Language detection using N-gram modal

Project Outline and ingredients:
Given a set of documents the aim of the project is to predict the language for the documents. The project scopes for two languages *viz.* English and Spanish.

Preprocessing: A simple rule for Preprocessing of filtering everything except letters [a-z] after converting the entire document into lowercase is applied.

Training data: For training a **modal for Spanish language**, a document consisting of **74946** lines of **majorly Spanish** language text and the **modal** trained for the **English language** consists of **67276** lines **all of the English** text.

Test data: 8 different files for both the languages were used to test if the prediction of the language on the basis of the created modals works fine.

Later Modifications: Potter Stemmer for preprocessing, Linear interpolation for probability smoothing and Trigrams approach.

The first step for creating the modal is to **preprocess** the training data, after applying the preprocessing techniques the total number of words found were 606688 for the English language and 635509 for the Spanish language.
If the word based approach is chosen, assuming that both the language modals will have **at least** 635509 words, the **space complexity** will be the **order of square of (635509)**. In contradiction to that, if a character level model is created, the total number of characters will be **27** including the '**$**' sign which indeed **reduces the space complexity** drastically. Thus, it is more **preferable** to create a modal at the **character level**.
Note: $ sign is considered in the vocabulary because the possibility of a bigram or a trigram starting with a $ sign is an important factor that contributes to the prediction of the language.

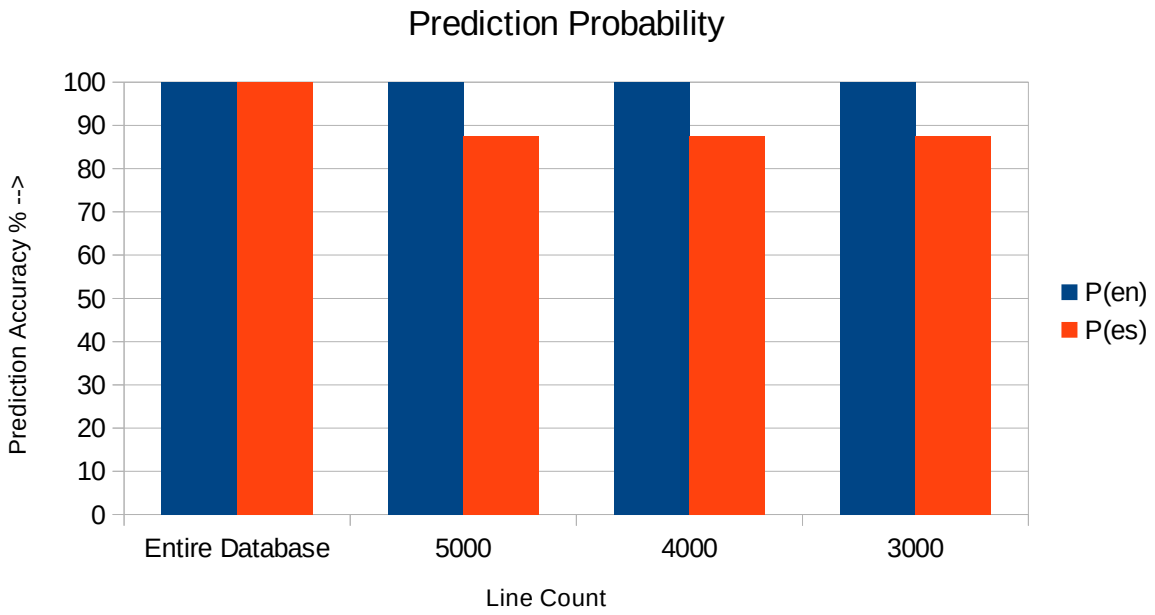
Suggested Changes to the proposed scheme:
The preprocessing gives the result of all the lowercase letters which is highly preferable for the English language, but on close observation of the retrieved words for the Spanish language, the characters that have special meaning for the pronunciation are filtered out. For instance words like **Andía** is transformed into **anda** and **capitán** is transformed into **capitn** which leads to a decrease in possible occurring bigrams. Also, the training data for the Spanish language contains several lines that contains English and thus is a combination of English as well as Spanish language.

Minimum number of words for prediction of languages while testing:
Assumption: The modals are trained using the entire test data provided. I.e all_en.txt and all_es.txt. The files to be tested are having either Spanish or English and not both together.

Two separate files were generated smallEng.txt and smallEs.txt which had 20 preprocessed tokens and the program was successful in predicting the correct language of the documents.
The test was also performed for single words like hello and hola and the languages were predicted perfectly even for a single word but there is always a possibility of getting the wrong answer if there are less number of tokens in the text. So pessimistically 30 tokens are needed to predict a language using our trained modal. (Depends on the corpus)

Comparing different modals:
If many modals are created for English and Spanish the only means of comparing them is by using them to predict the language of a few files. A common test data should be chosen and then the probabilities should be checked on the document level for each file in test dattaset. The one with the best probability for English language prediction will be having good rules for preprocessing and vibrant data for training the modal in comparison with the other modals for English language and vise versa.

Minimum training data for precise prediction



Graph 1.1

The predictive percentage in this case has two types of variants:
1) Predictions on the basis of lines.
→ This approach shows that the number of tokens for given number of lines are substantially lower in Spanish modal than that of the Modal.
For instance for given number of lines, the number of words are as follows:

Number of lines	Number of Words English	Number of Words Spanish
Entire Database	606688	635509
1000	9050	7293
2000	18482	14638

3000	28377	21520
4000	38182	29384
5000	48008	36880
6000	58207	42951

Table 1.1

2) Prediction on the basis of number of words.
When ever the number of lines are adjusted such that the number of tokens have minimum difference in the training data, the probability stands out to be 100% in most of the cases. Even though these many tokens show good accuracy there are chances of error in prediction and thus it is preferred to train a model with more literature.
When to stop?
→ When the prediction is correct even if there is a huge number of difference in tokens.

Number of tokens	Number of Lines (En)	Number of Lines (Es)
4300	500	630
9000	1000	1250
15500	1700	2100

Table 1.2

The notable difference in the number of tokens is because of the rules of preprocessing where the special symbols that add meaning to the language are removed.

So, in order to get precise accuracy of the language the training data is adjusted such that both the language modals are trained using the almost similar number of words. In our case **26000 lines of training data**, there are enough words for both the modal to predict the languages correctly.

Prediction of 3rd language:
The prediction of the french document from the source: *http://www.gutenberg.org/files/36460/36460-0.txt*
was made using the following test cases and the bi-gram modal.

Number of lines for training modals	Test result prediction
26000	English
30500	English
35000	English
72500	Spanish
Entire data set	Spanish

Table 1.3

The prediction was correct for modals having more training data. As modal doesn't know any language specifically, the words of french language have more similarity to Spanish language thus its prediction is justified.

Modifications and optimizations

Using tri-grams over bi-grams:

Assumptions: The given testing data file is composed of a **single language**.

Using tri-grams in place of bi-grams proved to be more effective in predicting language precisely even with small amount of training data.

Note: No modifications were made for preprocessing the training data, the same method was used.

The approach gives precise results with just **12000 lines** of the training data. It also proves that French is closer to Spanish than that of English even with this small amount of data.

Following extra reference was used for the of the french language prediction using tri-grams:

http://www.gutenberg.org/cache/epub/11049/pg11049-images.html

Using linear interpolation technique for smoothing.

The prediction for the Spanish using the Bi-gram Modal is even better with less training data using this technique.

It gives 100% prediction probability which **was once 87.5%** in **add one technique**.(Graph 1.1)

Using porter stemmer tool:

The combination of bigram modal for language prediction, linear interpolation technique for probability smoothing and porter stemmer tool for preprocessing of the input yields the best results in prediction of the language. Different modals were created on the basis of length of training data and even for small training data of **10000 lines the prediction** of language was on point.

The testing results are 100% accurate when modals are created using the entire training data set and the change in prediction of french language is as follows for modals with smaller training data.

Number of lines for training modals	Test result prediction
26000	Spanish
30500	Spanish
35000	Spanish
72500	Spanish
Entire data set	Spanish

Table 1.4

Observations: The different tools and techniques used for prediction yield better prediction results but has different execution time listed in **Table 1.5**.

Tool for tokenization	Tool For Pre-processing	Execution time of the program
-	.split() and re.sub()	7-8 Seconds
Potter Stemmer	.split(), potterstemmer.stem()	40-42 Seconds
Potter Stemmer	potterstemmer.stem(), word_tokenize()	115-120 Seconds

Table 1.5