FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Modeling User Influence and Expertise for News Sources in Social Media

José Miguel Martins

Master in Informatics and Computing Engineering

Supervisor: Eduarda Mendes Rodrigues (FEUP)

Second Supervisor: Luís Sarmento (Sapo Lab)

June, 2011

Modeling User Influence and Expertise for News Sources in Social Media

José Miguel Martins

Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: Jaime dos Santos Cardoso (PhD)

External Examiner: Sérgio Aleixo Matos (PhD)

Supervisor: Eduarda Mendes Rodrigues (PhD)

11th July, 2011

Abstract

Influence plays an important role in a physical world community. There are people, the influential, that are considered authorities in certain topics by the rest of the community. They have the power to affect the behavior and thoughts of others. It was already proved that, influence is likewise exerted in online communities. There are many advantages in creating ways to measure influence propagation in social networks. Some examples are viral marketing, expertise finding and computational journalism.

The present dissertation presents a pioneer study in what concerns the Portuguese Twitter community analysis and a new system to build real-time datasets from Twitter. During the dataset creation phase we built a system to crawl data from Portuguese Twitter users. This system, TwitterEcho, allows a complete and continuous crawling process, robust enough to support the whole Portuguese community on Twitter.

Throughout this dissertation we describe the system implementation details, with more incidence on the client part, since it is a distributed system. Additionally, we present the results of studies done with data retrieved by the system.

In the second part, we present the influence model proposed in this dissertation. The model was developed based on models described in the literature or used in commercial applications. We detail the processes of gathering the metrics used to influence measurement. This involves the application of data-analysis techniques to the content produced by Twitter users.

Finally, we show the results obtained by the model after its application on a representative dataset created from TwitterEcho. we also present a preliminary validation of the model, and propose further ways to evaluate the model.

Resumo

A influência desempenha um papel importante numa comunidade do mundo físico. Há pessoas, chamadas influentes, que são consideradas peritos ou autoridades em determinados assuntos pela restante comunidade. Elas têm o poder de afectar o comportamento e pensamentos dos demais. Foi já provado que, a influência também é exercida em comunidades online. Por isso, são muitas as vantagens em medir a propagação da influência numa rede social. Alguns exemplos são o marketing viral, a procura especializada e o jornalismo computacional.

A presente dissertação representa um estudo pioneiro no que concerne a análise da comunidade Portuguesa no Twitter e apresenta um novo sistema para criar datasets a partir do Twitter. Durante a fase de criação do dataset, foi construído um sistema para recolher informação dos utilizadores Portugueses do Twitter. Este sistema, TwitterEcho, permite um processo completo e contínuo, suficientemente robusto para recolher informação de toda a comunidade Portuguesa.

Ao longo desta dissertação são descritos detalhes da implementação do sistema, com maior incidência sobre o cliente, já que se trata de um sistema distribuído. Para além disso são apresentados resultados de estudos feitos a partir da informação recolhida por este sistema.

Numa segunda parte apresentamos o modelo de influência proposto nesta dissertação. O modelo foi desenvolvido com base noutos, descritos na literatura ou usados em aplicações comerciais. Serão detalhados os processos para a recolha das métricas necessárias para a quantificação da influência. Isto envolve a aplicação de técnicas de análise de dados ao conteúdo produzido pelos utilizadores do Twitter.

Por fim serão também mostrados os resultados obtidos pelo modelo após a sua aplicação sobre um dataset criado a partir do TwitterEcho. Esses resultados são alvo de uma validação preliminar e são propostos métodos para uma validação e avaliação mais completa.

Acknowledgements

Although a dissertation assumes an individual and solitary work, the present dissertation would not be accomplished without the help of a lot of people.
In first place I would like to thank my supervisors, Prof. Eduarda Mendes Rodrigues and Prof. Luis Sarmento. They were always present, with good ideas to overcome the problems my project went through. I would also like to thank Gustavo Laboreiro and Jorge Teixeira from SAPO Lab for the help provided in several technical issues. I am grateful to Eduardo Oliveira due to his fantastic work on TwitterEcho that was determinant to the success of this thesis.
Finally, and not less important, I am very grateful to my family and friends because always supported me and kept me focused on my task.

"Obrigado a todos!"

José Miguel Martins

“Education makes machines which act like men and produces men who act like machines.”

Erich Fromm

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

API	Application Programming Interface
CGI	Common Gateway Interface
CPAN	Comprehensive Perl Archive Network
CSS	Cascading Style Sheets
DB	Database
GMT	Greenwich Mean Time
GUI	Graphical User Interface
HITS	Hyperlink-Induced Topic Search
HTTP	Hypertext Transfer Protocol
ID	identifier
JSON	JavaScript Object Notation
LDA	Latent Dirichlet Allocation
PHP	PHP: Hypertext Preprocessor
REST	Representational State Transfer
RSS	Really Simple Syndication
SAPO	Serviço de Apontadores Portugueses Online
SMS	Short Message Service
UTC	Coordinated Universal Time
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 Context and Motivation

Since the *boom* of online social networks, many entities have been studying how influence propagates through these networks and determining who are influential users. The present thesis was motivated by the Reaction Project [REA11] and carried out in SAPO Lab at Faculdade de Engenharia da Universidade do Porto. The Reaction Project operates in the area of computational journalism which evolves all the activities of journalism aided by software, i.e. detection, production and diffusion of news.

1.1.1 Influence

Influence has been object of study of a vast set of fields. Sociology, communication, marketing and political sciences are some examples. The notion of influence plays an important role when it comes to understanding how society and corporations operate and behave. For instance, the way information propagates, consumer trends and how people vote [CHBG10]. However, the concept of influence is not consensual and several authors define it differently. Azman et al. [AMW10], define influence as the ability of an individual to produce effects in the decision making process of another. This definition only considers the existence of influence when a process of decision making occurs. Rashotte [Ras06], provides other definition of social influence as the change of behavior, thoughts or feelings of someone resulting of an interaction with another person or group. People adjust their actions and their beliefs by the behavior or opinion of the majority or of people they consider expert in some particular subject. There is another concept reported in the literature and that is often linked to influence: homophily [WLJH10]. This concept reflects a trend, whose existence was verified in [LTH+10], in which people create stronger links with people who share the same interests or the same social profile. So, influence is exercised more intensely between people considered similar in terms of education, beliefs, values and interests. The presence of influence in the physical world is clearly observable. According to [ALTY08], people before buying a product or requesting a service, take the advice from family, friends, or a expert on the subject, rather than be influenced by advertising or marketing campaigns.

1.1.2 Social Media

Media can be defined as an instrument of communication. By that definition, social media is a social instrument of communication. With the growing importance of the Web 2.0, websites that provide information and allow participants to interact and rate the content are more and more used [KLPM10].
The activity of social networking represents the grouping of individuals into specific groups, like small rural communities or neighborhoods. Social networking is an activity that takes a different dimension if done online. So, social networking websites function like an on-line community of Internet users. Depending on the social network website, users may or not share the same interests or the same benefits of being subscribed.
Users, most of the times called "nodes" in social network context, are linked to other users by relationships, the "edges". Those relationships may vary from website to website, but most of them present two main kinds of relationships: implicit or explicit. Explicit links are those which are created when, for instance, users declare themselves friends of other users in Facebook, or when a user decides to follow someone in Twitter. Implicit relationships are created when one user interacts with another through any kind of action. "Like" action in Facebook or "reply" in Twitter, are examples of implicit links on social networks.
Beyond this classification, explicit links can be also divided in directed and indirected links. In Facebook, for example a friendship relationship is an indirect link because the relationship is reciprocal. Differently, in Twitter's case users can "follow" other users and not receive the following relationship back. Because of this, Twitter establishes directed explicit links between their users.
The growth of the social media is evident and it becomes more and more important to characterize the dynamics behind the behavior of an on-line community.

1.1.3 Applications of influence detection methods

There are few doubts that social influence is becoming a complex and subtle force that governs the dynamics of social networks. Studying the patterns of influence can lead to a better understanding of why certain trends and innovations are adopted earlier and easier than others. However, the study of influence patterns is an hard task, mainly because that study lacks readily quantifiable metrics and it is very difficult to reproduce components as the choices of individuals or their behavior in a lab. Most of the studies done in this area provide ways to detect the existence of influence but not ways to measure it. For that reason, there is a clear need of methods and techniques to analyze and measure the influence. However, the task of transposing the reality to an abstract concept is not trivial. Due to this, there is not an unique or a standard model to define the influence. And many models that has been used do not represent correctly the dynamics of influence in a community. Influence models, when applied to a social network, can benefit several areas such as: viral marketing, personalized recommendations, information search, Question and Answer systems, computational journalism, among many others. Below, we provide additional information about the application of influence models on these areas.

Viral Marketing Recently there has been a growing interest in measuring influence of users in a social network for marketing purposes. Many advantages has been found in this area [CWY09], because a social network can be seen as a representation of a real community. Finding the top influential users it is possible, trough propagation maximization algorithms [KKT03], to advertise a product to most of the network, only by convincing the most influential. However, all of the studies made in the area assume that they will receive as input a labeled graph with the edges with influence scores. The task of labeling the edges with the scores has been largely ignored [GBL10].

Personalized Recommendations As was verified in [WLJH10] the concept of homophyly is present in the context of social networks. This phenomenon indicates the formation communities which shares the same socio-demographic, behavioral and interpersonal characteristics. This indicates the existence of stronger links and more reciprocity between users with affinity for similar topics. By detecting the most influential users of one topic, it is possible to recommend personalized content to people who are influenced by them.

Information Search In a conventional search through a traditional search engine like Google or Yahoo!, the results are shown ordered by the authority of the web pages. Commonly, algorithms like PageRank [ALTY08] are used to this job. These algorithms sort the results by the number of *inlinks* (links to that web page), among other metrics. That way is not possibly to guarantee the credibility of the author. If the search was made by the authority of the users rather than the web page, it would be possibly to, at least, say that the information was posted by an user that the community consider an expert in that subject.

Question and Answer Systems The Community Question Answer portals (ex: Yahoo! Answers) emerged to serve the growing need to obtain specific information about some subject in a quick way. That way the user is spared the effort to seek for the information through a lot of pages. However, they have been losing popularity due to the lack of quality and expert validation of the answers. With a way to measure the influence of users who posted an answer would be possible to assign a score of "credibility" or "expertise" to their answers and with this, revitalize and boost this kind of system [BAL+09].

Computational Journalism Computational journalism is an area that has gained prominence in the last years. It consists in the application of technology and software in all the activities journalism is involved [DFS10]. This area would benefit a lot with the existence of influential detection methods, both at an early stage of news detection as a late stage of spreading the news through the network. It is in this context that appears the present dissertation.

Beyond the presented areas, with the appearance and fast massification of social networks, the use of dataset created from them, allows the empirical validation of the theories of influence propagation in the physical world [CHBG10].

1.2 Project

The work developed at SAPO Lab involved the production of some deliverables that are briefly described in this section. During the first part of the dissertation we focused on the creation of the dataset. This dataset contains the necessary information that our influence model needs. So, in this phase we created a Perl module. The goal of this module is, once installed on a machine, to constantly retrieve information about users, *tweets* and network links information from Twitter service. This module is the client part of a system developed at SAPO Lab in cooperation with Eduardo Oliveira. The whole system consists in a distributed application, in which the central server manages a priority list with users, and sends some to the distributed clients. Once the client crawlers retrieve information, the server stores it in a database. Besides that, the server also has the objective to increase the user list, adding only the ones that are Portuguese. This is done through the information the user has on his public Twitter profile or, in the doubtful cases, through a syntactic analysis applied on the content he produced. For further information about the server implementation we refer you to the thesis of Eduardo Oliveira [Oli11]
The second part of the dissertation focused on the analysis of the Portuguese *twitosphere* with more incidence on the influence propagation dynamics.
The process to calculate the influence scores of one user is the following: retrieve relevant data from database; metric build through *tweet* content analysis; influence calculation

from the the value of the metrics and score normalization. A very simplistic interface prototype was also built. It allows the empirical validation of the results and accessing some relevant information about the influence score calculation.

1.3 Goals

The main goal of the present dissertation is to propose and formulate an influence model that can be applied to the Portuguese reality. After the formulation of the model, it will have to be implemented. On the implemented models, experiences will be made with a representative dataset to validate them. In this phase, a GUI prototype will be built so that the results can be seen in a more friendly way.
In short, the goals of this thesis are:

- to formulate and implement a model to detect influential users.
- develop part, more precisely the client application of a system (TwitterEcho) to crawl Twitter information about Portuguese users.
- perform influence model validation experiences with data collected from TwitterEcho and propose other ways to evaluate the model.
- develop a GUI prototype that allows the visualization of the results obtained by the influence detection model.

1.4 Thesis Structure

The present thesis has this and 5 more chapters.
In chapter 2 we provide background information about Twitter. In chapter 3 we refer some of the most recent studies made in influence detection area. In chapter 4 we explain and show results of the method used to build a representative dataset of Portuguese Twitter users. In chapter 5 we propose a model, to measure the the influence propagation in Twitter. Finally, in chapter 6, we draw conclusions about the work done during the dissertation.

Chapter 2

Twitter

2.1 Introduction

Twitter [Twi11c] is a *microblogging* social media service. It allows the registered users to exchange messages up to 140 characters, called *tweets*. This tool has the particularity of directed links between the users in following and friend relationships. It introduces the concept of "following", which means that if user A follows user B, then A will receive in real-time the updates that B makes to her "status", i.e. tweets posted by B.

Twitter appeared in mid 2006 by the hand of Jack Dorsey. The initial concept and keyword of this social media service was **mobility**. Users would update their status whenever and wherever they wanted. For that they would use:

- Twitter webpage or its API
- SMS by phone
- Instant Messaging
- Email
- Third-party Twitter client applications

Initially it appeared as a social media service for mobile devices, hence the emphasis given to the use of SMS and the 140 characters limit. Users update their "status" writing "what they are doing at the moment". However, it became a little absurd to let other people know what they were doing every minute. So, Twitter changed the slogan to "what's happening?". With this change, Twitter became a very popular vehicle to share news and real-time covering of events. In many recent cases, Twitter was the most important medium between the event and the public. It was massively used during the Iran elections [BE09], to report over the violence cases. More recently, it was used to transmit the news

of riots in Egypt.
In September 2010, Twitter, released official information stating that there were 175 million users registered publishing about 65 million *tweets* per day.
Despite the amount of registered users, was stated in [RGAH09] that only 10% of the users produce 90% of the Twitter content. This phenomenon was called "passivity", and its concept will be described with more detail in section 3.1.3.2.

Beyond the characteristics that make Twitter a powerful mean to spread news, it has become a study object of corporations willing to advertise their products and services. The idea behind viral marketing is that if the most influential users of a network are convinced, then a chain reaction would be triggered and all users will be convinced [GBL10]. This can be achieved with a relatively low investment. The problem is: given a representation of the network structure with the links between the nodes already labeled with an influence score, we want the minimum set of users that would maximize the influence propagation towards the network.
However, the opposite can happen too. Which means that consumers can use Twitter to denounce bad quality of the service or product of a corporation. This happened in Portugal in the "Ensitel" case [Mac10]. Ensitel is a company that sells mobile phones and has stores spread throughout the Portuguese territory. A Ensitel costumer, one day felt that she wasn't well treated at the store and she decided to post that situation in her private *blog*. Ensitel moved a lawsuit against the costumer in order to erase all the content she published about them. This triggered an unprecedented wave of discontent and rapidly the Portuguese social media was flooded by protest voices against Ensitel. With its image tarnished, Ensitel had no other choice than give up the judicial process and publicly apologize.

The focus of this thesis is in the Portuguese *twittosphere* analysis. Below we present the possible actions in Twitter that would be used as metrics to characterize influence.

2.2 Message Format

As was explained before, Twitter allows users to update their "status" whenever they want. The "status" can be seen as a message that users send to their followers, which is called *tweet* in Twitter jargon. The Twitter-API [Twi11b] allows developers to retrieve stored data like profile information of users, their *tweets* or their followers list. If we use the Twitter-API to request the *tweets* of one user, we will receive an array with the following information [YW10] .

user_id contains the unique identifier of the user who posted the message.

id identifier of the message. Two different messages posted by two different users, may have the same identifier.

created_at date when the message was posted.

text content of the message. This field is limited to 140 characters. With this, it is possible to retrieve information like mentions, *retweets* or replies.

source contains the Twitter client that was used to create the message. Beyond the Twitter website (twitter.com), clients like TweetDeck, Tweetie, UberTwitter or TwitterFon can be used.

truncated messages that exceed the 140 characters are truncated. This field is boolean type and indicates if the message was truncated or not.

in_reply_to_status_id if the *tweet* is a reply of another message, this field contains the identifier of the message it is replying to. Otherwise this field is null.

in_reply_to_user_id same as the latter. If the current *tweet* is a reply this field contains the user identifier to whom the message replies. If it is an ordinary message this field is null.

favorited identifies if the message was included in the favorite list of someone.

2.3 Possible Actions

Influence can be measured as the action of affecting someone else's actions. In this section the diverse actions users can do, will be explained. Some of the actions are already built into Twitter, but some are continually being adopted by the community. Recently Twitter released an upgrade of its website that allows shortcuts for some of the actions here presented.

2.3.1 Tweet

The basic user action in Twitter is updating the profile status. The new status will be sent to all of user's followers. So, this can be seen as a spreading of information through your network. Every time you update your profile status, you are sending a *tweet* that will be put in the followers' *timeline*.

2.3.2 Follow

If a user wants to receive content from other people, she needs to use the follow action. This action can be performed by simply clicking the "Follow" button of someone's profile. More recently, Twitter included in their website a recommendation system in which some popular users of the most diverse themes are recommended to you. This action is required to receive others content in your timeline. However it is allowed to see the last *tweets* of a user if he has not set his account private. In this case you will need to send a following request to the user.

Analogously, at any time a user may decide to stop receiving one's content. He needs to perform the "unfollow" action, which, as the latter, can be achieve by simply clicking the "unfollow" button in the user profile.

The "follow" concept existent in Twitter represents a directed link [LTH+10] which is a characteristic of this social media service. It means that a graph representing the network would have to be a directed graph. On the other hand we have social networks like Facebook whose graph representation would be an undirected graph.

From this we can conclude that the "follow" action establishes 2 relationships. A following relationship directed from the user that follows to the user that is followed. And a friendship relationship directed from the user that is followed to the user that follows. A reciprocal following action would result in 4 edges between those two nodes.

2.3.3 Reply

Reply action can be defined as an answer to someone else's *tweet*. A reply continues to be considered a *tweet*, but the information stored is different. In other words, as we see in 2.2 the fields: in_reply_to_user_id and in_reply_to_status_id will be filled with the information about the user who is replying and the *tweet* that is being replied.

To perform a reply the user writes a message with the following format:
@username (content).

This action has born by spontaneous and massive use of its users. Twitter realized that, and included the "reply" button on its interface. A reply can be seen in the *blog* paradigm as a public comment to a post.

2.3.4 Direct Message

Direct messages differs from the replies in permission aspects. While a reply can be seen by anyone, and it is like a normal *tweet*, the direct message is private and only the transmitter and the receiver have access to its content. Also a reply can be perceived as a comment to a *tweet*, and the direct message works like a private chat environment.
To prevent spam purposes, Twitter decided to prevent users that do not follow you, to

direct message you.
As the reply, the direct message feature was added to Twitter's website to allow a more intuitive way to direct message someone.

Nevertheless it is possible to perform this action simply by sending a message with the following format:
d @username (content)

2.3.5 Retweet

The act of *retweeting* someone's message has become very popular in Twitter. It is a mean to maximize propagation of information. It also maintains the original source of the content, since the content may be propagated through many depth degrees. It means that a *tweet* can be *retweeted* many times and by many users.
Suppose that you encounter a *tweet* that you consider relevant to be seen by your followers. You can use the "retweet" button existent in Twitter's website, or simply by typing a message with the format:
RT @username (original content)

This feature provided a preponderant metric to analyze the propagation dynamics of the network. It is possible to retrieve metrics relative to how is the information flow inside the network.
When a user *retweets* other's content, that signifies that the content is considered relevant or of interest by users. It represents influence being exerted, since the simple *tweet* action by the user triggered a post-action by the user that *retweeted*.

When compared to *blog* paradigm, *retweets* may be seen as a citation of other author. However the whole content of the "citation" comes with it.

2.3.6 Attribution

Attribution action is not currently officially recognized by Twitter. However it has been widely used to attribute the information credit to an extern entity. For instance, when you want to cite a traditional media like a newspaper that does not have Twitter account you use the attribution.

In short, when the source of the information of you are posting has been published somewhere else, attributions are used. To do that simply send the message with the following format:

(content) via @username (content)

2.3.7 Mention

Another popular action in Twitter is mention. It consists in including in your message the username of the user you want to mention. The difference between the reply and the mention action is only related to the location of the usernames. In a reply the usernames comes in the beginning of the content of the message. The message format was described before in 2.3.4. Differently in a mention the username doesn't come at the beginning of the message. Several users can be mentioned in one *tweet*. The standard format of a mention is:
(content)+ @username (content)

Which means that, for instance the tweet "@user1 @user2 @user3", does not represent a mention of 3 users but a reply to user1 mentioning user2 and user3.

2.3.8 Hashtag

Twitter's search service uses the tag concept to present the most relevant tweets tagged with a certain word. This concept is widely used in Twitter and the users use it to assign its content to a topic.
Supposing one is writing a message about the elections in his country. He may identify his message with, for instance, the *hashtag* #elections. If the posted message was considered relevant by the community will most likely be in the top of the tweets about elections.

A message with *hashtags* does not follow a standard format, since you can use *hashtags* wherever you want in your message text. So the regular expression for a message with hashtags will be:
(content)* (#hashtag)+ ((content)* (#hashtag)*)*

Twitter also have a trend analysis service that continually informs you about the most discussed topics worldwide or in your region.

2.3.9 Follow Friday

Following the *hashtag* concept, has been recurrent the use of pre-established labels to recommend users. The phenomenon appeared under the name of *Follow Friday* and pre-

conized that every Friday users who want, should recommend his most relevant friends (people he is following). This was achieved by adding a specific *hashtag* followed by the usernames of the users one wants to recommend. The movement gained supporters and has been widely used.
However we noticed some inconsistence in the choice of the *hashtag* to represent the recommendation. The more used are: #followfriday, #ff, #FollowFriday depending on the community it is used.

Despite the fact of not being recognized by Twitter as an official action, this action can be considered an important metric to characterize the influence or the popularity dynamics of this social media.

Twitter

Chapter 3

State of The Art

3.1 Influence Measurement

Discovering a relation between the concept of influence in the physical world and the influence in a social network has been a constant challenge of prior studies in this field. This chapter presents a literature review about influence detection methods on Twitter and on the *blogosphere*.Throughout it, some intuitive properties indicative of influence will be identified. For each one of these indicators, metrics will be mapped to them and will allow the measurement of those indicators. Metrics are based on the possible Twitter users actions. Beyond that, some approaches and models of influence measurement will be presented.

3.1.1 Metrics

Agarwal et al. [ALTY08] identified a set of properties considered good indicators of influence. Although, the properties identified are related to the *blogosphere*, they can be adapted to Twitter paradigm.
Those properties can be estimated and quantified from a set of statistics that can be "easily" collected. In order for someone to be considered influential in a virtual world, they take as assumption that he/she must be recognized by others, generate a lot of activity around him/her, has innovative ideas or consistent points of view and is eloquent. Taken into account these properties, the possible actions in Twitter were identified and mapped to the properties.

Recognition this property indicates if a user and the content she spreads are considered relevant by other users. In the *blog* paradigm this property can be quantified from the number of *inlinks* that the posts have. One *inlink* represents a link to a post, existent in the post of another user [ALTY08]. In Twitter context, the *inlink* concept cannot be applied. So, it is necessary to find other metrics that allow the measurement of recognition. According to the possible actions in Twitter, *retweets*, mentions, attributions and recommendations, can be used as metrics indicators of recognition. Furthermore, the recognition score of a user in Twitter can be described by the sum of the *retweets*, mentions, attributions and recommendations that the user received.

Activity Generation indicates the capacity of users to generate activity around the content they publish. According to [ALTY08], the metric that has been used to quantify this indicator is the quantity of comments that one content receives. Transposing this notion to the Twitter context, the capacity of activity generation of a user can be translated by the number of replies that the tweets of the user received. This number must be normalized by the number of tweets. In [CHBG10] the author considers that a spam filter must be applied to the replies. Another method to measure activity generation is by the comment rate. In the Twitter context, that can be given by the number of replies per hour.

Novelty innovative ideas or breaking news that represent novelty exert a stronger influence in a community. On the *blogosphere* this indicator is inversely correlated with the number of *outlinks* that one post has in its content. The *outlinks* are the links to another post of another blog. This metric can be easily mapped to the Twitter context as being the *retweets* and attributions made by the user. The higher this number, more indicative is that the information spread was not originally created by the user. In the *blogosphere* analysis an issue is put, that has to be with the length of the post [ALTY08]. The higher the post length, higher is the probability of containing *outlinks*. This issue has no expression when it comes to the Twitter paradigm.

Eloquence the eloquence represent a person ease of communication and the ability to "persuade" his interlocutors. This property is very difficult to quantify simply by using statistics. In the *blogosphere* an heuristic based on the size of the posts has been used [ALTY08]. However, in the case of Twitter that cannot be applied, since the maximum size of a message is 140 characters.

3.1.2 Classic academic solution

When appeared the necessity to measure influence propagated through a social network, were developed many and diverse methods of doing that.
This has been object of study since the appearance of *blogs*. However, there is no definitive method to do the task of influence measurement. The academic solution has been the simple sum of the external citation that a post received. Being the *blog* influence score the sum of all citation to all the posts normalized by the number of posts. However, this corollary assumes that there is a positive relation between the number of citations and the influence score [Gil04]. In [Raz09] the author verified that a citation content may be

negative, positive or neutral. That means that a user can cite another criticizing him or praising him or any of the previous. To assign this "feeling" metric, a semantic analysis of the content of the message would be necessary, which is outside the scope of this dissertation.
The method of counting the sum of citation is too simplistic to be considered a complete metric to measure influence. One citation can be more valuable than another. This way, the academic solution evolved to a method in which the relative value of a citation is calculated. For instance, let us assume that user A cited user B. That action will be taken into account as one citation multiplied by the current influence score of user A. So, the weight of one's citation depends on the position the user who cited has in the rank. One example of the application of this algorithm is Technorati [Tec11]. This tool constantly verifies almost 2 million *blogs* and outputs sorted lists ordered by the global authority. To compute the authority scores, Technorati counts all the *inlinks* in the *blogosphere*. On the other hand we have the *twittosphere*. In this context the academic solution that has been used to assign influence scores to the users has been the follower count [WLJH10]. There are many examples of web applications that use the previous heuristic such as Twinfluence (http://twinfluence.com/) and Twitaholic (http://twitaholic.com/). This kind of approach has the assumption of the bigger the audience of a user, higher is his influence. However, some recent studies suggest that this assumption is wrong and, for that reason, the size of a user audience cannot be used **alone** to quantify his influence. In [CHBG10], Cha et al. have done a representative study with a large amount of data: 54 million users, 2 billion follow links, 1.7 billion *tweets*. The have calculated three kinds of influence score. One created from the number of followers, other from the number of replies and the last with the number of *retweets*. A ranked list was created, based on those three values. Using the *Spearman's rank correlation coefficient*, they verified that for the top 10% influence users, the number of followers has a weak correlation with the remaining metrics. With that they concluded that the follower count is not representative of influence but of popularity. This mix of concepts (influence vs popularity) has been the main focus of failures and lack of precision of the actual tools of influence detection [ALTY08].

3.1.3 Influence Models

A model, by definition, is an abstract concept of the reality. A model will be always related to an imprecision, because it is very difficult to foresee all the variables existent in the real world.
In this chapter the most recent and important influence characterization models and algorithms are presented. According to [CLH^{+}10], there are two kinds of methods to model influence. Based in: snapshots or temporal dynamics. The first represents the influence characterization through a static log with the actions that occurred in the network within

a certain amount of time. Models created from this kind of method are called static models. These models have higher imprecision associated, because they do not analyze the propagation of influence dynamically. Nevertheless, they are computationally lighter and provide an easy and fast test environment.
The latter allows to build continuous models. These models are more precise and the values of its parameters are continually updated. The disadvantage of these models is in the testing tasks. It could became very problematic to run tests with huge networks and a large amount of data.

Models can differ also, in the kind of influence they measure. They can model the global influence of a user, or the influence of the user on a given topic. The global influence is the one that is exerted upon the network in a general way. The topical influence advocates that one user can be very influential in one topic, but can also be a nullity in terms of influence on a different topic. The work done by Cha et al. [CHBG10] analyzed the dynamics of the influence score of an user over time and over topics. In this they concluded that the influence score of a user is transversal in the more diverse topics, despite the big "competition" that exists in Twitter. "Only costs 140 characters to post a message".
The influence is not spontaneously nor accidentally gained. Still one user in order to maintain the influence gained must be constantly in interaction with his audience.

3.1.3.1 PageRank and HITS

PageRank and HITS are two popular and effective algorithms to rank websites by authority, and they are based on the link structure. The intuition behind PageRank is that the importance of an website is proportional to the combined importance of its neighbor websites [GBL10].
The problem in classifying the authority of *blogs* is different from the classification of webpages. The *blogs* are sparsely connected in the *blogosphere*, for that reason it is not recommended to use website ranking algorithms to find influential *bloggers*. The temporal aspect is very important on the *blogs*' domain. While a website gains influence over time, a *blog* post looses importance over time. This will result in a denser adjacency matrix in the website network than in the *blogosphere* [ALTY08]. In [KLPM10], Kwak et al., sorted the users by the number of followers and by the PageRank classification and encountered two similar lists. With this result, they concluded that PageRank, as HITS which uses a similar heuristic, are not originally prepared to model influence of an online community.

3.1.3.2 Influence-Passivity

This section presets the characteristics and the operation mode of the Influence-Passivity algorithm. This algorithm was proposed in 2010 and was developed by Romero et al. in cooperation with HP labs [RGAH09].

Assumptions Influence-passivity algorithm assumes that there is passivity in the information propagation in a social network. To prove that, they presented a mean value of 1 *retweet* in 318 posts for a regular user of Twitter. This algorithm also assumes that the influence score of a user depends on the "quality" of the people he influences. This assumption can be divided in 4 rules:

1. The influence score of one user depends on the number of users he influences and on their passivity score.
2. The influence score of one user depends on how much attention the user can get from the users he influences.
3. The passivity score of one user depends on the influence score of who he tries to influence and do not achieve that.
4. The passivity score of one user depends on how much influence he rejects compared to the amount of influence rejected by other users.

Operation Mode The Influence-Passivity algorithm receives as input a graph $G = (N, E, W)$. N represents the users. E on represents the links between users. Finally W represents the weight values of each link.

The value of w_{ij} in each e_{ij} represents the rate of influence that i exerts upon j, normalized by the total influence that i tries to exert upon j.

The algorithm iteratively and simultaneously computes the passivity and influence score. The obtained result is the calculation of the values of I_i and P_i that represent, respectively, the relative influence score and the relative passivity score compared the the rest of the network.

Acceptance Rate: represents the amount of influence j accepted from i, normalized by the total influence accepted by j from the whole set of users in the network. This parameter can be calculated by:

$$u_{ij} = \frac{w_{ij}}{\sum_{k:(k,j)\in E} w_{kj}}$$

This value can be viewed as the quantity of loyalty and dedication that j has towards i.

Rejection Rate: represents the amount of influence that i rejects from j, normalized by the total influence that the whole network users rejected grom j. Can be determined by:

$$v_{ij} = \frac{1 - w_{ij}}{\sum_{k:(k,j)\in E}(1 - w_{kj})}$$

The acceptance rate and rejection rate values are used to compute the following values:

$$I_i \leftarrow \sum_{j:(i,j)\in E} u_{ij} P_j \tag{3.1}$$

$$P_i \leftarrow \sum_{j:(j,i)\in E} v_{ij} I_j \tag{3.2}$$

The above equations represent the assumptions enumerated in the last section. Having these values is easier to present the structure of the whole algorithm. The main steps of the algorithm are detailed below.

Pseudo-code

```
$I_0 \leftarrow (1,1,1,...,1) \in R^{|N|}$
$P_0 \leftarrow (1,1,1,...,1) \in R^{|N|}$
for $i = 1$ to $m$ do
Updates $P_i$ through operation 3.2 e o valor $I_i - 1$
Updates $I_i$ through operation 3.1 e o valor $P_i$
for $j = 1$ to |N| do
$I_j = \frac{I_j}{\sum_{k\in N} I_k}$

$P_j = \frac{P_j}{\sum_{k\in N} P_k}$
end
end
return $(I_m, P_m)$
```

In what concerns the metrics used as influence indicators, in Influence-Passivity algorithm the numer of *retweets* and mentions were used. According to the author, this algorithm can achieve very satisfactory results. This can also be applied to topical influence modelling, if the algorithm operates over a sub-graph representing the universe of users that *tweeted* about one topic.

3.1.3.3 Twitter-Rank

TwitterRank is an algorithm proposed by Weng et al. in [WLJH10]. The intuition behind it is, given a certain topic, the social influence of one user is the sum of the social influence of his followers. The figure 3.1 presents in a general form the used methodology.

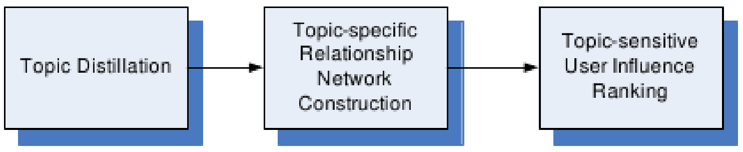
Topic Distillation
Topic-specific Relationship Network Construction
Topic-sensitive User Influence Ranking

Figure 3.1: Graphical representation of the approach of the *TwitterRank*

In this study the existence of *homophily* was discussed. The authors concluded that there is a certain reciprocity between users who share the same interests. In other words, actions between similar users occur more frequently than among users with different interests. This phenomenon is present in the social networks and, fundamentally in Twitter.

For the topic distillation of the *tweets*, the study suggest 2 methods. The first, more simplistic, is by using the *hashtags* to distribute the tweets by topics. The authors found this method not efficient and not sufficient for this job. Although we must take into consideration that the use of *hashtags* was not massive at that time. For that reason they used the second method proposed, also proposed in a topic influence detection context [NC08], which is the LDA. The LDA, firstly presented by Blei, et al. [BNJ03] in 2003, treats each document as an word counter array. Based on this assumption, each document is represented as a probability distribution over several topics.

In the Twitter context is not possible to consider each *tweet* as a document, since the goal here is to discover the topics of interest of each user. For that, all the *tweets* of one user are concatenated in a single document. Applying the LDA on that document, the following results are obtained:

1. A matrix DT, where D is the number of users and T is the number of topics. This matrix contains the number of times that one word existent in the content of the user *tweets*, were mapped to a certain topic.

2. A matrix WT. W is the number of unique words and T is the number of topics.

3. Z, an unidirectional array with size N. N represents the number of words contained in all the *tweets*. z_i is the mapping to one topic for the word w_i.

This model bases itself in a PageRank intuition. The influence score of one user is proportional to the sum of influence score of his followers. Despite the influence calculation of the users influence in a social differs from the authority calculation of one website, as described in 3.1.3.1, they also share some similarities.

As model input, a graph $D(V,E)$ is given. V contains all the users and E represents the set of all links between the users. TwitterRank introduces the concept of influence transition which is characterized by 3.3.

$$P_t(i,j) = \frac{|T_j|}{\sum_{a:s_i\, follows\, s_a} |T_a|} * sim_t(i,j) \tag{3.3}$$

In which $P_t(i,j)$ represents the influence transition probability between i and j. $\sum_{a:s_i\, follows\, s_a} |T_a|$ sums the number of all *tweets* published by the friends of i. The result of $sim_t(i,j)$ indicates the similarity between the user i and user j in topic t. The can be determined by 3.4.

$$sim_t(i,j) = 1 - |DT'_{it} - DT'_{jt}| \tag{3.4}$$

This model uses as metrics the number of *tweets* and the follower count, which does not seem representative of a user influence. According to [YW10] would be interesting to modify this model replacing the follower count metric by other metrics like mentions, replies or *retweets*.

3.2 Commercial Applications

Today there are many websites that offer services in the area of social networks analysis. There are many that claim to rank users by its influence. However, the metrics used are inconsistent and already proved wrong. This is the case of the number of followers and the activity rate. Although the existence of these websites, there are 3 that we consider do their job well. They are Klout, Twitalyzer and TweetGrader. Further information about these applications is presented below.

3.2.1 Klout

Klout [Klo11] is a service that measures the overall on-line influence of an user, using data from Facebook and Twitter. Klout assigns a score from 1 to 100 to each user, using

a total of 35 variables. Aside from this, they also provide a set of features such as «who influences whom» and «what kind of influencer am I». In their website they refer which are the metrics used to calculate the global influence score. They divided the influence score in 3 categories: True Reach, Amplification and Network score.
The authors of the influence algorithm believed that influence is the ability to drive people to action. And they defined the action as a reply, retweet, comment or a click. The process to get one's influence score starts by retrieving the metrics, then the scores are generated and normalized. After that, a weight is applied to each different kind of influence score and the global influence score is computed. It is important to refer that the weights are given by a machine-learning engine.
The 3 categories used by Klout to characterize the influence are explained below:

True Reach Score represents the size of your engaged audience and is based on those of your followers and friends who actively listen and react to your messages. Inactive and spam accounts were eliminated and to calculate this type of influence score only individual relationships actions were taking into account.
Factors measured: followers, mutual follows, friend, total retweets, unique commenters, unique likers, follower/follow ratio, followed back %, mention count, list count, list followers count.

Amplification Score is the likelihood that a user messages will generate actions. Measures the ability of the user to create content that rapidly spreads through the network and that compels others to respond. They divided the Amplification Score itself in 3 categories: engagement, velocity and activity.
Factors measured: Unique Retweeters, Unique Messages Retweeted, Likes Per Post, Comments Per Post Follower Retweet %, Unique @ Senders, Follower Mention %, Inbound Messages Per Outbound Message, Update Count.

Network Score indicates how influential your engaged audience is. This measures how influential are the users who reply, retweet and follow you.
Factors measured: List inclusions, Follower/Follow Ratio, Followed Back %, Unique Senders, Unique Retweeters, Unique Commenters, Unique Likers, Influence of Followers, Influence of Retweeters and Mentioners, Influence of Friends, Influence of Likers and Commenters.

3.2.2 Twittalyzer

Twittalyzer [Twi11a] claims to be the most robust tool for Twitter analysis. It uses a large amount of metrics to calculate its indicators. Among them there is the Klout 3.2.1 score. Klout is also used by Twittalyzer to determine which topics one user is most likely to discuss in Twitter.

When analyzing one user profile and his actions, Twittalizer provides 31 scores. For instance, **velocity** which is an indication of the relative frequency at which a user publishes updates in Twitter. **Generosity** is the percentage of updates in which a user retweets other people. However these metrics are not directly related with influence. There is the metric named by Twittalyzer as **influence**. It is likehood that some user will either:

- retweet something the user has written
- reference the user

Twittalyzer is a complete tool that offers a large amount of information to perfectly characterize a Twitter user. Still to the task of measuring one's influence it can become confusing to determine which metric indicates that. The **impact** score is the most relevant metric and it is the main metric presented when analyzing a user. The impact score is a combination of the following factors:

- The number of followers a user has
- The number of unique references and citations of the user in Twitter
- The frequency at which the user is uniquely retweeted
- The frequency at which the user is uniquely retweeting other people
- The relative frequency at which the user posts updates

When compared to the other methods of influence calculation the impact score seems to be appropriate to characterize influence.

3.2.3 TweetGrader

TweetGrader [Twe11] is a website and a tool that tries to measure the power, reach and authority of a Twitter account. In other words, what kind of impact will cause your tweets. TweetGrader does not provide much information about its algorithm. Still they exposed the factors used by the algorithm to determine the influence score. However these factors after calculated are combined with some weighting system that was not revealed by them.

Below are the algorithm factors and a brief detail about what they represent.

Number of Followers More followers leads to a higher Tweet Grade (all other factors being equal). However this metric is easy to game, so they assign a low weight for this factor.

Power of Followers If you have people with a high Tweet Grade following you, it count more than those with a low Tweet Grade following you.

Updates More updates generally leads to a higher grade - within reason. It means that over a certain threshold your score will not get higher even if you continue tweeting at every minute.

Update Recency Users with less elapsed time since their last tweet, generally get higher grades.

Follower/Following Ratio The higher the ratio, the better. However, the weight of this particular factor decreases as the user accrues points for other factors. So, once a user gets to a high level of followers or a high level of engagement, the Follower/Following ratio decreases.

Engagement The more a given user's tweets are being retweeted, the more times the user is being referenced or cited, the higher the twitter grade. Further, the value of the engagement is higher based on who is being engaged.

The grade calculation is done by a percentile calculation of the combination of all metrics. It means that if someone has a score of 80 it will have a higher influence score than approximate 80% of the users. Further, the ranking is created by simply sorting the grade of all the users. So if you have a ranking of 3000 it means that there are 2999 users with a higher score than yours.

Chapter 4

Twitter Crawler

4.1 Introduction

In order to validate the influence model built during the present thesis, we need a dataset. This dataset has to be representative of the Portuguese Twitter community and comprehend a wide temporal range.
The dataset must contain a set of messages published by users and some additional information about them. However, this task is not trivial.
The quality of the dataset is directly related to the quality of the final results. Because of that, we decided to build a system that would allow crawling Twitter data in real-time. The data collected by the system would allow the influence model to calculate metrics used to compute the final influence scores.
This system, called **TwitterEcho** was developed in cooperation with Eduardo Oliveira [Oli11] and retrieves the tweets published by Portuguese users, as well as diverse information about the network structure (followers and friends graphs).

4.2 Specification

4.2.1 Vision

The task of building a system to crawl Twitter data faces several restrictions imposed by Twitter-API. There are limits on the amount of data that can be retrieved using Twitter-API. Rate limits are more and more tight and Twitter has made it clear that its policy is to control the access to data. Currently, the REST API, used in this project, has a limit of 150 API calls per hour for anonymous users and a limit of 350 calls per hour for authenticated users. Consider a system that is crawling, at its limit, information about a list of users. If we have an authenticated user, we can request 350 calls per hour. The *GET users/lookup* method of the Twitter-API, allows retrieving profile information, as well as the last tweet,

of 100 users within 1 API call. That means that in one hour it allows retrieving information about 3500 users. This number is too low to ensure that the Portuguese *twittosphere* is fully covered.
Because of that, it was necessary to build a distributed system to complete this task. The basic idea is to have many thin clients sending requests. In it there are 3 main entities: the Server, the Client and the Twitter-API.
A simple diagram of TwitterEcho architecture and how the diverse entities communicate is presented below.

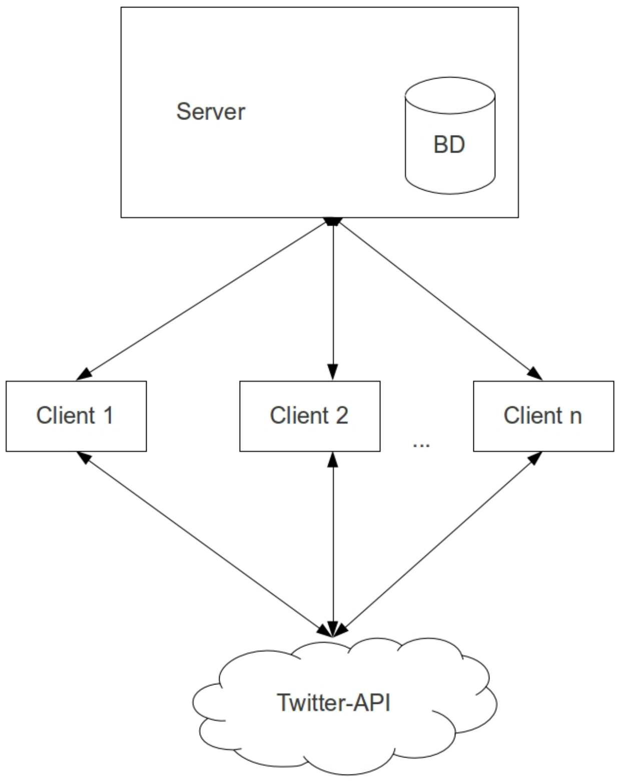

Figure 4.1: TwitterEcho architecture

The server's job is to maintain the database and manage a list of users. The database stores all the information retrieved from the Twitter-API and some additional metadata. The management job the server does, consists in prioritizing the list of users to be sent to the clients and increasing that list of users. However, as we only want Portuguese users to be inserted in our list, the server filters out the foreign users.

Clients periodically ask the server for a list of users to crawl. Once the list is received, depending on the crawler type, each sends a request to the Twitter-API which sends back the response. After that, the response is encoded in JSON and sent to the server. The server then parses the JSON message and stores the information in a database.

Generally, the client consists of:

- 4 executable scripts (featuring the two kinds of crawler lookup/links in UNIX and Windows systems);
- one Perl Module that contains the methods used by the executables and can be installed in any machine with Perl 5 installed;
- a script to assign the executables to crontab in UNIX and task scheduler in Windows. Lookup, by default, will run every minute and links every 10 minutes (more information about these values provided in 4.3.2).

4.2.2 Twitter API

The acronym "API" stands for "Application Programming Interface". It provides methods to accomplish any task that you can do with its website. Actually, the website also uses the Twitter-API to perform the actions. Programs that use Twitter-API methods communicate with it over HTTP.

Currently there are 3 distinct Twitter-APIs: two REST APIs and one Streaming API. The reason there are two different REST APIs is that Summize, Inc. was an independent company that provided search services for Twitter data. Later the company was acquired by Twitter and rebranded as Twitter Search. It continues to exist, still Twitter is willing to unify their REST APIs. The Streaming API is completely different from the other APIs because it allows long-lived connections and it is based on a completely different architecture. To make distinction between the 3 APIs we will refer to them as the REST API (the one used in the TwitterEcho Project), Search API and Streaming API.

The Twitter REST API methods allow developers to access core Twitter data. This includes update timelines, status data, and user information. The Search API methods give developers methods to interact with Twitter Search and trends data. The Streaming API provides near real-time high-volume access to tweets in sampled and filtered form.

The Search API is the easiest one to use. However the calls it makes to the system are not authenticated. It means that the rate limit is calculated by IP and not by username. So, the limit would be 150 calls per hour, which is extremely low and would compromise a robust and scalable system. The methods provided by the Streaming API do not guarantee full coverage of data to build a complete dataset to accomplish the second part of the dissertation. The REST API is the most consistent and complete API to retrieve Twitter data. Being a RESTful resource it provides the developer the choice of response format, which can be JSON, XML, RSS and Atom.

4.2.2.1 Authentication

There are two ways to make requests to Twitter-API. It can be done anonymously or with an authenticated user. As was previously explained, if the request is made by an authenticated user a larger amount of data can be retrieved. To build the clients we need to authenticate them when sending API requests in order to get the rate limits extended. OAuth is an open protocol to allow secure API authorization in a simple and standard method from desktop and web applications.

First of all, we need to register a client application. By doing that, Twitter-API will provide a consumer key and a consumer secret for that application. The architecture behind this consumer key and secret is also used by protocols such as ssh, under the form of public and private keys.

Registering a Twitter application is quite easy and intuitive. It suffices to answer a set of questions about the application to be developed. Once we have the consumer key and secret, we use them in conjunction with an OAuth library. There are OAuth libraries for many different languages. In our case, we used a Perl module, named Net::Twitter, to connect to Twitter-API and it already contained the methods for an OAuth authentication.

Every time the application wants to make a request it will need to be authenticated. And for that a tokens file is needed. The tokens file contains access information for the user and for the application. In a practical way when, in your code, you include the consumer key and secret for the application registered, you will be given a url and a prompt to fill with a pin. That pin is achieved by browsing to the given url and, after login with a Twitter account, give the application permission to register that account to that application.

Further the tokens file can be stored locally and be read when the application needs to make a request. Below is a representation of OAuth process flow.

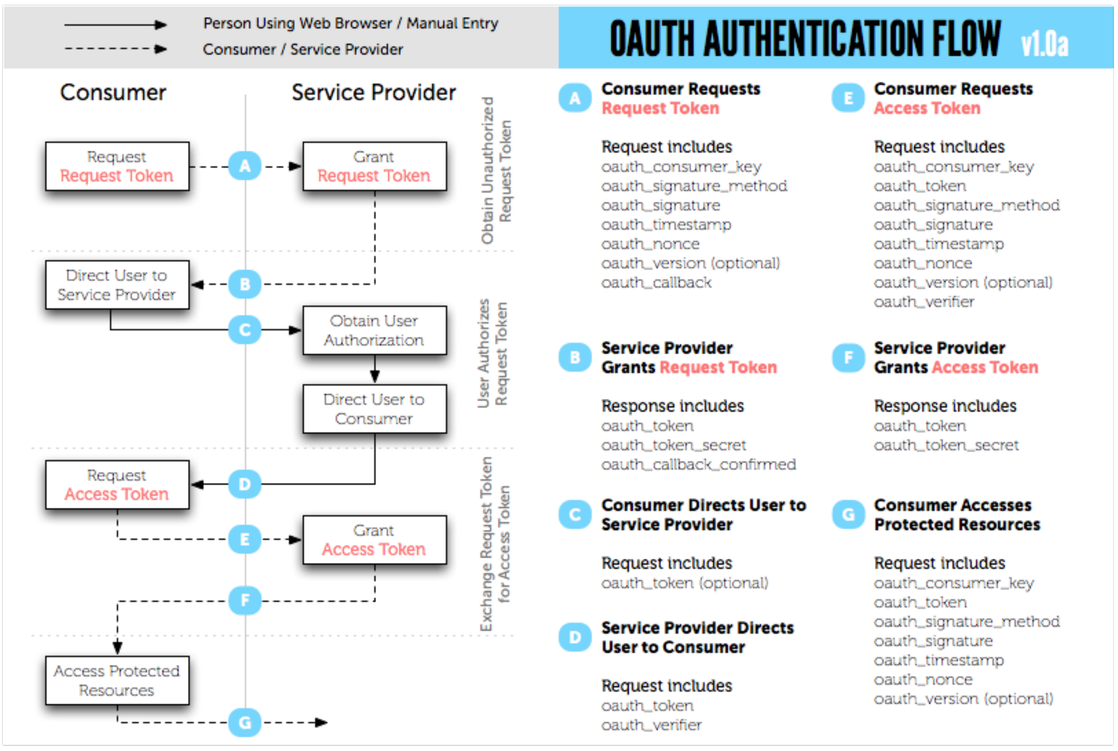

Figure 4.2: OAuth diagram [Twi11b]

4.2.2.2 Rate limits

The default rate limit for calls to the REST API varies depending on the authorization method being used and whether the method itself requires authentication.

- Anonymous calls are based on the IP of the host and permits 150 requests per hour. This classification includes unauthenticated requests (such as RSS feeds), and authenticated requests to resources that do not require authentication.
- OAuth calls allows 350 requests per hour.

The REST API does user and IP-based rate limiting. Authenticated API calls are measured against the authenticating user's rate limit. Unauthenticated calls are deducted from the calling hosts IP address allowance.
Rate limits are applied to methods that request information with the HTTP GET command. Generally, API methods that use HTTP POST to submit data to Twitter are not rate limited, however some methods are being rate limited now.

In the clients we use 3 API requests: *lookup_users*, *friends_id* and *followers_id*. The request *lookup_users* allow retrieving data and the last tweet of up to 100 users, spending 1 API call. The *friends_id* and *followers_id* methods consume, each one, 1 API call to retrieve either the follower or the friend list of one user.

Of utmost importance is that the rate limit never gets reached, because we do not want to risk being blacklisted by Twitter. For that we include that verification in the client code. There is one request (rate_limit_status) that does not consume API calls that allows users to know how much API calls remains for the current hour.

4.2.2.3 Scheduling the clients

Following a explanation about the period in which each type of crawler will run, is presented .
Consider that the lookup client per run, takes around 20 seconds to crawl data of 500 users, consuming 5 API calls. Assuming a overhead of 10 seconds it would be possible to run a lookup client every 30 seconds. Which means that with the 350 API calls per hour, we would run lookup 70 times within an hour. Thus, it means that the crawler would be active for 35 minutes running almost uninterrupted. With this we would have the crawler inactive for the last 25 minutes. So, if one user posts two tweets within this time, only the last one would be caught.
For that reason it is more efficient not to waste all the API calls in the beginning and distribute them over the whole hour. So, we decided to run each crawler every minute. This will only consume 300 of the 350 API calls, but all the tweets are expected to be caught. Except in the case when one user sends more than one tweet per minute. Since the objective is to capture a glance of the overall community interaction, that fact is not relevant.

Now regarding the links client case, it takes about 2 minutes to crawl the follower and friend list of 25 users. Each links run consumes 50 API calls, which means that would be possible to run 7 instances of the links client before the rate limit is reached. That means that it would be possible to run links every 8.57 minutes. However, in this case we do not have the risk of loosing information like in lookup's case. Imagine that a user decides to follow another and, in the next minute, chooses to "unfollow" the same user. That information is likely to be lost by the crawler, still this information is not relevant to our study. In fact, the possible propagated influence is retrieved from tweets and not from the following action.
So we choose to run links every 10 minutes. This ensures a continuous crawling process

and that the rate limit is never reached.

In case two types of crawlers were to run in the same machine at the same time, at the end of the first 10 minutes, we would consume 100 API calls. That means that in the first 35 minutes the rate limit would be reached and for the next 25 minutes the two types of crawler would became inactive. Because of the reason stated above, it is recommended not to install the two types of crawler simultaneously in the same machine. In summary, we scheduled links to run every 10 minutes and lookup every minute, one in each machine.

4.3 Implementation

4.3.1 Crawler Database

The Crawler's database is located at the server. It stores all the information retrieved by the clients plus metadata such as error and ping logs or information to the user list growing processes that run on the server.
In this section we describe the tables that are directly related with the information retrieved by the clients and used further to create the influence model dataset. Some of the tables described below have additional attributes that are not discussed here, because they are not relevant to the work performed in this thesis.
Table 4.1 contains information about Twitter users.
Table 4.2 contains information about the tweets posted by the users in database.
Table 4.3 contains the identifiers of the all user's follower list.
Table 4.4 contains the identifiers of the all user's friend list.
Finally, table 4.5 stores information about the number of tweets, followers and friends of every user over time.

Table 4.1: users table

Attribute	**Description**
id	Represents the internal Twitter identifier of a user
screen_name	The account username. It is used very often preceded by '@' when retweeting, mentioning and replying a user.
name	Supposedly the real name of a user.
location	Location inserted by the user. This does not have validation by Twitter, so this can be a real or fake location.
profile_image_url	Is the link to the image that the user has in his profile
created_at	Timestamp variable with the date of profile creation
time_zone	Its more or less like location but the input is validated. Very useful to determine the nationality of a user. It contains the name of the city of the user timezone. For Portuguese users it contains in most of cases «Lisbon».
description	Small text written by the user describing himself.

Table 4.2: status table

Attribute	**Description**
status_id	Represents the internal Twitter identifier of a tweet
user_id	The identifier of the user that sent the current tweet.
created_at	Timestamp of the creation of the tweet.
text	Content of the message. It is through the text that the influence model gathers the metrics to influence estimation.
source	Contains the description of the interface used by the author of the tweet. This can be the Twitter website, mobile phone or any other existent Twitter client.
in_reply_to_user_id	Contains the identifier of the user that is being replied by the current tweet, if that is the case. Otherwise this field will be null.
in_reply_to_status_id	Contains the identifier of the tweet that is being replied by the current tweet, if that is the case. Otherwise this field will be null.
in_reply_to_screen_name	Contains a string with the screen_name of the user that is being replied by the current tweet, if that is the case. Otherwise this field will be null.
retweet_count	Contains the number of times the tweet had been retweeted. However it only counts retweets up to 100 times. It means that if a tweet was retweeted more than 100 times, the retweet_count value will still be 100.

Table 4.3: followers table

Attribute	Description
user_id	Represents the internal Twitter identifier of a user
followers	It is an ordered list of user_ids that follows the user. The reason why the list is ordered by id is to easily know if the follower list suffered any change.
date_insert	Timestamp representing the date when the user's follower list was checked. This is very important to create graphs of followers over time.

Table 4.4: friends table

Attribute	Description
user_id	Represents the internal Twitter identifier of a user
friends	It is an ordered list of user_ids that user follows. The reason why the list is ordered by id is to easily know if the friend list suffered any change.
date_insert	Timestamp representing the date when the user's friend list was checked. This is very important to create graphs of friends over time.

Table 4.5: statistics table

Attribute	Description
user_id	Represents the internal Twitter identifier of a user
statuses_count	Contains the number of tweets published by the user.
followers_count	Contains the number of followers the user has.
friends_count	Contains the number of friends the user has.
last_date	Timestamp representing the date when the user's statistics were inserted into database.

4.3.2 Client

The client program consists of a Perl module, that can be easily installed in any machine. Once installed, the executables that use the module methods will continuously send requests to the Twitter-API and send the data back to the server.
In section 4.3.2.1 we introduce Net::Twitter, the Perl library responsible for the connection with the REST API, OAuth authentication included. The Crawler module was built using Net::Twitter.
In section 4.3.2.3 and in section 4.3.2.4 we explain how the different types of crawler operates.

4.3.2.1 Perl Module - Net::Twitter

There are many libraries written in many languages to provide access to Twitter-API. To write the clients we choose Perl. Perl is an interpreted, dynamic programming language. Its use is very common to perform simple tasks regarding data mining and text processing. Thus, its usage became massive in the late 90s as a CGI scripting language due to its parsing capabilities.

To use Twitter-API methods within Perl, we used the Net::Twitter module that is available at CPAN [CPA11]. It allows the use of REST API and Search API methods. It also allows the user to authenticate using OAuth in order to increase the rate limit when using the REST API.

This module provides the developer almost all the methods stated in REST API. In this present dissertation only three methods were used and, with them, we can retrieve all the necessary information.

The methods that we used were:

lookup_users Returns extended information about up to 100 users, specified by either ID, screen name, or combination of the two. The user's most recent tweet (if the authenticating user has permission) will be returned inline. This method accepts user IDs or screen names as either a comma delimited string, or as an ARRAY ref. It also accepts arguments in the normal HASHREF form or as a simple list of named arguments.

friends_id Returns a ARRAY reference of numeric IDs for every user followed the specified user.

followers_id Returns a reference to an array of numeric IDs for every user following the specified user.

Net::Twitter also provides an error handler, throwing exceptions whenever anything goes wrong. This feature was very important in the debugging phase, providing information about rate limit reaching, Twitter API overload or errors in the request.

To retrieve relevant and complete information from Twitter we use three API methods. Based on that, 2 types of client were created, lookup and links.

4.3.2.2 Crawler Module

The Crawler Module is a Perl Module created to provide the client's executables the methods they need to perform their tasks.
The module follows the Perl modules format specified by CPAN [CPA11]. Because of that, it can be easily installed in any machine, as long as Perl 5 and the dependent modules, such as Net::Twitter, are installed.

Some of the main methods are described below.

post_error this method sends to the server an error message. Is called whenever anything went wrong in the crawling process. This is very important to the logs system, since in the server it is possible to know which crawling processes are getting troubles.

get method used to retrieve a list of users to be crawled. The method takes as argument the hostname of the machine attempting to request the list of users and the version of the client running there, as well as the type of client running. This information is useful to the server, allowing a ping system. It is possible to know what machines are running what client type and when. If the version of the client is not the most recent, the server will reject the request made by it. The get method varies with the type of crawler. If it is the lookup, get will provide a list with a maximum of 100 users, in case it is links, it will provide a list with a maximum of 25 users.

post_lookup method whose mission is to encode, using JSON, the data retrieved from Twitter-API. Then the encoded message is sent back to the server. This is done to standardize the flow of information between the server and the clients. As the get method, this one also sends the server information about the host running the server and the client version.

post_links similarly to post_lookup, this method also encodes with JSON the data retrieved from Twitter-API and sends it back to the server.

lookup_users this method is responsible for authenticating the crawler client using OAuth, and connecting to the server using the get method to request a list of users. Then uses the Net::Twitter lookup_users method to collect data about the users received, before sending it to the server using post_lookup. This method repeats the above steps 5 times. It means that, per run, lookup_users will crawl information about 500 users, consuming 5 API calls. It contains error handling functions and will not permit the crawler to make API requests if the rate limit is reached.

friends_followers_users this method, as the latter, connects and authenticates to Twitter-API. After that, uses the get method to retrieve a list of users. For each user it will use the Net::Twitter methods followers_id and friends_id to build two lists. Each list is sorted by user_id and then sent to the server using post_links. This method will retrieve the followers and friends list of up to 25 users per run, consuming 50 API calls. It contains error handling functions and will not permit the crawler to make API requests if the rate limit is reached.

The methods described allow the executable scripts to accomplish their tasks. In the current version there are 2 kinds of client scripts, each one with two versions to allow running it under Windows and UNIX operation systems. Using an additional script, crontab_config.pl, it is possible to set any of the client types in crontab in UNIX. If the same authentication is used, it is recommended not to use both of the crawlers at the same time in the same machine, due to rate limit issues.

4.3.2.3 Lookup

One of the clients type is *lookup*. This crawler is responsible for retrieving profile data of a set of users. The data includes:

- users' Twitter id
- users' defined name
- users' location
- users' profile image url
- users' last tweet (includes all the information that characterizes a tweet, see 2.2)

When enabled in crontab, lookup will run every minute for about 20 seconds. Assume that the user priority system, existent in the server, is not active and that every user sent by the server have at least one tweet. The lookup crawler, would retrieve information and the last tweet of 500 users per minute. Therefore, ideally one lookup client will get information and one tweet of 30.000 users in one hour. With this, if you maintain this lookup client on throughout a complete day, it would be enough to crawl information about the whole Portuguese *twittosphere*.

However, the dataset built will only contain one random tweet per user which is not representative and enough to study the dynamics and propagation of influence through an online community.

Now consider a distributed system with 5 lookups crawling information. Assuming also that all the users update their status every minute. We can continually crawl all the tweets of 2500 users. With this we would need 200 clients to crawl the whole Portuguese *twittosphere*. Although the assumption is totally unrealistic.

Let us assume, in a more realistic approach, that each user publishes a tweet every hour. At the time of this study there were 62.460 users in the database, which means 62.460 tweets per hour. Having 2/3 clients running would be enough to continually collect all the data produced by the users.

However, there are users that tweet every minute and there are others that tweet once a week. Because of that, the server has a priority system in order to send first to the clients the most active users.

4.3.2.4 Links

Links is the client responsible for creating the social network graph. It retrieves information about the users' list of followers and friends. As was described in 2.3.2 the following action represents an unidirectional relationship. Assume a network with three users: A, B and C. In this community A follows B and C, on the other hand C follows A and B follows no one. For the user A the follower list will contain C and the friend list will contain B and C. The follower and friend lists are important to create a graph representation of the network, each edge representing a following relationship.

Each run of the links client will consume 50 API calls. Calling the method friends_id or followers_id for one user will consume 1 API call. In this process, the links crawler will receive a list of 25 users. For each one, 2 API requests are made. The running time of the links process is, in average, 2 minutes.

Due to rate limit issues 4.2.2.2, links will run every 10 minutes. Thus, in 1 hour with 1 links crawler it is possible to build the network of 150 users. What means that would be necessary 28 links clients running for 24 hours to build the entire Portuguese *twittosphere* network.
Having a more reasonable number of links, let us say 5, the system would need 5 days and half to build the complete network graph.

4.3.3 Server

The server of TwitterEcho was developed by Eduardo Oliveira [Oli11]. This server is responsible for storing the information crawled by the clients into the database, as well as maintain and manage a list of users.
At the start of the system, we provided to the server a list of about 2.000 users. It is not certain that all the initial users were Portuguese but the majority is. With this seed, the server started its job, sending sub-lists to the clients. Once the information of those users were retrieved, the server is ready to start its user list growing processes.
Those processes consists in a bunch of scripts running periodically. These scripts analyze the tweets retrieved and uses regular expressions to search for mentions in the text. Those mentions represent usernames that will likely be inserted in the users list. We only want Portuguese users to be inserted, since the purpose of the database is to aid the Portuguese *twittosphere* analysis. To achieve this, the server side script asks the Twitter website for profile information. In this case the Twitter-API is not used. If the user has his location and timezone defined, the script will look for clues that identify the nationally of the user. If this analysis is inconclusive, the user will be put in a queue to be further analyzed. A sample of 100 tweets posted by the user is retrieved, and a language analysis script indicates if the user is Portuguese or foreign. This process is very accurate [Oli11], and with this we assure only Portuguese users are inserted into database. Besides the list growing by mentions, there is also a process that attempts to add users that are in the follower list of some users. More information about the TwitterEcho server provided in [Oli11].

4.3.4 Summary

In this section we summarize the whole crawling process. First of all, we need to provide the server a seed with a set of users considered Portuguese. The server starts by sending those sub-lists of users to the clients that are running.
If the running client is a lookup type crawler, the server will send it 5 lists with 100 users each. In case the crawler is links, the server sends a list with 25 users each time.
The client receives the list of users and sends requests to Twitter-API through Net::Twitter module, depending on the type of client. The information retrieved is encoded with JSON and sent back to the server. After that, the server parses the encoded JSON data and stores it in the database.
Periodically, some server-side scripts, analyze the tweets retrieved and the follower lists in order to add users to the list of users. The users to add are analyzed and their "nationality" identified.

4.4 Crawl Analysis

TwitterEcho server uptime at the time of this thesis is 55 days. Within this time it retrieved information about 62.464 users in a total of 2.510.403 tweets. When comparing with relevant studies done on the online influence area, this number is quite acceptable since the current dissertation only involves the Portuguese community.
The list of users is 31 times bigger than the initial seed, containing more than 60 thousand users. This represents a fast growing list, although the number of registered users has been stabilizing recently.

In order to optimize the crawling process it is important to analyze the volume of data posted throughout the day to identify peaks of activity. Using the stored data we analyzed the tweet distribution per hour (see figure 4.3). In the chart is visible the low amount of tweets published at night. The content published starts to raise at dawn and has its maximum at and after dinner. Then it abruptly decreases as the night begins. From this information, the TwitterEcho system would need more lookup clients enabled during the day with special incidence between 19:00 and 22:00. Note that the times we are speaking here are UTC format. Which means that in Portugal corresponds to the GMT during the winter time, but in summer time it is 1 hour latter. Since the crawler has been running before and after the change from winter to summer time, the diagram may contain a deviation up to 1 hour.

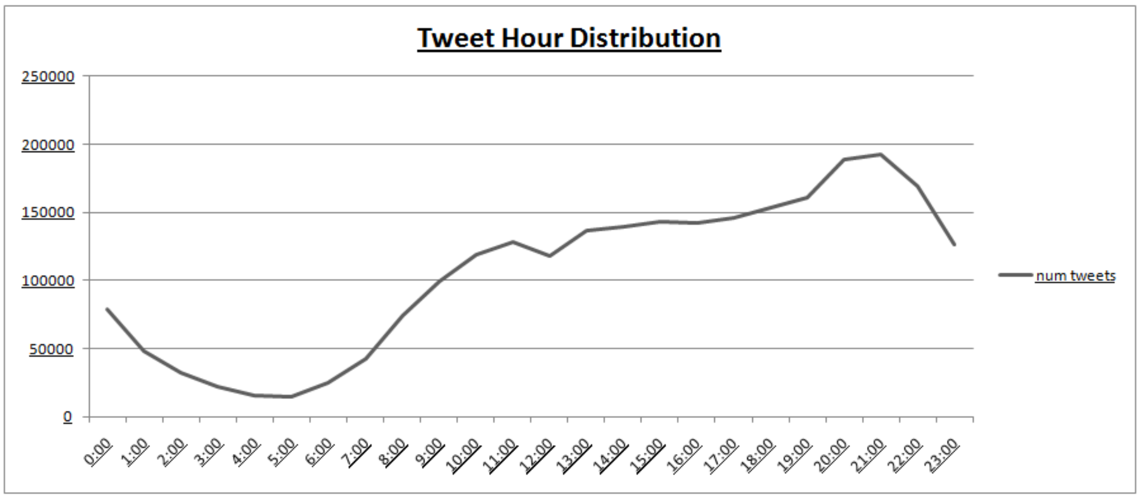

Figure 4.3: Number of tweets produced in each hour of the day

A second aspect to consider regarding the scheduler is to analyze the global activity of the Portuguese community. Chart 4.4 represents the distribution of users with a given number of tweets posted. Note that the chart is based on the table statistics 4.5 from the database. This means that the tweet count is the one stored by Twitter and not by our own 2 month tweet database.

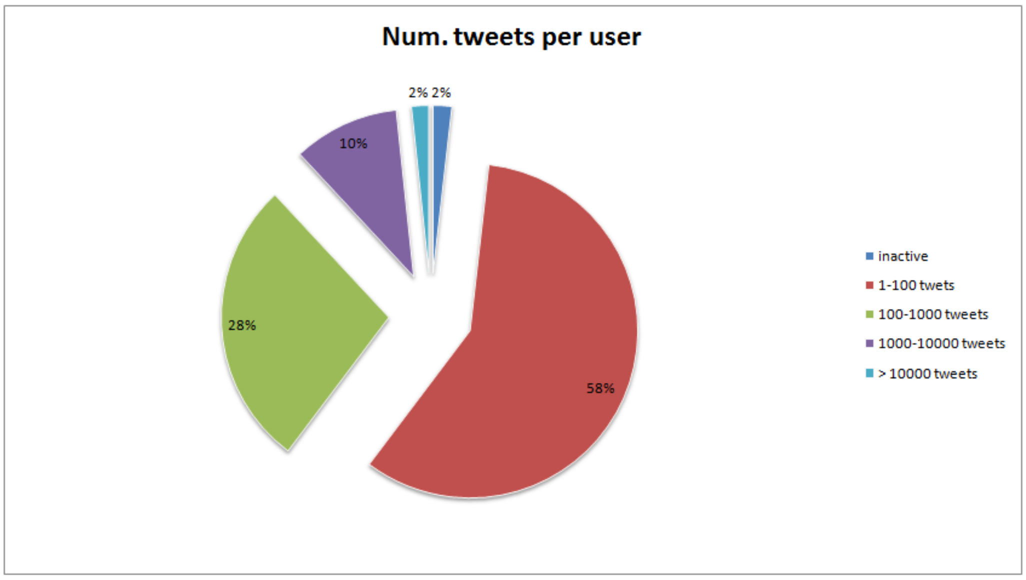

Figure 4.4: Percentage of users that produced certain amount of tweets

From the chart we can conclude 3 things:

Inactive users around 2% of the Portuguese community on Twitter does not tweet. This is an indicator of inactivity but cannot be taken into account alone. There are Twitter users that do not tweet, but read carefully the messages of their friends. They are just observers.
Through a view of the chart we notice that the percentage of users with no tweets is not that big, as was stated in [RGAH09], although the database does not contains all the Portuguese users. One reason to explain that is that the system privileges the mentioned users to entry the database. To be mentioned a user must be engaged in the community which is not possible without interaction from both sides. This could mean that an active and engaged user will have more possibilities to enter our database.

Very active users On the other hand we have a small percentage of users that have more than 10.000 tweets. The most active user has more than 500.000 tweets. In table 4.6 we manually check the 10 most active users.

Therefore, we can concur with what was stated in [CHBG10] that «activity does not represent influence». The top active users are news feeds and spam accounts, most likely controlled by bots, that do not interact with the Twitter community.

Low activity there is a large percentage of users that published less than 100 tweets. This number, plus the "update rate" shown in table 4.7, could be an indicator of inactivity.

Table 4.6: Most active users

Username	Description
@news_portugal	Seems a bot that posts messages with news from online newspapers appending to that a hashtag with the name of the newspaper and a url to the full new at the newspaper.
@NewsPortugal	As the previous it also aggregates news from online newspapers and spreads them with the same format.
@NoticiasPlaneta	Similar to the previous. Also publishes tweets of online newspapers around the world with special incidence in the Portuguese case.
@agendaportugal	It the tweet of an online job finder. The messages consists of a job offer with the link to the entry on their website.
@egomesduarte	Spam account.
@NoticiasMundo	Similar to the first top three active users. Every 30 minutes updates all the news from several news websites.
@news_pt	Portuguese news feed.
@_EDconsulting	Spam account.
@portugalnews4u	Compilation of Portuguese news.
@feedsdenoticias	Feed of the main news. Appends a tag indicating the news theme.

Another relevant aspect is to know how the Portuguese community is growing. Chart 4.5 represents the time when the Portuguese Twitter users created their accounts. It is clear that 2009 was the year of the *boom* of Twitter in Portugal. Between 2010 and 2011 there are few differences, noting a slight decrease of the number of registrations in the first semester of 2011. The tendency is for a stabilization of the number of records in the coming years, with the most interested people already registered. To complement the information given by 4.5, the next chart was built from the data collected from TwitterEcho. It 4.6 presents the growth of Portuguese Twitter users over time.

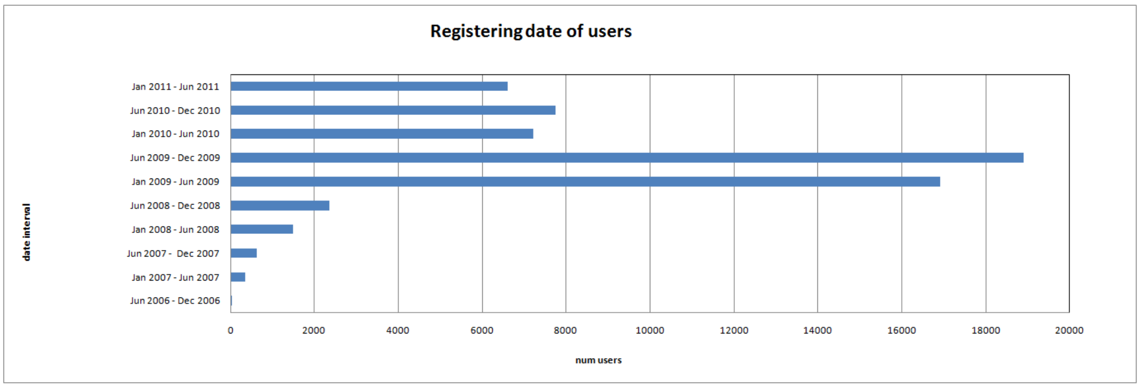

Figure 4.5: Number of registered users by time interval

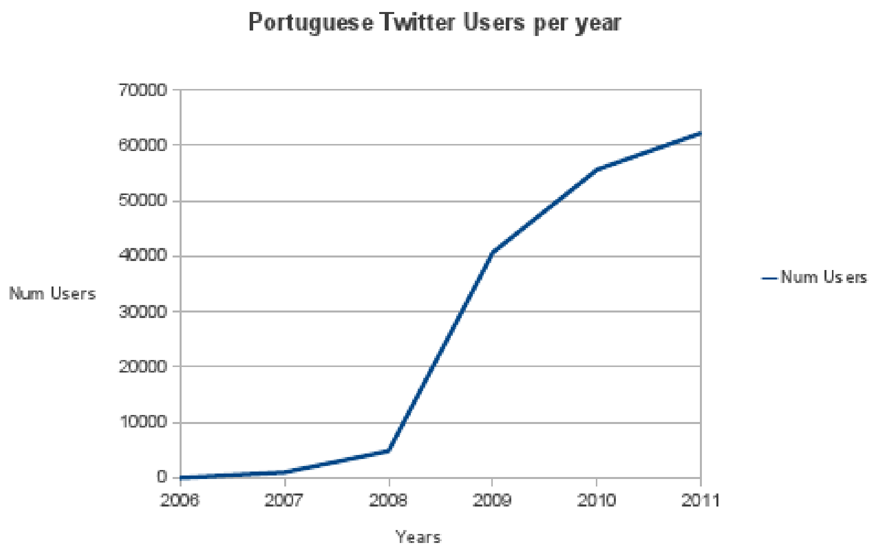

Figure 4.6: Portuguese Twitter users over time

Now we focus on the frequency that a user updates his status. The following table 4.7 represents the amount of distinct users that have tweeted within the date specified. Note that the test was done in 14th June 2011.

Table 4.7: Status update frequency

Range	**Number of users**
Yesterday	9.863
Last 5 days	15.256
Last 10 days	18.945
Last 15 days	21.207
Last 30 days	25.846
Since the beginning	58.963

Considering that the Portuguese *twittosphere* has 100.000 users, from the table we can conclude that only about 1/3 of the Portuguese *twittosphere* frequently publishes content. This is a very high value compared to what was stated in other studies. [HP09] refer that 10% of the users produce 90% of the content.
To try to prove that, we count the number of users that have statistics and we get 59.934. Calculating 10% of that value we get 5993 users.
If the statement above applies to Portuguese Twitter community, the top 5993 active users would produce 90% of all content.
We sum the number of tweets of each user and we get 48.868.580 tweets. The 5993 most active users published 43.340.514 which corresponds to 88,7% of all content.

So, we conclude that the statement done in [HP09] is correct and applies to Portuguese *twittosphere* as well.

To conclude this chapter, it is important to say that TwitterEcho is scalable and, depending in the number of clients, is enough to continually crawl all the actions in the Portuguese Twitter community.

Chapter 5

Influence Modeling

In this chapter we propose a model to represent the influence of Twitter users. The model, is based in some metrics that we consider indicators of influence, based on what was stated on the literature and used recently on similar models. In section 5.1 we expose a general vision of the whole influence measurement system and we detail the model specification. Section 5.2 provides implementation details of the diverse modules of this system. Finally in section 5.3 we discuss the results obtained and we validate/evaluate the model.

5.1 Specification

5.1.1 Vision

In this section we explain the processes of the influence detection system and the data flow between the diverse constituent modules. The methodology and architecture used for developing this system were similar to the ones used in [Klo11] and [Twi11a].
The language chosen to develop the modules was, once more, Perl due to its parsing capabilities and ease of use of regular expressions. To develop the website we used PHP embedded with HTML and CSS.

There are 4 modules that work separately and sequentially. This means that, with the exception of the interface module, each module has one executable script and, to obtain the final output, they need to run in a certain order. Next, we detail the operation mode of each module. The diagram in 5.1 shows the flow of information between the various modules.

Community module responsible for connecting to the database 4.3.1 and build the dataset necessary for the influence calculation. Information about users, tweets, followers and friends is then stored in files, because this makes it more lightweight in terms of memory.

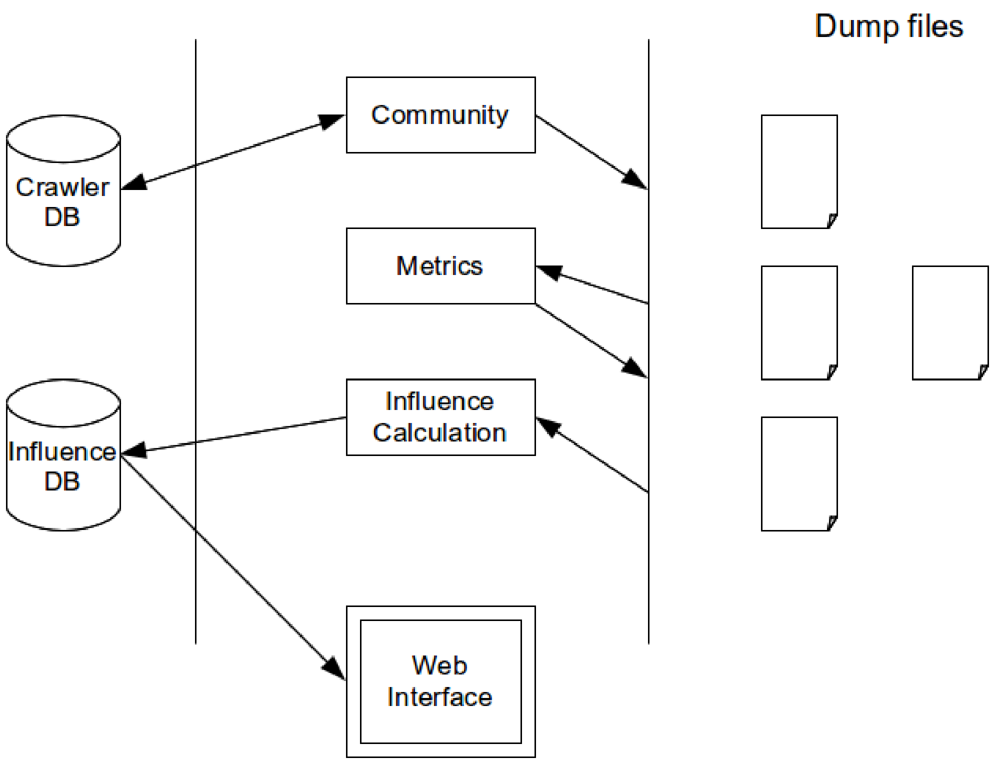

Figure 5.1: Influence Detection system

Metrics this module starts by reading the information stored by the Community module. Then uses that information to derive the influence metrics. Once retrieved, the metrics are also stored in files.

Influence Calculation this module starts by reading the data stored by the Metrics module. Uses the metrics retrieved to apply the model defined and explained later in 5.1.2. It outputs 3 different influence scores for each user, representing Recognition, Reach and Engagement influence. Then, based on these 3 values, a global influence score is calculated. The Influence Calculation module stores the results obtained in another database.

Web Interface website that allows the visitor to see the results obtained by the influence model. Reads the influence information stored in the database and presents a list with top influential users, allowing the display of more details about the score given.

5.1.2 Model Specification

The model proposed in this dissertation is based in some existent influence detection algorithms [RGAH09] and the intuition behind some websites [Klo11] with the same objective.

Through this section we expose and justify the assumptions made to create the model. Here, we describe the details of the model parameters and relate them to other similar models. Further explanations about the *modus operandi* of the system can be found in section 5.2.

The first assumption, is that global influence may be divided in indicators of its existence. We identify 3 indicators: recognition, engagement and reach. So, we define influence as a weighted combination of these indicators. Each indicator is explained below:

Recognition measures the credibility and authority the community assigns to a user. For that, recognition score is split in 3 different scores: retweet score, mention score and recommendation score.
These scores are weighted and combined to produce the recognition score. In the present model we assign more importance to retweet score, but if the recommendation becomes more used in the near future, it may be necessary to adjust the weights and assign more importance to recommendation. We can define recognition as the combined capacity of one user being recommended and mentioned by the community, and having his content spread through the network of his followers within many degrees. This indicator multiplies the number of retweets/mentions/recommendations by the number of distinct users that have performed that action. Which means that an user with 100 retweets from one user will be as influential as a user with 10 retweets made by 10 different users.

Engagement measures the capacity of a user to interact with the rest of community. This could also be called activity generation, which means, it is the users' ability to create activity around him and his content. In our model we define engagement as the number of replies given to the user, minus the replies posted by the user.
Here we want to measure the amount of activity, the content published by the users, attracts. For that, we measure the number of replies received, that can be seen as comments to the user's posts. We subtract the number of replies done by the user, because if the user comments other's content, there is an increased probability of being replied due to reciprocity. The subtraction operation is weighted, having the replies given to the a user a higher weight than the replies posted by the user.

Reach the reach score measures the size of the users' audience. But measuring the number of followers of one person does not represent much about his influence [CHBG10]. Because of that, this parameter has a low weight when compared to the other two indicators. The algorithm tries to eliminate "game possibility", in other

words, if someone follows another, he raises the probability of being followed back. In Twitter that phenomenon was spotted in [JSFT07]. There are many users with a huge follower list but, in the same way, they have a huge friend list too. In many cases the difference, in terms of size, between the follower and friend list is minimal.
Because of this reason, we define Reach score as the number of followers minus the number of friends. With that we try to transform the influence gaining process into a meritocracy, where the cheating possibility is very remote. To prevent the possibility of negative reach score, we only apply the subtraction operation when the follower list is bigger than the friend list, otherwise reach score would be zero. In other words, reach score is the subtraction of the number of followers by the number of friends forcing the minimum score to be zero.

In order to determine the diverse influence scores, we store a set of variables **for each user** that describe his actions since the start of the crawling process. Those variables include:

- replies made **by the user**, containing the screen_name of the replied users and the number of times he replied that user.
- replies made **to the user**, containing the screen_name of the users who replied him and the number of times each one replied the user.
- retweets made **by the user**, containing the screen_name of the retweeted users and the number of times he retweeted that user.
- retweets made **to the user**, containing the screen_name of the users who retweeted him and the number of times each one retweeted the user.
- mentions made **by the user**, containing the screen_name of the mentioned users and the number of times he mentioned that user.
- mentions made **to the user**, containing the screen_name of the users who mentioned him and the number of times each one mentioned the user.
- recommendations made **by the user**, containing the screen_name of the recommended users and the number of times he recommended that user.
- recommendations made **to the user**, containing the screen_name of the users who recommended him and the number of times each one recommended the user.
- hashtags used by the user, containing the number of times hashtag was used by him.
- the number of times the user was replied.

- the number of times the user was mentioned.
- the number of times the user was recommended.
- the number of times the user was retweeted.

As we can see from the information stored, the model can be easily adapted to calculate the relative influence as proposed in [CLH+10]. We have complete and extensive information about, for instance, who retweets who and how many times they have done so.
From this, it is possible to determine who you influence and by whom you are most influenced. The model proposed could evolve using the concept of relative influence and influence propagation. In other words, consider 3 users: A, B, and C, ordered by their influence score. Suppose A recommends C, that recommendation would have much more impact than if C, with no recommendations, recommends B.
Such influence propagation dynamics that could be easily implemented within the proposed model.

The model created considers the cheating possibility. Imagine a system that ranks users by their influence taking into account only their activity. It would be relatively easy for a programmer to develop a script to spam and, with that, get into the top of the rank: because of this reason we completely remove the activity measure from our model, since was already proved that activity does not correlate with influence [CHBG10] .
Another way to cheat, is a user mentioning himself. This action is possible in Twitter. However in this case the model just ignores that mention, since you cannot influence yourself.

Initially, recognition score was measured using the "action" count. It means that having a user B retweeting user A 100 times, would be the same that having 100 users retweeting user A, each one, once. The first case represents one user influencing another in a large scale. The latter represents one user exerting low influence on many users.
Using the first proposal, the two users with higher recognition score were 2 spam accounts, that mentioned one another in all produced content. To eliminate this game possibility we used the second proposal.

5.2 Implementation

In this section we explain how the influence detection system was implemented. Each module was developed independently and each one does a job that the next module will continue. This means that the modules work iteratively, using the work done by the previous by reading the state from a dump file. With this we can have a more lightweight system in terms of memory. Also, it allows running only the module we want, without needing to run the whole system which is quite heavy.

5.2.1 Community

This module is responsible for connecting to the Crawler Database 4.3.1 and retrieving information about the stored users, as well as their tweets and relationships (followers/friends). In this module we created two Perl classes named User and Tweet. These classes are responsible for storing information about **users** and **tweets**. This includes for users: id, screen_name, profile_imgage_url and description. From tweets we store: user_id, status_id, created_at, text, retweet_count, in_reply_to_status_id and in_reply_to_user_id. This data is from users table 4.1 and status table 4.2.

Once this information is created by User and Tweet instantiation, we created a list with the **uppercase screen_names** of all users. This is done because an user can, for instance, mention user "PauloQuerido" with a tweet like: "hello @pauloquerido", and Twitter accepts that as a mention.
So, we need a list with all screen_names with capital letters to help the detection of screen_names in tweet analysis in 5.2.2. Also, it improves the efficiency of the script when searching for the existence of the user mentioned in our database.

Beyond users and tweets' information, this module also retrieves the **number of followers and friends** for each user. This data is retrieved from statistics table 4.5.

Before the script dies, it "dumps" to files the content retrieved from the database in a standard format used widely in this project. Since we are using Perl, the format we choose to represent the information is a HASHREF of HASHREFs. With this, the next module running will start in the final state of this module. This helped a lot in the development and debugging phase, because it allows to run only the module that we want to test.

5.2.2 Metrics

The module Metrics is responsible for analyzing each tweet and collect some metrics that are implicit in the tweet text.
Firstly, the executable script starts by reading the files created by the module Community 5.2.1. This process is really fast, it takes only 1 second to retrieve information of more than 64.000 users.
However this is the most computationally heavy module in this system. To a complete metric analysis for 2.700.000 tweets it takes almost 10 minutes. But this issue can be

considered irrelevant because this module only needs to run once a day or even once a week.

The tweets' data is grouped by user. For each user's tweets we apply a set of regular expressions representing the actions in Twitter that we are using as metrics to characterize influence.

Hashtags we are using the following regular expression:

(#(word))+

If tweet's text contains that regular expression it will be stored in a Perl hashtable that as the format:
"[screen_name => [hashtag => num times used]]"
It means that for each user we are storing all the hashtags he used and in which frequency. With this information, it is possible to make a preliminary and simplistic topic distribution per user.

Retweets To for detecting a retweet we are using the following regular expression:

(RT(space)+@(word))

For each user's tweet if it contains the above regular expression we store the "word" in a Perl hashtable representing an user's screen_name. We call it "retweets_to" and it has the format:
"[screen_name => [screen_name => num times retweeted]]"
It stores for each user, the screen_names and the number of times he retweeted another user.

Replies To we use the following regular expression to detect a reply:

($\hat{@}$(word))

It means that if an user's tweet contains a @word in the beginning of its text it represents a reply to an user with screen_name equal to "word". We store it in a hashtable with the following format :
"[screen_name => [screen_name => num times replied]]"
So, for each user, we store users he replied and with which frequency it occurred.

Mentions To to detect the mentions made by one user we use the following regular expressions:

((word))* ; (^**(word))** ; **((#FF)||(#FOLLOWFRIDAY))/i**

For each tweet any @word that is not in the beginning of the text and does not contain the third regular expression, is marked as a mention. This metric is stored in a hashtable name mentions_to with format :
"[screen_name => [screen_name => num times mentioned]]"
It stores, for each user, the screen_names of the users he mentioned and how much times he did it.

Recommendation To to detect a recommendation in a tweet we use the following regular expressions:

((#FF)||(#FOLLOWFRIDAY))/i and (@**(word)**)

Every tweet that contains the first case insensitive regular expression, the following "@word"s are marked as a recommendation made by the user to users with screen_name equal to "word"s used. As the previous this is stored in a hashtable with format :
"[screen_name => [screen_name => num times recommended]]"

With exception for hashtags, to all of the hashtables created, a filter is applied to remove screen_names that do not exist in our database. To do that we benefit from the list of uppercase screen_names.
When all of those hashtables are filled, it starts building the main metrics which are the opposite relations. Instead of knowing the screen_name of the users a certain user retweeted, we want to know the screen_names of users that retweeted a certain user.
So we build 4 new hashtables: **retweets_from, mentions_from, replies_from and recommendations_from**. They are simply a transposition of the other hashes. For instance, imagine a user A retweeted user B 10 times. The retweet_to hash will present the data in the following format: [A=>[B=>10]]. On the other hand retweet_from presents the data with the format [B=>[A=>10]] and represents: B was retweeted by A 10 times. With the information provided by these new hashes we can build up the information really used by this version of the model. They are the **retweet_count, reply_count, mention_count and recommendation_count**. Each one contains, as the key of the hash, the user screen_name and as value the number of times he received a certain action (retweet, mention, recommendation or reply).

After this computation all of the built hashes are dumped in files, so the next module can use this to continue the task of detecting influential users.

5.2.3 Influence Calculation

This module takes the metrics stored in the dump files and starts the influence detection task. This task can be divided in 3 calculation processes. One for each type of influence score as stated in 5.1.2.
In this we use metrics in conjunction with a set of weights indicating the importance of each metric. We consider recognition influence as the most important type of influence, followed by engagement score. These represent influence in its definition, in other words, an action triggered by the behavior of an influential user is present.

Ideally the weights should be given by a machine learning system. With this, the system would improve the application of weights over metrics in order to retrieve more accurate results. Not having used machine learning we assign a weight of 70% to recognition, 20% to engagement and 10% to reach. Inside recognition score we assigned 60% to retweets, 30% to recommendation and 10% to mentions. To compute the engagement score we assigned 80% weigh to replies given to the user and 20% to replies posted by the user.
Once recognition, engagement and reach scores are determined, we normalize them. It means that we take the highest value in the sample, and we divide all the scores by that value. This normalization process is done because we want to combine (weighted) all 3 scores to determine the global influence score.

Finally, when all the scores are determined, the influence calculation model stores these values in a new database.

5.3 Influence Detection Analysis

In this section we present the results obtained by our model. Apart from this we discuss the results and validate them using 2 methods: one that correlates our model with some existent models; and other that tries to validate the perception of influence, ie comparing with a manual human classification.

The sample used contains data gathered by TwitterEcho from 21^{th} April 2011 to 15^{th} June 2011. This includes more than 2.3 million tweets and more than 63.000 users.

5.3.1 Results

From the experiment concluded with the dataset, we discover that "reply" is the metric with more adopters in the Portuguese community, with 17.250 distinct users replying someone. On the other hand recommendation is the less common action only used by 1340 users. However, that number can be explained due to the fact that recommendation appeared by user adoption and was not recognized by Twitter as an official action.
The amount of different users using hashtags, retweets and mentions is similar to the amount of people that used reply, with 14.246, 12.687 and 14.852 users respectively. It means that only between 1/6 and 1/5 users are really active in Twitter.

Next we present the users, with whom the community interacts the most. In terms of retweets received, we present the top 10 users in table 5.1. Those users can be considered producers of interesting content. It means that the content produced by them, is considered relevant and decide to spread it along the social network. With this, the information is propagated maintaining the source of the content. In the list we present, it is possible to identify bot accounts, starting by the first one, Publico. Publico is a Portuguese traditional newspaper. Its Twitter account only spreads through *twittosphere* the news that also are in the online version of the newspaper. In the top 10 retweeted there are also 2 more newspapers accounts not controlled by humans. This is to show that, you do not need to be actively publishing content to receive retweets, but if the content produced is interesting and with quality, you will likely receive many retweets.

Table 5.1: Most retweeted users

Screen_name	**Number of retweets**
Publico	2722
ChicodeOeiras	2166
fcancio	1882
PauloQuerido	1787
dita_dura	1408
itwitting	1301
G_L	1191
pedroaniceto	1185
supirinho	1161
JornalNoticias	1140

Regarding mentions, in 5.2 we present a table with the most mentioned users. The discrepancy between those users is huge with the most mentioned user with 8318 mentions and the the 10^{th} with 933 mentions. People with high popularity and whose contribution to community is considered relevant receive mentions from the other members. To be on the top 10 mentioned users you must be totally engaged with the community and produce quality content.

Table 5.2: Most mentioned users

Screen_name	**Number of mentions**
jamesdrodriguez	8318
egomesduarte	4122
ChicodeOeiras	1449
luis_grave	1385
supirinho	1295
pedroaniceto	1117
sandraalmeida	1036
vivi_marta	978
fcancio	943
pulsar_ana	933

In what concerns the reply count, we present in 5.3 the 10 most replied users. A reply can be seen as a comment to a message. So, in order to receive many replies an user must publish content that would generate activity. For instance, the most replied message in our database, has 33 replies and the content was a poll questioning: "What is the object immediately to the left of the computer?".

Table 5.3: Most replied users

Screen_name	**Number of replies**
ChicodeOeiras	3594
pulsar_ana	3528
BUGabundo	3044
fcancio	2873
brunum	2487
luis_grave	2277
pedroaniceto	2239
supercurlystic	2239
jamesdrodriguez	2176
supirinho	2145

The recommendation is a metric introduced in the model, because of its expansion and because there is no Twitter official action to recommend a user. The list with the most recommended users is in 5.4.

Apart from these results, regarding the number of actions a user has received, we present here the final results of the influence measurement algorithm. Since we are measuring 3 different kinds of influence in the next 3 tables we present the 10 most influential

Table 5.4: Most recommended users

Screen_name	Number of recommendations
MCeuRMP	262
jallima	230
Tisanas	213
kizapac	192
ChicodeOeiras	127
supirinho	127
joaosodre	119
tropicodelisboa	119
TudoAoContrario	119
sandraalmeida	116

users in relation to recognition, engagement and reach. Note that these scores are normalized. The normalization is doing by simply dividing all the scores by the higher score. Finally we present a table 5.8 with the final results of the global influence score, which is a weighted combination of the latter 3 scores.

Table 5.5: Recognition score

Screen_name	Recognition score
Publico	100
PauloQuerido	55
itwitting	26.92
JornalNoticias	21.33
pedroaniceto	18.17
fcancio	14.95
omalestafeito	13.48
jamesdrodriguez	12.72
dita_dura	12.32
ChicodeOeiras	11.96
JMF1957	11.84
SICNoticias	11.38
supirinho	9.02
der_terrorist	8.99
cvazmarques	7.62
tvi24_iol	7.14
cmjornal	6.93
ExpressoOnline	6.87
AldaTelles	6.08
G_L	5.73

Users with high recognition score 5.5, represent users that provide interesting content or breaking news to the community. It is very common to find newspapers, journalists and, especially, good communicators.

Table 5.6: Engagement score

Screen_name	Engagement score
ChicodeOeiras	99.97
pulsar_ana	98.14
BUGabundo	84.67
fcancio	79.92
brunum	69.18
luis_grave	63.34
pedroaniceto	62.28
supercurlystic	62.28
jamesdrodriguez	60.53
supirinho	59.67
sophieecore	57
_Andrea_D	55.74
Wonderm00n	55.13
trainmaniac	51.66
G_L	51.15
vivi_marta	50.51
vidademigalhas	49.6
isv5	48.18
MiGoncalves7	46.2
FatimaRD	45.59

People with high engagement score are totally engaged with the community, generating a lot of activity around them and their posts.

A high reach score, is associated to popular people from sports, music and comedy. This represents, the popularity of one people and not the influence. Thus, as we saw before, the bigger the audience of one user, more chances he has to be influential.

5.3.2 Validation

In order to validate our model we proceed with a series of commonly used methodologies. The first one consists of measuring the correlation between our results and some algorithms that, supposedly, do the same task.

So, for the top 20 most influent users according to our model, we measured their influence using Klout 3.2.1, Twitalyzer 3.2.2 and TweetGrader 3.2.3. After that we ordered the users by their score and produced 4 different ranks, one for each algorithm. These ranks are represented in table 5.9.

Having the rank distribution in several algorithms, we calculated a Spearman Rank correlation [Wei11] and Pearson correlation coefficient [otWoE]. These measures were to determine how correlated is the model developed with the other existent models and algorithms. So, first we determine the correlation between the 3 different algorithms and

Table 5.7: Reach score

Screen_name	Reach score
cristiano	100
cacelico	4.49
R9FALCAO	2.92
zerohora	2.55
MiKe1155	2.37
simao20sabrosa	1.96
portugalnet	1.35
IgorTiago_h	1.29
katyperrynet	1.2
genevievennaji1	1.16
DiscoLee	1.11
apombalivre	1.08
jamesdrodriguez	1.05
fguarin13	1.03
webmilionario	0.9
WelitonGaspar	0.9
Corpodormente	0.78
soccerportugal	0.78
fernandoalvim	0.73
funnyhumour	0.71

Table 5.8: Top influential users

Screen_name	Global influence score
Publico	80.48
PauloQuerido	49.51
ChicodeOeiras	27.56
fcancio	26.35
pedroaniceto	25.75
itwitting	21.83
jamesdrodriguez	21.1
pulsar_ana	19.42
supirinho	17.96
dita_dura	17.86
BUGabundo	17.48
JornalNoticias	17.33
omalestafeito	16.29
brunum	14.32
luis_grave	14.22
Wonderm00n	13.81
G_L	13.79
supercurlystic	13.47
vivi_marta	11.73
JMF1957	11.01

Table 5.9: Top users rank in different algorithms

Screen_name	**Mine**	**Klout**	**Twitalyzer**	**TweetGrader**
Publico	1	3	3	2
PauloQuerido	2	2	2	1
ChicodeOeiras	3	4	4	14
fcancio	4	6	7	10
pedroaniceto	5	8	5	9
itwitting	6	11	9	5
jamesdrodriguez	7	1	1	8
pulsar_ana	8	20	20	20
supirinho	9	12	10	12
dita_dura	10	8	8	3
BUGabundo	11	15	16	19
JornalNoticias	12	7	6	4
omalestafeito	13	14	19	7
brunum	14	19	18	15
luis_grave	15	13	12	18
Wonderm00n	16	5	15	13
G_L	17	10	11	11
supercurlystic	18	16	14	17
vivi_marta	19	18	13	16
JMF1957	20	17	17	6

we get the results presented in table 5.10.

We find that Klout and Twitalyzer have a very tight correlation. It can be explained because Twitalyzer uses the Klout score to determine its own score. Between those two and TweetGrader the correlation is weaker but still a positive correlation.

Table 5.10: Rank correlation

Algorithms	**Pearson**	**Spearman**
Klout-Twitalyzer	0.75	0.86
Klout-TweetGrader	0.36	0.59
Twitalyzer-TweetGrader	0.33	0.57

After that, we determined the Pearson and Spearman Rank correlation for our algorithm and the other 3. We obtained the results presented in 5.11.

The results obtained represent a weak correlation between our algorithm and TweetGrader, but still a positive correlation. Regarding Twitalyzer, the tool that claims to be the most complete Twitter metric analysis tool, we obtained a high correlation between it and our model. As Twitalyzer is also based on Klout, the correlation between Klout and our model is high as well. There are some factors that can explain not having better re-

Table 5.11: Correlation between our model and others

Algorithms	**Pearson**	**Spearman**
Mine-Klout	0.40	0.63
Mine-Twitalyzer	0.46	0.68
Mine-TweetGrader	0.16	0.40

sults. For instance, both in Klout and Twitalyzer, "jamesdrodriguez" pointed as the most influential between the users in the sample using data published by Portuguese users, and "jamesdrodriguez" is a Colombian football player that plays for F.C. Porto. What we measure is his influence within the Portuguese community. What Twitalyzer and Klout measures is his global influence around the world, which we believe can be even bigger in Colombia than in Portugal. Other factor that can lead to a deviation in the ranks is the information that they use. We computed the influence values within a 2 month sample. Klout, for instance, was launched in September 2009, which means that the data they collected since then is much more complete than the one we are using. This kind of validation allows us tho validate our model in terms of what is more advanced nowadays in influence calculation.

However, another validation was made. In this we try to validate our model in terms of influence perception. For that we sent a survey to 4 people with different ages and education level. In this survey we ask people to, given two Twitter users, chose the one they consider to be the most influential. This decision is based on the public profile of users and the intuition of the person answering the survey.
The survey consists in 10 random pairs of users from the final output of the Influence Model. This information is represented in the next table 5.12. Each row represents the comparison we made between two users, their rank and the difference between the ranks.

The rank represents the position that users occupy in the output list of our Influence Model. From all the answers we receive only 2 people said that "fguarin13" is more influential than "jonasnuts". Although those 2 users do not have a high value of rank difference, we are comparing a Portuguese user with a Colombian player of F.C. Porto. The results obtained for this pair, could be triggered by a mixture of concepts: popularity vs influence. On the other hand, when comparing those users, for instance, by Klout score we find that "fguarin13" has higher score than "jonasnuts". However, our model only uses the Portuguese dataset to output its final scores, which means that for the Portuguese community "jonasnuts" is more influential and for Worldwide community "fguarin13" is more influential.

In a general way, the results obtained provide a preliminary evaluation of our model. In chapter 6 we propose future evaluation experiments.

Table 5.12: Information about the validation survey

user 1	**rank**	**user 2**	**rank**	**rank difference**
pedroaniceto	5	claudiapinto_	53218	53213
Gaiathynaikos	15052	omalestafeito	13	15039
supercurlystic	18	minawolf	1859	1841
fcancio	4	BaiaVieira	426	422
JoseDePina	369	apombalivre	151	218
fguarin13	250	jonasnuts	41	209
itwitting	6	kamponez	85	79
ChicodeOeiras	3	Pinilla	45	42
abolapt	43	JornalNoticias	12	31
PauloQuerido	2	cvazmarques	32	30

Chapter 6

Conclusions and Future Work

In this chapter we draw conclusions about the TwitterEcho system, with more incidence over the clients, and about the results of the Influence Model.

Regarding TwitterEcho, we conclude that the system is robust and efficient enough to crawl the entire Portuguese community on Twitter. With some adjustments in the server and with more clients, we can aspire to collect data from a bigger community than the Portuguese one. Future work would be developing more types of client, beyond the lookup and links. Some examples are retrieving information given a certain keyword.
For instance, we give the client the words "elections" and "portugal". It would search for tweets related to those keywords and retrieve information about users that tweeted about those topics. Another example would be, retrieving information about trends. This means that, for instance, retrieving information about the users who post messages related to the trends in that moment.
We concluded from the results of this crawling system, that the active Portuguese community on Twitter is not that big, with about 20.000 relatively active users. In the present thesis we confirmed what was already confirmed to the USA Twitter community: only 10% of the users produce 90% of whole content. Also, 2009 was the *boom* year of Twitter in Portugal and is expected a decreasing of the number of subscriptions in the next years.

Future work in the influence model should be on weighting of the scores and validation and evaluation. The validation methods used in this thesis proved that the model mimics well the results of other algorithms with the same mission and it also obtained very good results regarding the influence perception.
However, some additional validations should be made. For instance, a good method to do this will be having the community doing the job for us. In other words, a recommendation system can be developed, based on the influence model. For instance, for each topic we

select a set of influential users, then we recommend those users to people who have the same interests. Then we could measure the "dropout rate", which represents the percentage of people that gave up to follow a recommended user. If people to be recommended are really influential, the user will most likely not give up the following relationship.
Another method that could be used is by incorporating a rate system in Twitter, in which Twitter user would be allowed to rate other users by their influence. However, this method would be susceptible to cheat possibility and would not be representative. To evaluate the method would be necessary to create a reference graph. This means we would need a graph with the users and with a manual human classification. This would be material to another master thesis.

Beyond validation and evaluation of the model, there are some aspects that, if implemented in our model, would simulate with better accuracy the dynamics of influence propagation in a social network. The model built during this dissertation is already prepared to support some of these changes. The most important one, as was referred in 3.1.3.2, is the concept of relative influence. This means to measure the influence user A exerts in user B, for instance. In the current model, we are only measuring the global influence of an user. However, the metrics stored allow that kind of measurement. For each user we are storing the screen_name and the number of times other users realized an action towards him.
In the current model, we are considering all mentions equally. But we know that is not correct. For instance if #1 rank mentions me it would be more valuable than if #5000 rank user mentions me. For that reason the first time the algorithm runs, it would ignore this factor. Thereafter, it would introduce the rank of the user doing an action in the influence calculations.

Another interesting study would be trying to improve TwitterRank [WLJH10]. It means, instead of using the follower and tweet count we still use our influence measuring model: with more incidence over the recognition and engagement score. After we measure influence for all the users, we would use the TwitterRank paradigm: "the influence of an user is dependent of the influence of his followers". This would be valuable to compare with the actual model and confirm what was stated in [YW10].

References

[ALTY08] Nitin Agarwal, Huan Liu, Lei Tang, and Philip S. Yu. Identifying the influential bloggers in a community. *WSDM*, February 2008.

[AMW10] Norhidayah Azman, David E. Millard, and Mark J. Weal. Issues in measuring power and influence in the blogosphere. *Web Science Conf.*, April 2010.

[BAL$^+$09] Jiang Bian, Eugene Agichtein, Yandong Liu, Hongyuan Zha, and Ding Zhou. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. *WWW*, April 2009.

[BE09] Alex Burns and Ben Eltham. Twitter free iran: an evaluation of twitter's role in public diplomacy and information operations in iran's 2009 election crisis. *Communications Policy & Research Forum 2009*, November 2009.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[CHBG10] Meeyoung Cha, Hamed Haddadi, Fabrício Benevenuto, and Krishna P. Gummadi. Measuring user influence in twitter: The million follower fallacy. *Association for the Advancement of Artificial Intelligence*, 2010.

[CLH$^+$10] Dan Cosley, Xiangyang Lan, Daniel Huttenlocher, Siddharth Suri, and Jon Kleinberg. Sequential influence models in social networks. *Association for the Advancement of Artificial Intelligence*, 2010.

[CPA11] CPAN. Cpan homepage. http://www.cpan.org/, 2011.

[CWY09] Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. *KDD*, June 2009.

[DFS10] Anna Daniel, Terry Flew, and Christina Spurgeon. The promise of computational journalism. *Media Democracy and Change: Refereed Proceedings of the Australian and New Zealand Communications Association Annual Conference*, July 2010.

[GBL10] Amit Goyal, Francesco Bonchi, and Laks V.S. Lakshmanan. Learning influence probabilities in social networks. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 241–250, New York, NY, USA, 2010. ACM.

[Gil04] Kathy E. Gill. How can we measure the influence of the blogosphere? *WWW*, May 2004.

[HP09] Bill Heil and Mikolaj Piskorski. Harvard business review - new twitter research: Men follow men and nobody tweets. http://smub.it/p1z, 2009.

[JSFT07] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, WebKDD/SNA-KDD '07, pages 56–65, New York, NY, USA, 2007. ACM.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. *SIGKDD*, 2003.

[Klo11] Klout. Klout homepa. http://klout.com, 2011.

[KLPM10] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591–600, New York, NY, USA, 2010. ACM.

[LTH$^+$10] Lu Liu, Jie Tang, Jiawei Han, Meng Jiang, and Shiqiang Yang. Mining topic_level influence in heterogeneous networks. *CIKM*, October 2010.

[Mac10] L. Machado. Ensitel, a public relations case study? http://lindamachadotweets.wordpress.com/2010/12/28/ensitel-a-public-relations-case-study/, 12 2010.

[NC08] Ramesh Nallapati and William Cohen. Link-plsa-lda: A new unsupervised model for topics and influence of blogs. *Association for the Advancement of Artificial Intelligence*, 2008.

[Oli11] Eduardo Oliveira. Twitterecho: crawler focado distribuído para a twittosfera portuguesa. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2011.

[otWoE] University of the West of England. Pearson's correlation coefficient. http://hsc.uwe.ac.uk/dataanalysis/quantInfAssPear.asp.

[Ras06] Lisa Rashotte. Social influence. In Ritzer G. (eds.) Blackwell Encyclopedia of Sociology, 4426- 4429, 2006.

[Raz09] Razorfish. Razorfish proposes social influence score. http://www.clickz.com/clickz/news/1712717/razorfish-proposes-social-influence-score, 2009.

[REA11] REACTION. Project reaction wiki. http://xldb.lasige.di.fc.ul.pt/wiki/Reaction, 2011.

[RGAH09] Daniel M. Romero, Wojciech Galuba, Sitaram Asur, and Bernardo A. Huberman. Influence and passivity in social media. *HP Labs*, 2009.

[Tec11] Technoratic. Technoratic homepage. http://technorati.com/, 2011.

[Twe11] TweetGrader. Tweetgrader homepage. http://tweet.grader.com/, 2011.

[Twi11a] Twitalyzer. Twitalyzer homepage. http://www.twitalyzer.com/, 2011.

[Twi11b] Twitter. Twitter-api documentation. http://dev.twitter.com/doc, 2011.

[Twi11c] Twitter. Twitter homepage. http://twitter.com, 2011.

[Wei11] Eric W. Weisstein. Spearman rank correlation coefficient. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/SpearmanRankCorrelationCoefficient.html, 2011.

[WLJH10] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 261–270, New York, NY, USA, 2010. ACM.

[YW10] Shaozhi Ye and Felix Wu. Measuring message propagation and social influence on twitter.com, 2010.