Tweet categorization by combining content and structural knowledge

J.M. Cotelo *, F.L. Cruz, F. Enríquez, J.A. Troyano

Department of Languages and Computer Systems, University of Seville, Avda. Reina Mercedes s/n, Seville 41012, Spain

Keywords:
Twitter
Tweet categorization
Ensemble learning
Knowledge combination

A B S T R A C T

Twitter is a worldwide social media platform where millions of people frequently express ideas and opinions about any topic. This widespread success makes the analysis of tweets an interesting and possibly lucrative task, being those tweets rarely objective and becoming the targeting for large-scale analysis. In this paper, we explore the idea of integrating two fundamental aspects of a tweet, the proper textual content and its underlying structural information, when addressing the tweet categorization task. Thus, not only we analyze textual content of tweets but also analyze the structural information provided by the relationship between tweets and users, and we propose different methods for effectively combining both kinds of feature models extracted from the different knowledge sources. In order to test our approach, we address the specific task of determining the political opinion of Twitter users within their political context, observing that our most refined knowledge integration approach performs remarkably better (about 5 points above) than the textual-based classic model.

1. Introduction

Twitter is a successful worldwide social media platform where millions of people frequently express ideas and opinions about a myriad of topics. Texts written in Twitter, called tweets, are characterized by having a very short length (140 characters) and often written using devices like smartphones with almost no revision before sending them, trading redaction quality and/or correctness for speed. Aside from the textual content, tweets (and Twitter itself) offer many other data and information that may serve as knowledge sources for solving different tasks of interest. In this work, we explore how the integration of these heterogeneous knowledge sources improves the overall performance when it is used for addressing the automatic tweet categorization task.

The widespread success of Twitter makes the analysis of the tweets a very interesting (and possibly lucrative) task. The amount of information in these texts is huge and tweets are rarely objective, becoming the target of large-scale analysis that could be really useful for marketing campaigns, public opinion determination or even inferring how a population responds for specific events. Branding [1,2], political analysis [3–5] or user profiling for market analysis [6] are examples of actual applications for the analysis of Twitter and other social media texts. Protection and detection against malware [7] is also an application of interest, as Twitter trending topics are also vulnerable to scamming, phishing or spamming.

Categorizing twitter messages is an interesting and valuable task. In this paper, we address the categorization of tweets, focusing on determining the political opinion of Twitter users within their political context. A collection of tweets not only provides textual information, but also provides structural information due to the relationship between users and messages, forming an underlying network. We discuss the analysis of both types of content, applying different approaches and yielding different feature models for each of them, and we propose several methods of combining these feature models (both structural an textual ones) in the classification process.

Although the approach presented in this article was originally designed for the tweet categorization task in mind, it can be applied to other social networks (e.g. Facebook, Google+, ...), as long as those social networks on which this approach is applied, may exhibit relevant structural features in its messages.

Fig. 1 shows a diagram of the overall process of our proposal, which can be described as follows. It starts with the retrieval phase, in which we collect tweets using an automatic topic-related retrieval method that ensures that collected tweets are politically relevant. Before continuing with the experimental process, those tweets are manually annotated by two independent annotators. The retrieval and annotation process, along with the reference to retrieval method, are described in Section 3. These politically relevant tweets are fed to two distinct pipelines in order to generate different feature models, being each pipeline focused on analyzing

* Corresponding author. Tel.: +34 676582325.
E-mail addresses: jcotelo@us.es (J.M. Cotelo), fcruz@us.es (F.L. Cruz), fenros@us.es (F. Enríquez), troyano@us.es (J.A. Troyano).

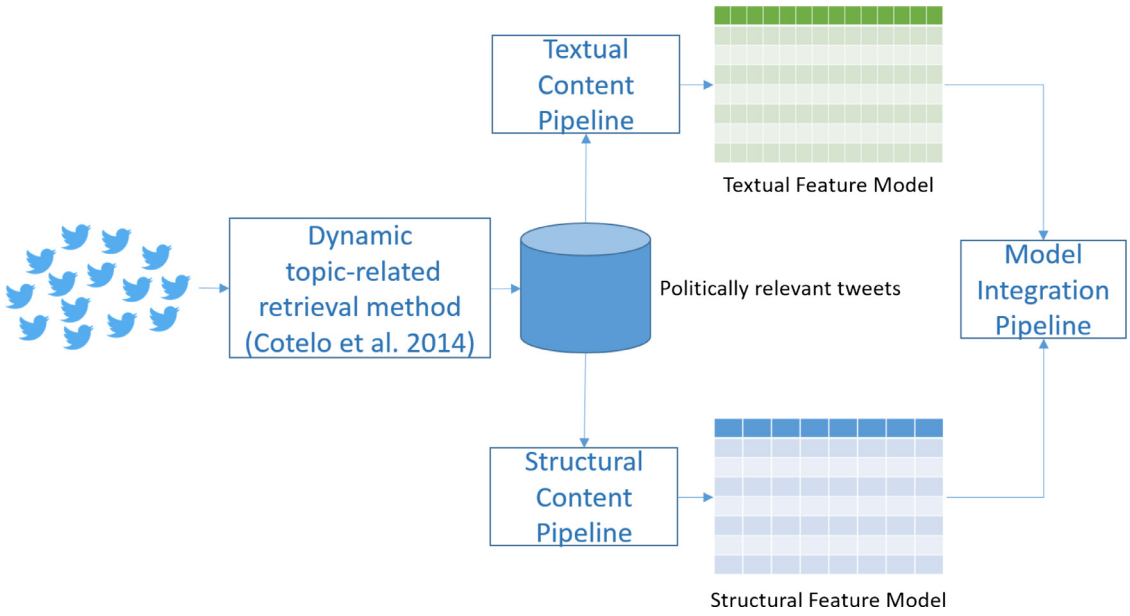


Fig. 1. Overall process of our proposal.

textual or structural content. After each structural and textual feature models are generated, both are fed to the model integration pipeline, which combines both models and improves the performance results.

This article is structured as follows: In Section 2, we discuss several other works that are related with the one presented in this paper. In Section 3, we define the specific task addressed in this paper and we characterize the dataset, describing how it is generated and its particularities. In Section 4 we described the whole experimental setup developed for evaluating our proposal. In Section 5, we address the extraction of knowledge from the textual content of tweets, discussing the adaptation of models such as the *Bag-of-Words* and its caveats, proposing an automatic feature selection process for improving the performance of that model. In Section 6 we discuss how to extract information from the structural content of tweets, exploiting the topological features of the underlying network formed by the users and messages. We propose a topological approach, consisting in generating a bipartite friendship graph based on the identification of two major kinds of users (*content creators* and *content consumers*). From this graph, we propose two different community feature models based on the *Louvain method* and *Spectral Biclustering* technique respectively. In Section 7, we combine both structural and textual feature models in three different ways: directly combining both feature models, using *Stacking Generalization* and a variation of our own that we called *Multiple Pipeline Stacked Generalization*. Finally, in Section 8, we summarize our efforts, review the main points of our work and discuss the importance of combining structural and textual content for tweet.

2. Related works

There are many works dealing with the classification of tweets, especially in the area of sentiment analysis (considering the subjective content of many of the tweets). Most of these works face polarity classification, i.e., deciding whether a tweet expresses a positive, negative or neutral opinion toward a given topic. For example, in [8] the polarity classification is performed using polarity lexicons (resources consisting of lists of negative and positive words), and later tweaking this polarities from the semantic context in which the words appear. In [9] the lack of Arabic polarity lexicons is supplied by using emoticons appearing in the tweets to build a polarity classifier on Twitter; the idea is taken from [10], where the same technique was applied to a corpus of tweets in English. In [11], automatic summaries of the opinions of a Twitter user are generated by integrating the various negative and positive opinions expressed by users on various topics. In all these studies the classification of a tweet is performed based solely on their textual content: in no case structural information inherent to Twitter is used, as might be other user's tweets or other tweets using the same hashtags, for example. In [12], the authors transform the textual content to a graph representation where nodes are tweets, users, hashtags and words. Nodes are labeled with polarities, taken from lexicons and from a supervised classifier previously trained. After that, they apply a label propagation algorithm described in [13]. Their proposal is evaluated using three datasets (one of them is from the political domain). Although the graph representation of textual content proves to be effective and it yields better results than other approaches that directly handle the textual content, the proposal only takes into account knowledge from textual content and do not extract any information from the structural knowledge that the underlying network offers.

Many authors have been interested in studying the behavior of users on Twitter in relation to politics. In [14] an analysis of the hashtags used in politics in Canada was performed to distinguish the different objectives for which they are used in the political context. In [15] a study of the different types of Twitter users was conducted to characterize the so-called opinion leaders; they tend to seek information, mobilize, and express opinions publicly, and have a great influence on the political tendency of their followers (we will use this idea in our work, see Section 6). In [4] the LIWC2007 resource (Linguistic Inquiry and Word Count; Pennebaker [16]) was used to determine emotional and cognitive characteristics of tweets related to federal election of the national parliament in Germany 2009. They conclude that there exist high correlations between the above results and dataset statistical metrics such as concentration and share, and even the mere number of tweets mentioning each candidate is a good estimator of the election results. In [17] they also try to measure whether there is a correlation between activity in social networks (Facebook and Twitter) and the results of the US presidential election of 2012. Based on the tweets mentioning Obama and Romney, Facebook official pages

of the candidates, and comments on these pages, the various citizens contributions were manually classified according to polarity. The final conclusion was that there is a strong positive correlation between the data obtained and the results of the elections.

These observations about the utility of the information contained in the tweets, information used to carry out electoral predictions or to estimate other political variables and traditionally been estimated using surveys, make the automatic classification of user tweets very interesting in political contexts. Beyond classifying the polarity of tweets, in [18] tweets related to US presidential election of 2012 were classified according to "purpose" of the authors: to point out hypocrisy or inconsistency; to point out mistake or blunder; to disagree; to ridicule; to criticize; to vent; to agree; to praise, admire, or appreciate; to support; to motivate or incite to action; to be entertaining; to provide information without emotion. A preliminary study, in which this information was annotated manually, confirmed the strong correlation between the distribution of tweets according to these categories and election results; however, the results reported for an automated purpose classifier were not good. We understand that their classification task has a level of detail that is too high, so in this paper we propose a simplification of the same idea: classifying tweets as "for" or "against" the different electoral candidates.

Our approach is novel because, instead of focusing on only one kind of content (textual content in this case), we integrate both explicit textual and underlying topological information of tweets at the same level. Extracting knowledge from the structural content of tweets and generating a valid feature model proved to be non-trivial. In our approach, it required the generation of a graph-based representation and inferring communities in order to generate an expressive feature model. Combining feature models from both textual and structural content proved to be successful by yielding very good performance but it also had its difficulties because the feature models were very different. We developed an interesting *ensemble learning* approach that we named *Multiple Pipeline Stacked Generalization* for specifically taking advantage of mixing different feature modules.

3. Task definition

We address the task of determining the political opinion from Spanish tweets whose contents are highly related to any aspect of Spanish politics. As in many other countries, the political situation is dominated by a handful political forces, in particular by two major parties: the conservative, liberal and Christian democratic *Partido Popular* (*PP*), and the social democratic *Partido Socialista Obrero Español* (*PSOE)*. Since the Spanish transition to democracy, these parties are the only ones that have held office and are going to be the focus of our case study.

We have generated a collection of tweets related to any of both major parties (PP or PSOE) using the dynamic retrieval method explained in [19]. Starting from a seed set, this retrieval method continuously collects data and periodically adjusts its keyword set, performing a graph-based analysis on the data collected in the previous iterations (using a sliding window). It guarantees a higher volume of tweets, introduces very low noise and it reacts to any unforeseen topic-related event during the retrieval time span. We performed the retrieval process during the presentation of the final draft of the amendments of the law that regulates abortion in Spain. This period spanned from 20th December 2013 to 23th December 2013. The proposed reform caused a great impact on the population of Spain and every major political party actively positioned regarding this matter.

From the whole collection of tweets, holding more than 100 k politically relevant tweets, we composed our final dataset, consisting in a random sample of 3000 manually annotated tweets that refer to the current government (PP at the time of the dataset collection) or the opposition (PSOE) party. The annotation process was made by two independent annotators and we check for tweets that with disagreement between annotators, ignoring those tweets and not counting them for the total 3000 tweets dataset.

Table 1
Dataset political opinion distribution.

	PSOE positive (%)	PSOE negative (%)	PSOE neutral (%)
PP positive	00.00	01.07	01.36
PP negative	01.02	04.00	46.14
PP neutral	02.51	18.83	25.07

Any tweet from this dataset express any positive, negative or neutral stance regarding PP and PSOE parties, so we define the task as classifying tweets into any of the nine combinatorial categories (the Cartesian product of the possible stances of PP and PSOE). The dataset was manually annotated, indicating the political stance of each tweet according the previous categories.

Table 1 shows the distribution of the political opinion in the tweets from the dataset and we observed an interesting phenomenon: most of classification classes have with very low representation, resulting that more than 90% of the dataset fits into three of nine political opinion classes. Moreover, users rarely praise any effort coming from a major political party, being most tweets either comments with little political opinion or negative criticism against any of both major parties, but rarely against both.

The low representation of the other six political opinion classes led us to evaluate a simplified or reduced version of the problem along the full version of the problem. Instead of taking into account all the classes, the reduced problem only considers tweets whose political opinion is one of the three major classes: totally neutral, PP negative/PSOE neutral and PP neutral/PSOE negative.

In summary, the task of determining the political opinion of tweets is performed against two version of the datasets, the full version and the reduced version. After determining the political opinion of tweets, the evaluation process is made by the direct comparison of the manually annotated political opinions and the computed ones.

4. Experimental design

With the above defined task in mind, we have developed an experimental setup for evaluating our proposal, making use of both versions of the dataset previously described. Our proposal combines information that comes from two different types of source knowledge, being each type of information processed in different pipelines. Thus, we implemented our experimental setup in three independent stages:

- **Textual content pipeline**. This pipeline performs an analysis over the actual textual content of the tweets, extracting information from the words according to a more conventional Bag-of-Words approach. The tweets are preprocessed before any textual content analysis.
- **Structural content pipeline**. This pipeline performs an analysis over the structural features of the tweets, extracting information from the underlying network topology, using a graph-based approach and a community model. We tested two community models: an affinity community model based on the *Louvain method* and a fuzzy belonging model based on *Spectral Biclustering*.
- **Model integration pipeline**. This pipeline performs a combination scheme on feature models coming from both previous pipelines and, depending on the schema used, applies an

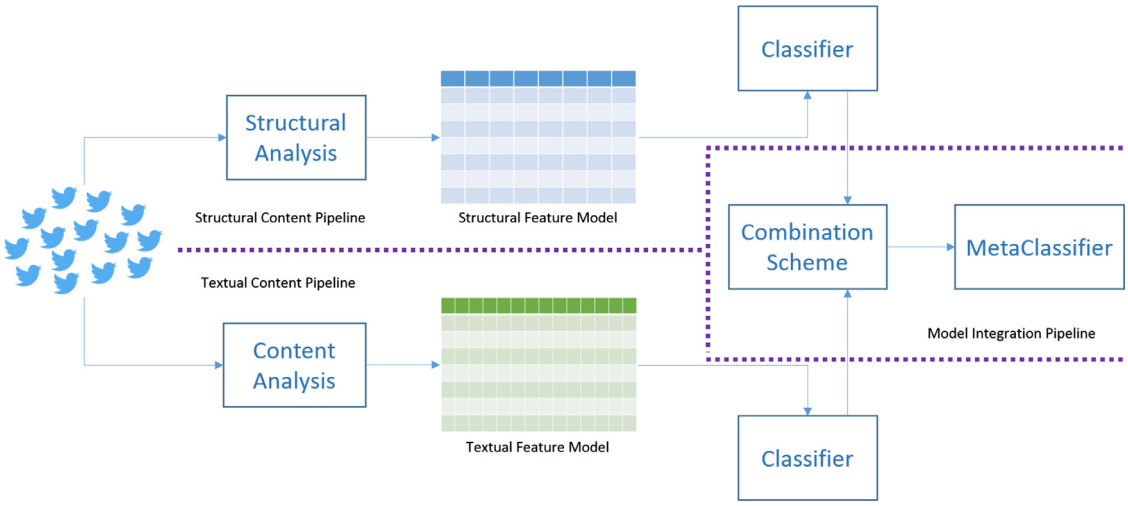


Fig. 2. Overall experimental design for our proposal.

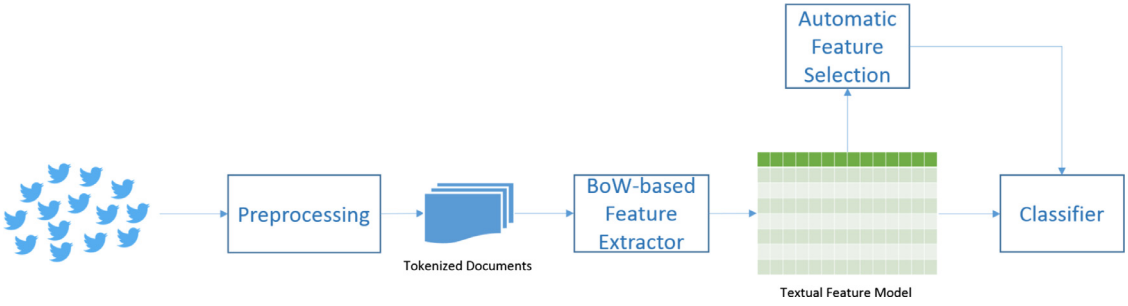


Fig. 3. Textual feature extraction pipeline.

ensemble learning metaclassification process on the relevance values extracted from classifiers from both previous pipelines.

Fig. 2 describes the whole experimental setup, showing the two pipelines for extracting the base feature models and the final combination stage, where our custom ensemble learning method takes place. Each particular pipeline is described in detail in the following sections.

On a side note, for the actual machine learning techniques and algorithms used throughout the whole experimentation process, we made use of the implementations already existing in the open-source project *scikit-learn*[1] [20]. This project provides all the documentation regarding to the implementation, including the references to the corresponding papers.

5. Textual content processing pipeline

Our pipeline designed for extracting features from textual content has similar stages to other text processing pipelines: preprocessing and tokenization, feature extraction, feature selection (if the resulting feature set is too large) and the classification stage. Fig. 3 shows the overall workflow of this pipeline.

5.1. Tweet preprocessing and tokenization

Dealing with tweet textual content differs from typical text processing in several aspects. On the one hand, special care must be taken during the tokenization process because tweets usually contain special elements like *hashtags* or *user mentions* that are quite relevant and have semantic value. On the other hand, tweets often are *"polluted"* with other elements that can be qualified as non relevant such as ASCII art, numeral and ordinals, date/time compounds and URLs. Those elements have to be removed with care without altering the rest of the extracted content.

The proper processing of the textual content of a tweet is crucial for posterior analysis. Our pipeline for processing tweets carefully addresses the points above mentioned and also performs more typical processing such as stopword and punctuation removal.

5.2. Bag-of-Words model

The feature model proposed in this paper for our textual representation is the well-known *Bag-of-Words* (*BoW*) model, commonly used as a standard for text classification and being appropriate for short texts like textual content of tweets. This model simplifies each document by representing it as the multiset of its words, disregarding grammar or word order but taking into account the multiplicity of the words within the document. When this model is used in vectorial form, it resembles to an histogram representation.

Despite its simplicity, this feature model is widely used in different applications where an input feature vector for training classifiers and the results obtained are often adequate. We chose this model because the nature of the textual content of tweets is brief and with low grammar complexity and, usually, no grammar correctness. Thus, traditional NLP approaches that rely on grammar analysis would give no useful information.

[1] http://scikit-learn.org .

Table 2
Cross-validated performance of the Bag-of-Words model.

Feature model	Full problem		Reduced problem	
	Accuracy (%)	AUC (%)	Accuracy (%)	AUC (%)
Dummy Random uniform	12.73	50.00	34.86	50.00
Dummy Random stratified	31.32	49.82	36.37	49.64
Dummy Most frequent class	46.13	50.00	51.24	50.00
Bag-of-Words	61.97	79.19	68.36	85.13
BoW-AFS (automatic feature selection)	77.37	90.15	88.38	96.30

5.3. Improving performance via automatic feature selection

Although the BoW model achieves some degree of success as a feature model for training the classifiers, there is room for improvement.

A common way of improving the BoW model consists in applying a *TF-IDF* weighting scheme to the vectorized documents, giving more relevance to some words than others and improving the overall performance of the model. Nevertheless, due the nature of the tweets textual content, applying a *TF-IDF* weighting scheme does not improve the results: the word co-occurrence within each document is very low and the tweet length very short compared to the large number of documents, thus rendering any weighting scheme based on word distribution is of little use.

A brief analysis on the BoW model vectorial data reveals an important issue also associated with the textual content. The dimensionality of the BoW model is quite high ($\approx 3k$ features) while the average words/tweet is about 10.41, thus resulting in a low data density ($\approx 3.34 \times 10^{-3}$) and low classifier performance. In order to palliate this issue, a dimensionality reduction process is applied to the BoW model. Generally speaking, there are two main ways to address dimensionality: *dataset transformation* techniques and *feature selection* techniques. Dataset transformation techniques try to reduce the dimensionality via data transformations, combining and transforming several data dimensions into fewer ones. Techniques typically rely on *matrix factorization problems* (*PCA, Kernel PCA, SVD* or *NNMF*) or *manifold learning* (*LLE, LTSA, Spectral Embedding* or *MDS*).

Though dataset transformation techniques perform well and usually improve performance, these techniques did not work well with our BoW model. No significant performance improvements were detected and classifiers generally performed worse, leading us to discard such techniques.

Feature selection techniques work differently as they are based under the central assumption that data contains many redundant or irrelevant features, which provide no benefit for the classification process. Feature selection techniques are better suited for our BoW model due to the sparseness of data and the fact that most of the words may not be relevant for our task, resulting in several features being uninformative. We have tested several feature selection techniques, being those based on *decision trees* the most successful for our BoW model.

We have defined an automatic feature selection step to process the BoW model, making use of a forest of *extremely randomized trees* with a high number of estimators. This type of forest is similar to a typical random forest but it differs in the way of how the thresholds are selected for each random subset: the thresholds are drawn at random for each candidate feature and the best are selected instead of looking for the most discriminating. This tends to further reduce the variance of the internal model. As it is shown in Fig. 3, this step is positioned in the pipeline right before the application of the SVM classifier with hyperparameter search.

5.4. Experimental results

We tested the performance of the BoW model against the two versions of the proposed task, including several dummy baselines for comparison purposes and the automatic feature selection step. Table 2 shows the cross-validated performance of this model, using a *Support Vector Machine* (*SVM*) classifier with hyperparameter search. Any cross-validation process is *stratified* (preserving the ratio of classes among the folds) with $k = 10$ folds.

Despite that the performance of the BoW model is quite superior to the dummy baselines the results are not very impressive. The BoW model surpasses the *most frequent class* baseline by $\approx +15\%$ for the full problem and $\approx +17\%$ for the reduced problem, achieving a moderate success (between 61% and 69% of accuracy).

With the previously mentioned automatic feature step, the BoW model experiments a huge performance improvement ($\approx$ from $+15\%$ to $+20\%$ respect bare BoW model) on both versions of the problem, achieving more than 88% of accuracy in reduced version of the problem. With this feature selection step, we can consider that the BoW model achieves a good performance and it is appropriate as an initial feature model for classification based on extracted textual content.

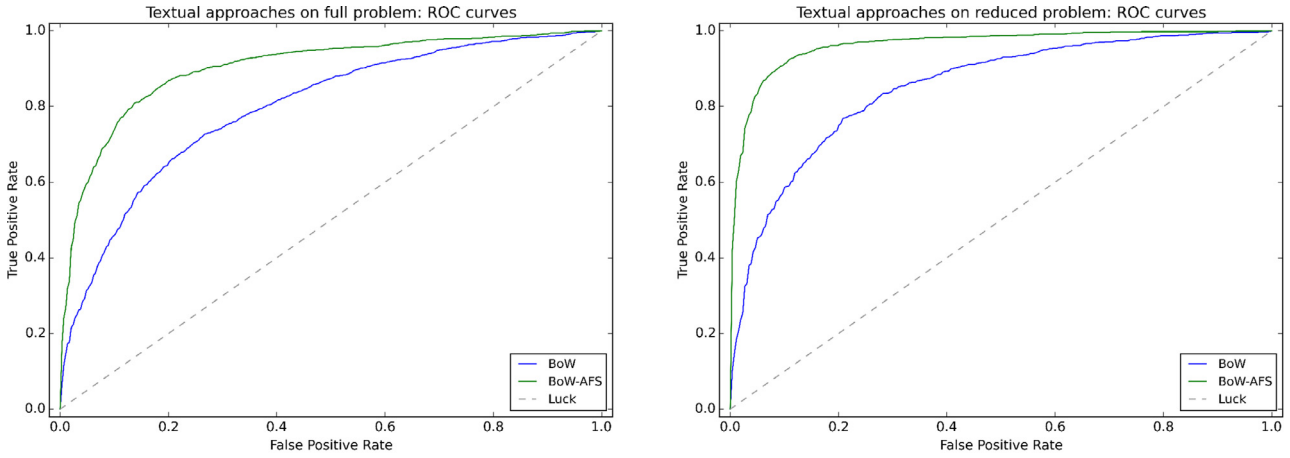
The receiver operating characteristic curves (ROC) of these models, for both full and reduced problem, are shown in Fig. 4a and b respectively, being the area under curve (AUC) values also shown in Table 2. ROC curves and their AUC values are consistent with the obtained accuracy values, yielding similar behavior.

6. Structural content processing pipeline

As we have mentioned before, the structural nature of tweets is an interesting and relevant source of knowledge, though this aspect is often overlooked and extracting structural knowledge from tweet collections is not straightforward. Despite the fact that there are constructs within tweets that establish some relationships such as *user mentions* or *hashtags*, the underlying network is very rich and complex, requiring additional effort and specific approaches for addressing that complexity.

The main idea relies on the fact that users may have explicit and implicit relationships between them, resulting in the creation of implicit communities in which members tend to share common interests despite the fact that most members do not know each other or have any direct contact. Thus, we devised a graph-based approach for discovering these implicit communities and characterizing users by using a feature model extracted from these communities.

Our approach for extracting meaningful structural features begins with building a graph-based representation of the existing user relationships, inferring a community model from that graph and generating a feature model from that community model. Thus, the resulting pipeline has the following stages: graph building, community detection, community feature extraction, feature selection (if the resulting featureset is too large) and the classification stage. Fig. 5 shows the overall workflow of this pipeline.



(a) Textual content models on full problem (b) Textual content models on reduced problem

Fig. 4. Receiver operating characteristic curves (ROC) for textual content models.

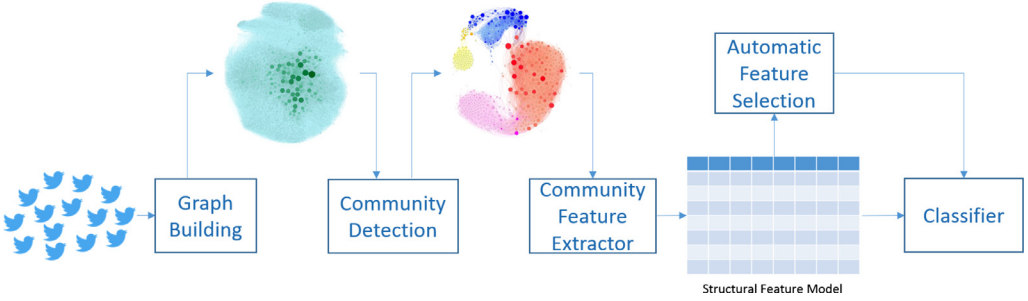


Fig. 5. Structural feature extraction pipeline.

Note that, in each structural approach, the proper detection of communities has been elaborated before the classifier learning phase, conceived as a feature extraction process, in order to include the knowledge from the social network as features for the learning process. We consider that this is necessary for evaluating our proposal because it is the only way to be sure that test instances have valid community values.

6.1. Building the graph-based representation

The first step of our graph building process consists in generating a direct friendship graph using all the users that appear in the tweet collection and their friends (users who they directly follow). The resulting graph may be huge and difficult to handle and it holds thousands of nodes that only have one incoming edge (friendship relationship), being mostly irrelevant nodes. Therefore, the friendship graph is pruned by removing irrelevant nodes with a low count of incoming edges and no outgoing edges.

This pruned friendship graph holds two kinds of users: original users that authored any tweet in the original dataset and new users inferred from the existing relationships within the network. We noticed that the original users are mostly *content consumers* while the new inferred users are mostly *content creators*. Content creators are nodes that act as sources of relevant content which we may identify as sources of political opinion. These users usually are mass media and politically active users and often hold a great number of followers but this is not a necessary condition. Content consumers are the rest of network nodes that consume the content that content creators generate by following them. Both roles are not mutually exclusive and any user may hold both roles.

From the pruned friendship graph, we build a bipartite graph that express this behavior taking into account that some nodes have both incoming and outgoing edges, potentially having both roles. Those nodes are transformed into two different nodes each one only having incoming or outgoing edges. Any isolated node resulting from the pruning process or the bipartite process is discarded. This bipartite graph is used as the base source for the community detection algorithms used in the next stage.

6.2. Community detection and feature model extraction

The feature model extracted from the bipartite graph greatly depends on the particular community detection process; different community detection methods yield quite different community models, each one with their own advantages and drawbacks. In this paper, we have tested two different community detection approaches: an approach based on computing an user affinity model and another approach based on modeling the community detection task as a biclustering problem.

The first approach consists on generating a feature model from the community model extracted from a similarity graph, being the resulting model able of computing the affinity of each user to each detected community.

The first step of this approach consists in generating a similarity graph of the content creator nodes from the bipartite graph, based on the idea that two content creator nodes are similar if they share common content consumers. We selected the *Dice* measure as an appropriate similarity measure for comparing content creator nodes via their sets of content consumers. Thus, we build a similarity graph whose edges between nodes have weight equal to the Dice measure between them.

Table 3
Cross-validated performance of the different structural models.

Feature model	Full problem		Reduced problem	
	Accuracy (%)	AUC (%)	Accuracy (%)	AUC (%)
Bag-of-Words	61.97	79.19	68.36	85.13
Community affinity	50.07	54.66	56.04	54.66
Community affinity-AFS	50.28	55.14	56.51	55.99
Biclustering relevance	59.97	77.21	68.75	77.21
Biclustering relevance-AFS	61.11	77.23	69.60	78.09

This similarity graph has to be further processed in order to reveal some kind of hidden structure. Any edge representing a low similarity value may be considered as uninformative, so every edge with a weight lower than some threshold is removed. In our work, we found that weight values inferior to 0.35 indicated low similarity between content creator nodes; this is direct result of using the *Dice* measure.

This processed similarity graph can be no longer a connected graph and different connected components must be inspected individually though in most cases, the connected component with greater number of nodes is also the only relevant connected component. After selecting the relevant connected components, we apply the well-tested *Louvain method* [21] for detecting communities on the processed similarity graph, assigning each content creator to a particular community.

Using this community model, we generate a feature model for each user representing the affinity to each community. For each user that appears in the dataset, we compute the proportion of their friendships to each community thus obtaining a vector of values whose sum is 1.

Though we consider this approach quite interesting, we felt that the feature model based on the communities extracted by the Louvain method was insufficient for revealing underlying structural knowledge. The major disadvantage of this first approach was that we did all the community analysis in a deferred way, slightly detaching content consumers and content creators instead of analyzing them jointly. Thus, we propose another approach that tries to address the joint analysis of content consumers and content creators.

This second approach is based on biclustering techniques that perform a simultaneous clustering of rows and columns of a matrix, being adequate for our task if we generate a model in which the content consumers and content creators are represented as rows and columns respectively. There are many biclustering techniques and describing all of them is out of the scope of this paper. The survey [22] is a good source for any interested reader on biclustering techniques. We have selected the *Spectral Biclustering* algorithm [23] for our problem because it computes a fuzzy community model with several degrees of membership instead of a one-to-one model.

Spectral Biclustering relies on the idea that the data matrix has a hidden checkerboard structure and the n rows and m columns may be partitioned into $n \times m$ biclusters. Each row will belong to m biclusters and each column to n biclusters with different degrees of membership and the algorithm uses those $m \times n$ biclusters for computing the most representative bicluster for each row and column element.

Using the reduced bipartite graph, we build a friendship matrix M where $M_{i,j} = 1$ if and only if the content creator i is being followed by the content consumer j (j has a friendship relationship with i). Applying the Spectral Biclustering algorithm to the matrix M, we generate representative biclusters that we can interpret as communities of both content creators and content consumers.

For each content creator i and its most representative bicluster b_i, we compute its intra-bicluster weight $w_i = \frac{\sum_{j \in b_i} M_{i,j}}{|j \in b_i|}$, being that the ratio of direct followers with the same representative bicluster of content creator i. These weights represent the relevance of the content creators within the community represented by that bicluster.

After that, for each user appearing in the dataset, we generate a feature model similar to the previous approach but in this case, the belonging measure to each community (bicluster) is done by summing of the weights of the directly followed content creators that belong to that bicluster.

6.3. Experimental results

We tested the performance of the feature models generated from both pure structural approaches against the two versions of the proposed task, testing the automatic selection step in both models and including the previous BoW model for comparison purposes. Table 3 shows the cross-validated accuracy of these models, using the same experimental process.

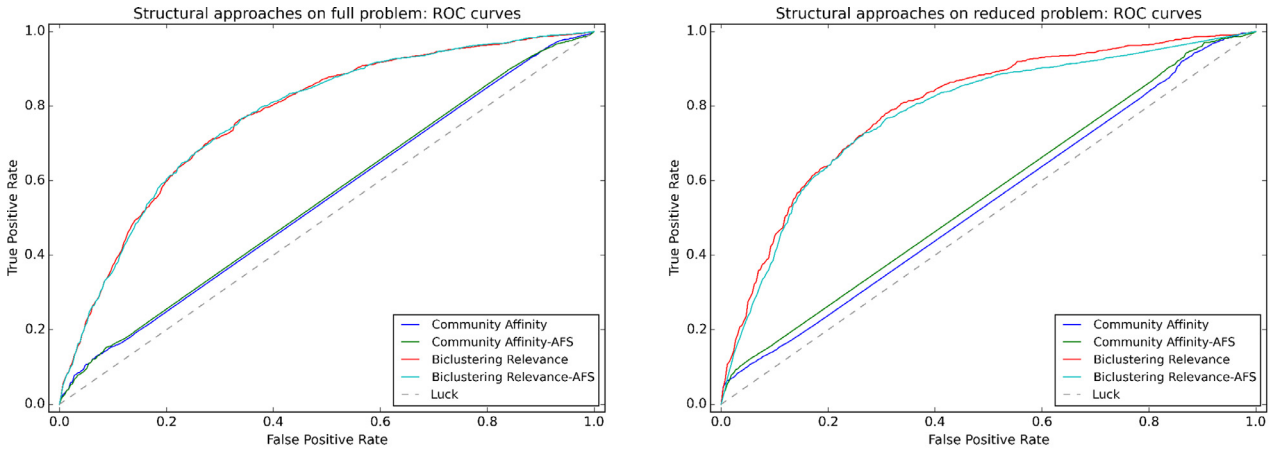
Table 3 shows the performance values of the different feature models. It is interesting that the feature model based on computing affinity values from communities extracted by using the Louvain method achieves this prediction power, moreover if we consider it does not take into account what users express in their tweets. Applying an automatic feature selection shows a slight improvement in performance but note that it is very difficult to further improve this feature model due the low number of features used.

Nevertheless, the feature model based on the Spectral Biclustering for obtaining the communities and computing the relevance within those communities proved being more effective. This approach performs remarkably better than the other structural approach and it yields similar results to the bare BoW model, indicating that the underlying network structure is, by itself, very valuable. Note that the automatic feature selection also slightly improves results but suffers the same issues than the other approach, namely the low number of features used.

The receiver operating characteristic curves (ROC) of these models, for both full and reduced problem, are shown in Fig. 6a and b respectively, being the area under curve (AUC) values also shown in Table 3. The ROC curves obtained by the models using the Louvain method, show that these models have issues when the underlying community structure. ROC curves and their AUC values are consistent with the obtained accuracy values, yielding similar behavior.

7. Model integration pipeline

In the previous sections we have shown how we have extracted knowledge from both structural and textual content, each type of content independently addressed and achieving different degrees of performance. In this section, we evaluate the idea of mixing the



(a) Structural content models on full problem (b) Structural content models on reduced problem

Fig. 6. Receiver operating characteristic curves (ROC) for structural content models.

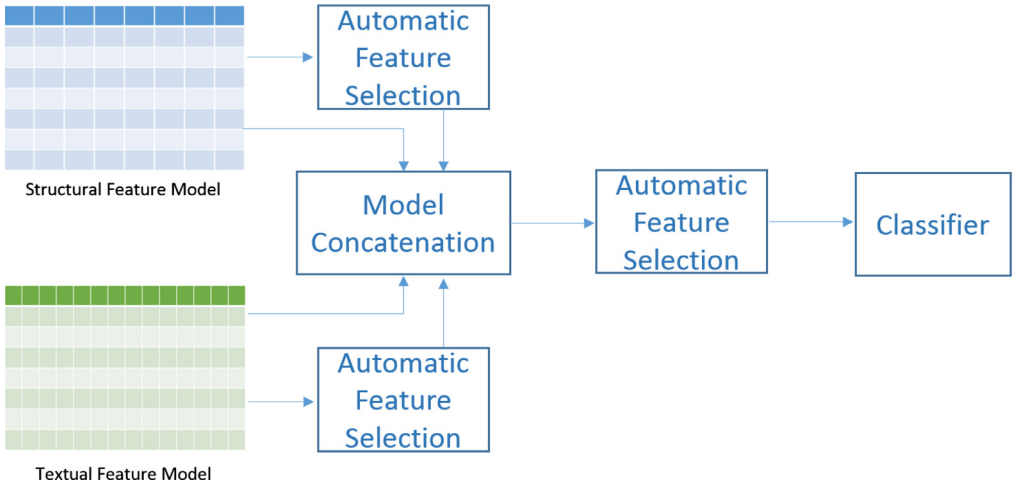


Fig. 7. Direct model combination pipeline.

best feature models of different nature, exploring several ways of combining these models in order to further improve the results.

We have tested three different approaches for feature combination: *Direct combination, Stacked Generalization* and our own proposal *Multiple Pipeline Stacked Generalization*. Each one of these approaches have different steps, so each pipeline is discussed in their respective subsections.

7.1. Direct combination

Our first mixing approach was a direct combination of both feature models into a larger feature model. This combination is done by simply concatenating the feature models into a single vector feature model. We apply an automatic feature selection step a priori, combining the pre-reduced feature models, and a posteriori, reducing the feature sets after the combining the feature models. We note the former model as *BoW-AFS + Bicl-AFS* while the latter is noted *Combined-AFS*. Fig. 7 shows the overall workflow of this pipeline.

We think that classifiers have difficulties when addressing the direct combination by concatenation. The feature models represent different kinds of knowledge thus confusing classifiers since features of different models behave differently and may tend to strongly disagree with each other.

7.2. Stacked generalization

The direct combination scheme did not yield very good results and it is clear that we needed other combination schemes that better handle data from models of different nature.

Ensemble learning methods use multiple learning algorithms to obtain better predictive performance. Though there are some methods that use the same feature model and/or algorithm, ensemble methods tend to yield much better results if there is a significant diversity among the models. Many ensemble methods promote such diversity among the models they combine but the ones in which we are interested are methods that allow different algorithms/feature models.

Stacked generalization or *Stacking* [24,25] involves training a meta-classifier on top of the outputs or predictions of several other classifiers. The underlying idea is that the meta-classifier learns how properly the classifiers learn the training data. Since our problem consists in combining information from very different nature, stacked generalization offer more flexibility than other ensemble methods such as *Bagging* or *Boosting*. Fig. 8 represents the overall workflow of this pipeline, showing how the classifier outputs are the inputs for the meta-classifier. The computation of base classifier outputs are made via internal cross-validation and we used 10-fold stratified internal cross-validation schema.

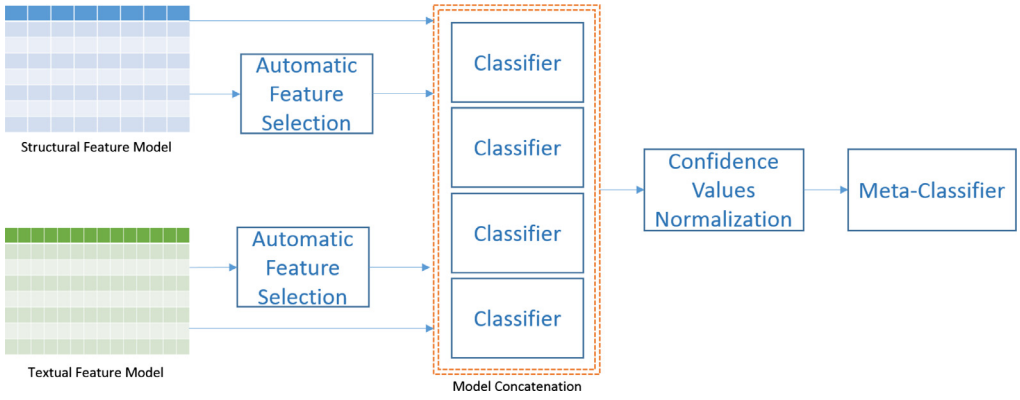


Fig. 8. Stacking Generalization combination pipeline.

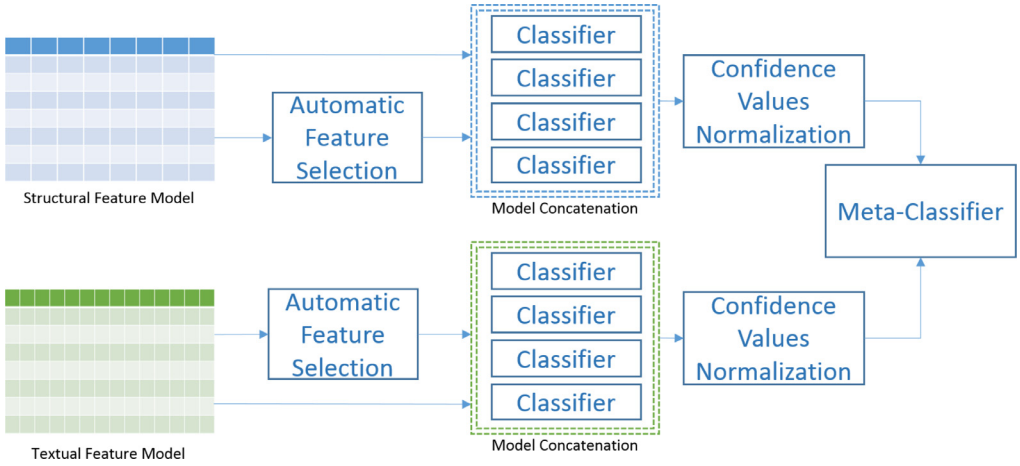


Fig. 9. Multiple Pipeline Stacked Generalization combination pipeline.

7.3. Multiple pipeline stacked generalization

In the previous section, we observed that the stacking technique is able to combine feature models from very different sources successfully. However, individual classifiers exhibit much of the same learning issues because they are fed with the complete feature model in a similar way as in the direct combination approach. Though the metaclassifier does its best to mitigate this situation and achieves some degree of success, we devised a variation of the stacked generalization technique that specifically tries to address this issue.

Our *Multiple Pipeline Stacked Generalization* approach is similar to traditional stacking but it addresses each original feature model in independent pipelines instead of feeding the complete feature model to each individual classifier. Each feature model is processed in a separate pipeline and each one has a proper set of individual classifiers, thus generating specific confidence values for each feature model. Furthermore, each pipeline may have different sets of individual classifiers with different parameters, potentially fitting better to each feature model.

As a result of this independent processing, the tier-1 feature model is larger than in stacking. Being N classes, M feature models and K_m classifiers per feature model, the tier-1 feature model will have $|N| \times \Sigma_{i \in M}|K_i|$ features while the stacking tier-1 model has $|N| \times |K|$ features ($|K|$ independent classifiers). The rest of the process (the meta-classifier learning stage and evaluation) is identical to traditional stacking. Similarly, the computation of base classifier outputs are made via internal cross-validation and we used 10-fold stratified internal cross-validation schema.

Fig. 9 represents the overall workflow of this pipeline, showing how our proposal has separate sets of tier-0 classifiers being independently trained and fed to the meta-classifier.

7.4. Experimental results

We tested the performance of the different model integration approaches tested against the two versions of the proposed task, including all the variants of the direct combination scheme and both the BoW and the Biclustering models for comparison purposes. Table 4 shows the cross-validated accuracy of these approaches.

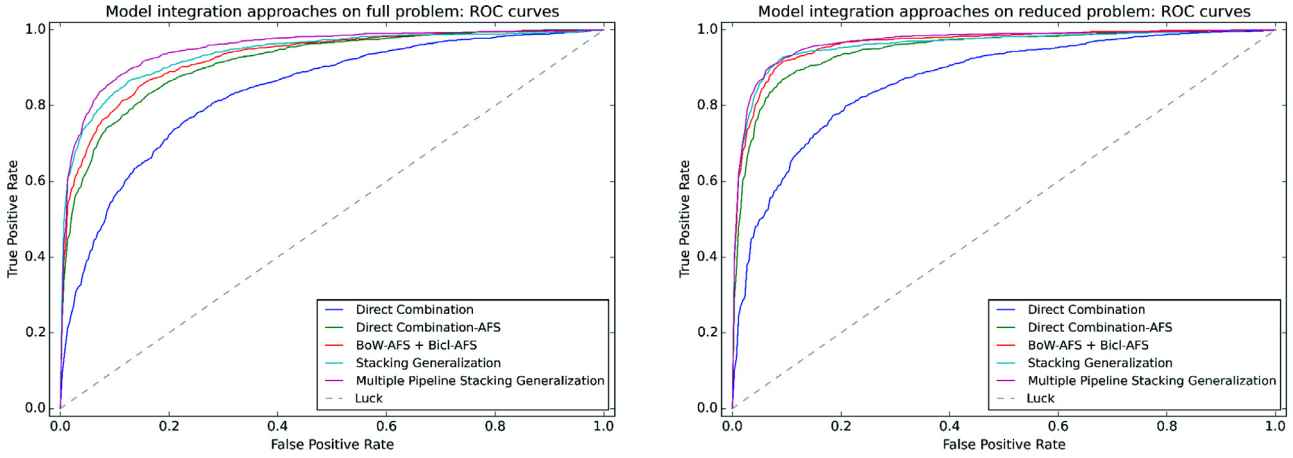
We observe that the direct combination scheme struggles to get better results in most variants, being the pre-reduced approach the only one getting better results. It surpasses any model regarding the reduced problem and its performance is on par with the best when addressing the full problem. As we mentioned before, classifiers have difficulties when addressing the direct combination by concatenation because the models represent very different kinds of knowledge.

The Stacked Generalization approach performs significantly better than any direct combination approach, even applying an AFS step. We experimentally found that the Stacking Generalization approach worked best when combining the BoW-AFS model and bare Bicluster model. Nevertheless, individual classifiers exhibit much of the same learning issues because they are fed with the complete feature model.

Our Multiple Pipeline Stacked Generalization, which used the same models used for Stacked Generalization, yields significantly

Table 4
Cross-validated performance of model integration approaches.

Integration scheme	Full problem Feature model	Reduced problem			
		Accuracy (%)	AUC (%)	Accuracy (%)	AUC (%)
None	BoW-AFS	77.37	90.15	88.38	96.30
	Bicl-AFS	61.11	77.21	69.60	79.78
Direct	Combined	64.79	83.62	71.98	86.75
	Combined-AFS	75.24	91.38	86.68	94.67
	BoW-AFS + Bicl-AFS	77.40	92.65	89.07	96.23
Stacking	Stacked Generalization	79.99	93.69	89.22	95.98
	Multiple Pipeline Stacked Generalization	82.36	95.05	91.22	96.71



(a) Integration approaches on full problem

(b) Integration approaches on reduced problem

Fig. 10. Receiver operating characteristic curves (ROC) for integration approaches.

better results than any other approach on both versions of the problem, proving that our variation was being more effective at combining both feature models.

The receiver operating characteristic curves (ROC) of the model integration approaches, for both full and reduced problem, are shown in Fig. 10a and b respectively, being the area under curve (AUC) values also shown in Table 4. ROC curves and their AUC values are consistent with the obtained accuracy values, yielding similar behavior.

In this work, each pipeline was set up using the best classifiers previously tested for each independent feature model and version of the problem. The hyperparameters for the classifiers are the same ones that we previously used when addressing the problem with each base feature model.

For the Stacked Generalization approach, we found that a combination of SVM-C, Random Forests and Logistic Regression classifiers worked well for both feature models and both versions of the problem.

For the Multiple Pipeline Stacked Generalization approach, when addressing the full version of the problem, we found that a combination of SVM-C, Random Forests, Logistic Regression and Multinomial Naive Bayes worked well for both feature models. However, in the reduced version of the problem we found that the best combination was SVM-C, Random Forest and Logistic Regression for the BoW model while the best for the Biclustering model was SVM-C and Random Forest.

8. Conclusions

In this paper, we propose an approach to the categorization of tweets within the political context, based on the novel idea of combining two different sources of knowledge: the textual content of tweets and the structural information of their underlying social network. This approach differs from the usual focus on textual content commonly found in other approaches in the current literature, and through experimentation, we found that mixing different feature models can yield very good results though it requires of an appropriate combination scheme.

We observed that, after preprocessing and tokenizing the tweets, generating a feature model based in the well-established *Bag-of-Words* was a sensible idea; despite of its simplicity, it achieves a moderate success and surpasses the baselines. This is mainly due to tweets are very short (140 characters max.) and have very little grammar complexity. We found that applying a *TF-IDF* weighting schema did not improve the results.

Nevertheless, this textual feature model substantially grew and it was suffering from dimensionality issues. Applying an automatic feature selection step based on a *forest of extremely randomized trees* was quite effective and with this reduction, the BoW model experiments a huge performance improvement.

When analyzing the structural information, our idea of transforming the underlying information of tweets into a bipartite friendship graph was quite successful when fed to the different community detection techniques. Though both techniques are interesting, the community model generated by the *Spectral Biclustering* is very superior; it yields better results and the model is fuzzier, allowing that any user belongs to many communities with different degrees of community membership.

After each knowledge is independently addressed, we discuss how to design the model combination stage and we test three different ways of mixing feature models from different types of knowledge: direct combination, *Stacked Generalization* and our proposed variant of Stacked Generalization named *Multiple Pipeline Stacked Generalization*. Results show that the pipeline with our

proposed method performs remarkably better than any other feature model and combination method, being effective when combining feature models from both types of content.

We can conclude that mixing both textual and structural knowledge is a good approach for determining the political orientation of tweets and can be applied to tweets from other domains. Extracting knowledge and generating good feature models is harder for structural content than for textual content and making use of both feature models proved to be non-trivial, as direct combination does not properly behave with feature models of different nature. Our proposed combination approach carefully tackles these issues and proved to be effective.

Acknowledgments

This research is partially funded by the national project TIN2012-38536-C03-02 from the Ministerio de Economía y Competitividad of Spain and the regional project P11-TIC-7684 MO from the Junta de Andalucía of Spain.

References

[1] M. Ghiassi, J. Skinner, D. Zimbra, Twitter brand sentiment analysis: a hybrid system using n-gram analysis and dynamic artificial neural network, Expert Syst. Appl. 40 (16) (2013) 6266–6282, doi:10.1016/j.eswa.2013.05.057.

[2] M.M. Mostafa, More than words: social networks' text mining for consumer brand sentiments, Expert Syst. Appl. 40 (10) (2013) 4241–4251, doi:10.1016/j.eswa.2013.01.019.

[3] M. Conover, J. Ratkiewicz, M. Francisco, B. Gonçalves, F. Menczer, A. Flammini, Political polarization on twitter., in: ICWSM, 2011.

[4] A. Tumasjan, T.O. Sprenger, P.G. Sandner, I.M. Welpe, Predicting elections with twitter: what 140 characters reveal about political sentiment., ICWSM 10 (2010) 178–185.

[5] I. Himelboim, S. McCreery, M. Smith, Birds of a feather tweet together: integrating network and content analyses to examine cross-ideology exposure on twitter, J. Comput. Mediated Commun. 18 (2) (2013) 40–60.

[6] K. Ikeda, G. Hattori, C. Ono, H. Asoh, T. Higashino, Twitter user profiling based on text and community mining for market analysis, Knowl. Based Syst. 51 (0) (2013) 35–47, doi:10.1016/j.knosys.2013.06.020.

[7] J. Martinez-Romo, L. Araujo, Detecting malicious tweets in trending topics using a statistical analysis of language, Expert Syst. Appl. 40 (8) (2013) 2992–3000.

[8] A. Babour, J.I. Khan, Tweet sentiment analytics with context sensitive tone-word lexicon, in: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 1, IEEE, 2014, pp. 392–399.

[9] S. Al-Osaimi, K.M. Badruddin, Role of emotion icons in sentiment classification of Arabic tweets, in: Proceedings of the Sixth International Conference on Management of Emergent Digital EcoSystems, ACM, 2014, pp. 167–171.

[10] A. Pak, P. Paroubek, Twitter as a corpus for sentiment analysis and opinion mining, in: LREC, 2010.

[11] S. Xie, J. Tang, T. Wang, Topic related opinion integration for users of social media, in: Social Media Processing, Springer, 2014, pp. 164–174.

[12] M. Speriosu, N. Sudan, S. Upadhyay, J. Baldridge, Twitter polarity classification with label propagation over lexical links and the follower graph, in: Proceedings of the First Workshop on Unsupervised Learning in NLP, Association for Computational Linguistics, 2011, pp. 53–63.

[13] P.P. Talukdar, K. Crammer, New regularized algorithms for transductive learning, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2009, pp. 442–457.

[14] T.A. Small, What the hashtag? A content analysis of Canadian politics on twitter, Inf. Commun. Soc. 14 (6) (2011) 872–895.

[15] C.S. Park, Does twitter motivate involvement in politics? Tweeting, opinion leadership, and political engagement, Comput. Hum. Behav. 29 (4) (2013) 1641–1648.

[16] J.H. Kahn, R.M. Tobin, A.E. Massey, J.A. Anderson, Measuring emotional expression with the linguistic inquiry and word count, Am. J. Psychol. (2007) 263–286.

[17] F.P. Barclay, Political opinion expressed in social media and election outcomes-us presidential elections 2012, J. Media Commun. (JMC) 1 (2) (2014).

[18] S.M. Mohammad, X. Zhu, S. Kiritchenko, J. Martin, Sentiment, emotion, purpose, and style in electoral tweets, Inf. Process. Manage. 51 (4) (2015) 480–499.

[19] J.M. Cotelo, F.L. Cruz, J.A. Troyano, Dynamic topic-related tweet retrieval, J. Assoc. Inf. Sci. Technol. 65 (3) (2014) 513–523, doi:10.1002/asi.22991.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[21] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech.: Theory Exp. 2008 (10) (2008) P10008.

[22] S. Madeira, A. Oliveira, Biclustering algorithms for biological data analysis: a survey, IEEE/ACM Trans. Comput. Biol. Bioinformat. 1 (1) (2004) 24–45, doi:10.1109/TCBB.2004.2.

[23] Y. Kluger, R. Basri, J.T. Chang, M. Gerstein, Spectral biclustering of microarray cancer data: co-clustering genes and conditions, Genome Res. 13 (2003) 703–716.

[24] D.H. Wolpert, Stacked generalization, Neural Networks 5 (1992) 241–259.

[25] L. Breiman, Stacked regressions, Mach. Learn. 24 (1) (1996) 49–64, doi:10.1007/BF00117832.