(12), 4986 – 5003 (2013)
17. Wang, X., McCallum, A.: Topics over time: a non-markov continuous-time model of topical trends. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 424–433. KDD '06, ACM, New York, NY, USA (2006)
18. Zhao, W.X., Jiang, J., Weng, J., He, J., Lim, E.P., Yan, H., Li, X.: Comparing twitter and traditional media using topic models. In: Proceedings of ECIR'11. pp. 338–349. Springer-Verlag, Berlin, Heidelberg (2011)