Twitter Early Prediction of preferences and tendencies based in neighborhood behavior

Emanuel Meriles, Martín Ariel Domínguez, and Pablo Gabriel Celayes

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

Abstract. In recent years, social networks have become increasingly massive. Consequently, they are a fundamental source of information and a powerful tool to spread ideas and opinions. Based on Twitter, this paper studies the problem of predicting the retweet preference of a user for a given tweet, considering how the tweet has been shared by that user's environment; and also the more global problem of predicting whether a tweet will be popular, based on the retweet behavior of central users. For both problems, we explore the evolution of prediction quality depending on the amount of information available over the time since a tweet is created, and derive insights about the trade-off between elapsed time and prediction performance.
For the user retweet prediction problem, this social prediction model achieves, for example, around 63.76% on F_1 score by using the first 15 minutes information, 75.2% by 4 hours, and 86.08% without considering any time window. In the case of popularity prediction, we achieve scores of 65.67% with 60 minutes of information, and 74.4% with 4 hours and 80.73% with no time window restriction, using the behaviour 15% of users considered as influencers. All these results are obtained without considering the content of the tweets. Next we incorporate features based on FastText word embeddings to represent the content of tweets. While such models alone attain an F_1 of around barely 50% for preferences and popularity prediction, when combined with social models they improve the popularity prediction, in most cases, more than 4%. For the case of preference prediction, FastText model is more useful in small time spans. We conclude that it is possible to reasonably predict the preference of a user retweet or how massive a publication will be, using only the information available during the first 30-60 minutes.

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

1 Introduction

In the last years, social media platforms like Twitter, Facebook and Instagram have gained increasingly massive adoption, becoming a central presence in everyday life and in public discussions, with a strong impact in the way people express themselves, get informed and influence each other. One of the most prominent of

these social networks is Twitter, an online real-time microblogging platform that enables its users to post, read and share short messages of up to 280 characters, known as tweets. Every time a user publishes a tweet, Twitter attaches to it a unique identifier and a creation timestamp. A frequently used function on the bird net is the "retweet", which allows a user to republish a tweet from another user within her own message stream (the "timeline"). A feature distinguishing Twitter from other popular social networks is that most of the shared content is public by default, and structured data is easily accessible through the official API, both about the way users interact with each other and the content that they share. This is perhaps one of the main reasons why most research work is done on Twitter.

The influence of social media can be studied both at an individual and at a collective level. In the first case, efforts are concentrated in understanding how the preferences and behavior of a given user are shaped by their previous social media activity. On the other hand, collective phenomena can also be studied and analyzed seeking to understand the mechanisms that cause publications to become popular among greater audiences. In our previous research efforts, we have worked on formulations of both of these targets as predictive modelling problems. In [2], we studied the problem of predicting the content preferences of a user based on the preferences of her neighborhood. Namely, given a tweet, how much we can predict whether or not a user will retweet it based on the retweets it gets among its nearby social connections. In [12], we took these ideas to the community level predicting popularity of tweets instead of individual preferences, and replacing the immediate set of neighbours of a user with a set of influencers. This can be regarded as a sort of common or global neighborhood in the sense that everyone is, to some extent, exposed to the activity of these very visible users. Regardless of whether the goal is predicting individual or global preferences, in both cases it is possible to base the predictions on social information (who shared which content, and how they are connected to other users) or on content information (what is the content about, which topics and concepts are discussed). Both our previous works started off focusing on what can be learned purely from social interaction information, to then extend the models with information about the content by means of diverse Natural Language Processing techniques.

Beyond social interactions and content characteristics, time is the natural next dimension to explore when trying to deepen our understanding of the formation of social preferences. *Who* shares a piece of information and *what* it is about are important, but the impact of these factors is strongly determined by *when* the activity happens. In the present paper, we tackle the problems of retweet and trend prediction using only information available within a limited time window since the creation of the analyzed tweet. For example: five minutes since a tweet was created, is it feasible to effectively predict if a user will retweet it? Or, 10 minutes later, can we give an accurate estimation of whether or not it will become popular? Temporal information can also be included among the features for prediction, considering not only whether or not users retweet a tweet,

but also when they do so. Extending our research in this way not only gives us a more profound understanding of the dynamics of social influence, but also makes the results much more applicable in an online early prediction setting, by providing us with a quantification of the trade-off between elapsed time and predictability.

The rest of this paper is structured as follows: Section 2 gives an overview of related works. Section 3 describes how we build the datasets from Twitter for our experiments. Section 4 describes time constrained models for prediction of retweet preference for individual users, while Section 5 introduces time constrained model for the prediction of popularity of a tweet. Finally, in Section 6 we present our conclusions and possible lines of future research.

2 Related Work

Along with the increased popularity and impact of social media, the research interest has grown both on modeling the preferences of users and the distribution of popular content. Many initial works focused on analyzing characteristics of the content and how they can be used to predict popularity and outreach [5, 14, 10]. Among these, [8] 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.

Another approach to study preferences and influence in social media is to focus on the interactions and connections between users. In [2] we studied the prediction of retweets for a given user, using the behavior of users in her second degreee social neighborhood (followed, and followed by followed) to build a classifier that determines whether or not she will *retweet* a given tweet. These models, based purely on social information, achieved a high predictive performance, with an average F_1 score of 87.6%. In this work we also explored extending the model with content features for the cases where the purely social models were not performing well. By using a topic modeling algorithm specifically adapted to tweets (TwitterLDA), an average performance improvement of 1.7% was obtained.

In [12], we extended the previous work to the problem of predicting popularity within a community of users, instead of just individual preferences. The target here was to build classifiers that can identify if a given tweet will become popular or not, based only on the activity of a selected set of influencers (i.e.: highly central and active users) on it. This work also explored improvements incorporating embeddings-based features to represent the content and different algorithms for selecting user influencers. Same as the previous work, we first studied the performance of a purely social prediction, obtaining an F_1 score of 79.2%, using only the retweeting behavior of the top 10% most central and active users as influencers. Adding content tweet features based on FastText embeddings, the performance increased up to 86.7%.

In the present work, we set off to refine the previous studies to take the time of retweeting activity in consideration. Similarly to the approach proposed in our paper, [15] develops models to predict the popularity of a given tweet based

on temporal information about the first minutes of activity, using a Bayesian approach. These models only use observations on the retweet times and the structure of the graph of retweets between users. [13] also explores the problem of early prediction of social activity but with a different goal: distinguishing tweets that become popular in an organic fashion from those that are sustained artificially by advertising or coordinated efforts like groups of bots or trolls.

3 Dataset

In this section, we describe the dataset used in this work for all experiments. The base dataset from previous work [2] could not be reused as a whole, because no information about the precise moment of publication of tweets was persisted during its tweet collection phase. 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 in this paper, we collected our data following the same steps detailed in previous work [2]. Data consists of Twitter users and the who-follows-whom relation between them. Even though we could have reused the data from the who-follows-whom relation from our previous experiments, we decided to rebuild the network with more recent connections, since otherwise there would have been a considerable time gap (around 2 years) between newly collected tweets and older social interaction information.

Back in our previous work, the idea was to create a representative subgraph 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 subsectionquently 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, a subgraph was extracted following the procedure below. Note that many users added in the last step might have no outgoing edges.

- 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 filtering produced the final graph $\mathcal{G}$ with $5,589$ vertices and $245,694$ 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 $9,441,950$ tweets. These tweets result from extracting the content written in Spanish from user's timelines in $\mathcal{G}$ for dates between September and October 2018. Also, in the begining of the tweet collection process, we fetched every tweet from the past that the Twitter API would give us, which at that moment were 200 tweets per user on its timeline.

Let's call $\mathbf{T}$ to our set of tweets. The main difference between this dataset and the ones used in our previous works is that we have a publication timestamp for tweets and retweets. Let's denote with $\mathbf{T}^{\theta}$ the set of content available at timestamp θ, this will be used in the next section for building the retweet-time-window.

3.3 Retweet-Time-Window

As mentioned before, the main idea in this work is to reproduce our previous works introducing the idea of time windows. To do so, for each "original" tweet (that is, a tweet that is not a retweet of any previous tweet) we take time windows of different widths, always with the publication timestamp of the considered tweet as a starting point. Let θ_0 represent the exact moment in which an original tweet t was published by its author user u, we call θ_0 the window start timestamp. So, for any time ω, we define the *retweet-time-window* $\mathbf{rtw}(t, \omega)$, for tweet t and width ω as the set of all the retweets of t made by any other user, that happen before $\theta_0 + \omega$. We also need to define $\mathbf{RTW}(\omega)$ which is the union of all retweet-time-windows of width ω. Formally, $\mathbf{rtw}(t, \omega) := \texttt{retweets}(t) \cap \mathbf{T}^{\texttt{timestamp}(t)+\omega}$ and $\mathbf{RTW}(\omega) := \bigcup_{t \in \mathbf{T}_o} \mathbf{rtw}(t, \omega)$, where $\texttt{timestamp}(t)$ is the time when t was published, $\texttt{retweets}(t)$ are all the retweets of t among the users in our graph $\mathcal{G}$ and $\mathbf{T}_o$ is the set of all original tweets in $\mathbf{T}$.

User selection : Inactive users will be 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 $3,911$ active users in $\mathcal{G}$. We restrict the analysis to those users, also removing content shared only by inactive users from $\mathcal{T}$.

4 Early prediction of retweet preferences

In this section we extend the experiments from previous work [2], for the different retweet-time-windows using the new dataset described in 3. We make a summary

from the previous work about how to build the model for prediction. Later, we describe how to select the users for prediction and how to build all the necessary datasets. We finish the section with the results describing the variation of predictive performance depending on the size of the chosen time window.

4.1 Experimental setup

We aim to predict, for a given user u and a given *tweet* t, whether or not u will share t based on information about which users in the environment of u have retweeted t *so far*. Since the process of feature extraction, modeling, and parameter tuning is computationally expensive, these experiments are performed on a selected subset of users. We begin by describing the criterion with which these users were chosen. We then describe how we generate, for each user u, a neighborhood of users E_u and a set T_u of potentially interesting *tweets*. Then, we describe the feature extraction process based on T_u and the partitioning into sets of *training*, *tuning* and *evaluation*. Finally, we explain the process of training classifiers and tuning their parameters.

Users for prediction : Following the methods applied in [2], but with the new data obtained in Section 3, we take the top 1000 users with the highest Katz centrality coefficient [6] in the graph in $\mathcal{G}$ and, on the other hand, the top 1000 users with the highest retweeting activity. We restrict our analysis to users belonging to both sets, which leaves us with a set U of 211 users. For comparison, in the previous work this selection consisted of 194 users.

User's environment : We consider as environment a second degree of users which are followed by u. This is, we take all users (other than u herself) to 1 or 2 steps forward from u in the directed graph G, formally:

$E_u = \left(\bigcup_{x \in \{u\} \cup \texttt{followed}(u)} \texttt{followed}(x)\right) - \{u\}$.

Visible tweets : The Twitter API doesn't provide explicit information about whether or not a user viewed a given tweet, but we can at least take a universe of *potentially viewed* tweets. We simply define this to be the set of all the tweets written or shared by the users followed by u. We exclude from this set those tweets *written* by u herself, since our focus is in recognizing interesting external content, and not on studying the generation of content from a particular user. Formally this set is defined as:

$T_u := \left(\bigcup_{x \in \{u\} \cup \texttt{followed}(u)} \texttt{timeline}(x)\right) - \{t \in T | \texttt{author}(t) = u\}$,

where $\texttt{timeline}(x)$ is the set of tweets in T that were written or shared by x.

In addition, we need to define the subsets of tweets that are visible at time ω since their creation. Formally:

$\mathbf{RTW}_u(\omega) := \{t \in T_u | t \in \texttt{timeline}(x, \texttt{timestamp}(t) + \omega)$ for some $x \in \{u\} \cup \texttt{followed}(u)\}$, where $\texttt{timeline}(x, \theta)$ is the set of all those tweets in $\texttt{timeline}(x)$ that were written or shared by x until time θ.

Features for users' environment Now, we can build the set of features and target vectors needed for the predictive model centered in user u until the time ω. Given $E_u = \{u_1, u_2, \ldots, u_n\}$, we define for each tweet $t \in \mathbf{RTW}_u(\omega)$ the following vector of boolean features:

$v_u^{\omega}(t) := [\texttt{tw\_tl}(t, u_i, \omega)]_{i=1,\ldots,n}$,

where $\texttt{tw\_tl}(t, u, \omega) := \begin{cases} 1 & t \in \texttt{timeline}(u, \texttt{timestamp} + \omega(t)) \\ 0 & \text{otherwise} \end{cases}$

Note that the content of tweet t is not considered, we only include the information about who retweeted t until time ω since its creation.

Finally, the target vector $y_u(t)$ is a boolean vector which indicates, for each tweet $t \in \mathbf{RTW}_u(\omega)$, whether the user u has retweeted t.

Building train, tuning and test sets : As usual for any machine learning problem, a train and test set should be available. We also build a tuning set, to validate the models using unseen data, but different from the test set. To do so, we decide to randomly split T_u in 3 datasets for every user u: training (T_u^{tr}), tuning (T_u^{tu}), and evaluation (T_u^{ev}). The resulting sets contain 70%, 10% and 20% respectively, against T_u dataset. Let y_u^{tr}, y_u^{tu} and y_u^{ev} be the corresponding target for each of these datasets. Now, let $T_u^{tr}(\omega)$, $T_u^{tu}(\omega)$, $T_u^{ev}(\omega)$ be the corresponding previously defined train, tuning, and test set with the information of the retweet-time-window ω, as we defined in the previous Section . Next, we show the available retweet information in $T_u^{tr}(\omega)$for different windows ω. In the following list, we show the percentages of retweets available against the total amount of retweets for the full T_u^{tr} set without windows. These percentages are averages between our test users, which have a total retweet activity of $4,468.38$ in T_u^{tr}.

1s	2s	10s	30s	2m	5m	15m	30m	60m	90m	120m	240m	540m
0%	0%	0.1%	2.18%	8.72%	15.93%	27%	35.47%	44.89%	51%	55.66%	66.72%	78.28%

Adding content features : To analyze the content of each tweet, we decided to use the FastText [4] implementation of *word embeddings*. One of its most interesting features for our work is the possibility of assigning vectors to words not seen in the vectorization model training, looking for matches at the n-gram level of characters to vectorize those words out of vocabulary. This makes our vectorization more robust for handling misspelled words that are common on social networks. We use a 300-dimension pre-trained model, included in the FastText [1] library. Even though FastText only provides vector representations of single words, a vector representation of a document (tweet) can be easily obtained by taking the average of the vectors of its component words. In our previous work for prediction of single user retweets [2], we did not use FastText, but in Section 4.2, we report those experiments.

4.2 Results

We will now analyze the results obtained from training and evaluating user-centered classifier models using the feature vectors described in the previous section. We start with the purely social models and we finish with models that also consider the content.

[1] `https://FastText.cc/docs/en/crawl-vectors.html`

Social Prediction : For the social model centered in a user, we use SVC [2] included in the scikit-learn[9] Python package, with `GridSearchCV` for hyperparameter optimization.

Then, for each user u in our set U of test users, we evaluate the classifier obtained on the evaluation set, obtaining the results that can be seen in Table 1. Then, we considered different widths for the retweet-time-window, ($\omega \in \{2, 5, 15, 30, 60, 120, 240, 540\}$ minutes). This analysis seeks to answer the question of how much time is needed to accurately predict the retweet behavior for user u. This set of experiments shows, on the one hand, that when no window is considered the F_1 score obtained is 86.08; this result is comparable to the results obtained in the previous work [2] of 87.68. On the other hand, analyzing the results obtained for the different values of ω, we see that with only 60 minutes of data, we start off with a high performance of 69.08. From these results we can say that 60 minutes are enough to ensure an acceptable prediction.

Social Model				**Social + FastText**			
time	**Av. F_1**	**Av. Prec.**	**Av. Rec.**	**time**	**Av. F_1**	**AV. Prec.**	**Av. Rec.**
2m	53.76	43.18	80.09	2 m	55.38	47.95	78.65
5m	60.45	45.01	93.42	5 m	62.05	48.82	92.17
15m	63.76	49.32	91.45	15 m	64.47	51.43	90.36
30m	66.29	52.93	89.83	30 m	66.89	53.61	89.94
60 m	69.08	57.19	88.35	60 m	70.05	58.04	88.60
90 m	70.64	59.96	87.27	90 m.	71.36	60.57	87.82
120 m	72.07	62.43	86.45	120 m	72.61	62.83	86.91
240 m	75.2	68.24	84.98	240 m	75.52	69.02	84.98
540 m	78.65	75.96	82.61	540 m	78.92	76.28	82.61
n. w.	86.08	94.56	80.17	n. w.	86.24	93.78	80.93

Table 1. Performance of social models over the set U of selected users.

Adding Content-based Features As we see in the results shown in Table 1, adding content features barely increases the performance between 0.15 and 1.62% for bigger time windows. However, the content plays a more significant role in prediction performance for smaller windows. It's an interesting result, given that in presence of less social behaviour information, for example for 2 minutes of information, the content features improve the F_1 measure by 1.5. It is important to note that in the previous work [2] FastText had not been used.

We remark that in our current work when we combine the social model with FastText, it hardly obtains a 0.15% of improvement in the F_1 score, relative to

[2] `Support Vector Classifier`, which is the name given in `scikit-learn` to `Support Vector Machines` (SVM)

the pure social model. In our previous work, the improvement achieved by the Twitter LDA model combined with the social one did not reach 0.3%.

5 Early prediction of tendencies

In this section, we explain how to extend the experiments from previous work in [12], for different retweet-time-windows using the new dataset described in 3.

5.1 Experimental setup

This section describes how to build models capable of accurately predicting the acceptance that a tweet t could have over a general audience of users ($U_G \subset \mathcal{G}$), based only on the reaction of a set of influencers ($U_I \subset \mathcal{G}$) to the publication. Also, we show how to set up models for this purpose over a selection of users and tweets from the $(\mathcal{G}, \mathbf{T})$ dataset defined before. Similarly to the previous section, we start with predictive models based only on social features, and then we extend them to use word embeddings features.

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

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.

Trending tweets We call a tweet *trending* if we consider it popular enough to possibly become a trending topic. This term 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 take a given tweet as popular, we built and analyzed a histogram of how many retweets each tweet in $\mathbf{T}$ 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 14, allowing us to implement more accurate models. Therefore, we consider a tweet *trending* if it was retweeted at least 14 times[3] .

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 $\mathbf{T}$ that showed up on the *timeline* of at least one user in U_G, defining $T' := \left(\bigcup_{x \in U_G} timeline(x)\right)$.

[3] In the previous dataset used in the experiments in [12] this number was 13.

Influencers detection: As in the previous work, 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, Betweenness, Closeness, Eigenvector centrality and Eccentricity included in `igraph` Python package[3]. 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.

In the same line as in [12], 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*. We found that $\alpha = 0.5$, is the best choice.[4]

This analisys 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.

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. To 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 $V_t^\omega := \left[\, i_{t1}\ i_{t2}\ \ldots\ i_{tn} \,\right]$, where n is the number of influencers, and each i_{tj} is 1 if the tweet t was retweeted by the influencer j before $\texttt{timestamp}(t) + \omega$, and 0 otherwise. Formally, suppose that we wanted to make predictions with the information available at ω time after the tweet t is created. Then, if j is an influencer $\mathbf{RTW}_j(\omega)$ returns the set of m tweets in the timeline of j until time $\texttt{timestamp}(t) + \omega$. Grouping in a matrix all the vectors associated with the m tweets, the input for the model becomes for each features row t:

[4] To compare the performance of these options a subset of 500 random tweets from $\mathbf{T}$ was set aside, as a tuning set. This sample called T_{SI} is removed from $\mathbf{T}$ to avoid considering them as part of the test set, where trending prediction models will be evaluated later.

$$features_t := \left[\, i_{t\,1} \ldots i_{t\,j} \ldots i_{t\,n} \,\right]_t \text{ where } i_{tj} = \begin{cases} 1 \text{ if } t \in \mathbf{RTW}_j(\omega) \\ 0 \quad \text{otherwise} \end{cases}$$

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 $R^{\omega}(t)$ be a function that returns the number of retweets in U_G for the tweet t until the time $\texttt{timestamp}(t) + \omega$; we define the target vector as follows:

$$classification = \begin{bmatrix} r_1 \\ \ldots \\ r_m \end{bmatrix} \text{ where } r_t = \begin{cases} 1 \; R^{\omega}(t)(s) >= \textit{golden value} \\ 0 \qquad \text{otherwise} \end{cases}$$

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 **??** shows how we split the set $\mathcal{G}$ in two disjoint parts, U_I (influencers) and U_G (common users). For all the 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.[5]

To determine well-formed training and test sets for tweets, we drop from the $\mathbf{T}$ dataset the tweets posted by users in U_I named T_I. Besides, it is also necessary to cut from $\mathbf{T}$ the set T_{SI} used previously in this section to detect influencers. The remaining tweets, i.e. $T_G = \mathbf{T}' - T_I - T_{SI}$ are split again. To do so, T_G is randomly split in training T_G^{train} (75%) and test T_G^{test} (25%) datasets to evaluate prediction models. Now, let $T_G^{train}(\omega)$ and $T_G^{test}(\omega)$ be the corresponding previously defined train and test sets, with the information of the retweet-time-window ω. Next, we show the available retweet information in $T_G^{train}(\omega)$ for different windows ω. In the following list, we show the percentage of retweets available against the total amount of retweets for the T_G^{train} without windows:

1s	2s	10s	30s	2m	5m	15m	30m	60m	90	120m	240m	540m
0%	0%	0.04%	0.7%	3%	7.7%	15.1%	21.1%	29.3%	35.7%	40.4%	53.6%	68.5%

Adding Content-based Features To achieve an increase in the quality of trending tweet prediction, we apply NLP techniques to extend the purely social model with content-based features, in a similar way as in the previous section.

5.2 Results

Now, we describe how we build our predictive models and the results obtained with and without content analysis for different time-retweet-windows. We will

[5] Note that we split this dataset, to make the test fix set, when we vary the number of influencers taken to make prediction.

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. The experiments were run on two servers for 2x14 core CPU E5-2680 v4 2.40GHz servers, with 128Gb of RAM.

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 $\mathbf{T}$ 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.

time	Top-10% inf.				Top-15% inf.				Top-20% inf.			
	FT	Base .	Soc.	S+FT.	FT	Base .	Soc.	S+FT.	FT	Base .	Soc.	S+FT.
15m		25.62	46.72	53.06		39.12	52.83	57.32		42.83	58.44	63.95
30 m		32.22	61.54	64.61		39.23	62.91	66		42.15	64.54	67.93
60 m		23.38	64.61	68.98		32.06	65.67	69.47		48.66	67.87	71.21
90 m		32.38	65.17	69.78		32.51	68.68	70.32		47.01	70.77	73.12
120 m	50.71	35.72	69.78	73.18	50.71	39.62	71.66	74.73	50.71	50.08	73.66	75.53
240 m		39.2	73.7	75.99		45.21	74.4	76.32		53.96	76.43	78.45
540 m		43.46	75.25	80.66		51.38	77.51	82.27		56.23	79.93	84.07
n.w.		46.06	77.85	82.69		54.31	80.73	84.72		57.44	82.22	89.17

Fig. 1. Performance in F_1 score for all models, evaluated over U_G, using top 10%, 15% and 20%1 of users as influencers. Columns: FT=FastText ;Model ; Base = Baseline; Soc= Social Model; S+FT. =Social and FastText combined model.

Social and Combined Models Now we show the results obtained from training and evaluating trend prediction models with the features described in Section 5.1. We used Support Vector Machine models for classification, more precisely the `SVC` an in the previous section.

We decided to focus on the experiments considering 10%, 15% and 20% of $\mathcal{G}$ as influencers. We tested models for time-retweet-windows size in minutes $\omega \in \{15, 30, 60, 90, 120, 240\}$ and also without any window (**n. w.**), that is, with all collected retweets in the dataset. In table 1, we report the results for all tested models using the F_1 score. The models reported are FastText **FT**), the baseline

described in the previous section (**Base**), social (**Soc.**), and finally, the social combined with FastText (**S+FT.**). In the table we can observe that the results obtained in the previous work [12], with another dataset for the social model with 10% of influencers ($F_1 = 79.2$), is comparable with the results obtained in this work (77.85). In all cases, the social model performs better than the baseline. The results also show that the minimum window to obtain reasonable predictions with the pure social model is around 120 minutes, taking 10% of users as influencers and this can be reduced 60 minutes with 20% of influencers.

Now, we combine the social model with the 300-dimensional vector from FastText model described in the previous Section 4.2. In Table 1, column **S+FT.** , we can observe that the best results are obtained by combining a 20% of influencers to train the social model with FastText. This combination achieves an F_1 score of 89.17. The results obtained for the combined model shows that, an ω between 30 and 60 minutes, achieves an F_1 score over 70%.

6 Conclusions and Future Work

As a general conclusion, we confirm that the information about social connections between Twitter users and their first hours of activity on a tweet, can be essential to determine if it is preferred by a particular user or becomes popular among the general audience. In both cases, we obtained a good performance without analyzing the content.

Particularly, for predicting a user's preference with 60 minutes of information we obtained a performance around 69.08%, and after adding content information performance grows slightly to 70.05%. We observe that, considering more time of environment behavior, the social model obtained high performance levels that are only barely improved when combining it with FastText content features.

For the popularity prediction problem, considering the top 20% most central and active users as influencers, we need 90 minutes of their behaviour information to obtain F_1 scores greater than 70%. If we extend the models with content features, these performance levels are achieved earlier, using between 30 and 60 minutes of information.

In all cases, the results seem 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, which indicates that content still has a level of importance when it is considered within a certain social context, specially early in the lifetime of tweets, when social information about them is still limited. FastText seems particularly well suited for dealing with content from Twitter, mostly 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. One of the next experiments will be to add features that

take into account the time elapsed after the original tweet has been published. Another line of research is to try to change to Deep Learning models and replace the SVM models. Additionally, we have the idea to use a Long Short-Term Memory (LSTM) neural network, representing the activity on tweets as a temporal series. For influencers detection, alternatives such as [7] and [1] could be applied to improve the selection of relevant users.

We also propose to research 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) [11].

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 apply 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. 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)
3. Csardi, G., Nepusz, T.: The igraph software package for complex network research. InterJournal **Comp. Systems**, 1695 (2006), `http://igraph.org/python/`
4. 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/`
5. Hochreiter, R., Waldhauser, C.: A genetic algorithm to optimize a tweet for retweetability. Mendel pp. 13–18 (2013)
6. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **VOL. 18, NO. 1**, 39– 43 (1953)
7. 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)
8. 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)
9. 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/`
10. 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)

11. 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)
12. Silva, M., Domínguez, M., Celayes, P.: "analyzing the retweeting behavior of influencers to predict popular tweetswith and without considering their content". In: Communications in Computer and Information Science, Springer, 5th International Conference on Information Management and Big Data (SimBig 2018). Springer, ISBN 978-3-030-02840-4 (2018)
13. Varol, O., Ferrara, E., Menczer, F., Flammini, A.: Early detection of promoted campaigns on social media. EPJ Data Science **6** (03 2017). https://doi.org/10.1140/epjds/s13688-017-0111-y
14. 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
15. Zaman, T., Fox, E.B., Bradlow, E.T.: A bayesian approach for predicting the popularity of tweets. CoRR **abs/1304.6777** (2013), `http://arxiv.org/abs/1304.6777`