Analyzing the Retweeting Behavior of Influencers to Predict Popular Tweets, with and without Considering their Content

Matías Gastón Silva, Martín Ariel Domínguez, and Pablo Gabriel Celayes

FaMAF, Universidad Nacional de Cordoba, Córdoba, Argentina
{mgs0113, mdoming, celayes}@famaf.unc.edu.ar

Abstract. Twitter and social networks in general, participate more and more in everyday life. This is why they have become a fundamental source of information that reflects the ideas and opinions of their users. This paper shows how the most influential users, called influencers, can be decisive in defining whether a publication becomes popular or not, regardless of its content. To achieve this, we build a dataset of Spanish-writing users sampled from Twitter, along with the content generated and shared by them within a year. In a first phase, we use different algorithms to detect users who are ”influencers”. In a second phase, we train a binary classifier to predict if a given tweet will be a trending publication, based on information about the activity of the influencers on the given tweet. We obtain a model with an F_1-score close to 79%, based on the retweeting behavior of a 10% of the users dataset considered as influencers. Finally, we add two Natural Language Processing (NLP) techniques to analyze the content: Twitter-LDA topic modeling, and FastText word embeddings. While both models alone have an F_1 of less than 50% for trending prediction, FastText combined with the social model reaches an 86.7% score. We conclude that while analyzing the content can help to predict the popularity of a tweet, the influence of a user's environment in the retweeting decision is surprisingly high.

Keywords: retweet prediction, Social Network Analysis, Machine Learning, LDA, FastText, word embeddings

1 Introduction

The evolution of technology and the constant growth of its infrastructure allow us to be connected to our social networks, anytime, anywhere. Because of this state of permanent communication, social networks today are a vast reservoir of valuable information. One example of how this data is used to the advantage of businesses is the marketing field, where this kind of information is used to learn about the tastes and needs of the population to promote brands. In this sense, influencers have been acknowledged as message replicators and, as so, they are also used as marketing tools. Political campaigns are another instance of the use of social network data. Campaigners develop massive communication strategies that direct specific messages, even fake news, based on profiled users. Therefore, the analysis of these data becomes essential to understand social phenomena and its impact on how a piece of content can be massively spread.

This work attempts to contribute to the understanding of how publications in social networks become popular. In particular, we concentrate on trying to quantify the importance of the behavior of central users in the propagation of information. More specifically, this work is done on Twitter, an online real-time social network, where users can post, read and share information in multiple formats, mostly in the form of short text messages (originally 140 characters, extended to 280 characters in late 2017). In this case, we only analyze written content. Twitter tags each post with a unique timestamp and places the publication on the timeline of its emitter. The users and their timelines are mostly public and can be downloaded through the public API provided by Twitter. On this social network, users have a front page where they can find posts from the people they follow. If someone thinks a message is of interest or likes the content, she can republish it over her timeline. This action is called *retweeting* and represents, at least for us, acceptance of the tweet[1]. The repetition of the retweeting action by multiple users on a given post is the way in which a publication becomes ”popular” in Twitter. Consequently, the subject of the tweet becomes a trending topic.

To address the issue of how a tweet becomes a trending topic, in a first phase, we evaluate different algorithms to effectively detect influencers, which will allow us to rank them by importance. In a second stage, we separate a part of the most influential users and use their retweeting activity to train a binary classifier over tweets. The set of selected features refers to whether a portion of these central users has shared the tweet or not. The target binary variable is whether each publication is popular or not. A tweet is defined as popular if it has been retweeted more than a certain number of times, which we will establish opportunely. The model obtained is evaluated on a set of unseen tweets, reaching an F_1 score of 79.2% in predicting which tweets are popular. Note that these predictions were made without taking into account the content of publications. Subsequently, we add two NLP techniques to analyze the content: word embeddings, with the FastText [10] algorithm, and a Twitter-specific adaptation of the Latent Dirichlet Allocation (LDA)[30] topic modeling technique. The result of combining the model based on central users behavior with FastText, reaches a performance of 86.7%, taking 10% of the users ranked as influencers.

Summarizing, the present work was carried out in the following phases:

- Construction of datasets: a set of Twitter users, the network of follower relations among them and a set of tweets produced or shared by them.
- Selection of an influencer detection algorithm.

[1] We assume that acceptance is the most usual way to use a retweet. However, it is true that not always a retweet represents acceptance, in some cases a retweet could be used to be ironic about a publication, or also, to make visible some topic with which we disagree.

- Study of the network of selected users and detection of most relevant ones in terms of activity and network position, splitting users in two groups: a set of ranked influencers and a set of regular users.
- Comparison of models to learn and predict general retweeting preferences on a dataset of tweets, based on information about the influencers set.
- Study of possible improvements to social prediction models, introducing NLP techniques such as topic modeling and sentence embeddings.

The rest of this paper is structured as follows: In Section 2, we analyze related works in the area, comparing them to our work. In Section 3, we describe how we build the datasets from Twitter for our experiments. Next, in Section 4, we describe the details of the construction of our social model for prediction of popular tweets. We also include information on how we add content-based features using the Twitter LDA topic modeling and FastText word embeddings. Finally, Section 5 contains the analysis of the results obtained and in Section 6, we present our conclusions and possible lines of future research.

2 Related Work

Along with the evolution of social networks, the academic studies based on them have increased in quantity and quality, with many works studying the problem of predicting popular or viral content.

A recurring topic among these works is the analysis of the content of the publications as in [11, 28, 22]. In particular, in [11], a genetic algorithm is proposed to optimize the composition of the message to increase its outreach. In this case, the authors take a different approach from ours, generating a simulation over an artificial network similar to Twitter, where nodes decide in a deterministic way whether or not to retweet a given message. Here, the focus is on the generation of content, without considering social features. Among these purely content-based works, [17] is more closely related to our study. They develop purely content-based models for predicting the likelihood of a given tweet being retweeted by general users. The performance of their models is reported only through ROC curves, without providing any overall performance score to establish a precise quantitative comparison to our model. However, a visual comparison between their ROC curves and the ones produced by our final models indicates a higher AUC score in our results. This study also provides a feature importance analysis, which produces very revealing insights about what makes a tweet popular.

Another point of view, more similar to ours, is the focus on the social environment of users rather than the content being spread, which can be found in [29, 26]. In [29], the authors work with different mechanisms to infer when people are likely to initiate a new activity. After the experiments, the conclusion was that the testimonial comments of neighbors were more relevant than promotion messages showing the advantages of such activity. In addition to the increase in registration, permanence was also improved more by peers influence than by typical promotion. As expected, without any promotion, the inscription and permanence rates were much lower than the ones in the scenarios described above. This case is a practical experiment that only shows the conclusions, but no models are provided at the end of the investigation. In [6], the author predicts retweets from a given user based mostly on the retweeting behavior in her second-degree social neighborhood with an average F_1-score of 87.9%. Our work tries to expand this idea to a more general model, focusing on a community instead of a single user.

Finally, the work in [27, 18] conducts trendy research that analyzes the flow of fake information. Here the authors evaluate the propagation of fake news over Twitter and find out that this kind of news is more viralized than real ones. Another revealing insight was that the propagation was faster for publications with fake information. Once again, this work gives more importance to the content, but it also captures the idea of influencing users by a synthetic environment with fake content or users.

3 Dataset

In this section, we describe the dataset used in this work for all experiments. The base dataset (social graph and tweets) is taken from the previous work [6]. We extend this base with more content (almost double), keeping the same social graph of users. We explain the construction of our dataset in two steps: first building the social graph of users and then getting content shared by them.

3.1 Social Graph

To perform the experiments of this paper, we reuse a dataset created for the previous work [6], which contains Twitter users and the who-follows-whom relations between them. Back then, the idea was to create a minimal representative dataset of Twitter where all users would have a similar amount of social information about their neighborhood of connected users. The decision was to build a homogeneous network where each user has the same number of followed users.

To this end, a two-step process was performed. Initially, a large enough *universe graph* was built, which was subsequently filtered to obtain a smaller but more homogeneous subgraph.

The *universe graph* was built starting with a singleton graph containing just one Twitter user account $\mathcal{U}_0 = \{u_0\}$ and performing 3 iterations of the following procedure: (1) Fetch all users followed by users in $\mathcal{U}_i$; (2) From that group, filter only those having at least 40 followers and following at least 40 accounts; (3) Add filtered users and their edges to get an extended $\mathcal{U}_{i+1}$ graph.

This process generated a *universe graph* $\mathcal{U} := \mathcal{U}_3$ with $2,926,181$ vertices and $10,144,158$ edges.

For the second step, in order to get a homogeneous network (note that many users added in the last step might have no outgoing edges), a subgraph was taken following this procedure:

- We started off with a small sample of seed users S, consisting of users in $\mathcal{U}$ having out-degree 50, this is, users following exactly 50 other users.

- For each of those, we added their 50 most socially affine followed users. The affinity between two users was measured as the ratio between the number of users followed by both and number of users followed by at least one of them.
- We repeated the last step for each newly added user until there were no more new users to add.

This procedure returns the final graph $\mathcal{G}$ with $5,180$ vertices and $229,553$ edges, called the homogeneous K-degree closure ($K = 50$ in this case) of S in the universe graph $\mathcal{U}$.

3.2 Content

The content dataset is composed of $1,636,480$ tweets inherited from previous work extended with a set of $2,237,287$ new tweets. These tweets result from extracting the content written in Spanish from user's timelines in $\mathcal{G}$ for dates between March 2016 and February 2017. This does not mean that we have all the tweets of every user in this period of time. Due to the limitations of the API (30 days at the moment of collecting the data) it is imposible to fetch old tweets.

4 Experimental setup

In this work, we aim to build models capable of accurately predicting the acceptance that a tweet t could have over the general audience of users ($U_G \subset \mathcal{G}$), based only on the reaction of influencers ($U_I \subset \mathcal{G}$) to the publication. This section describes how we set up models for this purpose over a selection of users and tweets from the $(\mathcal{G}, \mathcal{T})$ dataset defined before.

First, we start with the predictive model based only on social features. Then we move on to explain how additional content-based features were incorporated to improve predictions, giving details about NLP techniques, namely an adaptation of LDA topic modeling to Twitter and sentence embeddings based on the FastText algorithm.

4.1 Social Prediction

The primary focus of this work is to predict if a tweet t will have enough retweets from general users to consider it as *trending tweet* based on information on which of the influencers from U_I has shared it.

Even though the dataset is homogeneous enough considering connections, there are still inactive users in the network. Users that only use the social network in passive mode without engaging in any tweeting or retweeting activity are omitted. As regards the content dataset, as expected, most of the tweets are shared only by its author. This behavior causes an imbalance in the classification that affects the performance. It can be fixed filtering out those irrelevant tweets.

Therefore, we begin this section with an explanation of our filtering processes to select relevant users and tweets. After that, we detail how we proceed to get the influencers U_I from $\mathcal{G}$ and which algorithms we use to that purpose. Finally, we explain the feature extraction and dataset splitting for training and testing the models without any data overlap between those tasks.

User selection As mentioned before, the inactive users are omitted in this experiment because they are unpredictable by nature. We consider that a user in our dataset is passive if she has less than ten retweets in her timeline. Filtering those out leaves us with a set of only 3626 active users in $\mathcal{G}$. We restrict the analysis to those users, also removing content shared only by inactive users from $\mathcal{T}$.

Trending tweets We call a tweet *trending* if we consider it popular enough to possibly become a trending topic. This consideration is related to the number of retweets it earns over the general public U_G. To get the *golden value* of retweets considered enough to consider a given tweet as popular, we analyzed and built a histogram of how many retweets each tweet in $\mathcal{T}$ receives.

Initially, we wanted to use the value in the 90th percentile as our golden value, but given the fact that most tweets are shared only by their author, this value turned out to equal 1. So we decided to discard all the tweets with less than 3 retweets, which caused this percentile to increase to 13, allowing us to implement more accurate models. Therefore, we consider a tweet *trending* if it was retweeted at least 13 times.

On the other hand, it is important to remark that the experiments carried out make sense only within the context of U_G users, keeping in mind that the goal of this work is to analyze the influence of the U_I group over general users. That is why we are interested only in those tweets from $\mathcal{T}$ that showed up on the *timeline* of at least one user in U_G, defining $T' := \left(\bigcup_{x \in U_G} timeline(x)\right)$.

Influencers detection Much effort has been made by the research community in influencers detection [7, 19, 25, 1]. However, most of the works are based on supervised methods, which are not applicable in our case, since we do not have a labeled corpus of influencers.

We decided to use the ideas included in [1], which proposes a combination of three types of features: network centrality, activity level and profile features. Since we didn't have any extended profile information in our dataset, we focused on centrality and activity. This has the advantage of making the results more generalizable to other social networks without depending on specific information that might be available only in Twitter, and for certain users.

To measure the *centrality* of a user we apply an average of metrics computed by the following algorithms: PageRank[20], Betweenness[9], closeness [23], Eigenvector centrality [3] and Eccentricity [4] included in `igraph` Python package[8]. The *activity* level of a user is computed simply as the average of the number of tweets and the number of retweets posted by users.

To decide the best option to rank users as influencers, we compared different weighted combinations of centrality and activity measures, $\alpha * Centrality + (1 -$

$\alpha) * Activity$, where α controls the importance given to *centrality*. In Figure 1, we can see that the best results were obtained for a simple mean of both metrics ($\alpha = 0.5$). To compare the performance of these options a subset of 500 random tweets from $\mathcal{T}$ was set aside. This sample called T_{SI} is removed from $\mathcal{T}$ to avoid considering them as part of the test set, where trending prediction models will be evaluated later.

In Figure 1, we show the results for the different alternatives. Each curve is plotted using the selected ranking and running the purely social prediction over T_{SI}, splitting $75\% - 25\%$ for training and test. The y-axis details F_1 score for prediction, while the x-axis reflects the number of influencers, chosen with the evaluated ranking, used for social feature extraction in the models, detailed later in this Section.

Figure 1 reveals that a very central user would be useless for this study if she has a low level of activity and, similarly, a very active user has no value as an influencer if she is not sufficiently well connected. The comparison of these results indicates that the best choice for measuring the influence level of users is the average of centrality and activity.

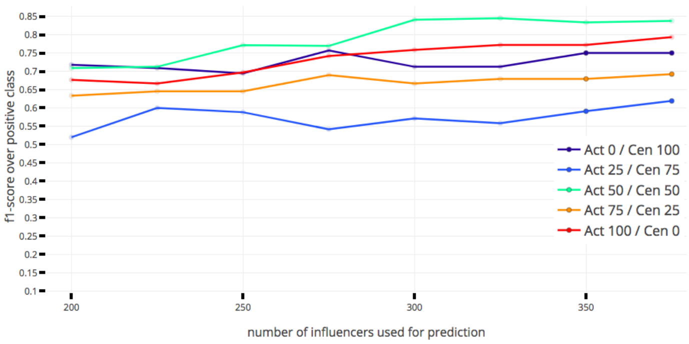


Fig. 1: Comparison of alternatives of influence detection where *Act* involves features related with Activity and *Cen* those related with Centrality. The curves correspond with the pure social model performance prediction over T_{SI}.

Now that we have selected our metric, we apply it to $\mathcal{G}$ without these 500 tweets from T_{SI}, to get a ranking of all users by level of influence. We take the top 25% as our set of *influencers* and call it U_I, the rest of the users are considered the general audience and called U_G. The goal of the social models described later is to predict the level of acceptance of tweets among the general audience U_G, based on knowledge about the activity of the influencers U_I on them. The idea for the experiments described in the following sections, is to vary the number of influencers taken from U_I to predict the popularity of tweets.

Social Features As mentioned earlier, we need to train a classifier model to make predictions. For that purpose, it is necessary to define the feature vector and the target vector. For the feature vector, in the social based model, we only consider the retweeting behaviour the selected influencers have over tweets from the training set. For each tweet t, we can define a binary vector $T_t := \left[\, i_{t1}\; i_{t2}\; \ldots\; i_{tn} \,\right]$, where n is the number of influencers, and each i_{tj} is 1 if the tweet t was retweeted by the influencer j, and 0 otherwise. More formally, let the function $TM(j)$ return the set of tweets in the timeline for influencer j. Grouping in a matrix all the vectors associated with the m tweets, the input for the model becomes:

$$features := \begin{bmatrix} i_{11} & i_{12} & \ldots & i_{1n} \\ \ldots & \ldots & \ldots & \ldots \\ i_{t1} & i_{t2} & \ldots & i_{tn} \\ \ldots & \ldots & \ldots & \ldots \\ i_{m1} & i_{m2} & \ldots & i_{mn} \end{bmatrix} \text{ where } i_{tj} = \begin{cases} 1 \text{ if } t \in TM(j) \\ 0 \quad \text{otherwise} \end{cases}$$

Note that the content of tweet t is not considered, we only include the information about which of the users in U_I retweeted t. Now, as part of the supervised method, we use the following objective vector, calculated over the training set of tweets. Let $RT(t)$ be a function that returns the number of retweets in U_G for the tweet t; we define the target vector as follows:

$$classification = \begin{bmatrix} r_1 \\ \ldots \\ r_t \\ \ldots \\ r_m \end{bmatrix} \text{ where } r_t = \begin{cases} 1\; RT(s) >= \mathit{golden\ value} \\ 0 \qquad\quad \text{otherwise} \end{cases}$$

4.2 Splitting the Dataset

To evaluate the performance of our models, we divide our dataset of tweets into two parts, one for training and another for evaluation. As usual, these datasets are not overlapping. In other words, the evaluation data is not seen by the training algorithms.

Regardless of the chosen number of influencers for prediction, we want the training and evaluation datasets to remain disjoint. In this sense, as we explained previously in this section, the left diagram in Figure 2 shows how we split the set $\mathcal{G}$ in two disjoint parts, U_I (influencers users) and U_G (common users). For the all other experiments of this paper, U_I is defined as the 25% best-ranked users from $\mathcal{G}$, using the average of centrality and activity to detect influencers (Figure 1).

To determine well-formed training and test sets for tweets, we drop from the $\mathcal{T}$ dataset the tweets posted by users in U_I named T_I. In addition, it is also necessary to cut from $\mathcal{T}$ the set T_{SI} used previously in this section to detect influencers. The remaining tweets, i.e. $T_G = \mathcal{T}' - T_I - T_{SI}$ are split again. To do so, T_G is randomly split in training (75%) and test (25%) datasets to evaluate prediction models. For clarification, the right diagram in Figure 2 describes these splits.

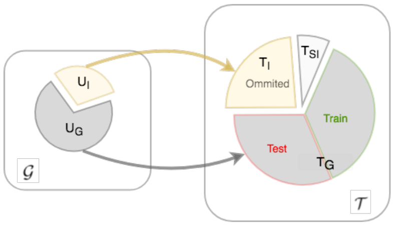


Fig. 2: The left chart distinguishes general users (set U_G) from influencers (set U_I). The right chart shows how to obtain training and test datasets

4.3 Adding Content-based Features

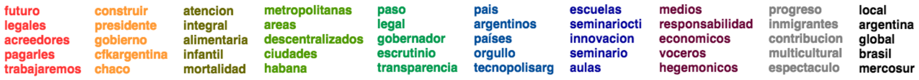
To achieve an increase in the quality of trending tweet prediction, we apply NLP techniques to to extend the purely social model with content-based features. Representing text content with vocabulary-based representations such as TF-IDF introduces problems of efficiency and overfitting due to large dimensionality. That is why it is convenient to use more compact vector representations that somehow manage to encode semantic similarity between texts. Trying the most popular algorithms for this task, we found that Twitter-LDA as a topic extractor and FastText as a sentence embedder were the options that best fit in our experiments. Both are described later in this section.

Preprocessing To begin with, we enumerate the sequential transformations performed to turn a tweet into a vector of numeric features describing its content.

- **Normalization** In the first step, we remove the following for normalizing purposes: URLs, accents, unusual characters, numbers and stopwords
- **Tokenization** Next, we convert the text to lowercase, split it into tokens and apply lemmatization for Spanish language to all words. We use the `spaCy` package[12] for this stage. The resulting representation as a sequence of normalized tokens is the basis for both Twitter-LDA and FastText representations.

Twitter-LDA Twitter-LDA[30] is a variant of the classic LDA topic modelling algorithm used in [6], specially tuned for short text documents like tweets. The LDA model enables us to discover a given number of underlying topics within a given corpus, generating a representation of each topic as a probability distribution over the words. Additionally, it reduces dimensionality by representing the each text with a topic-based distribution. The Twitter-LDA adaptation modifies the assumptions of LDA by restricting each tweet to just one relevant foreground topic, and adding an extra ”phantom” topic of background words used to model uninformative vocabulary in each tweet. Moreover, tweets are grouped by user during the training phase, allowing the model to pick up more topical patterns than it would by treating short texts in isolation.

We experimented with different numbers of topics on the training dataset: 5, 10, 15, 20 and then incrementally by adding 10 topics up to 80. In all cases, we validate the experiments only using the training set. The best results are obtained using 10 topics. This produces a one-hot encoded 10-dimensional representation of tweets, where the coordinate corresponding to the topic assigned to a tweet is set to 1, and all the rest are set to 0. Some examples of the resulting topics and their top-five words are shown in Figure 3. Note that words that represent a topic bear a semantic relation between them. The first topic in the Figure groups “legales” (legal), “acreedores” (creditors) and “pagarles” (to pay) which belong to the same semantic field.



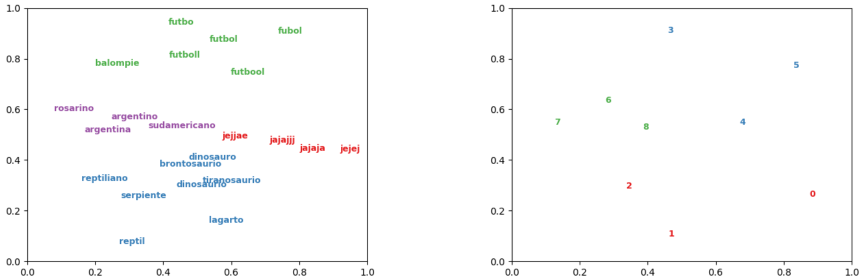
futuro	construir	atencion	metropolitanas	paso	pais	escuelas	medios	progreso	local
legales	presidente	integral	areas	legal	argentinos	seminariocti	responsabilidad	inmigrantes	argentina
acreedores	gobierno	alimentaria	descentralizados	gobernador	países	innovacion	economicos	contribucion	global
pagarles	cfkargentina	infantil	ciudades	escrutinio	orgullo	seminario	voceros	multicultural	brasil
trabajaremos	chaco	mortalidad	habana	transparencia	tecnopolisarg	aulas	hegemonicos	espectaculo	mercosur

Fig. 3: An example of top words in 10 Twitter-LDA topics from twitter dataset

FastText *Word embeddings* refers to a family of different techniques that associate vector representations to input words. Conceptually, the idea is to map a discrete large-dimensional bag-of-words representation of a corpus into a continuous space of fewer dimensions. The resulting representations have the property that words with similar meanings correspond to nearby vectors as we can see in the left plot of Figure 4. As a consequence, this kind of representation improves efficiency and reduces overfitting without loss of information.

In this work, we use the FastText implementation [10] of word embeddings, which is presented as an alternative to the traditional Word2Vec model [15]. One of its most prominent features is the possibility of assigning vectors to words not seen during the model training, looking for matches on character n-grams to vectorize those out-of-vocabulary words. This makes it more robust for handling misspelled words that are commonly found in social media text. We use a pre-trained model of 100 dimensions, included in the FastText library from [10]. Although word embeddings models provide vector representations for single words only, convolution functions can be applied to obtain vectors of the same dimensionality that represents whole sentences or paragraphs. In the case of FastText library, a given text is represented as the average of all the vectors of its component words.

The left plot in Figure 4 shows some examples of Spanish words with similar meaning which are plotted in the same color, and are close to each other. For example, the words “jajaja” and “jejej” are different ways to indicate laughing. The right side plot of Figure 4 shows the distance of FastText vectors for the tweets at the bottom of the Figure. We plotted with the same color tweets with similar meanings: tweets 1 and 2 are very close to each other (in English: *“it can’t be possible, lol”* and *“no way, lol”*, respectively). For the 2D visualization of the 100 dimension FastText vectors we used the Multi-Dimensional Scaling algorithm included in `scikit-learn.manifold` package. As expected, tweets representations are close if their content is similar.



Numbered examples of tweets: 0."jajajaj que risa que me da" - **1.**"esto no puede ser jejeje" - **2.**"no hay forma de ser jajjja" - **3.**"Restos de tiranosaurios fueron encontrados en Neuquén." - **4.**"Este es otro tweet que habla de dinosaurios y cocodrilos" - **5.**"restos dinosaurios dar lugar lagartos y lagartijas" - **6.**"Hablar de fútbol en argentina es muy difícil" - **7.**"Uruguay dejo un buen papel ante argentina y empato el partido para lograr su clasificacion." - **8.**"sigue complicado el inicio del torneo argentino de futbol la afa cada vez peor"

Fig. 4: Two-dimensional visualization of FastText vectors for selected examples of words (left) and tweets (right).

5 Results

Now we describe how we build our predictive models and the results obtained with and without content analysis. We will compare our models to a baseline built from a purely social model where users considered influencers are selected randomly instead of using an influencer detection algorithm. With this we want to show the utility of using an algorithm to detect influencers, and the relevant information those provide for learning about the behavior of general users.

5.1 Baseline

As a baseline, we use a model that is sufficiently demanding to be compared with our proposals. We decided to use the same kind of features as in the pure social version, but randomly selecting a set of 25% of the users from $\mathcal{G}$ as the set of influencers U_I .

To make a fair comparison with our models we do a new split from the dataset $\mathcal{T}$ to T_I and T_G with the content of users in the random selection of U_I and U_G respectively. In turn, a $75\% - 25\%$ train-test split is performed on T_G for the training and evaluation of the baseline models under the same conditions as in the social alternative. We keep the datasets disjoint and evaluate over general users with influencer behavior data as input.

Following the same pattern as in the other social models, we then proceed to evaluate the social baseline over increasingly large numbers of users from U_I taken as the source of social features. In this case we do not have a ranking of users to draw the top ones from, so we make these selections randomly as well. In order to calculate the baseline performance, for each value of the number of source influencers (let us call this k), we randomly select k users from U_I, and train and test a model using the train-test split of T_G. To avoid lucky and potentially misleading results, we repeat this process five times for each value of k, reporting the average $F1$-score.

The results of the baseline score can be seen in Figure 5. As expected, the yield curve of the baseline is always much lower than the performance of the pure social model with detection of influencers.

5.2 Social Models

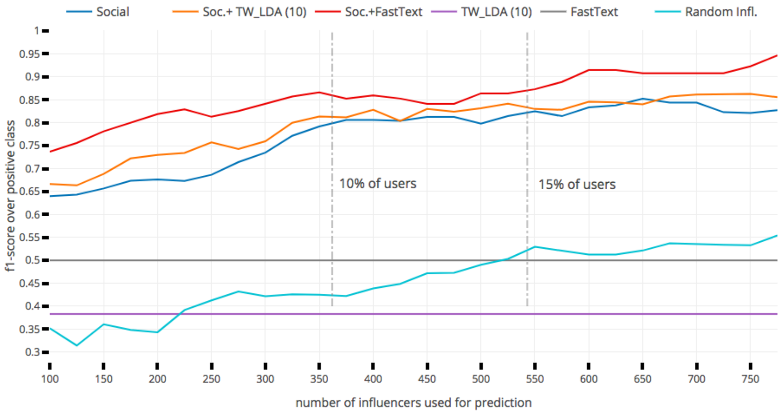


Fig. 5: F1-score on experiments with and without content analysis.

Now we show the results obtained from training and evaluating trend prediction models with the features described in Section 4.1. We used Support Vector Machine models for classification, more precisely the `SVC` implementation from `scikit-learn` [21], combined with its `GridSearchCV` class for search of optimal hyperparameters through cross-validation over the training set.

We decided to focus on the experiments considering 10% and 15% of $\mathcal{G}$ as influencers. These values would still return relevant results to our purpose while letting the trained model with enough information. That is the reason why in Figure 5 we put the vertical lines showing these values. There, we can see that considering 10% of the user space as influencers we have an F_1-score near to 78% over the test data. Details about scoring can be seen in Table 1. In this figure, we can also observe the comparison with the baseline model. Here, we confirm that not all users bring the same information. There is a group that can exert influence over their social environment and another that shows the follower's behavior despite the content.

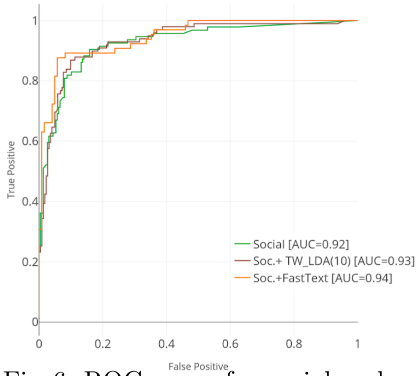


Fig. 6: ROC curve for social and combined models, using top-10% of U as influencers ($\subset U_I$).

Model	F_1	Pr .	Rec.
Baseline	42.5%	29.8%	63.8%
TW-LDA(10)	38.3%	30.5%	51.5%
FastText	49.5%	34.3%	92.1%
Social	79.2%	75.8%	82.9%
Soc.+TW-LDA(10)	81.4%	79.1%	83.8%
Soc.+FastText	86.7%	88.7%	84.6%

Table 1: Performance evaluations over U_G, using top-10% of U as influencers ($\subset U_I$). TW-LDA(10) refers to model Twitter-LDA with 10 topics.

5.3 Social + NLP Models

In this section, we present improved models that add content-based features to the Social Model. Looking for improvements in the scores, we try two alternatives for content analysis: Twitter-LDA[30] and FastText[10]. We apply the first option to discover topics among the tweets and tag each of them by its topic. On the other hand, FastText is used to provide compact dense vector representations of tweets in a way that captures semantic similarities between their content. The feature vectors for combined models are built as follows:

Social+Twitter-LDA: the feature vector of the Social Model is extended by appending the 10-dimensional boolean vectors from Twitter-LDA Model described in Section 4.3.

Social+FastText: In this case, the vectors of social features described in the previous section are extended by appending the 100-dimensional vector from FastText Model described in Section 4.3.

Even though the purely content-based models performed poorly (even worse than the baseline in some cases), the combined models using content-based and social features obtained the best scores. In Table 1, we compare the baseline with the two new models. The improvement of Twitter-LDA[30] alternative was about 2% over the Social model, obtaining almost the double of performance over the baseline. On the second model, with FastText[10] embeddings we also improved the performance. This time the increase was about 8% over the social model, which makes this model the best fit in our experiments with an 86.7% efficiency ceiling. Also, Figure 5 shows the performance of combined models using different numbers of influencers from U_I. It is clear that the FastText combined model obtains the best performance. Finally, in Figure 6 we include ROC curves for social and combined models, which makes it possible to compare our work to the previous content-based work in [17]. In the social and combined cases, we use the full set of influencers U_I for the social features.

6 Conclusions and Future Work

As a general conclusion, we confirm that the information about social connections between Twitter users and their activity can be essential to determine which content becomes popular. We obtained a surprisingly high performance without analyzing the content, which seems to suggest that the source of information has a stronger influence than the actual content when it comes to spreading it across the network. The purely content-based model was far below from the social-based pure model scoring, which reinforces the idea that sometimes our contact lists can provide more information about us than our timeline. Anyway, the combined model with content analysis increased the performance significantly (especially when using FastText word embeddings), which indicates that content still has a level of importance when it is considered within a certain social context. FastText seems particularly well suited for dealing with content from Twitter, specially because of its ability to obtain representations for unseen or misspelled words.

This research opens many doors to evolve the model. The most relevant to us are described next.

A possible improvement is training the model exclusively with tweets published earlier than the tweets used in the test stage. Keeping in mind the temporal variable, using techniques such as Early Prediction [13], we could make a model capable of predicting popularity with the information available on the first minutes of the tweet creation. Later, we can improve this by using Deep-Learning[2]. For influencers detection, alternatives such as [16] and [1] could be applied to improve the selection of relevant users.

We also propose to conduct research about the aggregation formula for sentence embeddings. We have used a simple average of the vectors of the component words, but there are other more sophisticated functions, such as the weighted average by the inverse document frequency (IDF) [24]. Furthermore, we shall test other embedding models such as Doc2Vec [14] and compare results. Additionally, instead of using the default 100-dimensional pre-trained Spanish model from FastText, we can consider other possibilities such as using a model trained on the Spanish Billion Word Corpus from [5]. To that end, we can train a custom model on our dataset of tweets, or attempt to combine both datasets somehow.

Finally, an interesting line of open research is trying to replicate the experiments for other social networks such as Facebook and Instagram, and see to what extent our conclusions are applicable to those. In particular, the pure social model can be extended to any network of users sharing content, which makes it possible to evaluate it even in image-based networks such as Instagram. However, we are limited by the availability of data to build datasets.

References

1. Azcorra, A., Chiroque, L.F., Cuevas, R., Anta, F., Laniado, H., Lillo, R.E., Sguera, C.: Unsupervised scalable statistical method for identifying influential users in online social networks. Scientific Reports **8**, 6955 (05 2018)

2. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning **2**(1), 1–127 (2009), also published as a book. Now Publishers, 2009.
3. Bryan, K., Leise, T.: The $25,000,000,000 eigenvector: the linear algebra behind google. SIAM Review **48**, 569–581 (2006)
4. Buckley, F., Harary, F.: Distance in graphs. Addison-Wesley (1990)
5. Cardelino, C.: Spanish billion word corpus and embeddings, `http://crscardellino.me/SBWCE/`
6. Celayes, P.G., Domínguez, M.A.: Prediction of user retweets based on social neighborhood information and topic modelling. In: Advances in Artificial Intelligence: Proceeding of 16th MICAI. LNCS, Springer, Mexico (2017)
7. Cossu, J.V., Dugué, N., Labatut, V.: Detecting real-world influence through twitter. 2015 Second European Network Intelligence Conference pp. 83–90 (2015)
8. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal **Comp. Systems**, 1695 (2006), `http://igraph.org/python/`
9. Freeman, L.C.: A set of measures of centrality based on betweenness. Sociometry **40**(1) (Mar 1977)
10. Grave, E., Mikolov, T., Joulin, A., Bojanowski, P.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017n. pp. 427–431. Spain (2017), `https://fasttext.cc/`
11. Hochreiter, R., Waldhauser, C.: A genetic algorithm to optimize a tweet for retweetability. Mendel pp. 13–18 (2013)
12. Honnibal, M., Johnson, M.: An improved non-monotonic transition system for dependency parsing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1373–1378. ACL, Portugal (2015), `https://spacy.io/`
13. J.E.Smith, Tahir, M., Sannen, D., van Brussel, H.: Making early prediction of the accuracy of machine learning applications. In: Lughofer, E., Sayed-Mouchaweh, M. (eds.) Learning in non-stationary environments: Methods and Applications, pp. 121–151. Springer, New York (2012)
14. Lau, J.H., Baldwin, T.: An empirical evaluation of doc2vec with practical insights into document embedding generation. In: Proceedings of the 1st Workshop on Representation Learning for NLP. pp. 78–86. Association for Computational Linguistics (2016)
15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs 1301.3781** (2013)
16. Morone, F., Min, B., Bo, L., Mari, R., Makse, H.A.: Collective Influence Algorithm to find influencers via optimal percolation in massively large social media. Scientific Reports 6 p. 30062 (Jul 2016)
17. Nasir, N., Gottron, T., Kunegis, J., Alhadi, A.C.: Bad news travel fast: A content-based analysis of interestingness on twitter. In: WebSci '11: Proceedings of the 3rd International Conference on Web Science (2011)
18. Naveed, N., Gottron, T., Kunegis, J., Alhadi, A.C.: Bad news travel fast: A content-based analysis of interestingness on twitter. In: Proceedings of the 3rd International Web Science Conference. pp. 8:1–8:7. WebSci '11, ACM, New York, NY, USA (2011)
19. Neves, A., Vieira, R., Mourão, F., Rocha, L.: Quantifying complementarity among strategies for influencers' detection on twitter1. Procedia Computer Science **51**, 2435 – 2444 (2015), international Conference On Computational Science, ICCS 2015

20. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Tech. rep., Stanford University (1999)
21. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011), `http://scikit-learn.org/`
22. Pennacchiotti, M., Popescu, A.M.: A machine learning approach to twitter user classification. In: Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia. vol. 11. Spain (2011)
23. Sabidussi, G.: The centrality index of a graph. Psychometrika **31**(4) (1966)
24. Sanjeev Arora, Yingyu Liang, T.M.: A simple but tough-to-beat baseline for sentence embeddings. In: Proceeding of International Conference on Learning Representations, ICLR 2017, April 24 - 26, Toulon, France (2017)
25. Simmie, D.S., Vigliotti, M.G., Hankin, C.: Ranking twitter influence by combining network centrality and influence observables in an evolutionary model. J. Complex Networks **2**(4), 495–517 (2014)
26. Uddin, M.M., Imran, M., Sajjad, H.: Understanding types of users on twitter. CoRR **abs/1406.1335** (2014)
27. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
28. Vougioukas, M., Androutsopoulos, I., Paliouras, G.: Identifying retweetable tweets with a personalized global classifier. In: Proceedings of the 10th Hellenic Conference on Artificial Intelligence, SETN 2018, Patras, Greece, July 09-12, 2018. pp. 8:1–8:8 (2018). https://doi.org/10.1145/3200947.3201019
29. Zhang, J., Brackbill, D., Yang, S., Centola, D.: Efficacy and causal mechanism of an online social media intervention to increase physical activity: Results of a randomized controlled trial. PM Reports **2**, 651 – 657 (2015)
30. 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 the 33rd European Conference on Advances in Information Retrieval. pp. 338–349. ECIR'11, Springer-Verlag, Berlin, Heidelberg (2011)