A modular approach for lexical normalization applied to Spanish tweets

J.M. Cotelo , F.L. Cruz, J.A. Troyano, F.J. Ortega

Department of Languages and Computer Systems, University of Seville, Avda. Reina Mercedes s/n, 41012 Seville, Seville, Spain

Keywords:
Twitter
Text normalization
Domain adaptation

A B S T R A C T

Twitter is a social media platform with widespread success where millions of people continuously express ideas and opinions about a myriad of topics. It is a huge and interesting source of data but most of these texts are usually written hastily and very abbreviated, rendering them unsuitable for traditional Natural Language Processing (NLP). The two main contributions of this work are: the characterization of the textual error phenomena in Twitter and the proposal of a modular normalization system that improves the textual quality of tweets. Instead of focusing on a single technique, we propose an extensible normalization system that relies on the combination of several independent "expert modules", each one addressing an very specific error phenomenon in its own way, thus increasing module accuracy and lowering the module building costs. Broadly speaking, the system resembles to an "expert board": modules independently propose correction candidates for each *Out of Vocabulary* (OOV) word, rank the candidates and the best one is selected. In order to evaluate our proposal, we perform several experiments using texts from Twitter written in Spanish about a specific topic. The flexibility of defining resources at different language levels (core language, domain, genre) combined with the modular architecture lead to lower costs and a good performance: requiring a minimal effort for building the resources and achieving more than 82% of accuracy compared to the 31% yielded by the baseline.

1. Introduction

One of the most important challenges that we are facing today is how to process and analyze the large amount of information on the Internet, and especially social networking sites like Twitter, where millions of people express ideas and opinions on a daily basis about a myriad of topics.

Texts written in Twitter, called tweets, are characterized by having a short length (140 characters) that is very small compared to the size of traditional genres. Furthermore, most of these tweets are typically written using devices like smartphones, resulting in that those tweets are often hastily written (some users even in a real-time fashion as with instant messaging chat) with almost no revision before sending them, trading redaction quality and/or correctness for speed.

Consequently, the users of these networks have developed a new form of expression that includes SMS-style abbreviations, lexical variants, letters repetitions, use of emoticons, etc.

The widespread success of Twitter makes the analysis of the tweets a very interesting (and possibly lucrative) task. The amount of information in these texts is huge and tweets are rarely objective, becoming the target of large-scale analysis that could be really useful for marketing campaigns, public opinion determination or even inferring how a population responds for specific events. Branding (Ghiassi, Skinner, & Zimbra, 2013; Mostafa, 2013), political analysis (Conover et al., 2011; Himelboim, McCreery, & Smith, 2013; Tumasjan, Sprenger, Sandner, & Welpe, 2010) or user profiling for market analysis (Ikeda, Hattori, Ono, Asoh, & Higashino, 2013) are examples of actual applications for the analysis of Twitter and other social media texts. Protection and detection against malware (Martinez-Romo & Araujo, 2013) is also an application of interest, as Twitter trending topics are also vulnerable to scamming, phishing or spamming.

But the problems mentioned above lead to inherent difficulties processing those texts using traditional NLP analysis. Current NLP tools may have problems to process and understand these short and noisy texts, being unsuitable for NLP tasks as Opinion Mining, Topic Modeling or any other characterization task. Generally speaking, any NLP task used for analyzing tweets would greatly benefit from a normalization process, because NLP analysis is quite sensitive to the quality and length of the input texts. Text Summa-

* Corresponding author.
E-mail addresses: jcotelo@us.es (J.M. Cotelo), fcruz@us.es (F.L. Cruz), troyano@us.es (J.A. Troyano), javierortega@us.es (F.J. Ortega).

rization (Jabeen, Shah, & Latif, 2013) and Ontology-based Sentiment Analysis (Kontopoulos, Berberidis, Dergiades, & Bassiliades, 2013) are examples of process that rely on tweet normalization for proper operation.

Typically, normalization approaches focus on working with a single technique, motivated by solving the task for a subset of possible error phenomena. The main issue is that those approaches are not adaptive enough, being cumbersome to use them for another text domains or other media, specially where other error phenomena are more common than the originally addressed ones. In some cases, performance suffers too mach or the adaptation process is too complex to perform. In this paper, we present a modular approach for the normalization task with these issues in mind, providing a system that is more resilient and easier to implement and expand than rest of more traditional approaches that usually appear in the current literature.

1.1. Related works

Although our concern is to address texts written in Spanish, most of the existing works that tackle the lexical normalization problem deal with SMS or social media texts primarily written in English. Though Spanish and English are different languages, many of the ideas and general processes of lexical normalization approaches intended for English texts may be adapted for Spanish texts.

A survey about the works on the bad use of language on the Internet is presented in Eisenstein (2013). This study reviews and criticizes the different types of NLP approaches that tackle this problem, dividing them into two opposed categories: normalization, and domain-adaptation. The aim of the works on normalization is to improve the quality of texts by converting the OOV terms into valid phrases; while works focused on domain adaptation are based on adapting the NLP tools in order to make them able to process this kind of bad language. Common approaches for normalizing noisy text rely on rule-based techniques (Sidarenka, Scheffler, & Stede, 2013), statistical language models (Yang & Eisenstein, 2013) or in a mixture of both (Costa-Jussa & Banchs, 2013).

A two-phase method for solving SMS abbreviations is proposed in Pennell and Liu (2011). The first phase consists in a character-level machine translation model, instead of a word-level system, that learns mappings between characters (letters, group of letters and symbols) instead of learning mappings between words or phrases. The second phase is intended to include information about the context in order to refine the abbreviation normalization, using a word-level language model (Shannon's noisy channel model described in Shannon (1948)). In Han et al. (2011) and Han, Cook, and Baldwin (2013) a study of the out-of-vocabulary words on Twitter is carried out, analyzing the different unorthodox uses of the language in this social network. Regarding the observations of the study, the proposed technique is focused on the lexical normalization of OOV words composed of just one term. This normalization is performed in an unsupervised fashion, taking into account morphophonemic variations of words and the context where the words occur. They show an extension of the work in Han et al. (2013) with a more detailed explanation and a thorough experimentation. In Han, Cook, and Baldwin (2012), the same authors tackle the same problem by means of a normalization lexicon based on a dictionary of pairs of OOV terms and *In Vocabulary* (IV) terms, ranked through a string similarity algorithm that takes into account the morphophonemic similarity of the pairs, demoting as well the noisy pairs.

Though it is not easy to find works for other languages than English, there are several remarkable lexical normalization efforts for Spanish texts that are worth mentioning: Porta and Sancho (2013) proposes an approach based on weighted finite-state transducers (FST) that are statistically combined in a three step composition (recognized variants, possible variants and language model). These FST are generated from custom-made rules and a word trigram language model. Also they use several lexical resources like the Diccionario de la Real Academia Española (DRAE) Spanish dictionary and a web page corpus (Wacky). It is not uncommon that Spanish users make use of common English words, so the authors of this work also used a lexicon composed of the 100k most frequent English words from British National Corpus for handling these words. Gamallo, García, and Pichel (2013) proposes a lexical approach using diverse lexical resources, generating correction candidates and classifying them into two categories: primary and secondary. Those candidates are generated either using an array of simple correction rules (capitalization, character repetition and common orthographic errors), a normalization list (based on textual data), the DRAE dictionary, a proper noun dictionary collected from Wikipedia or a language model based on a corpus of newspapers RSS. Ageno, Comas, Padró, and Turmo (2013) propose an approach based on different processing modules (classified into three groups: single words, multiword terms and regular expressions) for generating different correction candidates for each unknown word. They also use the *foma* toolkit for doing approximate searches in order to find similar single or multiword terms. The system selects the candidate by using a voting scheme weighted by module precision. Some of the resources used are an acronym list, a multicharacter emoticon list, an onomatopoeia list, several dictionaries (with variants) and proper noun lists (Spanish and English languages).

Instead of focusing on a technique for addressing the normalization task, in this paper we propose a modular normalization system that relies on the combination of several independent "expert modules". Each module is designed to address a very specific error phenomenon, thus increasing module accuracy and lowering the module building costs. In essence, the system behaves as an "expert board": when an OOV is detected, one or more modules propose a correction proposal, a ranking is made and the best one is selected. This way, instead of relying on an single and multi-purpose technique, our system allows that each module implements its own technique.

Our "expert board" proposal is easily expanded for addressing other error phenomena (via adding other specific-purpose modules), it is very robust against hard and ambiguous error phenomena, it reduces the designing and building costs and the experimental results show very good performance.

1.2. Structure of this work

In Section 2, we characterize the problem by manually annotating a statistically significant sample of a corpus 3.1 millions of tweets. The result of this characterization is a statistical distribution of the different error phenomena that can be found in Twitter.

In Section 3, we describe the highly modular architecture of our system. We divide our architecture in several stages: *preprocessing*, *detection*, *candidate generation* and *candidate selection*. *Preprocessing* and *detection* stages are described in Section 3.1, while *candidate generation* and *candidate selection* are described in Sections 3.2 and 3.3 respectively.

In Section 4, we identify several conceptual major categories for terms and we take a modular approach for the lexical resource generation. In this section, we discuss the particular process and cost for generating each lexicon.

In Section 5, we evaluate our system under different perspectives and we provide several metrics to perform this evaluation. We show the performance of our system with different modules activated, addressing different error phenomena. We also provide

Table 1
Detected term distribution commonly found in Twitter media.

Term type	Detected (%)	Description
Language IV	68.76	Valid terms found in the Spanish vocabulary resource
Rest	31.24	Terms not found in the Spanish vocabulary resource
OOV	28.82	Invalid terms not found in any resource
Specific IV	15.82	Valid terms found in some specific-purpose resource (genre, domain, etc.)
REGEX	53.33	Valid terms recognized using regular expressions (mentions, hashtags, dates, etc.)

an analysis of error phenomena and an analysis of the selection step that is the basis for tuning the candidate selection process.

In Section 6 we summarize our efforts, review the main points of our work and propose several directions for improving our system and further our research.

2. Problem characterization

In order to characterize the problem and evaluate our system, we have compiled a dataset composed of 3.1 millions of tweets written in Spanish that are related to the *2012 UEFA European Football Championship*. The dataset was generated using the dynamic retrieval process described in Cotelo, Cruz, and Troyano (2014). The idea behind this collection is to have a dataset in Spanish containing a large quantity of tweets with the typical quality issues in Twitter texts.

As an initial step of our work, we carried out an analysis over the term distribution in our tweet dataset, using vocabulary resources as knowledge for determining valid terms. Table 1 and Fig. 1 represent the results of the analysis, showing that only a 68.76% of the terms were recognized in a Spanish language vocabulary resource, being that ratio named as *Language IV* in the table and the figure.

A significant share of the terms not found in Spanish vocabulary were OOV while the rest of terms were found in some specific-purpose vocabulary or were something different than a word, like dates or twitter-specific constructs (hashtags or mentions).

Before doing any normalization process we did a characterization of the existing OOV error phenomena. In order to perform this characterization, we obtained a statistically significant sample of the dataset ($\alpha = 0.05,$ error bound $= 1\%$) composed of tweets that contained at least one OOV. Each tweet was analyzed and each OOV within it was manually tagged indicating the OOV term, the error phenomenon associated and its corrected form. In some cases, multiple phenomena were found. This manually annotated sample is used as the evaluation dataset used in the experimentation process.

Characterized error phenomena were classified into several categories:

- *Orthographic errors* (`ORT`): Spelling errors, word segmentation issues, lack or misuse of diacritics, lack of capitalization.
- *Texting Language* (`TXT`): Ad-hoc acronyms and abbreviations, letter omission, omission of parts of speech, logograms and pictograms. For instance, "*x2*" is an ad hoc acronym of "*por dos*", meaning "*twice*" in English.
- *Homophonic confusion* (`HOMO`): Common phonological changes and other spelling variants of phonological origin.
- *Unrecognized Onomatopoeia* (`ONO`): Rare onomatopoeia or spelling variants of onomatopoeia that were not easily recognized.

Table 2
Characterization of error phenomena commonly found in Twitter media.

Phenomenon	Ratio (%)	Examples
Orthographic errors	27.51	Sacalo → sácalo, trapirar → transpirar,...
Texting Language	7.92	x2 → por dos, q → que, aro → claro,...
Homophonic confusion	8.52	kasa → casa, caxo → cacho,...
Unrecognized onomatopoeia	5.96	jajajajjajja → ja, jum → um,...
Character repetition	15.25	siiiiiiii → si, quiiiiieeeeroooo → quiero,...
ASCII Art	13.80	♠♥oO._.Oo ...
Free inflections	6.90	gatino, besote, bonico, ...
Other errors	6.73	htt, asdfasdfasdf, ...
Other language	4.00	flow, ftw, great, lol, ...
Multiple phenomena	3.41	diass → días, artooo → rato, ...

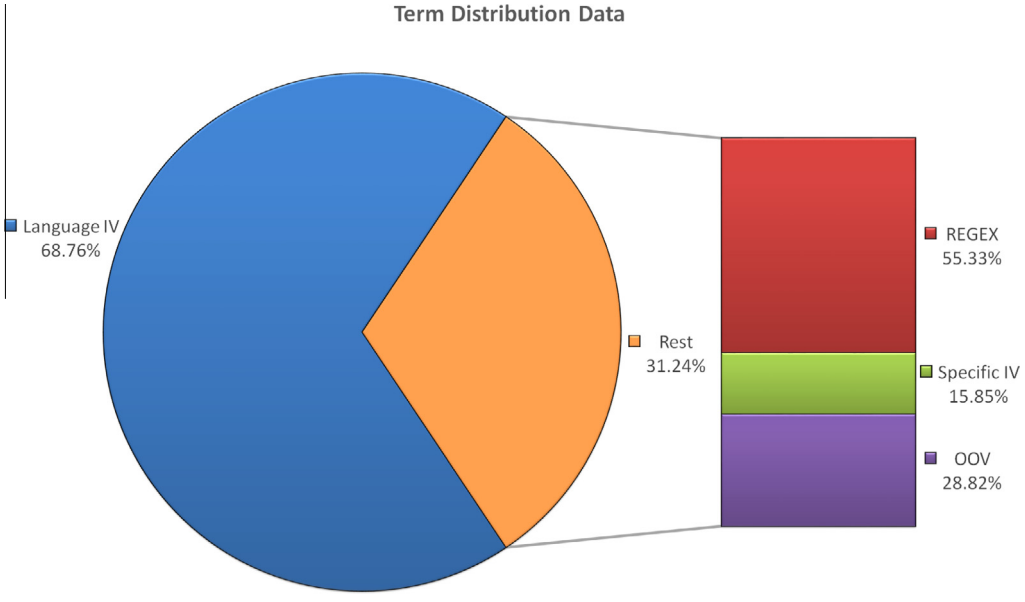

Fig. 1. Detected term distribution commonly found in Twitter media.

- *Character repetition* (REP): Unnecessary character repetitions usually intended to give emphasis.
- *ASCII Art* (ASC): Special use of available characters to reflect situations, emotions, mood, other nonverbal communication or even aesthetic uses.
- *Free inflections*: Accepted spelling variants usually bound to some specific region or collective. Not an error phenomena in the proper sense, but they are *a priori* OOVs. For instance, in some regions, "*gatino*" is an accepted spelling variant of "*gatito*", meaning "*kitty*" in English.
- *Foreign terms* (FT): Foreign words that may or not be accepted within the Spanish language context.

Table 2 illustrates this characterization and provides some examples for each phenomenon (some of the have been mentioned before). It is observed that most of the errors fit into the major categories described above and those errors are associated with the fast and informal writing style in Twitter, usually done from a mobile device.

In order to address the different challenges posed by the identified phenomena, we have taken a lexical approach for building the system proposed in this work. Our proposal has a modular architecture, constituted by several components that are easily configurable, allowing the system to easily adapt to different domains or applications. Our approach is mainly composed of three types of components:

- **Resources**: lexicons, corpora and any other language resources. It also includes resources that contain specific knowledge of the media used.
- **Rules**: rules for handling common phenomena found in this type of media as excessive character repetition, acronyms or homophonic errors.
- **Lexical distance analysis**: traditional lexical distance analyzers for handling common orthographic errors found on it.

Table 3
Extract of the transformation rules used in our system.

Matching	Processing	Example	Phenomenon
^ [ck]n$	con	kn, cn, → con	Texting Language
x([aeiouáéíóú])	ch\1	xaval, coxe → chaval, coche	Texting Language
((\w)(\w))\1+(\2—\3)?	\g⟨1⟩	sisisisisisi, nononononono → si, no	Character repetition
^ t[qk]m+$	te quiero mucho	tkm, tqm → te quiero mucho	Texing Language

In essence, our system examines each word at a lexical level and determines whether it is an OOV or not, using the available resources. If it is the case, the system generates a set of correction candidates and finally selects the best correction candidate for each case. A benefit of this lexical approach is that the cost of generating these resources is quite low compared to the cost of typical tagged data for more traditional pipelines.

3. System architecture

The existing normalization works are appropriate but they are usually designed to address a specific scenario, being very costly to adapt them to another domain or language. Our approach takes a different path: proposing a system that focuses on flexibility, modularity and lower manual workload.

Our system exhibits a highly modular architecture, composed of several components that interact with the pipeline, other components or external resources. Fig. 2 shows the processing workflow of the system and how its components are interconnected.

Conceptually, we divide the system in several stages that are described below: Preprocessing, detection, candidate generation and candidate selection.

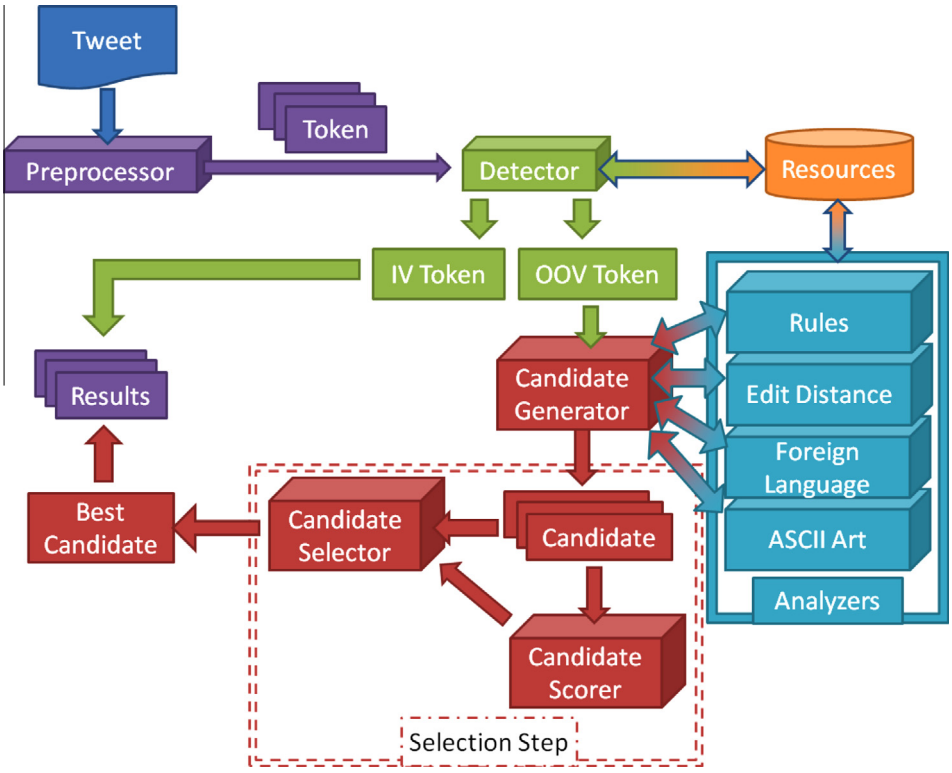

Fig. 2. Architecture and processing steps of the proposed system.

3.1. Preprocessing and detection

As in the majority of NLP processing pipelines, the first step of our approach consists in performing some kind of preprocessing to the raw text. Our preprocessing module performs the typical initial processing step done in lexical analysis, generating a stream of tokens from each tweet while taking into account special terms like Twitter constructs (hashtags and usernames), numerals, dates, URLs and emoticons. These special terms are carefully detected and processed. It also performs encoding-related cleaning routines as treating characters wrongly encoded, discarding characters from private Unicode planes and normalizing characters to their Unicode normal form.

The next stage consists in detecting the *coarse* nature of each resulting token. Basically, the detection module tries to determine whether a token is an OOV or not, by checking if it belongs to any external resource or it conforms to some well known constructs like usernames, hashtags, dates, numerals, emoticons or even URLs. As external resources, we use a set of lexicons, each one providing known forms used in Twitter, the Spanish language, well known emoticons or even colloquial inflections. For construct detection, we use an approach based in regular expression pattern matching.

After this detection stage, every OOV token is sent to the next step of the pipeline while the rest of detected terms are only marked and sent to the end of the pipeline.

3.2. Candidate generation step

After a token is marked as OOV, it is sent through the pipeline to the candidate generator module that controls the candidate generation stage. The candidate generator module is linked to several *analyzer* modules that actually perform the bulk of the candidate generation work.

Given an OOV token, the candidate generation module instructs each analyzer module to perform some "error guessing" process and to generate candidates for correcting the provided OOV token. The specific underlying process varies from each analyzer, being each one of them specialized in some specific error phenomena. Furthermore, some analyzers use external resources to perform their work and every analyzer module provides some kind of *basic scoring* indicating some degree of confidence for each correction proposed.

The modular architecture allows to have an arbitrary number of different analyzers, giving total flexibility in designing the specific system for any other context. More specifically, for our work we have implemented the following modules:

The **Edit distance** module works very similar to distance-based suggestion scheme commonly found in spell checkers; the main difference is that it takes into account multiple lexicons instead of a single one. This module is the most general and tackles most of the error phenomena, nevertheless it deals with *Orthographic errors* better than any other implemented module. The string distance used is the Damerau–Levenshtein distance that, given two strings, it expresses the number of lexical operations (insertion, deletion, substitution or adjacent transposition of characters) to transform one string into the other. For instance, the Damerau–Levenshtein distance between the words *puerta* and *perro* is 3:1 elimination and 2 substitutions. Its implementation is based on Levenshtein automata (Schulz & Mihov, 2002) and lexicons. The confidence values assigned to the generated candidates is in the range of [1–3] and is inversely proportional to the distance between the candidate and the OOV.

The **Transformation rules** analyzer module holds a collection of hand-crafted rules. These rules are intended to inject human knowledge into the system and each one represents some kind of *"well defined"* error. Using these rules, it generates a set of candidates by matching the OOV token against the rules and performing a lexical transformation of the original OOV token according to that rule, generating a candidate of correction. It is possible to generate more than one candidate due to multiple rule matchings, but their number is usually limited to a few. These rules are intended to address phenomena that the edit distance module does not correctly treat like *Character Repetition* or *Texting Language*. The ruleset generation process is described in Section 4 and Table 3 shows some example of rules, using Python's regular expressions. This module always assigns the maximum confidence value of 3 for the generated candidates.

The **Foreign language** detection module tries to identify which language the OOV token belongs to. If the module determines that the language of the OOV token is different from Spanish, it is marked as a *Foreign language* token. Notice that the module does not actually perform any correction; tokens from other languages have to be marked but not corrected. The language analyzer module uses a trigram language guessing module implemented in Python 3 (Phi-Long, 2012) as backend. The confidence values assigned to the generated candidates is in the range of [1–3] and is directly proportional to the confidence value provided by the underlying estimator.

The **ASCII Art** detection module tries to identify whether the OOV token is some kind of ASCII Art, an unregistered emoticon or a variant of a known one. It works by using a mix of several hand-crafted regular expressions and a list of emoticons. This module behaves in a similar way to the *Foreign language* module in the sense that it only marks the tokens; no correction is provided for tokens detected this way. This module always assigns the maximum confidence value of 3 for the generated candidates.

After candidates are proposed by the analyzers, the candidate generation module performs a validation and filtering step, that is intended to remove incorrect and/or duplicate candidates, according to a set of validation rules and language resources.

Recapitulating, the candidate generation step performs this way:

1. Given an OOV token, the candidate generation module send the token to the analyzer modules.
2. Each analyzer module proposes one or more candidates as possible corrections for that OOV token.
3. The candidate generation module filters and validates all the candidates proposed by the analyzer modules.
4. The set of all the resulting candidates are sent to the next step of the pipeline.

3.3. Candidate selection step and summary of the process

This is the final step in the processing pipeline where, for each OOV token, we choose the best possible correction among the proposed candidates.

In order to choose the best candidate, we assign a numeric value to each one, using the candidate scorer module. This module computes a score by normalizing the confidence values associated to each candidate. After that, the candidate selector module sorts the candidates by score and selects the best one, using several in-cascade criteria to resolve tie-ins.

The whole system workflow can be summarized as follows: the system generates a token stream from the tweet using the preprocessing module. For each token, it determines whether a token is an OOV or not using the detector module. If the token is an IV, no further processing is done because it is a *valid form*. Otherwise, if the token is an OOV, the candidate generator module creates a tentative list using the analyzers previously

described. As final step, the candidate selector module selects the best candidate for correction using the scores provided by the candidate scorer module.

Table 4 shows some examples of the system output, showing a proposed correction for each input tweet. For the sake of clarity, only the resulting text is shown, omitting all the token and candidate information.

4. Resources employed

For the evaluation of our proposal, we have constructed a system for the following scenario: Spanish tweets during the *2012 UEFA European Football Championship* (Euro2012) event. Throughout this section, we describe the process and costs for generating the resources used by the system.

We performed an analysis for determining the very nature of the vocabulary used in those texts. This analysis brought us the idea that the vocabulary may be split in several separate conceptual categories, being able to independently address different aspects of the vocabulary used in the texts.

We identified three major conceptual categories: *Language*, *Genre* and *Domain*. Terms belonging to the *Language* (Lang) category are terms that belong to the language in a general sense, not being associated to any specific context, domain or media for our purposes. Most frequently used Spanish forms fall into this category. The *Genre* category is composed by terms whose use is typically confined to the Twitter and the Internet context. These terms are usually specific terminology or expressions commonly used in both Twitter and The Internet. The *Domain* category is made of common football terms and other terms referring to the *Euro2012* event such as player names, teams, stadiums where matches are played.

As a result of the previous categorization, we have compiled and used several specialized lexicons for the detection and analysis stages in our system, being all of them in raw text format and containing one entry per line. This approach facilitates the task of generating any lexicon, only having to focus in a specific aspect, and improves flexibility by easing the adaptation process to newer contexts.

According to our experiments, our system can be configured from scratch for a new scenario (language, domain and genre) by investing approximately 20 h of manual effort. Once that initial job is made, adapting the system to a different domain (any domain not related with football) only requires of manually composing another domain resource.

The specific quantity of time required to compose another domain resource is quite dependent of the domain itself, but usually range from minutes to a few hours. In some cases, this effort may be reduced if other existing resources are identified to be adapted, or even if automatic procedures can be applied to accelerate the task of build this domain resource.

Table 4
Examples of system output.

Source tweet	Corrected output
RT @axestrella7: fds estupendo: Partidzo d España con mi amr. tqmmmmmm:)	RT @axestrella7: Fin de semana estupendo: partidazo de España con mi amor. Te quiero mucho:)
iiiiiiiker iiiiiker iiiiiiiker!!!! q crack *.* #VamosEspaña	Íker Íker Íker! Qué crack *.* #VamosEspaña
@SergioRamos ers un crack menudopenalty mas bien tirado demostrando q lo de la otra vez solo fue mala suerte	@SergioRamos eres un crack menudo penalty mas bien tirado demostrando que lo de la otra vez sólo fue mala suerte

Table 5
Lexicons used for our proposed system.

Lexicon	Entries	Description
Lang	1250796	Common forms from Spanish. Based on `LibreOffice` dicts.
Genre	40	Common forms related to Twitter and internet. Handcrafted.
Domain	2710	Forms related to football and Euro2012 event. Handcrafted.
Emoticons	320	Commonly used emoticons. Handcrafted.

In general terms, the time required for constructing this domain resource may be reduced if automatic construction methods are used.

Table 5 shows stats about all the lexicons used.

The generation of these lexicons was straightforward and was done with relatively low effort and costs. The process and cost for each generated lexicon were as follows:

- **Lang**: The OpenOffice Spanish dictionaries are in Hunspell format so we used the munch/unmunch tools (also provided within the Hunspell package) in order to extract all the forms, applying morphological transformations in the process. It took about 90 min to do the whole process.
- **Genre**: This small-sized lexicon was entirely handcrafted and it contains most frequently used forms related to Twitter (including Spanish loanwords and variants that come from English). It took about 75 min to do the process.
- **Domain**: This medium-sized lexicon was generated by adding the crawling of several lists of players, teams, countries and locations from several sport specialized sources to a list of common football terms. It took about 4 h to compose this lexicon: composing the list of common football terms, doing the crawling and the necessary preprocessing.
- **Emoticons**: This small-sized lexicon was generated from several lists of well-known emoticons, being the most extensive the one from Wikipedia.[1] This process took about 60 min, partly due to the revision of all the included emoticons. Emoticons conceptually fall into the *Genre* category but they were separated during the lexicon generation for the sake of simplicity.

For the transformation rules analyzer module, we hand-crafted a set of 71 rules. The syntax used in the ruleset file vaguely resembles a CSV format: there is a rule per line, holding information about the matching and the transformation process. Despite that crafting process was significantly costlier than the lexicon generation, it only took about 15 h to craft and refine the ruleset, being a relatively low cost for custom-made rules addressing a great variety of diverse error phenomena. Moreover, this ruleset is only tied to the language so it can be used for other domains or genres.

The language detector module uses a trigram character language model implementation as backend and uses dictionaries as backoff strategy just in case of insufficient data for language estimation.

5. Evaluation of the system

In this section we evaluate our approach measuring the system performance under different perspectives: *module-wise*, *phenomenon-wise* and *candidate selection step* analysis. We define several metrics for measuring the system performance, each one tries to measure a relevant aspect of our processing pipeline. These met-

[1] http://en.wikipedia.org/wiki/List_of_emoticons.

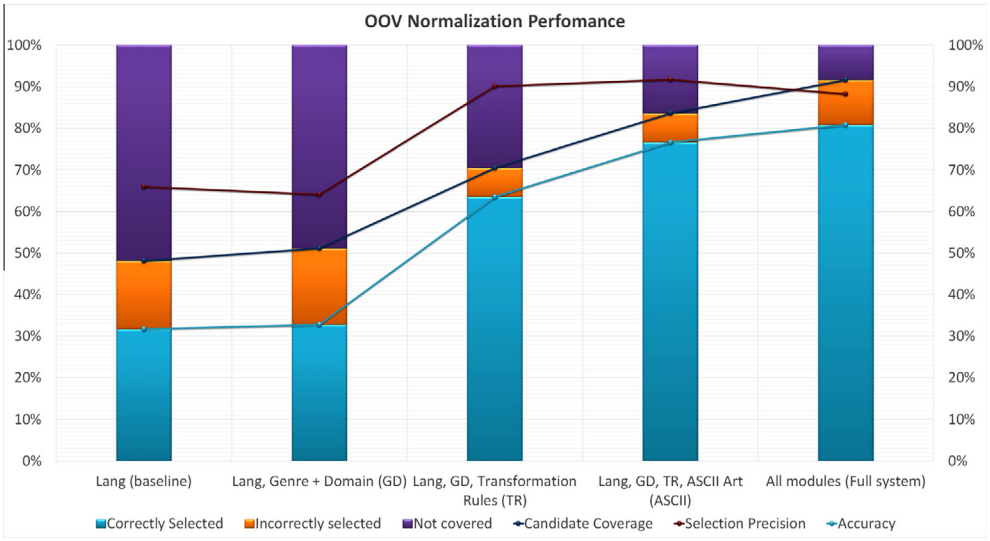

Fig. 3. System performance with different modules activated.

Table 6
System performance with different modules activated.

Active modules	Candidates/ OOV	Candidate coverage (%)	Selection precision (%)	Total accuracy (%)
Lang (baseline)	43.01	48.07	65.92	31.68
Lang, Genre + Domain (GD)	94.00	51.16	64.05	32.77
Lang, GD, Transformation Rules (TR)	94.21	70.48	90.13	63.52
Lang, GD, TR, ASCII Art & Emoticons (ASCII)	94.91	83.62	91.68	76.66
All modules (Lang, GD, TR, ASCII, Foreign language)	95.22	91.65	88.20	80.83

Table 7
Module correct candidate contribution per OOV error phenomenon.

Modules	ASC (%)	HOMO (%)	ONO (%)	ORT (%)	FT (%)	REP (%)	TXT (%)
Edit distance	0.00	92.11	6.38	86.31	0.00	57.98	15.79
Transform rules	0.00	92.11	63.38	56.65	0.00	99.16	53.95
ASCII Art	94.44	0.00	0.00	0.00	0.00	0.00	0.00
Foreign language	0.00	0.00	0.00	0.00	100.00	0.00	0.00
Default strategy	5.56	0.00	34.04	0.00	0.00	0.00	0.00

rics are described and formalized in Section 5.1. The annotated dataset sample used for evaluation purposes is described in Section 2.

In Section 5.2 we show the system performance with the different modules being activated incrementally. Also, we analyze the *contribution* of each module to the system performance overall.

In Section 5.3 we analyze the different performance values of the *full system* regarding each type of underlying OOV error phenomenon. Not all the phenomena are equally difficult to tackle and in this section we analyze this situation, providing an explanation for each phenomenon and its peculiarities.

In Section 5.4 we evaluate the flexibility of our system when it is adapted to a new domain. In this case, we have performed experiments with a corpus composed by tweets related to the proposed reform to the abortion law in Spain on 20th December 2013. The performance results on this new domain are very similar to the main experiment, showing that our system is able to tackle different domains with the minimal effort of composing another domain resource. The rest of the resources and components of the system remains the same.

In Section 5.5 we provide an analysis of the *candidate selection* step within our processing pipeline. This analysis set the basis for a modification of the current selection step, consisting in the addition of *a posteriori* classifier-based step to further refine the candidate selection process. That refinement led to significant performance improvements.

5.1. Metrics

As we mentioned above, in order to measure the performance of our system, we provide several metrics specifically designed to evaluate different aspects of our proposal. A description for each metric is below:

- *Candidate Coverage*: it measures how many times an OOV have been "covered" by the system, having the proper correction within the proposed candidate set. In essence, it measures the ability of the system for generating correct candidate solutions but not necessarily selecting them as the proposed correction.
- *Selection Precision*: it measures how many times the proper candidate has been selected from the generated candidate set, provided that the proper correction candidate is within the proposed candidate set. It measures how accurate is the candidate selection step.
- *Accuracy*: it measures how many times an OOV has been correctly addressed, meaning that the system generates and selects the proper correction for that OOV. This metric relates to the traditional accuracy metric used in Information Retrieval and represents how much the system actually fully corrects OOVs.

In addition to the explanation of the metrics, we provide a formalization for them as follows:

Definition 1. Given the following provisions:

- Let $T_{dataset}$ the set of all tweets from the dataset and let OOV_t the set of detected OOVs in a tweet $t \in T_{dataset}$.

Table 8
Full system performance regarding different error phenomena.

Phenomenon	Selection precision (%)	Candidate coverage (%)	Total accuracy (%)
ASC	92.22	100.00	92.22
HOMO	100.00	92.11	92.11
ONO	78.72	100.00	78.72
ORT	85.90	86.31	74.14
FT	62.96	100.00	62.96
REP	99.15	99.16	98.32
TXT	79.59	64.47	51.32

- Let C^t_{oov} the candidate set generated by the system for an OOV token, $oov \in OOV_t$ detected in a tweet $t \in T$.
- Let $corr^t_{oov}$ the tagged correction for the $oov \in OOV_t$ detected in tweet $t \in T$.
- Let $csel^t_{oov}$ the candidate selected from C^t_{oov} by the system.

The metrics used to analyze the performance of the system are the following:

$$\text{Candidate Coverage} = \frac{\sum_{t \in T_{dataset}} \left| \{ corr^t_{oov} : corr^t_{oov} \in C^t_{oov}, oov \in OOV_t \} \right|}{\sum_{t \in T_{dataset}} \left| \{ oov : oov \in OOV_t \} \right|}$$

$$\text{Selection Precision} = \frac{\sum_{t \in T_{dataset}} \left| \left\{ csel^t_{oov} : csel^t_{oov} = corr^t_{oov}, csel^t_{oov} \in C^t_{oov}, oov \in OOV_t \right\} \right|}{\sum_{t \in T_{dataset}} \left| \{ corr^t_{oov} : corr^t_{oov} \in C^t_{oov}, oov \in OOV_t \} \right|}$$

$$\text{Accuracy} = \frac{\sum_{t \in T_{dataset}} \left| \left\{ csel^t_{oov} : csel^t_{oov} = corr^t_{oov}, csel^t_{oov} \in C^t_{oov}, oov \in OOV_t \right\} \right|}{\sum_{t \in T_{dataset}} \left| \{ oov : oov \in OOV_t \} \right|}$$

5.2. System performance module-wise evaluation

Fig. 3 and Table 6 show different OOV normalization performance values of the system against the evaluation dataset.

When activating different analyzer modules, it is observed that the *accuracy* of the system improves significantly as more modules are activated, but increasing the number of generated candidates per OOV as well.

This performance increment is mainly due to the candidate generation role of modules. Each module contributes with their own candidates, typically increasing the number of different candidates to be considered in each candidate set thus having a great impact on the system *candidate coverage*. Larger candidate sets composed of candidates from different modules tend to have the proper correction within themselves because each candidate has been generated to address a particular error phenomenon. In contrast, this increment of *candidate coverage* comes at the expense of introducing noise and larger computational requirements.

Table 7 shows the correct candidate contribution of modules to each OOV error phenomenon. Notice that several modules may independently propose the same candidate (causing some overlapping) and some phenomena are not fully covered, thus the sum of the columns does not have to be 100%.

It is worth mentioning that we used a threshold of $k \leqslant 2$ for the edit distance module because most of the correct candidates are within that range. Though is true that selecting a higher k includes more candidates, most of the newly included candidates are not a valid solution and usually they have a low confidence score and will not be selected.

5.3. System performance phenomenon-wise evaluation

The full system performs differently depending on the underlying error phenomenon of each OOV, being some phenomena easier to *normalize* than others. Table 8 and Fig. 4 detail the system performance (all modules activated) regarding each OOV underlying error phenomenon.

Some error phenomena as ASC, HOMO and REP are well understood and easier to normalize and thus, very effectively addressed. Other phenomena as ONO, ORT are also well understood but harder

Table 9
Full system performance on *abortion* evaluation dataset.

Phenomenon	Selection precision (%)	Candidate coverage (%)	Total accuracy (%)
ASC	76.67	100.00	76.67
HOMO	95.45	95.65	91.30
ORT	87.50	87.13	76.24
REP	100.00	100.00	100.00
TXT	96.63	94.98	86.96
Overall	94.40	91.81	82.71

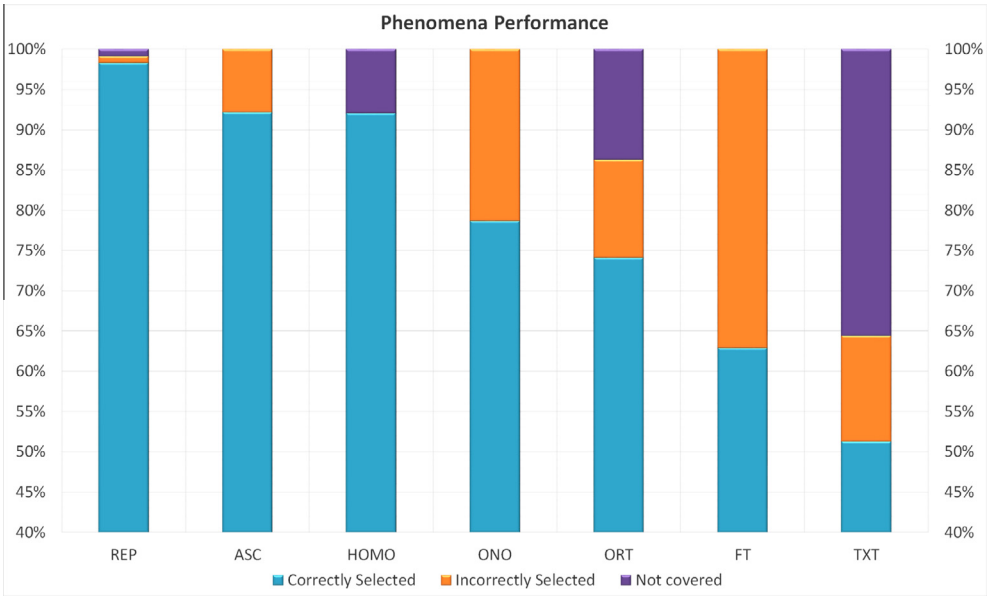

Fig. 4. Full system's performance for each analyzed phenomenon.

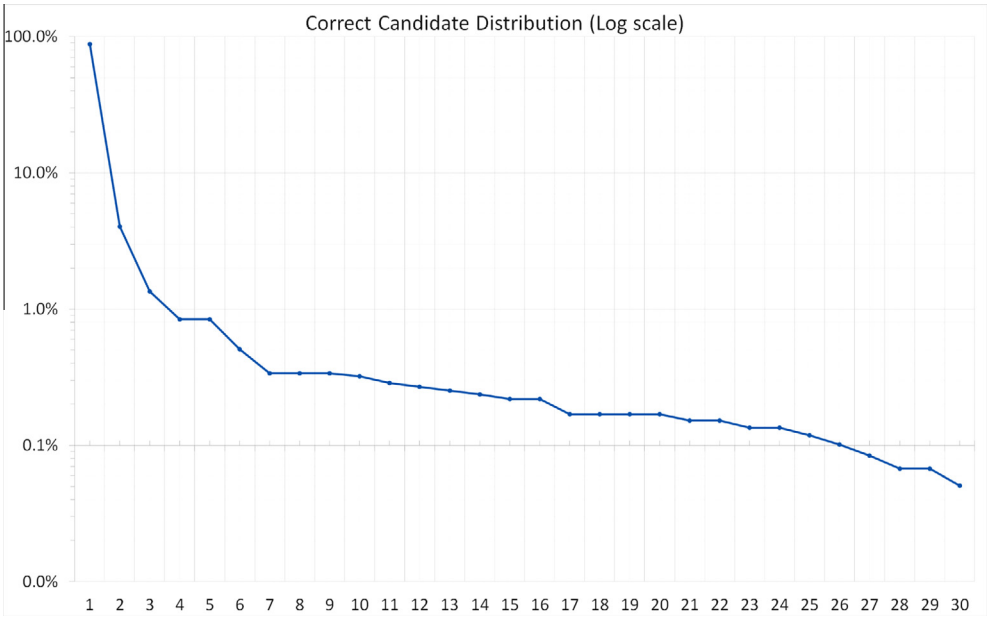

Fig. 5. Correct candidate ranking distribution.

Table 10
Maximal performance gain for a perfect candidate selection process.

Candidates considered	Selection precision upper bound (%)	Accuracy upper bound (%)
Candidates at position $p \in [1,3]$	93.60	85.78
Candidates at position $p \in [1,5]$	95.28	87.33
All candidates	100.00	91.65

to normalize, mainly due to factors as lower candidate coverage and easier confusion with other phenomena.

The FT phenomena yields lower performance because it is easily mistaken for ORT phenomena. In Spanish tweets, it is commonly found that people substitute Spanish words for "equivalent" terms from other languages and sometimes these terms are very similar to valid Spanish words. That leads to a high confusion between FT and ORT phenomena, yielding a poor selection precision.

The TXT phenomena achieves lowest performance. It is harder to address because new ad hoc acronyms are constantly created and each valid language term has multiple SMS-like abbreviated variants.

5.4. System domain adaptability evaluation

In order to measure the flexibility of our system, we have collected another dataset of different nature: Spanish tweets during the presentation of the final draft of the amendments of the law that regulates abortion in Spain. This period spanned from 20th

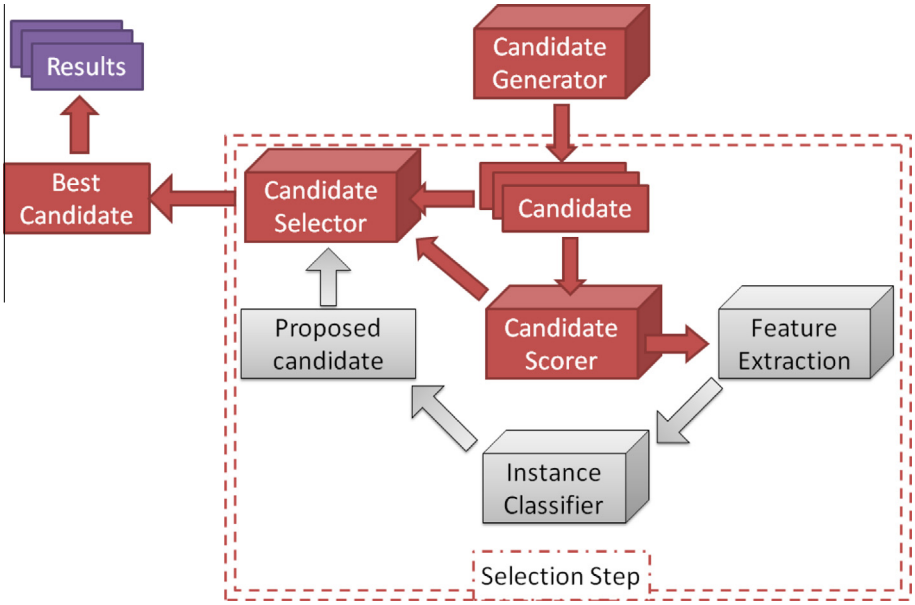

Fig. 6. Inclusion of *a posteriori* classification step into the selection step for the proposed system.

Table 11
System performance with different modules activated.

Active modules	Candidate coverage (%)	Selection precision (%)	Total accuracy (%)
Lang, GD, Transformation Rules (TR)	70.48	90.13	63.52
All modules (Full system)	91.65	88.20	80.83
Full + Classification	91.65	89.21	81.76
Full + Classification + Feature reduction	91.65	90.39	82.84

December 2013 to 23th December 2013. The proposed reform caused a great impact on the population of Spain and every major political party actively positioned regarding this matter. The generation process for this *abortion* dataset was the same as with the *Euro2012* dataset: the dynamic retrieval process described Cotelo et al. (2014).

Since the language and the platform remain the same, addressing this new dataset only required to generate a new *Domain* resource; the rest of resources and parameters were the same as in the *Euro2012* experiment. This new *Domain* resource contained the names of the most relevant politicians and ministers, specific terms regarding the abortion topic and other specific terms regarding Spanish legislative system. Composing this new resource required only 1 h of manual effort.

In order to generate the evaluation dataset, we proceeded very similarly to the *Euro2012* dataset: we obtained a statistically significant sample from the *abortion* dataset ($\alpha = 0.05$, error bound $= 1\%$) made of tweets that contained at least one OOV. This evaluation dataset was manually annotated like the annotation process in the *Euro2012* experiment.

Table 9 shows the performance values of the observed phenomenon when our system is configured for addressing this new domain. The overall performance values of the system falls in the same range of the previous experiment, being only a slight performance difference (1.88%) respecting the previous experiment. This difference may be attributed to the fact that the quality of the tweets in the *abortion* dataset is greater and the OOV phenomena is slightly easier to address.

We can conclude that with a minimal manual effort (about 1 h) of resource generation, our system is able to successfully adapt to a new domain.

5.5. Tuning the candidate selection step

The selection step, as previously shown in Section 3, is very straightforward: it generates a ranking of candidates using the confidence levels provided by each analyzer module and selects the candidate with better score. The goal of this step is to choose a candidate that is the proper correction (*correct candidate*) of the OOV.

The overall performance of the candidate selection step is calculated using the *Selection Precision* metric previously mentioned and the results are shown above. These results are equivalent to the number of *correct candidates* that have been ranked first. Fig. 5 shows the distribution of *correct candidate* positions within the ranking.

The distribution shown in Fig. 5 resembles to a power-law distribution: it is easy to see that most of the *correct candidates* are within first positions followed with a long tail. Only taking into account the first 5 positions of the ranking, more than 95% of the correct candidates are covered.

We came to the conclusion that a reranking of the candidates that lie within the first positions could improve the overall performance of the system. Table 10 shows different *Selection Precision* and *accuracy* performance upper bounds if this reranking process is made.

We devised a variation from the original candidate selection step process, extending it with an auxiliary classifier-based process. This extension fine tunes the selection process, providing rerankings for the first n elements of the original scoring-based candidate ranking. Fig. 6 represents this extension within the original selection step process, only showing the modified piece within the previously shown architecture.

We selected an ensemble *Random Forest* classifier to perform this extension. A *Random Forest* classifier is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the provided data, making use of averaging to improve the predictive accuracy and reducing over-fitting.

This *a posteriori* classification step tries to maximize the selection precision performance of the system. Based on the experimental data discussed above, we only considered the first 5 candidates for the classifier-based process.

We observed a significant improvement using this extension to the selection step with the full system. Table 11 and Fig. 7 show the

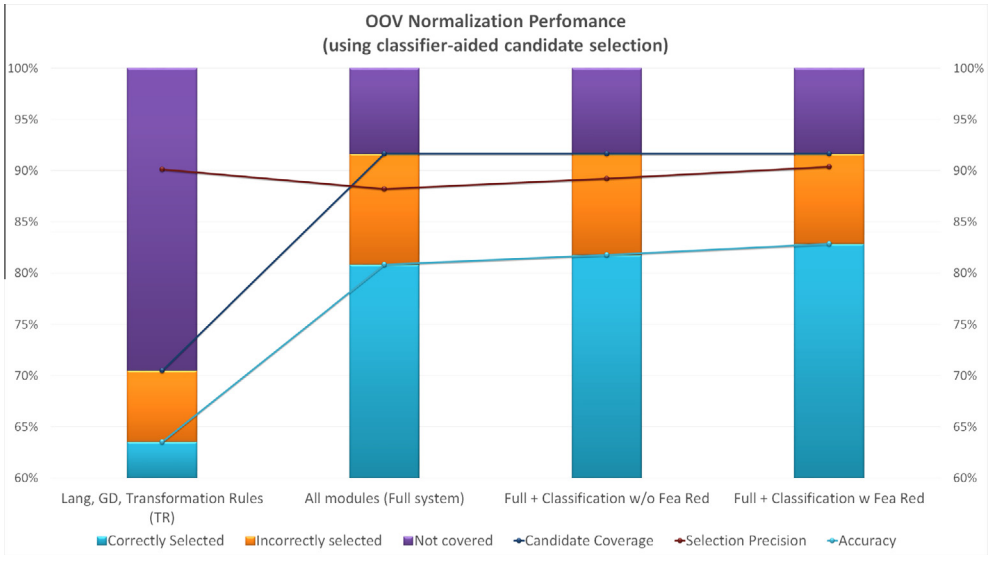

Fig. 7. System performance with *a posteriori* classification step.

system performance with the proposed classifier-based extension, showing that the classifier-based extension indeed refines the previous candidate ranking.

If an automatic decision-tree based feature reduction is done, the system yields a *Selection Precision* improvement from 88.20% to 90.39%, leading to an *accuracy* improvement from 80.83% to 82.84%. This 2% improvement is quite significant; it is very difficult to surpass the 90% *Selection Precision* score due to the difficulty of the task at these levels.

It is worth mentioning that we used a k-fold cross-validation scheme ($k = 10$) in the classifier evaluation process avoiding overfitting.

6. Conclusions and future work

In this paper, we have proposed a novel resource-based lexical approach for addressing the lexical normalization of tweets problem. This approach is based on the idea that our system must behave as an "expert board" and proves to be more resilient and easier to expand than typical single technique approaches that usually appear in the current literature.

Our proposal is based on an extensible architecture made of independent modules, each one focused on addressing specific error phenomena. This focus allows to lower the costs of making any module while increasing accuracy. Furthermore, expanding the system for addressing other error phenomenon only requires the addition of specifically tailored module for that phenomenon.

The combination of the modular architecture and the lightweight resources makes this approach very easy to adapt to different domains, genres and error phenomena while having a very low cost to implement. Once the system is implemented for some language, it only takes about 4 h to adapt it for a new domain and genre, which further lower the effort needed for adapting the system to another normalization context.

We have shown that increasing the number of specialized processing modules does actually increases the overall performance of our system but introduces some noise, making the candidate selection step a little harder. In order to palliate this effect, we proposed an improvement to the existing candidate selection step consisting in introducing a feature-based classifier to perform some candidate reranking. The results show that when we use an *a posteriori* classification step for reranking candidates, our system performs significantly better due to an increment of the precision in the candidate selection step.

Including the improvement mentioned above, the overall performance of our system is quite significant: it achieves more than 82% of accuracy compared to the 31% yielded by the baseline.

The proposed system also have its drawbacks. As we mentioned before, increasing the number of modules that participate in the pseudo-democratic candidate proposal also add significant noise the decision making process and this noise increase is inherent to this kind of ensemble schema. This noise imposes a loose upper bound on performance but modules can be designed to reduce overlapping as possible, incurring in less noise penalty.

Also, special care must be taken when designing the preprocessing stage, as in any lexical-level based approach. Proper tokenization and OOV detection process has to be quite good for proper system functioning since the rest of the stages (candidate proposal and selection) are after the preprocessing stage in the system pipeline.

Our system has room for improvement in several directions. Currently, we use a standard regex-based approach for tokenization. Including a segmenter module into the preprocessing stage would yield better tokenization, thus improving the rest of the system.

Each OOV token is independently addressed at lexical level, not taking into account the rest of the tweet. Both generation and selection stages would greatly benefit if any context information, automatically discarding candidates that would not fit due to morphosyntactical constraints.

Another interesting research direction consists on improving the candidate scoring and selection methods. In addition of using ad hoc heuristics, modules can make use of machine learning methods for establishing confidence values, improving the selection step.

Currently, we are devising a way of combining unstructured information (text) with structural information (via hashtags, retweets, mentions and replies). We think that combining these two types of information and applying machine learning methods may have a significant impact for the analysis of tweets.

Acknowledgements

This research is partially funded by the national project TIN2012-38536-C03-02 and the regional project P11-TIC-7684 MO.

References

Ageno, A., Comas, P. R., Padró, L., & Turmo, J. (2013). The talp-upc approach to tweet-norm 2013. In *Proceedings of the tweet normalization workshop at SEPLN 2013*. Sociedad Española para el Procesamiento del Lenguaje Natural.

Conover, M., Ratkiewicz, J., Francisco, M., Gonçalves, B., Menczer, F., & Flammini, A. (2011). Political polarization on twitter. In *ICWSM*.

Costa-Jussa, M. R., & Banchs, R. E. (2013). Automatic normalization of short texts by combining statistical and rule-based techniques. *Language Resources and Evaluation, 47*, 179–193.

Cotelo, J. M., Cruz, F. L., & Troyano, J. A. (2014). Dynamic topic-related tweet retrieval. *Journal of the Association for Information Science and Technology, 65*, 513–523.

Eisenstein, J. (2013). What to do about bad language on the internet. In *Proceedings of NAACL-HLT* (pp. 359–369).

Gamallo, P., García, M., & Pichel, J. R. (2013). A method to lexical normalisation of tweets. In *Proceedings of the tweet normalization workshop at SEPLN 2013*. Sociedad Española para el Procesamiento del Lenguaje Natural.

Ghiassi, M., Skinner, J., & Zimbra, D. (2013). Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert Systems with Applications, 40*, 6266–6282.

Han, B., & Baldwin, T. (2011). Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies. HLT '11* (Vol. 1, pp. 368–378). Stroudsburg, PA, USA: Association for Computational Linguistics.

Han, B., Cook, P., & Baldwin, T. (2012). Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning* (pp. 421–432). Association for Computational Linguistics.

Han, B., Cook, P., & Baldwin, T. (2013). Lexical normalization for social media text. *ACM Transactions on Intelligent Systems and Technology, 4*, 5:1–5:27.

Himelboim, I., McCreery, S., & Smith, M. (2013). Birds of a feather tweet together: Integrating network and content analyses to examine cross-ideology exposure on twitter. *Journal of Computer-Mediated Communication, 18*, 40–60.

Ikeda, K., Hattori, G., Ono, C., Asoh, H., & Higashino, T. (2013). Twitter user profiling based on text and community mining for market analysis. *Knowledge-Based Systems, 51*, 35–47.

Jabeen, S., Shah, S., & Latif, A. (2013). Named entity recognition and normalization in tweets towards text summarization. In *2013 Eighth international conference on digital information management (ICDIM)* (pp. 223–227). IEEE.

Kontopoulos, E., Berberidis, C., Dergiades, T., & Bassiliades, N. (2013). Ontology-based sentiment analysis of twitter posts. *Expert Systems with Applications, 40*, 4065–4074.

Martinez-Romo, J., & Araujo, L. (2013). Detecting malicious tweets in trending topics using a statistical analysis of language. *Expert Systems with Applications, 40*, 2992–3000.

Mostafa, M. M. (2013). More than words: Social networks' text mining for consumer brand sentiments. *Expert Systems with Applications, 40*, 4241–4251.

Pennell, D., & Liu, Y. (2011). A character-level machine translation approach for normalizationof sms abbreviations. In *IJCNLP* (pp. 974–982).

Phi-Long (2012). Python 3.3+ implementation of the language guessing module made by Jacob R. Rideout for KDE.

Porta, J., & Sancho, J.L. (2013). Word normalization in twitter using finite-state transducers. In *Proceedings of the tweet normalization workshop at SEPLN 2013*. Sociedad Española para el Procesamiento del Lenguaje Natural.

Schulz, K., & Mihov, S. (2002). Fast string correction with levenshtein-automata. *International Journal of Document Analysis and Recognition, 5*, 67–85.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal, 4*, 379–423.

Sidarenka, U., Scheffler, T., & Stede, M. (2013). Rule-based normalization of german twitter messages. In *Proceedings of the GSCL Workshop Verarbeitung und Annotation von Sprachdaten aus Genres internetbasierter Kommunikation*.

Tumasjan, A., Sprenger, T. O., Sandner, P. G., & Welpe, I. M. (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment. *ICWSM, 10*, 178–185.

Yang, Y., & Eisenstein, J. (2013). A log-linear model for unsupervised text normalization. In *EMNLP* (pp. 61–72).