# TweetSafa: Tweet language identification

***TweetSafa: Identificación del lenguaje de tweets***

**Iosu Mendizabal**
(IIIA) Artificial Intelligence Research Institute
(CSIC) Spanish Council for Scientific Research
iosu@iiia.csic.es

**Jeroni Carandell & Daniel Horowitz**
(UPC) Universitat PolitÃ¨cnica de Catalunya
(URV) Universitat Rovira i Virgili
(UB) Universitat de Barcelona
jeroni.carandell@gmail.com
daniel.horowitzzz@gmail.com

**Resumen:** Este artículo describe la metodología utilizada en la tarea propuesta en SEPLN 14 para la identificación de lenguaje de tweets (TweetLID), como se explica en (Iñaki San Vicente, 2014). El sistema consta de un preprocesamiento de tweets, creación de diccionarios a partir de N-Grams y dos algoritmos de reconocimiento de lenguaje.
**Palabras clave:** Reconocimiento de lenguaje, lenguaje de tweets.

**Abstract:** This paper describes the methodology used for the SEPLN 14 shared task of tweet language identification (TweetLID), as explained on (Iñaki San Vicente, 2014). The system consists of 3 stages: pre-processing of tweets, creation of a dictionary of n-grams, and two algorithms ultimately used for language identification.
**Keywords:** Tweet identification, tweet language.

## *1 Introduction and objectives*

Language identification is vital as a preliminary step of any natural language processing application. The increasing use of social networks as an information exchange media is making of them a very important information center. Twitter has become one of the most powerful information exchange mechanisms and every day millions of users upload tons of tweets.

The SEPLN 2014 TweetLID task focuses on the automatic identification of the language in which tweets are written, as the identification of tweet language is arousing an increasing interest in the scientific community (Carter, Weerkamp, and Tsagkias, 2013). Identifying the language will help to apply NLP techniques subsequently on the tweet such as machine translation, sentiment analysis, information extraction, etc. Accurately identifying the language will facilitate the application of resources suitable to the language in question.

The scope of this task will focus on the top five languages of the Iberian Peninsula:Spanish, Portuguese, Catalan, Basque, and Galician as well as English. These languages are likely to co-occur along with many news and events relevant to the Iberian Peninsula, and thus an accurate identification of the language is key to make sure that we use the appropriate resources for the linguistic processing.

The rest of the article is laid out as follows: Section 2 introduces the architecture and components of the system: the preprocessing state where the tweets are adapted to a better comprehension for our algorithm and the used algorithms. Afterwards, section 3 describes our results for the given problem. To conclude, in section 4 we will try to draw some conclusions and propose future works.

## *2 Architecture and components of the system*

We have presented two different approaches to the problems which have been presented in track one (constrained) and track two (unconstrained). Both of these methods share great part of the process in terms of the set of tweets being used to learn from, as well as the way incoming tweets are preprocessed and learned.

### 2.1 Pre-processing

The first step of this process, is to identify the noise present in all tweets regardless of the language. There are common issues related to regular text, such as multiple space characters, but also specific Twitter tokens like the user name tag or emoticons. After identifying this issues, we are able to remove them using mostly regular expressions. We have highlighted the main issues found in the

tweet domain and what our approach was towards it:

- Different case characters: All characters were lowercased, so they wouldn't interfere in the identification process, since the same character with different cases is treated as two different elements.
- Numbers, Emoticons: Since these kind of characters are presented equally in any language, they have been removed.
- Vowel repetitions: The vowel repetition is a common issue when dealing with chatspeak. These kind repetitions could damage the algorithm's performance, therefore they were completely removed and reduced to a maximum of two from the text using regular expressions.
- Multiple spaces: This is also a common issue when dealing with tweets. The regular expression formats the text from multiple spaces into one space character.

When working with N-grams, it is important to observe that not all special characters are to be removed from the text, since they could interfere in the identification process. Characters like the apostrophe, are more likely to appear in English and Catalan than in others, therefore this kind of special characters must not be considered as noise, and we save them for a better result.

## 2.2 N-GRAM distribution

To classify the tweets into languages using N-grams we have to extract meaningful distributions from each language. To do so, we created documents of concatenated tweets for each language: English, Spanish, Catalan, Portuguese, Galician, Basque, other and undetermined. Mixed labelled tweets such as the ones with 'en+es' as well as those with ambiguous languages 'en/es' are added to both languages they contain (in this case to both Spanish 'es' and English 'en'). Then we extract N-gram distributions in a dynamic way so that we can choose the number of N we wish.

## 2.3 Algorithms

Once we have N-gram distributions for each language, given a new tweet we want to classify we are going to find the most possible language by extracting the tweet's N-gram distribution and comparing it with the languages distributions. To do so, we took two different approaches.

### 2.3.1 Linear Interpolation

The first method tries to find out what the probability is of a sentence being generated by each language by multiplying the probability of the consecutive N-Grams of the sentence in their respective languages. The problem appears when we deal with a small finite dataset and there are therefore not enough instances to reliably estimate the probability, in other words, the sparse data problem appears. This means that if a corpus of a certain language does not have a certain N-Gram, a sentence with the latter would automatically have a probability of zero.

To avoid this problem in the computation of the probabilities of each tweet for the languages of our N-Gram distribution we use the linear interpolation smoothing method, also known as the Jelinek-Mercer smoothing (Jelinek, 1997),(Huang, Acero, and Hon, 2001). To be able to use this smoothing method we have to make a computation with our N-Gram corpus, the one generated with the 14991 tweets for the training purpose, to calculate the $\lambda$ values. We create a dynamic program to compute as many $\lambda$ values as the N-Grams we extracted from the training set. For instance, if we consider up to 5 N-Gram distributions for English we will compute 5 $\lambda$'s for each N-Gram up to 5, so all $\lambda_i$ corresponding to the i-Gram where $i \in \{1, ..., 5\}$. The probability of an N-Gram will be computed as follows:

$$\mathrm{P}(\mathrm{t}_n|t_1, t_2, ..., t_{n-1}) = \sum_{i=1}^{n} \lambda_i \hat{P}(t_n \mid \bigcap_{j=1}^{i-1} t_{n-j}) \quad (1)$$

For any n and where $\hat{P}$ are maximum likelihood estimates of the probabilities and $\sum_{i=1}^{n} \lambda_i = 1$, so P represent probability distributions.

The values of $\lambda$ are computed by deleted interpolation (Brants, 2000). This technique successively removes each max-gram (biggest n-gram) from the training corpus and estimates the best values for the $\lambda$'s from all other n-grams in the corpus by adding a confidence to the lambdas for the most proportionally seen N-Gram . The algorithm is

given in Algorithm 1.

set $\lambda_1 = \lambda_2 = \lambda_3 = ... = \lambda_n = 0$;

**foreach** *MAX-Gram* $(t_1, ..., t_n)$ *with count* $(t_1, ..., t_n) > 0$ **do**

depending on the maximum of the next values:

**case** $\frac{count(t_n)-1}{N-1}$: *increment* $\lambda_1$ *by* $count(t_1, ..., t_n)$

**end**

**case** $\frac{count(t_{n-1},t_n)-1}{count(t_{n-1})-1}$: *increment* $\lambda_2$ *by* $count(t_1, ..., t_n)$

**end**

**case** $\frac{count(t_{n-2},t_{n-1},t_n))-1}{count(t_{n-2},t_{n-1})-1}$: *increment* $\lambda_3$ *by* $count(t_1, ..., t_n)$

**end**

...

**case** $\frac{count(t_1,...,t_n)-1}{count(t_1,t_2,...,tn-1)-1}$: *increment* $\lambda_n$ *by* $count(t_1, ..., t_n)$

**end**

**end**

**Algorithm 1:** Deleted interpolation Algorithm.

### 2.3.2 Out-of-place measure

For this next method, for every n we will only consider a ranking list of n-grams ordered by most to least frequent and where only the order is preserved as opposed to the exact frequencies. We decided to do this because when it comes to comparing a single tweet (documents of only 140 characters) to distributions of each language, we cannot consider that the frequency distribution of the tweet n-grams will resemble the ones in the concatenated document. We can however, say that the most frequent have a higher probability of appearance, but not necessarily with proportional frequencies as in the document. For this reason, we used the out-of-place measure.

We decided to send this method as unconstrained because two of the parameters which we used, that will be discussed later on, were extracted from a previous work we did with a self downloaded corpus of tweets of different languages. We did this because it would take too long if we had to find the new values because of the huge search space.

This measure is a distance which will tell us approximately how far the tweet is from a language for a fixed n-gram. Given the tweet ranking $\{T_i^n\}_i$ and a language L n-gram ranking $\{L_j^n\}_j$, the distance is computed by the sum of the number of indexes that an element of T has been displaced in list D. So we sum $\mid i-j \mid$ for every $T_i^n$ in the tweet that is equal to $L_j^n$. In the case that an element in $\{T_i^n\}_i$ does not exist in list $\{L_j^n\}_j$, we suppose the best case, i.e. that the non appearing element is in the bottom of the list. This as we will discuss in section 4 might not have been such a good idea. Finally, to be able to compare different distances we need some kind of proportion of the out-of-place measure that we describe as:

$$\frac{outOfPlaceMeasure}{length(T^n_{i\ i}) * length(L^n_{j\ j})} \quad (2)$$

As we can see in Figure 1, the out-of-place measure is calculated for a tweet from an English dataset. The $m$ and $n$ parameters give us the maximum number of elements we allow for each list so that computational time does not get compromised by an unnecessary whole search of all the n-grams in a language (Cavnar, Trenkle, and others, 1994). This is the part of this algorithm that makes it unconstrained, since the parameters we used came from a previous similar project we did using self downloaded tweets and where we found that the values of $m$=80 and $n$=50 were best. To avoid possible divisions by

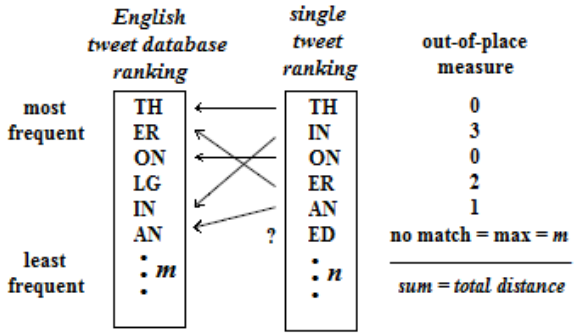


Figure 1: Example of an out-of-place measure

zero in equation 2, given that tweets are sometimes zero or very close (especially after the cleaning of html's, punctuation, etc), we supose that if the number of characters is smaller than three, the tweet is undetermined. Again, a bold affirmation which needs to be fine-touched in future work.

Finally, in the training process, we are going to reward each n-gram if it correctly

guessed a tweet. So if for example, a trigram labels a tweet correctly but the unigrams and bigrams do not, we reward the trigrams with one point where the others don not get any. We do this with all the tweets in the training set and in the end we get frequency of reliability of each n-gram. When the test is done on a tweet, a weighted voting is done using these confidence parameters so that the most voted languages counting the reliability weight wins.

## *3 Setup and evaluation*

The official result of our approach are the next ones: In the constrained category using the linear interpolation algorithm, section 2.3.1, we obtained a precision of 0.777, a recall of 0.719 and a F-measure of 0.736.

In the unconstrained category we used the out-of-place measure algorithm, section 2.3.2, and obtained the next results: precision of 0.598, recall of 0.625 and F-Measure of 0.578.

### 3.1 Empirical settings

Before submitting the final results we made different executions with different maximum N-Grams to know which was the one with the best results. Also because of the ambiguity of tweets with more than one language, for instance es+en, to compute this we take average value of all the probabilities of all the languages and then create a threshold. For the linear interpolation we used:

$$Threshold = \frac{maxProbability - Average}{\alpha} \quad (3)$$

Where the $maxProbability$ refers to the maximum of the probabilities of the languages and $\alpha < 0$ is the value of restriction that tolerates more or less the number of languages that may be suggested. The bigger the $\alpha$, the less tolerance to ambiguity of predicted languages for each tweet yet the more precise the result, while the smaller the alpha, the higher the recall yet smaller the precision.

For the ranking-based method, the threshold is chosen by running a search from 0 to 0.3 with intervals of 0.05. The most optimum found on the data set is 0.05.

### 3.2 Empirical evaluation

We ran experiments with different N-Gram values, from 1 to 8, and we set the $\alpha$ value to 10 which gave us the best results in the validation set.

In figure 2 we can see the results of the experiments we made using the linear interpolation method. We can observe how the results are going better while the N-Grams are going bigger, but the peak of the results are achieved with the 5-gram, from there on the results are slightly worst each gram we sum.

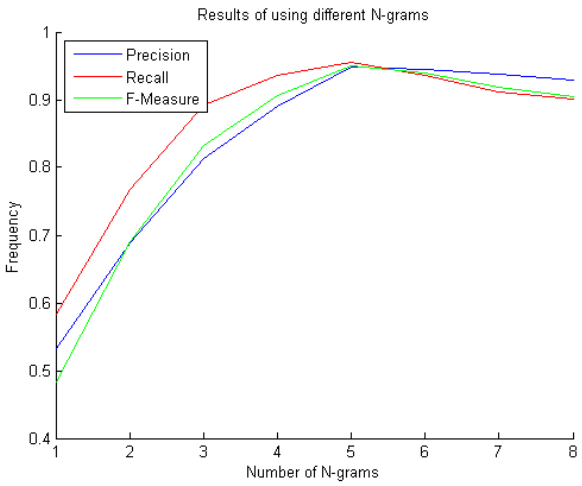


Figure 2: Results obtained for the training set with: Linear interpolation method.

Because of these results, we decided to send the 5-gram results for the test set given for the SEPLN 2014 task.

In the case of the ranking based method, we do not have to test different n-gram combinations since we obtain a reliability for each n-gram to be truthful. So if a certain n-gram were systematically wrong it would have a very low confidence which would not make it so influential. Finally we decided for computational reasons to use only 6 n-grams.

## *4 Conclusions and future work*

In this paper we have described our approach for the SEPLN 2014 shared task of tweet language identification (TweetLID). Our system is based on a pre-processing part taking into account the different accents can appear in different languages using language codifications in the N-Gram distribution state without erasing them.

Also we have two different algorithms the linear interpolation smoothing and the out-of-place measure. These algorithms obtain an F-measure of 0.736 and 0.578 respectively in the given test corpus of 19993 tweets. Our system ranked in the $3^{rd}$ best place among

the participants of the constrained track, using the linear interpolation algorithm, and $6^{th}$ in the unconstrained track, using the out-of-place measure.

Among the mistakes we made was to underestimate numerical digits in languages, which we removed. In the English language, numbers are often used to shorten text, thus making us lose great part of words for example; "to forgive someone" might be written as: '2 4give som1'. This is true in many internet alphabets which are emerging such as Arabizi(the arabic chat language).

For possible future work for the ranking-based method it might be interesting to consider the distribution of the length of words in each language since it can be a very determining characteristic. Also in this method, the out of place measure should have penalized more severely the non non-appearing characters in the document list instead of supposing it could be found on the last element of the list.

Finally we have to stress the importance the pre-processing of tweets as one of the key parts in the project.

## *References*


Brants, Thorsten. 2000. Tnt: A statistical part-of-speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 224–231, Stroudsburg, PA, USA. Association for Computational Linguistics.

Carter, Simon, Wouter Weerkamp, and Manos Tsagkias. 2013. Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. *Lang. Resour. Eval.*, 47(1):195–215, March.

Cavnar, William B, John M Trenkle, et al. 1994. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175.

Huang, Xuedong, Alex Acero, and Hsiao-Wuen Hon. 2001. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.

Iñaki San Vicente, Arkaitz Zubiaga, Pablo Gamallo José Ramom Pichel Iñaki Alegria Nora Aranberri Aitzol Ezeiza VÃctor Fresno. 2014. Overview of tweetlid: Tweet language identification at sepln 2014. In *In TweetLID @ SEPLN 2014*.

Jelinek, Frederick. 1997. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, USA.