An approach for discovering keywords from Spanish tweets using Wikipedia[b]

Daniel A. Hernández[a], Juan C. Roldán[a], David Ruiz[a], and Fernando O. Gallego[a]

[a]University of Sevilla, ETSI Informática, Avda. Reina Mercedes s/n, Sevilla E-41012, Spain - {dayala1, jcroldan, druiz, fogallego}@us.es

[b]Supported by the Spanish R&D&I program under grant TIN2013-40848-R.

KEYWORD

Twitter, Social Media Analysis, Wikipedia, Keywords Discovery

ABSTRACT

Most approaches to keywords discovery when analyzing microblogging messages (among them those from Twitter) are based on statistical and lexical information about the words that compose the text. The lack of context in the short messages can be problematic due to the low co-occurrence of words. In this paper, we present a new approach for keywords discovering from Spanish tweets based on the addition of context information using Wikipedia as a knowledge base. We present four different ways to use Wikipedia and two ways to rank the new keywords. We have tested these strategies using more than 60000 Spanish tweets, measuring performance and analyzing particularities of each strategy.

1. Introduction

The rapid growth of social networks in the form of user generated content presents new opportunities. Users generate large amounts of uncategorized data in the form of short messages. While ordinary documents from websites provide contents about a specific topic with informational purposes, Twitter messages usually include one or two thoughts about recent events, products, services and brands because of their social component. Users share their points of view with their friends, and give their opinions about competing groups (Xie et al., 2012).

This information is useful for business, because it allows them to get an idea of what people think about their products, services and brands. Community managers must be aware of the opinion of a company or product in order to improve it, and market analysts study the information that could affect the reception and success of products or services, as the popularity of a company compared to its competitors, the opinion about new products, etc. These opinions, which can be found online, have a considerable and increasing impact on consumer interest (Yubo Chen and Wang, 2011; Hennig-Thurau et al., 2014).

The volume of the available information in Twitter makes it impossible to manually observe and analyze the former. Its study is possible thanks to Listening Platforms, which provide analytical information. This information usually includes several statistically relevant keywords (SR-Keywords) that appear with a higher than average frequency. However, Twitter messages have unique characteristics (Hu and Liu, 2012) that cause problems when SR-keywords are used.

There is a context that can not be retrieved with statistical methods but adds information about what is being said. Usually, disambiguation is performed in a supervised way by deducing their context. For example, if we analyze Spanish tweets about the recent regional Andalusian elections (March 22th, 2015) we could obtain as SR-keywords the words *Podemos*, *Andalucía* and *Izquierda*. *Podemos* could mean the emerging political party, or could be the verb *poder*, or even part of a slogan. *Andalucía* could mean the autonomous community or be part of the name of an entity such as *Junta de Andalucía*. *Izquierda*, even knowing the political context, could

refer to left-wing politics or to the of the name of the political party *Izquierda Unida*.

In order to solve this problem, in this paper we present an approach for discovering keywords from Spanish tweets using Wikipedia as a knowledge base to enrich the text with context information. We call these knowledge based keywords (KB-keywords). Furthermore, we show four strategies for using Wikipedia to retrieve relevant knowledge: using each SR-keyword to retrieve Wikipedia articles, using each SR-keyword with the addition of the query given to the listening platform, using the entire tweets set and using several tweets groups. We rank the KB-keywords using two different rankings: measuring the intensity of the relationship with our relevant knowledge and measuring the importance of the concept represented by the KB-keywords.

We present the experimental results of each combination of strategies, including the execution time of each strategy when using different inputs. Results show considerable differences in both performance and KB-keywords. The first two search strategies have a much lower execution time but use less information and require the previous extraction of SR-keywords. The last two ones have higher execution times but use all the available text from the tweets and don't require SR-keywords. The second ranking results in KB-keywords that represent more abstract concepts.

The rest of the paper is organised as follows: Section 2 presents related work, Section 3 describes the details about the functioning of our algorithm, Section 4 presents the experimental results when applying it and Section 5 summarizes our work.

2. Related work

In this section we define the requirements of our approach and compare it to other techniques using a comparison framework.

2.1 Requirements

We have considered the following requirements:

R1 Knowledge presence: Keywords extraction must take into account human knowledge about the analyzed text. This knowledge is stored and used to extract topics that do not appear in the original text.

R2 Completely unsupervised: The solution must not require any additional human input. Text corpora are also excluded, since the only available information is the group of tweets to be analyzed, and not documents (tweets in this case) used to calculate frequencies or other probabilistic data.

R3 Correlation analysis: The extracted terms must reflect the relationships between several words or parts in the original text. The presence of a certain group of words or phrases could have an indirect meaning which would not be present in the individual parts, since when they are grouped they show a context. For example, the words *vehicle* and *fly* used together could refer to planes, which are not mentioned explicitly.

R4 Group of numerous small documents: Unlike most common cases, and due to the maximum tweet length, the analyzed text is divided into a high amount of small documents instead of less big documents.

2.2 Existing techniques

We have identified the most frequent keywords discovery approaches. We also provide several recent use examples that confirm their relevance:

Technique	R1	R2	R3	R4
TF-IDF	No	No	No	Yes
LSA	No	No	Yes	No
Bayesian models	No	Yes	Yes	No
Knowledge based	Yes	Yes	Yes	Yes

Table 1: Comparison framework

TF-IDF ponderation : Let w be a word in a text with N words, $c(w)$ the number of occurrences of w in the former and $IDF(w)$ the inverse document frequency of w (representing its overall frequency), w is given the following weight: $p(w) = \frac{c(w)}{N} \cdot IDF(w)$. TF-IDF is the basis or point of reference for many document classification techniques (Zhang et al., 2011; Ko, 2012; Ren and Sohrab, 2013).

Latent Semantic Analysis : LSA consists in representing a document as a keywords documents matrix with TF-IDF weights. A single value decomposition (SVD) of this matrix results in three different matrices. Each of these matrices provides different statistical information about important words, topics and sentences (Zhang et al., 2011; Thorleuchter and Van den Poel, 2013; Thorleuchter and Van den Poel, 2012).

Bayesian models : The text input is used to create several probabilistic distributions of its words. The probability of each word is used as its weight. Words are probabilistically related in order to create a model used as classificator (Blei, 2012; Chen et al., 2012; Zhu et al., 2014).

Knowledge-based : An external source of knowledge (knowledge base) is used to enrich the text with relevant additional information (Chen et al., 2013; Hulpus et al., 2013). There already exist techniques that use Wikipedia as a source of knowledge, but while those approaches are based on the retrieved articles, ours is based on the linked articles. We also define a framework where different strategies can be used.

2.3 Comparison framework

Table. 1 shows a comparison between the aforementioned techniques. Our approach meets the established requirements, which the other techniques partially fail. The reasoning behind this evaluation is the following:

R1 Knowledge presence: Other approaches are mainly based on the frequence of occurrence of words or phrases in documents. This limits them to selecting words from the text they analyze.

R2 Completely unsupervised: TF-IDF needs a background corpus to calculate the IDF. LSA uses TF-IDF frequencies. Bayesian models do not need this kind of corpus and are unsupervised (Nenkova and McKeown, 2012).

R3 Correlation analysis: TF-IDF scores each word individually, without analyzing the relationships among them. LSA reflects relationships between words by assigning them to topics. Bayesian models connect words in order to create networks.

R4 Group of numerous small documents: TF-IDF can be applied to the total of words in the document. LSA and Bayesian models create structures based on the number of phrases or individual documents analyzed, which would be problematic when dealing with a relatively large number of documents, as is the case of social media.

The other techniques do not meet some of our requirements because they have not been developed with Social Media Analysis in mind, and are geared towards traditional documents analysis. Text from Twitter have unique characteristics that create new problems (Hu and Liu, 2012).

Documents usually have a small size and are limited to a low amount of characters. Short messages consist of a few sentences, with no context. Consequently, it is difficult to obtain a reliable similarity measure, and data matrices, which are essential for text processing, are sparse.

It is difficult for traditional models to find connections between keywords. Two messages could be related to the same topic, but not contain any shared word. Only by enriching the available information with semantic knowledge from an external source we can reveal their relationship.

Twitter also has a real-time nature, and its content is constantly updated. Twitter users post news and information several times each day to a point where almost any query will return messages from minutes or seconds ago. This results in a greater amount of documents.

Knowledge based models solve the lack of context by adding additional knowledge and selecting keywords from it.

3. Proposal

In this Section we first define the relevant concepts and elements of the vocabulary that we use to describe our approach.

3.1 Vocabulary and notation

Word : A single written unit of language. We denote them by $w_1, w_2, w_3...$ The finite set of words is denoted by $Words$.

Example: $w_1 = Podemos, w_2 = consigue, w_3 = la$.

Tweet : Sequence of words written by a Twitter user. We denote them by $t_1, t_2, t_3...$ The finite set of tweets is denoted by $Tweets$.

Example: $t_1 = [w_1, w_2, w_3, w_4...]$ = [*Podemos, consigue, la, cobertura, informativa, de, TVE, para, las, elecciones, andaluzas, como, el, partido, del, cambio, que, es]* (*Podemos achieves the media coverage of TVE for the Andalusian elections as the party of change it is*).

SR-Keyword (Statistically Relevant Keyword) : Sequence of words statistically relevant in a collection of tweets. We denote them by $k_1, k_2, k_3...$ The finite set of SR-keywords is denoted by $SR\text{-}Keywords$.

Example: Let $\{t_1, t_2, ..., t_n\}$ be a set of tweets, we extract from them the following set of SR-keywords: $\{k_1 = [w_1] = [$*Elecciones*] (*Elections*), $k_2 = [w_2, w_3] = [$*Partido, político*] (*Political, party*)$\}$.

KB-Keyword (Knowledge-Based Keyword) : Sequence of words that summarize a collection of tweets enriched with human knowledge, representing important ideas in that knowledge. We denote them by $kbk_1, kbk_2, kbk_3...$ The finite set of KB-keywords is denoted by $KB\text{-}Keywords$.

Example: Let $\{t_1, t_2, ..., t_n\}$ be a set of tweets, we extract from them the following set of KB-keywords: $\{kbk_1 = [w_1, w_2, w_3] = [$*Podemos, (partido, político)*$]$, $kbk_2 = [w_4, w_5, w_6] = [$*Pablo, Iglesias, Turrión*$]\}$.

Query : Sequence of words, usually of small length, given by a user to retrieve relevant information. It includes typical operators of information retrieval systems, such as logical operators. We denote them by $q_1, q_2, q_3...$ The finite set of queries is denoted by $Queries$.

Example: $q_1 = [w_1, w_2, w_3] = [$*Podemos, AND, elecciones*$]$.

Listening Platform : Let $Queries$ be the set of Query $Tweets$ the set of Tweet and $SR\text{-}Keywords$ the set of KB-Keyword, we define a Listening Platform as a function with the following domain and range: $f : Queries \rightarrow \mathcal{P}(Tweets) \times \mathcal{P}(SR\text{-}Keywords)$. We denote them by lp_1, lp_2, lp_3... The finite set of listening platforms is denoted by $ListeningPlatforms$.

Example: Opileak[1], Trackur[2] or Sysomos[3].

Knowledge base : Written data that reflects human knowledge. Different concepts or ideas are separated and stored in a knowledge source. We denote them by kb_1, kb_2, kb_3... The finite set of knowledge bases is denoted by $KnowledgeBases$.

Example: Articles in Wikipedia. For example, the articles *Elecciones* (*Elections*), *Podemos (partido político)* (*Podemos (political party)*) and *Pablo Iglesias Turrión*.

Document : Sequence of words that represent information about a concept or idea from a knowledge base. We denote them by d_1, d_2, d_3... The finite set of documents is denoted by $Documents$.

Example: Given the listening platform *Wikipedia*, we extract from it the following set of documents: $\{d_1 =$*Podemos (partido político)*$, d_2 =$ *Elecciones*$, d_3 =$*Pablo Iglesias Turrión*$, ...\}$.

Knowledge system : Let $Queries$ be the set of Query and $Documents$ the set of Document, we define a knowledge system as a function with the following domain and range: $f : Queries \rightarrow \mathcal{P}(Documents)$. We denote them by ks_1, ks_2, ks_3... The finite set of knowledge systems is denoted by $KnowledgeSystems$.

Example: A Lucene index with documents that correspond to Wikipedia articles.

Search strategy : Let $Queries$ be the set of Query, $Tweets$ the set of Tweet and $SR\text{-}Keywords$ the set of SR-Keyword, we define a search strategy as a function with the following domain and range: $f : Queries \times \mathcal{P}(Tweets) \times \mathcal{P}(SR\text{-}Keywords) \rightarrow \mathcal{P}(Queries)$. We denote them by se_1, se_2, se_3.... The finite set of search strategies is denoted by $SearchStrategies$.

Example: A strategy that creates a query per SR-keyword, using its words as the words of the query.

KB-keywords ranking : Let $KB\text{-}Keywords$ be the set of KB-Keyword and $Documents$ the set of Document, we define a KB-keywords ranking as a function with the following domain and range: $f : KB\text{-}Keywords \times \mathcal{P}(Documents) \rightarrow \mathbb{R}$. We denote them by ra_1, ra_2, ra_3... The finite set of scoring strategies is denoted by $Rankings$.

Example: A ranking that gives a score equal to the number of times a KB-keyword occurs in a collection of documents.

Ranked KB-keyword : Let kbk, s be a KB-Keyword and a real number, we define a ranked RK-keyword as the tuple (kbk, s). We denote them by rk_1, rk_2, rk_3... The finite set of ranked KB-keywords is denoted by $RankedKB - Keywords$.

Example: A tuple with the values $($*Elecciones*$, 3.14)$.

[1] http://www.opileak.es/
[2] http://www.trackur.com/
[3] https://sysomos.com/

3.2 Our approach

We have developed Alg. 1 to discover KB-keywords from tweets and SR-keywords given by a listening platform and the query used to extract them. This algorithm uses our search strategies and rankings. $createQueries$ is an algorithm that creates queries using search strategies, $askKnowledgeSystem(q)$ is an algorithm that queries the knowledge system using Query q, $calculateKB\text{-}keywords(D)$ is an algorithm that selects unranked KB-keywords from each set of Document in D, and $relationshipsRanking(k, D_i)$ and $importanceRanking(k, D_i)$ are algorithms that assign a score to KB-Keyword k using the set of Document D_i it has been calculated from, according to two different rankings.

Algorithm 1 KB-keywords Extractor

Input
  $initialQuery$: Query
  $tweets$: Tweet
  SR: Set of SR-Keyword
Output
  (RR_1, RR_2, RR_3, RR_4): Tuple of sets of Ranked KB-keyword using Relationship Ranking
  (IR_1, IR_2, IR_3, IR_4): Tuple of sets of Ranked KB-keyword using Importance Ranking
Variables
  D: Tuple of sets of Document (D_1, D_2, D_3, D_4)
  Q: Tuple of sets of Query (Q_1, Q_2, Q_3, Q_4)
  KB: Tuple of sets of KB-keyword (KB_1, KB_2, KB_3, KB_4)
  $score$: $\mathbb{R}$

– Initialization of tuples with empty sets
– Step 1: Using our search strategies the queries are created
$Q \leftarrow createQueries(initialQuery, tweets, SR)$
– Step 2: The knowledge system is queried to retrieve documents
for $i \leftarrow 1$ **to** 4 **do**
  for all $q \in Q_i$ **do**
    $D_i \leftarrow D_i \cup askKnowledgeSystem(q)$
  end for
end for
– Step 3: KB-keywords are obtained from the documents
$KB \leftarrow calculateKB\text{-}keywords(D)$
– Step 4: The KB-keywords are ranked
for $i \leftarrow 1$ **to** 4 **do**
  for all $k \in KB_i$ **do**
    $RR_i \leftarrow RR_i \cup \{(k, relationshipsRanking(k, D_i))\}$
    $IR_i \leftarrow IR_i \cup \{(k, importanceRanking(KB_i, D_i))\}$
  end for
end for

A knowledge system allows us to use queries and obtain sorted documents (Wikipedia articles in our case), and a search strategy determines how these queries are created. We have devised four different search strategies that create several queries to perform searches whose results are used to create a single, final collection of articles:

SE1 Keywords: We use every original SR-keyword (given by the listening platform) as a query to obtain Wikipedia articles as shown in Alg. 2 (Search strategy 1). Each search using the queries returns several scored documents. We discretize these documents using the scores given to them by the knowledge system and keep only the first group, so that only the most relevant documents remain, without including documents with a much lesser score or excluding documents which have a relatively good score.

Algorithm 2 Create Queries

Input
  $initialQuery$: Query
  SR: Set of SR-keyword
  $tweets$: Set of Tweet $(t_1...t_n)$
Output
  Q: Tuple of sets of Query (Q_1, Q_2, Q_3, Q_4)
Variables
  $words$: Set of Word
  $numberGroups$: $\mathbb{N}$

$Q \leftarrow (Q_1 = \emptyset, Q_2 = \emptyset, Q_3 = \emptyset, Q_4 = \emptyset)$
$words \leftarrow \emptyset$
for all $k \in SR$ **do**
  – Search strategy 1
  $Q_1 \leftarrow Q_1 \cup \{k\}$
  – Search strategy 2
  $Q_2 \leftarrow Q_2 \cup \{(k \cup initialQuery)\}$
end for
– Search strategy 3
for all $t \in tweets$ **do**
  $words \leftarrow words \cup t$
end for
$Q_3 \leftarrow Q_3 \cup \{optional(words)\}$
– Search strategy 4
for all $tGroup \in groupTweets(tweets)$ **do**
  $Q_4 \leftarrow Q_4 \cup \{optional(tGroup)\}$
end for

Example: Using the query *Podemos elecciones* (*Podemos elections*) we obtain, among others, the SR-keyword *Elecciones andaluzas* (*Andalusian elections*). Using this SR-keyword we query the knowledge system and obtain a set of documents. After the discretization we keep the documents shown in Tab. 2.

SE2 Keywords plus query: This strategy is the same as the Keywords strategy, but when creating queries with each keyword we add the original query text given to the listening platform as additional terms, as shown in Alg. 2 (Search strategy 2). This way we keep the results from being too unrelated to the original query.

Example: Using the same query, *Podemos elecciones*, and the same keyword, *Elecciones andaluzas*, we perform a search, adding the query to the keyword. The search input would then be *Elecciones andaluzas Podemos elecciones*. Using this input we obtain the documents shown in Tab. 3.

Result	Score
Asamblea Nacional de Andalucía	1.47
Frente Andaluz de Liberación	1.43
Liberación Andaluza	1.35
Coalición Andalucista	1.30

Table 2: Keywords strategy - elecciones andaluzas search results

Result	Score
Coalición Andalucista - Poder Andaluz	0.99
Frente Andaluz de Liberación	0.96
Asamblea Nacional de Andalucía	0.95
Liberación Andaluza	0.86
Coalición Andalucista	0.85

Table 3: Keywords plus query strategy - elecciones andaluzas podemos elecciones search results

SE3 Tweets group: This strategy consists in using the information found in Twitter, instead of the SR-keywords that represent it.

We join all the tweets and create a single text document with the unique words of each tweet. This document is then used to create a query for similar documents (Wikipedia articles) by using every word as an optional term as shown in Alg. 2 (Search strategy 3), where $optional(words)$ is an algorithm that transform the set of Word $words$ into a sequence of Word, adding the syntax that makes the former optional when querying the knowledge system.

There is no need to obtain keywords and all the available information is taken into account. Since this document can include tens of thousands of tweets with a great amount of unique words, this algorithm is far more time expensive.

Unlike the former two strategies, discretization can not be used. There is a single query and search, and discretization would result in retrieving a very small number of documents (around 2-6). This is why a fixed number of results is used.

Example: We join all the documents obtained from the query *Podemos elecciones* (around 60000), and create a single string of text with their unique words used as document. With this query as input, the knowledge system returns the most similar Wikipedia articles, shown in Tab. 4. We keep the first 1000 articles.

Result	Score
Izquierda Unida (España)	0.19
Gaspar Llamazares	0.19
Julio Anguita	0.16
Bipartidismo en España	0.15
Caso ERE en Andalucía	0.15
...	...

Table 4: Tweets group strategy - elecciones andaluzas podemos elecciones search results

SE4 Several tweets group: We use the same query creation process as in SE3. However, instead of creating a single document using all the tweets, we set up several documents, each of them corresponding to a group of tweets (randomly selected). The algorithm is as shown in Alg. 2 (Search strategy 4), where $groupTweets(tweets)$ is an algorithm that groups the set of Tweets $tweets$, returning several sequences of Word.

This strategy could be considered a generalization of the former, and it is more complex, since there is not only one search to perform, but several. The number of searches depends on the total number of tweets and the size of each group, which we denote by S. Performance depends on these parameters, since a higher amount of searches and bigger documents increase the required time.

Discretization can be used, since this time several searches are performed.

Example: Using the documents from the *Podemos elecciones* query, we create documents from groups of 200 tweets. Using the query resulting from the first group, we obtain the articles shown in Tab. 5.

Result	Score
Gaspar Llamazares	0.31
Izquierda Unida (España)	0.30
Elecciones al Parlamento Europeo de 2014 (España)	0.22
Podemos (partido político)	0.21
Julio Anguita	0.21

Table 5: Several tweets groups strategy - elecciones andaluzas podemos elecciones search results

Using these strategies we obtain different results, but always with the same format: a single collection of Wikipedia articles. As usual, these articles contain hyperlinks to other Wikipedia articles, which represent important ideas or concepts. Otherwise, there would not be Wikipedia articles about them. And that is how we defined KB-keywords: keywords that represent important ideas in the knowledge related to the information we analyze. Therefore, linked articles in our collection, specifically their title, will be KB-keywords.

In our Wikipedia articles collection we can find thousands of different KB-keywords, some of which appear in hundreds of articles, and some of which appear in only one. Consequently, we must sort them, give them a score and create a ranking.

Given a KB-keyword kbk_i corresponding to the title of article $article(kbk_i)$ linked in our collection of Wikipedia articles $documents$, we have devised two ways to obtain a score used to rank it:

RA1 Relationships ranking: This strategy is based on the number of articles in $documents$ that link to $article(kbk_i)$, rewarding the concepts that are strongly related to our knowledge data. Since some articles are very frequently linked in all contexts, such as *España* (*Spain*) or *Estados Unidos* (*United States*), we have calculated the inverse document frequency of every Wikipedia article title.

A KB-keywords would then be ranked as shown in Alg. 3, where $links\ (d, article(kbk_i))$ is an algorithm that returns $true$ if Document d contains an hyperlink to $article(kbk_i)$ and $false$ otherwise, NW is the total number of articles in the entire Wikipedia corpus and $linkingArticles\ (article(kbk_i))$ is an algorithm that returns the articles in the entire Wikipedia corpus that contain an hyperlink to $article(kbk_i)$.

Algorithm 3 Relationship Ranking

Input
  $documents$: Set of Document
  kbk_i: KB-Keyword
Output
  $score$: $\mathbb{R}$
Constants
  NW: $\mathbb{N}$

$score \leftarrow 0$
for all $d \in documents$ **do**
  if $links(d, article(kbk_i))$ **then**
    $score \leftarrow score + 1.0$
  end if
end for
$score \leftarrow score \cdot \log \frac{NW}{|linkingArticles(article(kbk_i))|}$

Example: We have a collection of Wikipedia articles as shown in Tab. 6. There are four KB-keywords: *Andalucía*, *Partido Popular*, *PSOE*, *Izquierda Unida* and *Partido Andalucista*. Let's suppose that the IDF of every one of them is equal to 1. The ranked KB-keywords would be then as shown in Tab. 7.

Article Id	Links	Title occurrences
1	Andalucía	7
1	Partido Popular	2
2	Partido Popular	1
2	PSOE	5
3	Partido Popular	1
3	Andalucía	2
4	Izquierda Unida	4
4	Partido Andalucista	2

Table 6: Articles collection example

KB-keyword	Score
Partido Popular	3
Andalucía	2
PSOE	1
Izquierda Unida	1
Partido Andalucista	1

Table 7: Relationships ranking results

RA2 Importance ranking: The former strategy rewards the number of relationships counting the number of pages in which each KB-keyword article is linked.

However, it does not take into account how many times the KB-keyword is mentioned in each article. Instead of the number of links, this strategy uses the total number of occurrences of each KB-keyword in order to reward the importance and weight of the concepts.

KB-keywords are ranked as shown in Alg. 4, where $numberOcurrences(kbk_i, d)$ is an algorithm that returns the number of occurrences of KB-Keyword kbk_i in Document d.

Algorithm 4 Importance Ranking

```
1: Input
2:     documents: Set of Document
3:     kbk_i: KB-Keyword
4: Output
5:     score: ℝ
6:
7: score ← 0
8: for all d ∈ documents do
9:     score ← score + numberOccurrences(kbk_i, d)
10: end for
```

Example: We have the same collection of articles, those in Tab. 6. Counting the number of occurrences of each KB-keyword, the results are shown in Tab. 8. As we can see, the scores and order are different.

KB-keyword	Score
Andalucía	9
PSOE	5
Partido Popular	4
Izquierda Unida	3
Partido Andalucista	2

Table 8: Importance ranking results

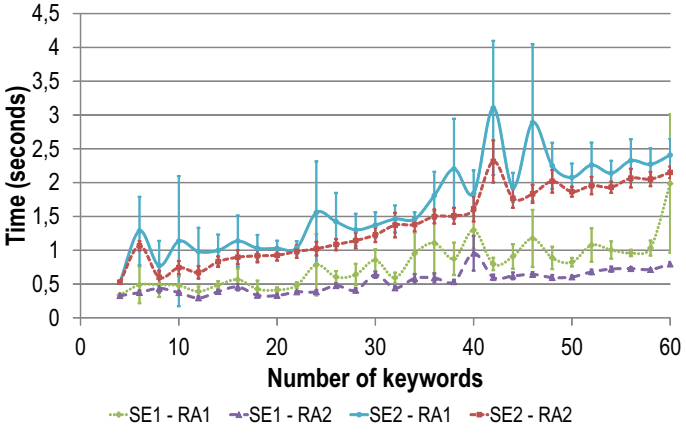
Once KB-keywords have been ranked, the process has finished. We have a new set of keywords.

4. Results and Discussion

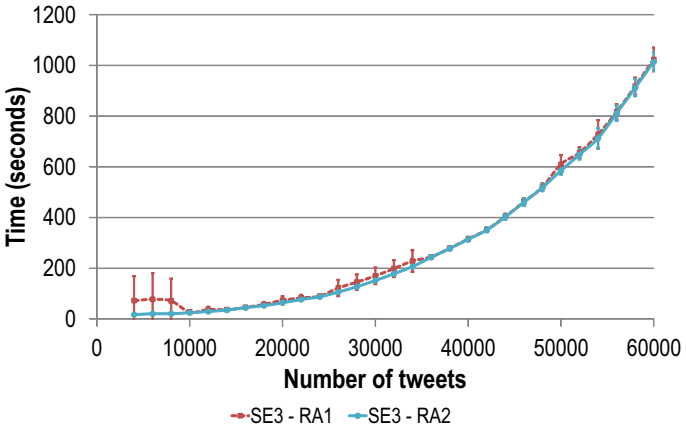
In this Section we discuss the results obtained when testing our approach, both performance-wise and functional-wise.

We have used two cases where we test the performance of each combination of strategies and rankings. We have analyzed tweets and SR-keywords obtained with Opileak using the query *Podemos elecciones*, calculating average execution time and confidence intervals ($\alpha = 0.05$). We have used Opileak as listening platform (SR-keywords are obtained as suggested in (Dubhashi and Panconesi, 2009)), Wikipedia as knowledge base, a Lucene index as knowledge system and a machine with 4GB of RAM and processor Intel Xeon 1.86 GHz. Tab.9 shows the 5 best SR-keywords.

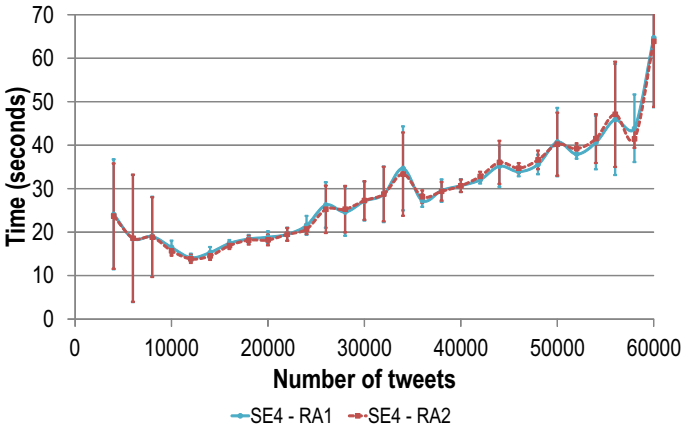
We have measured the execution time of SE1 and SE2, and both rankings using different amounts of keywords as input and 15 samples for each average time. The number of keywords ranges from 4 to 60 at intervals of 2.



(a) SE1 and SE2 performance



(b) SE3 performance



(c) SE4 performance

Figure 1: Performance using different strategies and rankings

	SR-Keyword
1	elecciones andaluzas
2	resultados
3	escaños
4	vía
5	PP

Table 9: SR-Keywords

The results are shown in Fig. 1a. We observe how execution time increases linearly as we expected, with outliers that cause a greater deviation in some cases, probably because of the indexation of documents and the different behaviour of queries with different terms. Using both search strategies, RA2 has lower execution times, and also lower standard deviations. This is probably due to the database access required to obtain the IDF of each KB-keyword using RA1. Still, the difference is small. Using SE2, execution times are higher because of the addition of words to each Lucene search, which increases the complexity of the latter. Deviations are lower, so the correlation index is higher.

We have measured the average execution times of SE3 and SE4, and both rankings using different amounts of tweets as input and 5 samples for each average time. The number of tweets ranges from 4000 to 60000 at intervals of 2000. When using SE4, groups had a fixed size of 200 tweets, so the number of groups ranges from 20 to 300, at intervals of 10.

The results are shown in Fig. 1b and Fig. 1c. Execution times are much higher, reaching 1000 seconds when using a single group and 60000 tweets. SE4 is much more efficient, reaching around 50 seconds with the same number of tweets. Even though the number of searches increases, the complexity of each search is much lower than that of a single greater search. The relationship between execution time and number of publications when using SE3 can be almost perfectly fitted by a cubic polynomial, while using SE4 results in a linear relationship. Again, RA2 is slightly more efficient, but since times are higher in this case, the differences are relatively smaller.

The KB-keywords we obtain are Wikipedia articles titles. In Tab. 10 we can see the first 10 KB-keywords obtained from the *Podemos elecciones* query when we apply each search strategy and ranking. SR-keywords are also shown in order to analyze the differences between KB-keywords and SR-keywords.

From the results, we can make the following observations:

- All strategies result in keywords which reflect the content of the Twitter publications. These were related to Podemos, the Andalusian regional elections and other relevant political parties in Andalusia.
- Since we use Wikipedia titles, SR-keywords reflect more clearly the colloquial language of tweets, but KB-keywords reflect in a more precise and readable way the relevant concepts, and there are no keywords such as *vía* in SR-keywords, line 4 (used when a tweet is a copy of another one).
- Sometimes, the same concept is repeated using different years. When we use SE1 and RA1, the KB-keyword *elecciones al parlamento europeo...* (*elections to the European Parliament...*) in RA1-SE1 lines 1-3, is repeated three times. In order to avoid this, KB-keywords would have to be filtered, removing those which are very similar to others. For example, we could ignore KB-keywords that include a year. However, this could cause the exclusion of relevant concepts or events.
- RA2 results in concepts that are overall more abstract, and are more frequently used, such as *candidato* (*candidate*) in RA2-SE1 line 2, *votos* (*votes*) in RA2-SE1 line 3, *partido político* (*political party*) in RA2-SE2 line 3 or *ciudad* (*city*) in RA2-SE3 line 2.

	SR-keywords	RA1				RA2			
		SE1	SE2	SE3	SE4	SE1	SE2	SE3	SE4
1	elecciones andaluzas	elecciones al parlamento europeo de 1979 (bélgica)	primavera europea	psoe	equo	elector	eldiario.es	municipio	upyd
2	resultados	elecciones al parlamento europeo de 1984 (bélgica)	podemos (partido político)	andalucía	partido popular	candidato	eta	ciudad	euro
3	escaños	elecciones al parlamento europeo de 2009 (bélgica)	elecciones al parlamento europeo de 2014 (españa)	partido popular	partido andalucista	votos	partido político	cia	ciu
4	vía	nacionalismo andaluz	izquierda política	izquierda unida	Psoe	papeleta	otan	eta	provincia
5	PP	voto (elecciones)	pablo iglesias turrión	izquierda unida (españa)	izquierda unida	votación	elecciones europeas de 2014	ucrania	política

Table 10: KB-keywords

- Using SE2, results are more related to the original query. Among them, we find *podemos (partido político)* (*Podemos (political party)*) in RA1-SE2 line 2, *pablo iglesias turrión* or *elecciones* (*elections*) in RA1-SE2 line 5.
- Top KB-keywords using SE3 and SE4 are similar. Using RA1, in both cases we find the KB-keywords *psoe*: RA1-SE3 line 1 and RA1-SE4 line4, *partido popular*: RA1-SE3 line 3 and RA1-SE4 line 2 and *izquierda unida*: RA1-SE3 line 4 and RA1-SE4 line 5.
- While SE1 and SE2 require previously obtained statistical information (SR-keywords), SE3 and SE4 only require the analyzed tweets.

5. Conclusions

In this paper, we propose an approach for extracting keywords from Spanish tweets. This approach takes context into account, and selects as keywords important concepts from the knowledge related to the text we analyze, and not from the text itself.

This knowledge is usually added to disambiguate the meaning of keywords. We automatize this process by using the knowledge available at Wikipedia to add context.

Querying an index with Wikipedia articles, we extract a collection of articles related to the available textual information, which represent the knowledge related to it. We then use the title of linked articles in the collection as keywords, which are obtained with purely statistical methods. Using Wikipedia articles titles as keywords also removes the possibility of ambiguity.

Our algorithm uses different strategies when searching articles and ranking KB-keywords, which lead to different results and performance.

We have used Spanish tweets and Wikipedia articles, but the process is completely independent of languages. In order to analyze tweets written in a different languages, the only necessary change would be replacing the

Index with articles in the required language.

6. References

Blei, D. M., 2012. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

Chen, Y., Li, Z., Nie, L., Hu, X., Wang, X., Chua, T.-s., and Zhang, X., 2012. A Semi-Supervised Bayesian Network Model for Microblog Topic Classification. In *COLING*, pages 561–576.

Chen, Z., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., and Ghosh, R., 2013. Discovering coherent topics using general knowledge. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 209–218. ACM.

Dubhashi, D. P. and Panconesi, A., 2009. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press.

Hennig-Thurau, T., Wiertz, C., and Feldhaus, F., 2014. Does Twitter matter? The impact of microblogging word of mouth on consumers' adoption of new movies. *Journal of the Academy of Marketing Science*, 43(3):375–394.

Hu, X. and Liu, H., 2012. Text analytics in social media. In *Mining text data*, pages 385–414. Springer.

Hulpus, I., Hayes, C., Karnstedt, M., and Greene, D., 2013. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 465–474. ACM.

Ko, Y., 2012. A study of term weighting schemes using class information for text classification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1029–1030. ACM.

Nenkova, A. and McKeown, K., 2012. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer.

Ren, F. and Sohrab, M. G., 2013. Class-indexing-based term weighting for automatic text classification. *Information Sciences*, 236:109–125.

Thorleuchter, D. and Van den Poel, D., 2012. Improved multilevel security with latent semantic indexing. *Expert Systems with Applications*, 39(18):13462–13471.

Thorleuchter, D. and Van den Poel, D., 2013. Technology classification with latent semantic indexing. *Expert Systems with Applications*, 40(5):1786–1795.

Xie, J., Emenheiser, J., Kirby, M., Sreenivasan, S., Szymanski, B., Holme, P. et al., 2012. Evolution of Opinions on Social Networks in the Presence of Competing.

Yubo Chen, S. F. and Wang, Q., 2011. The Role of Marketing in Social Media: How Online Consumer Reviews Evolve. *Journal of Interactive Marketing*, 25(2):85–94.

Zhang, W., Yoshida, T., and Tang, X., 2011. A comparative study of TF* IDF, LSI and multi-words for text classification. *Expert Systems with Applications*, 38(3):2758–2765.

Zhu, J., Chen, N., Perkins, H., and Zhang, B., 2014. Gibbs max-margin topic models with data augmentation. *The Journal of Machine Learning Research*, 15(1):1073–1110.