PLN Curs 2011-2012 Partial exam

A possible solution is the following:

Recent works have been focused on the increasing relevance of the social networks in the public image of politicians and political parties. Several of them have been focused on the opinion on candidates to presidency and on how this opinion changes.

We are planning to develop an application accessing a set of tweets containing the name of a specific politician and for each tweet determine if it is about the politician and if it is positive or negative. That is, the application input will be the name of a politician and his web page URL address, the output will be the tweets referred to the politician, indicating for each of them, if the politician valoration is positive or negative.

The problem is not simple, it presents several challenges. First, a name may identify more than one person, as in the following examples. Georges Bush is the name of previous president of the USA, but also his father's name. Clinton is the name of the Secretary of State in USA, Hillary, but also the name of the ex-president Bill. Zapatero is the name of the previous president of Spain, but is also the name of a profession (shoes maker).

Tweets also present several difficulties when applying natural language tools because text in them is very short, usually without capital letters, with lots of abbreviations, colloquial forms and special characters, such as emoticons.

Figure 1 presents several examples of Tweets. Twitter include additional meta-information that can be useful, such as the identifier, author and date.

Sample Tweets
Indians game tomorrow night at Victory Feild 7pm; come to Friday's b-4 or after to enjoy some amazing drinks, see you there!!
Mmm subway....
mummma bought me a subway :D
ii want some subway asap...!!!
@user1 <333 get me subway!
Ac milan RT @user2 @user3 Manchester United °°
Support.me.and.my.art....on."COAST.to.COAST".at.
PARIS-MILAN-ISTAMBUL.tour...than ks...luv.u...
#nowplaying - Scorpions: Always somewhere
scorpions - sly
Yeeeew and aaaakh, I found two little (thank God)
scorpions in my room tonite! : O
They tryna raise the MTA fares again smh
snooooooooooooooo-ooooop!! Was anybody else scared 2
sing that song pre-delta lol
how did she go? arik? or delta? cos theyre the ones that fly direct
apollo is toooo cute :)
Guna bck at the apollo

Figura 1: Examples of tweets

We ask you:

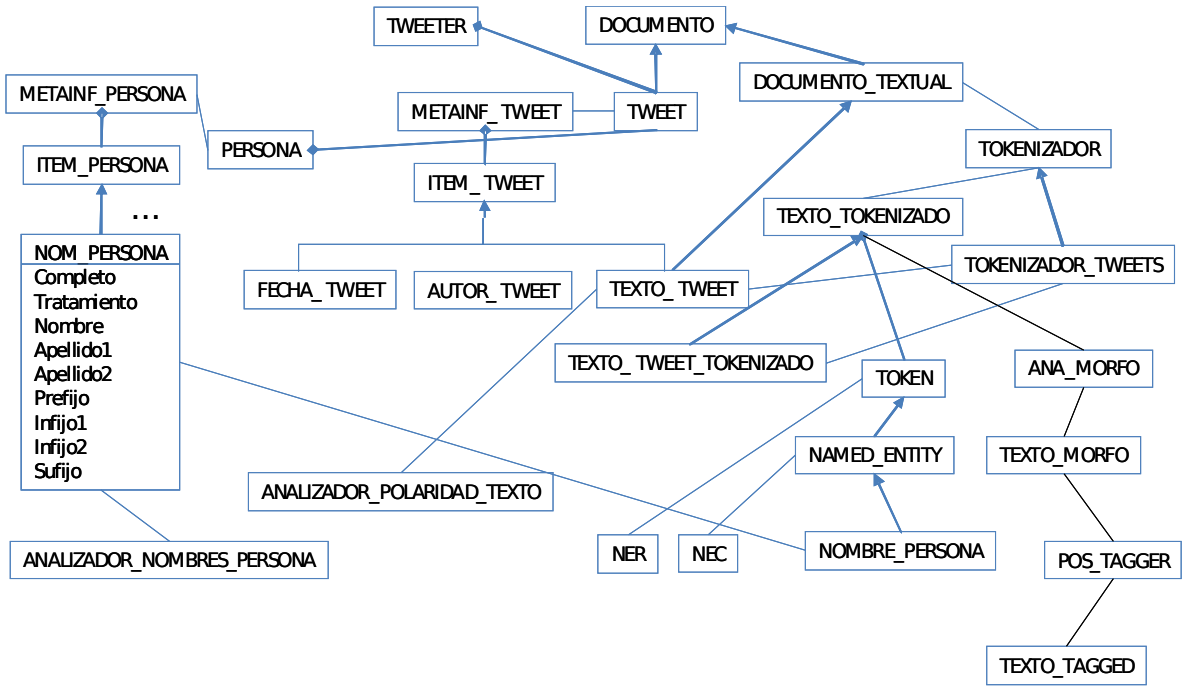
1. A list with main problems the processing of tweeds present, the previous introduction presents several of the problems, you have to complete them.

Answer:
Several of the most relevant problems are the following:

- Use of emoticons: :), :D
- Use of abbreviations: bck
- Repeating vocals: Yeeeeew, tooooo
- Onomatopoeia: aaaakh
- Colloquial forms Mum, Guna, asap
- New words (generated from combination of previous):tryna, tonite
- Irregular use of : !, ! !, ..., -
- Codes used as words b-4, @user2, <333, ºº
- Interjections not use in formal text Yeeeeew
- Most sentences have mistakes
- Specific writing for proper names:milan, PARIS-MILAN-ISTAMBUL
- Many tweets are continuations of previous tweets.

2. A representation in UML formalism of the main domain and application concepts

Answer:



We have represented the class PERSONA(PERSON) having the meta-information , METAINF_PERSONA that contains ITEM_PERSONA, being one of the items NOM_PERSONA (NAME_persona). Nom_persona is described by a set of attributes: complete_name, title (tratamiento), first name (nombre), last name (Apellido 1), etc.

We have represented the class TWEETER that consists of a collection of TWEETs. Each TWEET includes meta-information (METAINF_TWEET) that consists of the author (AUTOR_TWEET), the date (FECHA_TWEET) and the text (TEXTO_TWEET). TWEET is a subclass of the class DOCUMENTO (DOCUMENT). TEXTO_TWEET (the text of the tweet) is a subclass of the class DOCUMENTO_TEXTUAL (TEXTUAL_DOCUMENT).

The class PERSONA (PERSON) has associated a collection of elements in class TWEET, that consist of tweets containing its name.

We have also represented a several processors involved in the analysis of tweet text. First step in the text processing is called tokenization and consists of segmenting the text (belonging to the class DOCUMENTO_TEXTUAL) into words and sentences. This process is done by a tool belonging to the class TOKENIZADOR (tokenizer) and the resulting text belongs to the class TEXTO_TOKENIZADO (tokenized text). Probably, conventional tokenizer could not be applied and a specific tokenizer for tweeds will have to be developed. For this reason, we have defined the subclass TOKENIZADOR_TWEETS (TWEET-TOKENIZER) that tokenizes instances of the class TEXTO_TWEET (TWEET-TEXT), resulting instances of the class TEXTO_TWEET_TOKENIZADO (TOKENIZED-TWEET-TEXT). This specific tokenizor for tweets will not deal with the task of segmenting the text in sentences because tweets usually consists of only one sentence (maxim length is 140 characters).

The class TEXTO_TOKENIZADO (TOKENIZED-TEXT) includes elements of the class TOKEN. An specialization (subclass) of the class TOKEN is NAMED_ENTITY. The task of detecting names and organizations is called named entity detection. This process is performed by a Named Entity Recognizer (NER), that takes as input a token and determines if it is a named entity. The processor that determines the name entity class is a Named Entity Classifier (NEC), that takes as input a named entity. For this application a named entity recognizer for person names have to be included in order to recognize the politician name.

Second step involved in the text processing consists of performing morphological parsing (or analysis), that is the process of finding the constituent morphemes in a word (e.g., cat +N,+PL, for cats). This process is perform by morphological parsers (morphological analyzers). This processor has been represented by the class

ANA_MORFO, that transform instances of the class TEXTO_TOKENIZADO (TOKENIZED-TEXT) in instances of the class TEXTO_MORFO (MORPHO_TEXT, representing the text in which the several morphemes of the words have been analyzed).

The following step in the text processing consists of assigning words the syntactic category (part of speech) category. This task is performed by taggers, represented by the class POS_TAGGER, that takes as input the output of previous step (resulting of the morphological parsing). That is, the processor represented by the class POS_TAGGER transform instances of the class TEXTO_MORFO (MORPHO_TEXT) in instances of the class TAGGED_TEXT.

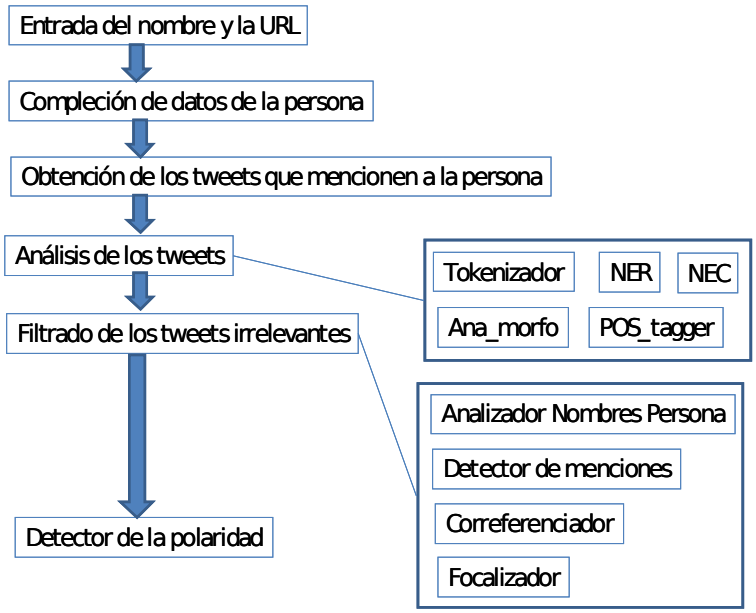
In fact, the morphological parser and the tagger are only needed by the Named Entity Recognizer (NER) and Named Entity Classifier (NEC), for obtaining the politician name.

The following steps, syntactic analysis and semantic analysis are not necessary for this application. They would not be easy to develop, either. What is also really needed is a processor capable to detect the teed polarity (if the opinion contained is positive or negative).

3. You have to propose an architecture of the application, indicating data structures and linguistic processors.

Answer

The functionality of the application will be the following: The user introduces the complete name of the politician and the URL address of the web page containing his description. Using the information in this web page the personal data is completed (nationality, date and place of birth, etc.) and alternative names (i.e., Nike names). Then, a set of tweets about the politician will be collected.



Each of the tweets will then be analyzed, determining the polarity (if favorable or not). Finally an evaluation of all the tweets analyzed and their evolution could be presented to the user.

4. Describe the linguistic processors used in previous question: which knowledge sources will use and how you will get them.

Answer:

Each of the tweets will be analyzed with the following processors:

- Tokenizer. It will be specific for tweets, or at least for social networks. It will have to distinguish between alphabetic tokens and the other token (containing other characters).
- Morphological parser. It will have to include a detector of emoticons, abbreviations and unknown words. We could define a grammar of emoticons (it could be represented by a Finite State Automata (FSA). For detection of abbreviations, a collection of common abbreviations as well as a set of rules could be used.
- NER, NEC adapted to the proper names as they appear in tweets. We will have to consider that capital letters are not used in most tweeds.
- POS_tagger adapted to tweets. An statistic existing tagger, as a HMM, could be adapted to tweeds, using manual tagged tweeds.
- Text segmentators are not appropriate for this application. Language selector is not needed, either.

In order to detect which tweets are relevant (are about the specific politician asked) the following processors will be used:

- Detector of mentions . A name entity recognizer of person names, extended with pronouns.
- Detector of correferences. Any conventional existing one.
- Person names analyzer.
- Focus detector. In order to detect if the tweed refers to the politician we can use a focus detector that gives a specific weight to each token representing a person name. This weight can be directly related to the frequency
- Polarity detector. In order to detect if the tweed express a positive or negative opinion, this detector can use two lists, one containing positive terms and the other negative terms. If positive terms are more than negative, then the

opinion is positive. Those lists can be obtained from the web.

5. One of the processors needed is a detector of person names. It can be built using a finite state automata (FSA): You have to describe how will you build it.

Answer:

It is easy to obtain gazetteers containing proper names for women and men. For each possible name we could built a FSA, calculate their union, determinize the resulting union and minimize it. As a result we will have a recognizer of female proper names recognizer and a male proper name recognizer. Then, from these two recognizers, we have to build a new one to recognize compound proper names. Will have to include preposition and articles (‘’, ‘de’, ‘del’, ‘de la’, etc.). Resulting automata could recognize names like José, José Antonio, José María, María, María José, María de las Mercedes, etc.). We could include also titles (‘Sr.’, ‘Sra.’, ‘Dr.’, ‘Dra.’, ‘Illmo.’, etc.). Finally, we will have to build an automata to recognize last names.