Universidade de Aveiro

DETI

Teoria Algorítmica da Informação

Lab Work nº 2

Grupo 3

Hugo Leal, 93059

Luísa Amaral, 93001

Tiago Rainho, 92984

28 de dezembro de 2021

Contents

1 Introduction

A Finite-Context model (FCM) is a type of Markov model, and with this model it is possible to collect statistical information from texts. This project's main goal is to use one or more FCM to determine the similarity between a given corpus of reference texts and a target text. By calculating the similarity, it is possible to analyze in which language or languages a text was written and where the segments of the languages are located in the text. To do this, the FCM models serve as descriptions for each language so that it can be estimated how many bits would be required to compress sample texts when compared to these models. The language of the model that will require the less amount of bits will most likely be the language in which the sample text is written, since the model requires fewer bits to describe this text.

To be able to obtain this goal, three different modules were developed. The first one, *lang*, serves as a building point for the FCM models representing each language and can compare a sample text with a model text to estimate how many bits it would be required to compress the sample text. The second module, *findlang*, expands on the previous module to determine in what language a sample text was written. Lastly, the third module, *locatelang* analyzes a sample text and finds the language in which different segments of the text were written.

Regarding the organization of this report, the section Method presents a detailed explanation of the methods and solution adopted to complete this project successfully. In the section Results and Discussion the results obtained from the execution of the developed modules are shown, analyzed and discussed. Lastly, in the section Conclusion the main conclusions that we gather from this work are presented.

2 Method

In order to solve the problem of measuring the similarity between sample texts and models representing languages, it is necessary to find information relative to the amount of bits necessary to describe the sample text using each of those models.

The amount of information is the estimated number of bits required to compress the sample text based on each language model. This means that if the amount of information is a large value, the compression will require more bits because the text has little similarities with the language model. This happens when the text doesn't possess contexts that are frequent in the language model. When this happens, it is less likely that the text was written in that specific language. In contrast, when the amount of information is smaller, fewer bits are required to compress the information, and thus there is more similarity between the language model and the text under analysis, meaning that it is more likely that the text was written in that language.

To achieve the goal of this project, three python modules were developed, the *lang* module, the *findlang* module and the *locatelang* module.

2.1 Lang

The method *train()* receives the text corpus and builds a FCM object that handles the parsing of this training text and the construction of the structure that holds the number of occurrences of characters of the alphabet after a certain context.

The FCM method *probability_e_c()* calculates the probability of event e happening on a given context, and it is used to calculate the number of bits required to compress the information.

In order to get the amount of information of a given text, taking into account all FCMs, the method *estimated_information()* was created which uses the method *amount_of_information()* that receives a FCM to calculate the average estimated number of bits necessary to compress

the given text. The returned value is normalized in order to keep the results meaningful.

Listing 1: Calculation of amount of information

```
# returns the mean entropy for the sample text
def estimated_information(self, text:str):
    total = 0
    sum_k_weights = 0
    for fcm in self.fcm_list:
        weighted_k = self.calculate_fcm_weight(fcm)
        total += self.amount_of_information(text, fcm) * weighted_k
        sum_k_weights += weighted_k
    return total / sum_k_weights
```

The *estimated_num_bits()* is used to get a list of numbers each representing the number of bits needed to compress every character of the given text. This method also takes into account all FCMs therefore returns normalized values which maintain their meaning.

Different length problem

When computing the *estimated_num_bits()* with FCMs with different sizes we end up with different length lists, these lists would not be accurate to compare, for that reason we extend the smaller lists with their last value because it is the value that makes sense to compare because it is approximately the same context.

Listing 2: Calculation amount of bits needed to compress a given text

```
# returns the minimum amount of compressed bits for each character of the
    given string
def estimated_num_bits(self, text:str):
    values = []
    sum_k_weights = 0
    for fcm in self.fcm_list:
```

```
            entropies = self.num_bits(text, fcm)

            # use weighted k to assign different weights on each k
            weighted_k = self.calculate_fcm_weight(fcm)
            weighted_entropies = [weighted_k * value for value in entropies]
            sum_k_weights += weighted_k

            values.append(weighted_entropies)

        lst = []
        for i in range(len(text)):
            total = 0
            for entropies in values:
                # use all fcms even when the larger ones (using their last value
                    -> tail)
                total += entropies[i] if i < len(entropies) else entropies[-1]
            lst.append(total/sum_k_weights)
        return lst
```

For each k-sized context in the text, the probability of an event happening after that specific context is calculated and the logarithm of this probability is saved. Having all these logarithms, the average is calculated and returned as the estimated number of bits required to compress the analysis text.

2.1.1 Normalization

When using multiple FCMs there must be a way of comparing them in order to prioritize the more reasonable over the least reasonable, one way of doing this is to assign different weights based on their characteristics such as the size of the sliding window (k).

In a first approach, the weight assigned to each FCM model in the calculations was the multiplication by the k value of the FCM. This meant that an FCM with a k value of 5 would have a weight of 5 in the calculations, and an FCM with a k value of 9 would have a weight of 9. As will be seen in the results sections of this report, this is not the best approach.

In the second approach that is adopted in the current implementation, the method *calculate_fcm_weight()* computes the weight assigned to each FCM based on its k. For a possibility of future work, multiple parameters can be adjusted in order to tweak certain languages if needed (for example, the sliding window size can be slightly higher in German than in English because on average the German words tend to be larger than English ones). The function which maps the weight for each FCM is represented in Figure 1.

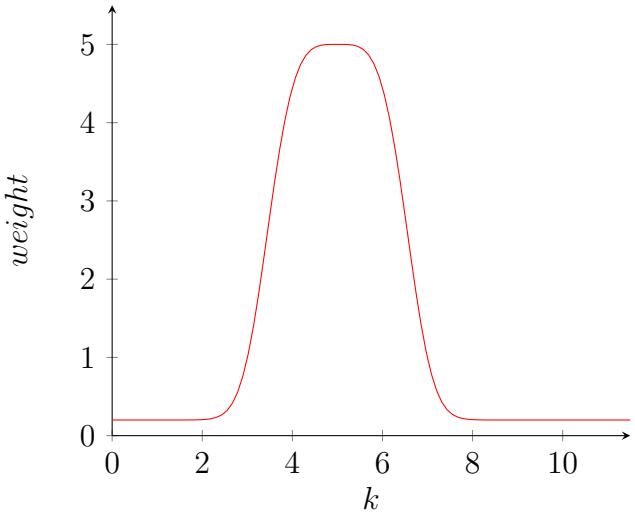

Figure 1: Weighted FCM function based on k

The adjustable parameters are the following:

- optimal k
- minimum k weight
- maximum k weight

- weight factor
- orientation strength
- thinness
- thickness

2.1.2 Running the Lang Module

In order to run this module we only need the files to train our model on, the k and *alpha* values, and the file to analyse.

```
python3 src/lang.py --files
    dataset/eng_AU.latn.Aboriginal_English.comb-train.utf8
  --k 10 --alpha 0.5 --t test_files/english_text.txt.txt
```

2.2 FindLang

The purpose of this module is to provide a guess for the language in which a text was written, using several example texts from several languages to train several FCM models. Each language will have at least one trained FCM model, having up to how many models the user decides to create.

Listing 3: Method for finding the language

```
def find_lang(language_classifiers:List[Lang], text:str, threshold:float) ->
    Tuple[Lang, float] or Tuple[None, float]:
    language_classifications = [(lang_classifier,
        lang_classifier.estimated_information(text)) for lang_classifier in
        language_classifiers]
    language, entropy = min(language_classifications, key=lambda t: t[1])
    return (language if entropy <= threshold else None, entropy)
```

For each language, the estimated amount of information required to compress the analysis text is calculated. Since each language can have multiple FCM models with different orders, the final value is obtained by performing a normalization of the multiple values of number of bits (since there will be one for each FCM model of the language). This normalization is explained in section 2.2.1.

After obtaining the values of estimated amount of information for each language in the data set relative to the text under analysis, the minimum amount of information value is found and the language corresponding to that value is selected. If that value is equal or greater than the threshold, then it is considered that the analysis text is written in that language. If it is smaller than the threshold, then no language is returned.

2.2.1 Normalization

Considering that a language can possess n FCM models, it will result in n values of amount of information. To obtain one single final value, the method *estimated_information* performs a normalization where it sums every value calculated multiplied by a weight given to the k order of the corresponding FCM model. This weight was decided based on the fact that better results were verified when the k values were situated between 3 and 6, so a bigger weight is given to values inside that range and a smaller weight is given to values outside that range. This increases the accuracy of the calculations, since better k orders end up being more significant in the calculations. Then, this sum is divided by the sum of all the orders of the FCM models of the language, and this value is returned as the estimated amount of information for that language. This normalization can be seen in Listing 2.1.1.

2.2.2 Running the FindLang Module

In order to run the *findlang* module it is only required to provide the file to analyse (file), the optional parameters are the languages in which to train the models (languages) which

are already pointing to specific files (the default value is all of the available languages), the threshold in which to limit our acceptance of the result (threshold). Finally it is also possible to override the k and alpha values.

When needing to use multiple FCMs for each Lang, just need to add more k and alpha values.

```
python3 src/findlang.py --file test_files/english_text.txt --k 3 4 5 --alpha
    0.1 0.2 0.3 --threshold 4 --languages portuguese english spanish
```

2.3 LocateLang

The goal of this module is to analyze a text containing segments written in different languages and return the character position at which each segment starts, as well as the language in which the segment is written. This module takes advantage of previously developed methods in order to find the languages in which the segments are written by calculating its amount of information.

For each character in the text under analysis, the estimated number of bits required to compress that character is obtained by calculating the symmetric logarithm value of the probability of that character occurring after that certain context. This is calculated for each language of the group of languages being used as models. A list of values with the size equal to the number of characters in the text under analysis will be obtained for each language. This means that if the text is being compared to the models of six different languages, then six lists of values will be generated.

The lists of values now have the estimated number of bits for each character, and each list represents a language. For each character, the language that is detected will be the language where the corresponding value for that character will be the minimum among all values, as long as it is above the threshold. This also allows this module to determine where a language starts and ends, since the detection is done character per character.

The *locatelang* module receives the training datasets for each language to train the FCMs in each Lang module. Each language can contain multiple FCMs with different K and α values, respectively.

For each language, the number of bits to compress the data is calculated for the given text. In case of a language containing multiples FCMs these values are normalized (Section 2.1.1) and then smoothed by calculating the average of values on a size N sliding window (Section 2.3.1).

The language of each character is obtained by getting the language with the minimum number of bits. In case the minimum value is bigger than the threshold, no language (None) is returned.

2.3.1 Smoothing values

After generating these lists, these values were smoothed in order to generate smoother lines in the produced graphs and to better visualize the results. The process of smoothing consists on deciding on the size of a window and iterating the values by slicing them using that established window size. Each time a new slice is selected, the average of the values in that slice is calculated and is saved in the previous place of the first value of that slice. This is repeated throughout the length of each list. In the end, the last $n - 1$ values will be left over, and the size of the sliding window will decrease by one at every slide, maintaining the same logic of calculating the average of the values in the window and saving that average in the place of the first value. These smoothed values are then used in the detection process.

2.3.2 Running the LocateLang Module

To run the module *locatelang* it is required to give a file with the text to be analyzed (*file*). There are also optional arguments that locatelang accepts, such as, *threshold* which is the value of the minimum number of bits necessary for a language to classify a text defaults to 4. It is also possible to choose which languages will be used as models (*languages*) in

case no language is provided all languages are used as models.

Regarding the usage of different FCMs, it is possible to give as arguments the k and respective α values. In case no values are given as arguments, the program will run with an FCM with $k = 5$ and $\alpha = 0.5$

gets as arguments the text that is going to be classified, a list of languages to classify the given text (list of Lang, see section 2.1), the threshold of the minimum number of bits necessary for a language to classify a text, a value of the smoothing window used when smoothing the given values for each language and a boolean argument "show".

Listing 4: Normalization of the FCM models values

```
python3 src/locatelang.py --file test_files/polish_texto_exemplo.txt
    --threshold 4 --show True
```

2.3.3 Calculating Accuracy

In order to compare different models we need to quantify its quality, for that purpose the function *calculate_accuracy()* was created that accepts as arguments the result of the *locatelang()* and the ground truth supposed to generate. Then the function returns the accuracy by looping though each character and comparing the *locatelang()* result with the ground truth, if it is equal then increment a counting variable, otherwise continue, when the loop ends then return the $counter/total$.

In order to get the accuracy for every text we want, we need to have a file in which to write the wanted result, for this reason the "truth.txt" file was created and every time the locatelang.py is runned the accuracy will be calculated in the end. This file is an array of tuples with the language on the left side and the starting character index on the right side, the whole index is assumed to be in crescent order.

The current default "truth.txt" is made for the test file "test_files/multiple.txt".

3 Results and Discussion

3.1 Lang module

The Lang module was run for two texts (Portuguese and German) and for each text the number of bits required to compress them was calculated based on a finite context model with $k = 5$ and $\alpha = 0.5$ trained with a Portuguese dataset. The resulting values can be seen in Table 1.

Portuguese text	4.449
German text	6.329

Table 1: Values of the estimated number of bits required to compress text of different languages for a Portuguese dataset

As it is possible to observe on Table 1, the text written in German needs more bits to be compressed in comparison to the text written in Portuguese. This is due to the fact that the FCM has more contexts present in the Portuguese text rather than the German one. This phenomenon occurs firstly, because the FCM was trained with a Portuguese dataset and secondly, due to the fact that German is a Germanic language and Portuguese is derived from Latin, which means that the languages have differences that are reflected when constructing their FCM models.

3.2 findlang module

The *findlang* module was run with different files with text to be analyzed, with a threshold of 5. Each file contains a text with only one specific language, namely, Portuguese with 1112 characters, Polish with 1317 characters, English with 297 characters, German with 598 characters and Italian with 382 characters. The module was also run for a file referred as "Multiple", which contains a multiple section of languages, as mentioned in the

beginning of section 3.3.

For each of these files, FCM modules were trained with different values of K and α and with the following languages: Portuguese, Polish, English, German and Italian as datasets. The output results can be seen in Table 2.

	Texts					
FCM	Portuguese	Polish	English	German	Italian	Multiple
$k = 3$	portuguese	polish	english	german	italian	english
$\alpha = 0.5$	2.82	2.61	2.6	2.68	3.68	4.56
$k = 5$	portuguese	polish	english	german	italian	None
$\alpha = 0.5$	3.74	3.0	2.9	2.94	4.05	5.28
$k = 3$ $k = 5$	portuguese	polish	english	german	italian	None
$\alpha = 0.5$ $\alpha = 0.5$	3.58	2.94	2.85	2.9	3.99	5.16
$k = 3$ $k = 5$ $k = 9$	portuguese	polish	english	german	italian	None
$\alpha = 0.5$ $\alpha = 0.5$ $\alpha = 0.5$	3.66	3.02	2.93	2.98	4.06	5.2
$k = 3$ $k = 5$ $k = 9$	portuguese	polish	english	german	italian	None
$\alpha = 0.1$ $\alpha = 0.1$ $\alpha = 0.1$	3.05	2.71	2.62	2.81	3.9	5.17

Table 2: Output results, identified language and respective amount of information for the different texts and respective FCM modules

Based on Table 2, it is possible to see that the module identified correctly the languages in which each text is written.

Comparing the values for FCMs $k = 3$ and $k = 5$ it is possible to observe that for all files the number of bits needed to compress the text is smaller for $k = 3$ in comparison to $K = 5$. This is due to the fact that, for smaller sized contexts (smaller k), there is more probability that a context appears on the text that is being analyzed, and so, a smaller number of bits is necessary to compress the given text.

Regarding the "Multiple" file, from Table 2, it is possible to see that the minimum estimated number of bits to compress the text was bigger than 5 with the exception of $k = 3$. Since the module was run with a threshold of 5, for the cases where the minimum value was bigger than 5, the returned output language was *None*. This is due to the fact that, no language compressed the text with an estimated number of bits smaller than the threshold value. When $k = 3$, the model trained with English compressed the text with an estimated number of bits smaller than 5. This can be due to the fact that, with smaller sized contexts, some languages of the file may have contexts similar to the ones on the English FCM. The English text is also more present than the other languages. That is another factor that can contribute to the smaller estimated value.

The *findlang* module was run with a normalization function, where FCMs with k values closer to 5 are more valued than FCMs with smaller or bigger k values than 5 (as mentioned in section 3.3). This can be observed in Table 2, by looking at run outputs with different k and same α FCMs. The values of $k = 3$ $k = 5$, are bigger than the values of $k = 3$ and smaller than the values of $k = 5$. This occurs, because the FCM values of $k = 5$ have more weight than the $k = 3$ ones. Nonetheless, the minimum number of bits are still influenced by the $k = 3$ ones.

When analyzing the values of $k = 3$ $k = 5$ $k = 9$, it is possible to observe that the estimated output values are close to the ones of $k = 5$. The reason for this similarity is based on the fact that the weights of $k = 3$ and $k = 9$ when normalizing, are close and small. Therefore, the amounts of information of $k = 3$ and $k = 9$ have very little influence on the values of $k = 5$.

From Table 2, by looking at $k = 3$ $k = 5$ $k = 9$ with different α values, it is possible to observe that a smaller α value gets a smaller number of bits necessary to compress each text. An explanation for this relates to the fact that FCMs built with smaller *alpha* values rely on empirical observations, and, inherently, the probabilities of the events are closer to the truth.

Table 3 refers to the output values of the *findlang* module run with a threshold of 5 and with two texts to be analyzed, one in English and another one in Middle English. Suppressed versions of each text can be seen in Figure 2 and Figure 3.

```
How good it was in you, my dear
Mr. Bennet! But I knew I should
persuade you at last. I was sure
you loved your girls too well to
neglect such an acquaintance.
```

Figure 2: Suppressed English text

```
SIÞEN þe sege and þe assaut watz sesed at Troye,
þe bor3 brittened and brent to bronde3 and askez,
þe tulk þat þe trammes of tresoun þer wro3t
Watz tried for his tricherie, þe trewest on erþe:
Hit watz Ennias þe athel, and his highe kynde,
þat siþen depreced prouinces, and patrounes bicome
Welne3e of al þe wele in þe west iles.
```

Figure 3: Suppressed Middle English text

For both, the FCMs were trained with an English and Middle English dataset, with different values of k and α as showed on the referred table.

	Texts	
FCM	English	Middle English
$k = 3$	english	middle_english
$\alpha = 0.5$	2.6	4.41
$k = 5$	english	middle_english
$\alpha = 0.5$	2.9	4.82
$k = 3$ $k = 5$	english	middle_english
$\alpha = 0.5$ $\alpha = 0.5$	2.85	4.75
$k = 3$ $k = 5$ $k = 9$	english	middle_english
$\alpha = 0.5$ $\alpha = 0.5$ $\alpha = 0.5$	2.93	4.77
$k = 3$ $k = 5$ $k = 9$	english	middle_english
$\alpha = 0.1$ $\alpha = 0.1$ $\alpha = 0.1$	2.62	4.85

Table 3: Output results, identified language and respective amount of information for texts in English and middle English and respective FCM modules

By analyzing Table 3, it is possible to see that both examples of the respective languages were guessed correctly by the *findlang* module. It is also possible to see that the Middle English outputs present a bigger number of bits required to compress the text in relation to the English ones. This can be explained by the fact that Middle English contains a bigger alphabet than English and so it contains more different contexts than English.

3.3 locatelang module

To test the *locatelang* module, a text with 6 different languages was created. This text starts with 753 characters of English, followed by 367 characters of Portuguese, 425 characters of Spanish, 316 characters of French, 283 characters of German and 221 characters of Italian.

"Straight, in the middle of the room, cramped in the freedom of its growth by no encircling walls or soon-reached ceiling, a shadowy tree arises; and, looking up into the dreamy brightness of its top—for I observe in this tree the singular property that it appears to grow downward towards the earth—I look into my youngest Christmas recollections! All toys at first, I find. Up yonder, among the green holly and red berries, is the Tumbler with his hands in his pockets, who wouldn't lie down, but whenever he was put upon the floor, persisted in rolling his fat body about, until he rolled himself still, and brought those lobster eyes of his to bear upon me—when I affected to laugh very much, but in my heart of hearts was extremely doubtful of him. Era uma manhã muito fresca, toda azul e branca, sem uma nuvem, com um lindo sol que não aquecia, e punha nas ruas, nas fachadas das casas, barras alegres de claridade dourada. Lisboa acordava lentamente: as saloias ainda andavam pelas portas com os seirões das hortaliças; varria-se devagar a testada das lojas; no ar macio morria à distância um toque fino de missa. Este rey moro tenía una hija muy hermosa y compasiva, llamada Casilda. Una esclava castellana contó á la hija del rey moro que los nazarenos amaban á su Dios, y á su rey, y á sus padres, y á sus hermanos, y á sus esposas. También contó la esclava á la hija del rey moro, que los nazarenos nunca quedan huérfanos de madre, porque cuando pierden á la que los concibió, les queda otra, llamada María, que es una madre inmortal. Tout le monde aime les centres commerciaux. En été comme en hiver, les gens se bousculent dans ces endroits fantastiques remplis de boutiques variées et d'animations diverses. En effet, même si vous ne souhaitez pas acheter quelque chose, il est toujours intéressant de passer du temps dans les centres commerciaux. Dann besuchen sie das Gebäude des Reichstags am Ufer der Spree. Hier wählen die Deutschen ihren Präsidenten. Außerdem trifft sich dort das Parlament und macht die Gesetze für Deutschland. Deutschland ist ein demokratisches Land: Alle Bürger Deutschlands dürfen das Parlament wählen. Davanti al mio letto si trova la mia televisione e una poltrona su un tappeto. Spesso mi siedo e guardo la televisione per ore. La mia stanza ha anche una

scrivania dove si trova un computer, che uso quando devo studiare."

This text was then analyzed in this module using one or more FCM models with different orders and different smoothing values to represent each language. The goal of this experiment was to verify what factors would affect the accuracy of the language detection and how.

Considering all this, it was possible to obtain graphs where each plotted line represents the comparison with a model language and to see how the estimated number of bits variate throughout the text for each language. In each chart, it is possible to see the plot of the calculated values of amount of information for each language. In the x-axis the position of each character in the text is represented, and in the y-axis the range of estimated number of bits necessary to compress a character is represented. The languages chosen to serve as models for similarity comparisons were the six languages present in the sample text: English, Portuguese, Spanish, French, German and Italian.

In Figure 4, the values were generated by comparing the sample text with representations of the six languages, with every representation being a single FCM model built with $k = 5$ and $\alpha = 0.3$.

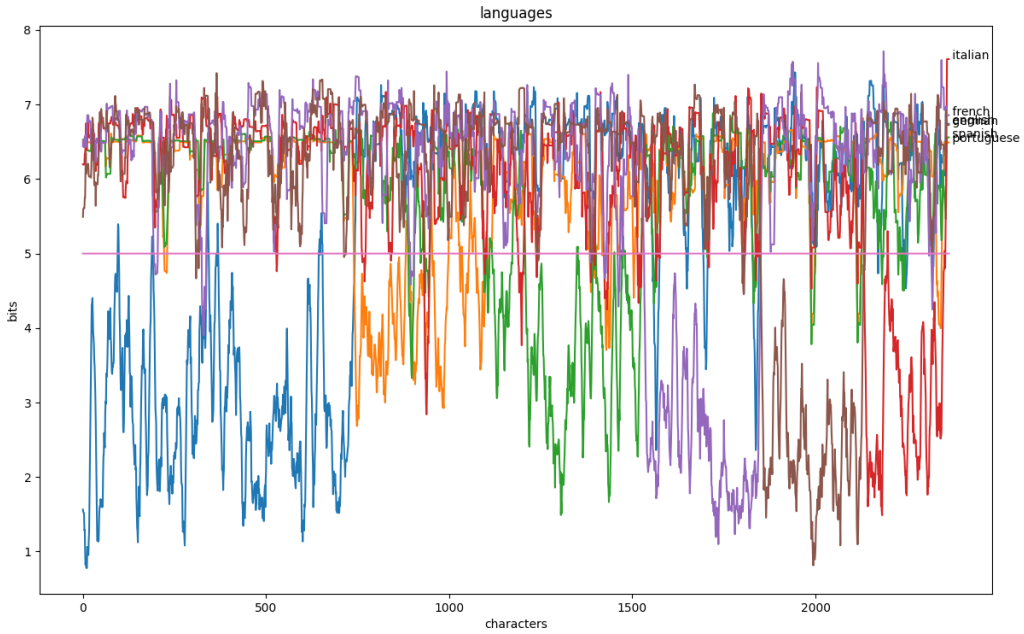

Figure 4: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$

In Figure 5 it is possible to better visualize the results of the detection, with the segments of the text being colored with the language that was detected in that segment. If the segment isn't colored, then no language was detected in that segment because all the values were above the threshold, and thus no relevant similarity was found between that sample text segment and one of the language models. It is also possible to see a list with information relative to where every detected language segment starts and ends. Lastly, the accuracy obtained in this detection is shown in the end, being 89.26% in this case.

Figure 5: Results of the detection, considering one FCM with $k = 5$ and $\alpha = 0.3$

In Figure 6, the values were generated by comparing the sample text with representations of the six languages, with every representation being a single FCM model built with $k = 9$ and $\alpha = 0.3$. The difference between Figure 6 and Figure 4 is that the first was generated using an FCM model with $k = 5$ and the second with $k = 9$. The value of α and the sample text are the same. When comparing with Figure 4, it is verified that the values of the number of bits are higher throughout the sample text in Figure 6. This shows that when changing the value of k, the detection accuracy is affected. With higher values of k, the contexts used to build the FCM model start being too specific, and harder to find in the sample text, which means the similarity between the language and the text decreases and the amount of information values increase, since more bits will be required to represent the information. As a consequence, the smoothed values might even pass the threshold more often, leading to a worse detection. In contrast, with smaller values of k the contexts are smaller and easier to find in other text and the number of bits required to compress the text is lower, translating in a higher similarity. However, if the value of k is too low, the contexts are too general and the amount of information will be low when comparing with every language model, making it harder to detect what language that text was written in. This means that there is a range, situated roughly between 3 and 6, where the detection achieves better levels of accuracy. In Figure 6 the accuracy was much lower

when compared to the first experiment, having decreased from 89.26% to 31.29%.

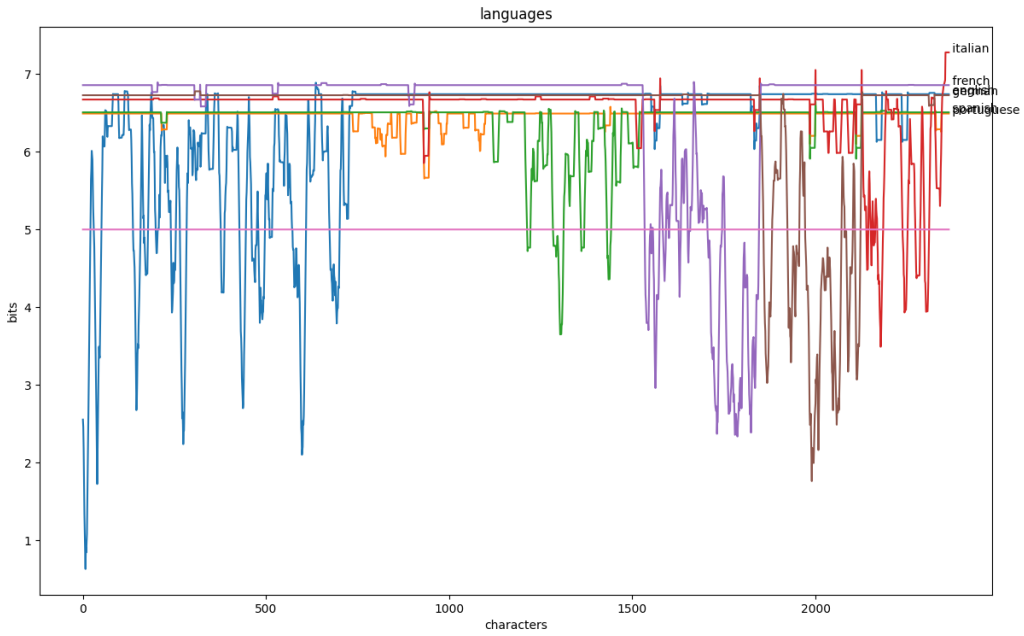

Figure 6: Plot of calculated amount of information for each language, considering one FCM with $k = 9$ and $\alpha = 0.3$

```
(venv) ➜  TAI_Lab2 git:(main) ✗ python3 src/locatelang.py --file test_files/multiple2.txt --show True --languages english portuguese spanish italian french german --k 9
 --alpha 0.3 --threshold 5
--------------------
Straight, in the middle of the room, cramped in the freedom of its growth by no encircling walls or soon-reached ceiling, a shadowy tree arises; and, looking up into th
e dreamy brightness of its top-for I observe in this tree the singular property that it appears to grow downward towards the earth-I look into my youngest Christmas rec
ollections! All toys at first, I find. Up yonder, among the green holly and red berries, is the Tumbler with his hands in his pockets, who wouldn't lie down, but whenev
er he was put upon the floor, persisted in rolling his fat body about, until he rolled himself still, and brought those lobster eyes of his to bear upon me-when I affec
ted to laugh very much, but in my heart of hearts was extremely doubtful of him. Era uma manhã muito fresca, toda azul e branca, sem uma nuvem, com um lindo sol que não
 aquecia, e punha nas ruas, nas fachadas das casas, barras alegres de claridade dourada. Lisboa acordava lentamente: as saloias ainda andavam pelas portas com os seirões
s das hortaliças; varria-se devagar a testada das lojas; no ar macio morria à distância um toque fino de missa. Este rey moro tenía una hija muy hermosa y compasiva, ll
amada Casilda. Una esclava castellana contó á la hija del rey moro que los nazarenos amaban á su Dios, y á su rey, y á sus padres, y á sus hermanos, y á sus esposas. Ta
mbién contó la esclava á la hija del rey moro, que los nazarenos nunca quedan huérfanos de madre, porque cuando pierden á la que los concibió, les queda otra, llamada M
aría, que es una madre inmortal. Tout le monde aime les centres commerciaux. En été comme en hiver, les gens se bousculent dans ces endroits fantastiques remplis de bou
tiques variées et d'animations diverses. En effet, même si vous ne souhaitez pas acheter quelque chose, il est toujours intéressant de passer du temps dans les centres
commerciaux. Dann besuchen sie das Gebäude des Reichstags am Ufer der Spree. Hier wählen die Deutschen ihren Präsidenten. Außerdem trifft sich dort das Parlament und ma
cht die Gesetze für Deutschland. Deutschland ist ein demokratisches Land: Alle Bürger Deutschlands dürfen das Parlament wählen. Davanti al mio letto si trova la mia tel
evisione e una poltrona su un tappeto. Spesso mi siedo e guardo la televisione per ore. La mia stanza ha anche una scrivania dove si trova un computer, che uso quando d
evo studiare. --------------------

Language locations:
 [(english, 0, 20), (english, 30, 50), (english, 137, 157), (english, 167, 177), (english, 199, 204), (english, 240, 253), (english, 265, 283), (english, 377, 386), (en
glish, 428, 444), (english, 461, 474), (english, 479, 496), (english, 510, 516), (english, 573, 608), (english, 674, 700), (spanish, 1213, 1221), (spanish, 1284, 1312),
 (spanish, 1360, 1368), (spanish, 1431, 1442), (french, 1534, 1549), (french, 1551, 1553), (french, 1557, 1574), (french, 1584, 1591), (french, 1600, 1600), (french, 16
27, 1632), (french, 1652, 1658), (french, 1706, 1745), (french, 1753, 1845), (german, 1858, 1881), (german, 1924, 1955), (german, 1972, 2071), (german, 2080, 2102), (ge
rman, 2108, 2123), (italian, 2139, 2144), (italian, 2151, 2154), (italian, 2159, 2163), (italian, 2169, 2184), (italian, 2238, 2251), (italian, 2273, 2285), (italian, 2
298, 2310)]
accuracy: 31.29%
```

Figure 7: Results of the detection, considering one FCM with $k = 9$ and $\alpha = 0.3$

In Figure 8, the values were generated by comparing the sample text with representations of the six languages, with every representation being a single FCM model built with

$k = 5$ and $\alpha = 1$. When comparing the Figure 8 with Figure 4, it can be seen that the accuracy reduced as well. This shows that the value of α chosen for the models affects the accuracy as well. For the first figure, the chosen α was 0.3, and in the second the chosen α was 1. This proves that the accuracy of language detection increases as the α value decreases, since there is more certainty when building the FCM model from the empirical observations. As the value of α increases, the probabilities of the events will be further away from reality, and the comparison with the language models might result in inaccurate values.

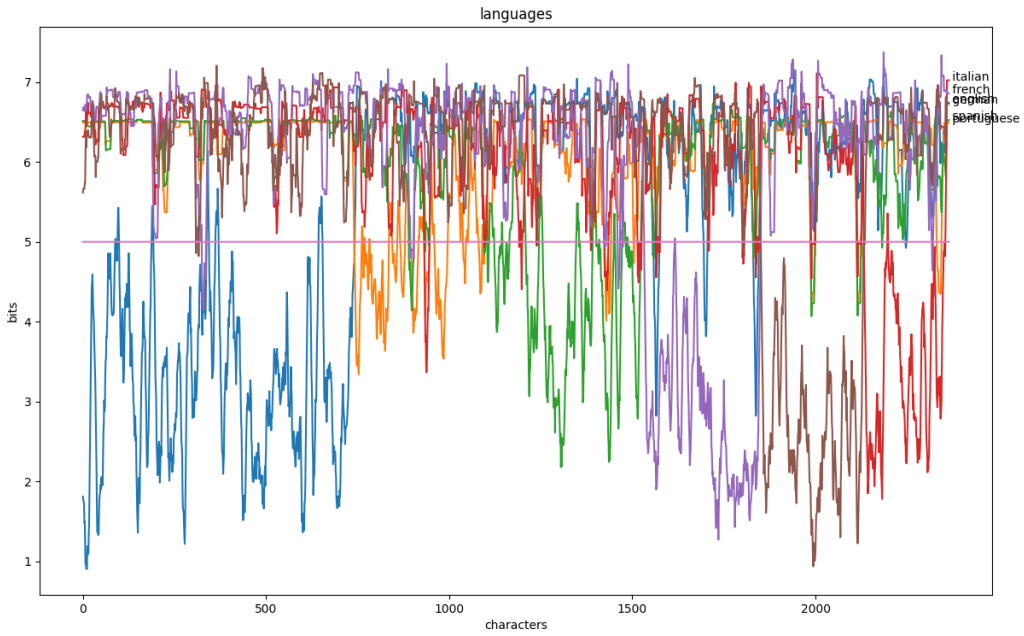

Figure 8: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 1$

Figure 9: Results of the detection, considering one FCM with $k = 5$ and $\alpha = 1$

However, the values chosen for k and α are not the only factors that affect the accuracy of the detection. Each language is represented by at least one FCM model, and these models are built upon sample texts from the data set that are written in that specific language. This means that as the size and content of the sample texts used to train the models increases, the more well-developed and complete the model is. With better constructed models, the accuracy of the detection increases since it will encounter contexts that has probably encountered before in a specific language, increasing the probability of the sample text being written in that language.

Another factor that affects the accuracy of the language detection is the difference between the languages itself. For example, languages belonging to the Romance languages' family are all derived from Latin and are very similar. This means that when analyzing the model text from the data set, these languages will have FCM models with similar contexts, making it harder to distinguish between them when trying to identify in what language a segment of text was written, since the values of amount of bits will be similar. One solution that was adopted to avoid this problem was to increase the number and variety of the model texts of these languages in the data set, so that the FCM model that represent these languages could be more catered to each specific language and could avoid more collisions between contexts and events.

The possibility of using multiple FCM models to represent each language was also explored. It was expected that the accuracy of the results would improve with this approach, but the experiments showed that it would be necessary to adjust how every FCM model is considered in the calculations.

In Figure 10, the values were generated by comparing the sample text with representations of the six languages, with every representation being composed of two FCM models built with $k = 5$ and $\alpha = 0.3$ and with $k = 3$ and $\alpha = 0.1$. In Figure 11 it is possible to visualize the language detected in the colored text segments.

Comparing with Figure 4, the difference between the implementations is that in Figure 10 the results were generated with an extra FCM model with $k = 3$ and $\alpha = 0.1$. As it is possible to see in Figures 11 and 5, the accuracy increases from 89.26% to 93.4% with the use of these two models.

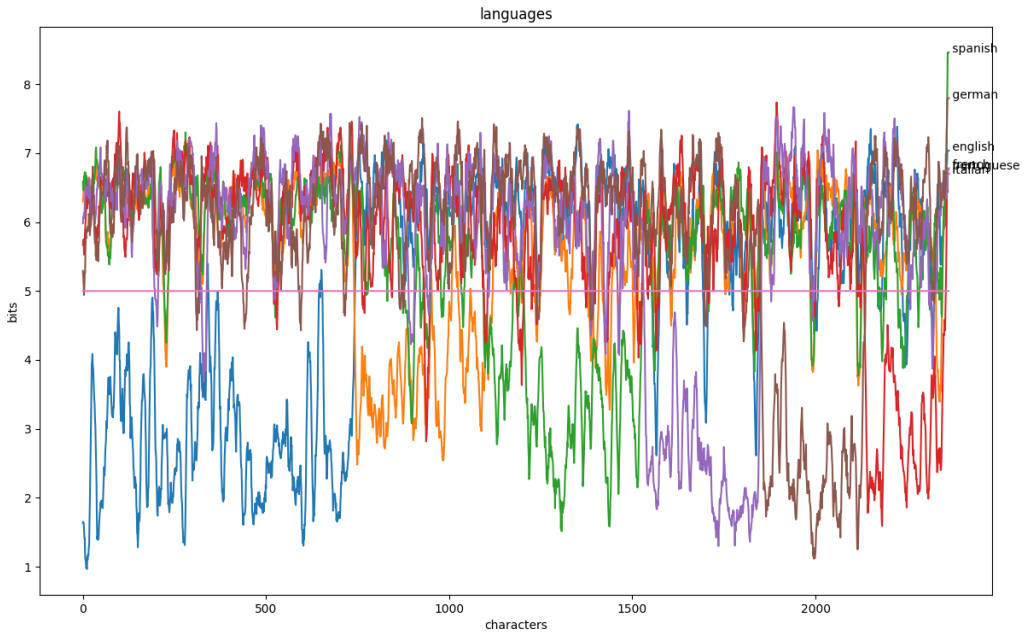

Figure 10: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 3$ and $\alpha = 0.1$

Figure 11: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 3$ and $\alpha = 0.1$

In Figure 12, the values were generated by comparing the sample text with representations of the six languages, with every representation being composed of two FCM models built with $k = 5$ and $\alpha = 0.3$ and with $k = 9$ and $\alpha = 0.1$. In Figure 13 it is possible to visualize the language detected in the colored text segments.

Comparing with Figure 4, the difference between the implementations is that in Figure 12 the results were generated with an extra FCM model with $k = 9$ and $\alpha = 0.1$. As it is possible to see in Figures 13 and 5, the accuracy decreases from 89.26% to 57.97% with the use of these two models.

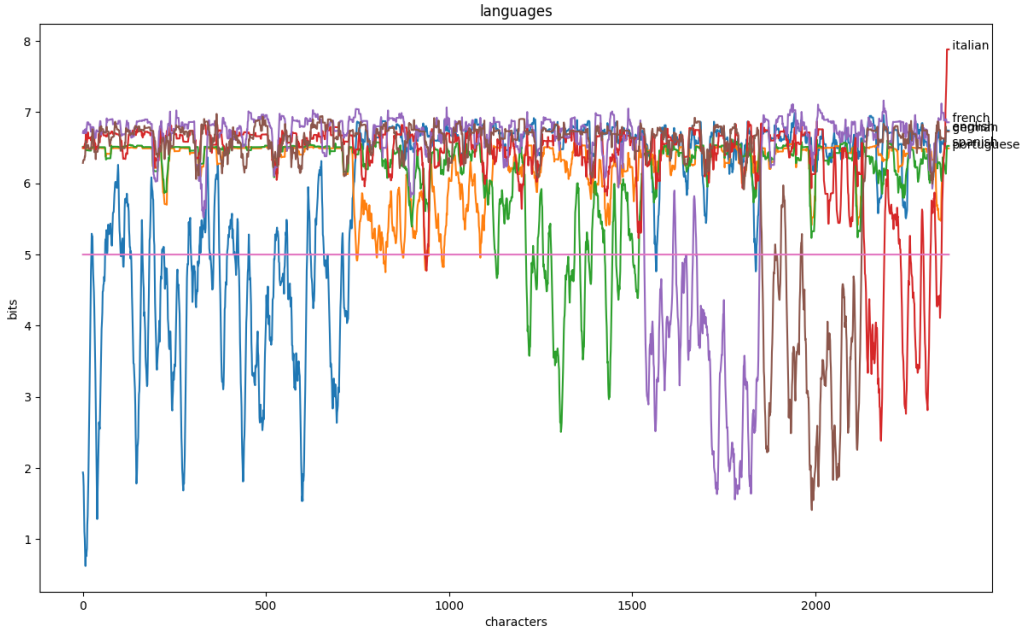

Figure 12: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 9$ and $\alpha = 0.1$

```
(venv) ➜  TAI_Lab2 git:(main) ✗ python3 src/locatelang.py --file test_files/multiple2.txt --show True --languages english portuguese spanish italian french german --k 5
 9 --alpha 0.3 0.1 --threshold 5

--------------------
Straight, in the middle of the room, cramped in the freedom of its growth by no encircling walls or soon-reached ceiling, a shadowy tree arises; and, looking up into th
e dreamy brightness of its top—for I observe in this tree the singular property that it appears to grow downward towards the earth—I look into my youngest Christmas rec
ollections! All toys at first, I find. Up yonder, among the green holly and red berries, is the Tumbler with his hands in his pockets, who wouldn't lie down, but whenev
er he was put upon the floor, persisted in rolling his fat body about, until he rolled himself still, and brought those lobster eyes of his to bear upon me—when I affec
ted to laugh very much, but in my heart of hearts was extremely doubtful of him. Era uma manhã muito fresca, toda azul e branca, sem uma nuvem, com um lindo sol que não
 aquecia, e punha nas ruas, nas fachadas das casas, barras alegres de claridade dourada. Lisboa acordava lentamente: as saloias ainda andavam pelas portas com os seirões
s das hortaliças; varria-se devagar a testada das lojas; no ar macio morria à distância um toque fino de missa. Este rey moro tenía una hija muy hermosa y compasiva, ll
amada Casilda. Una esclava castellana contó á la hija del rey moro que los nazarenos amaban á su Dios, y á su rey, y á sus padres, y á sus hermanos, y á sus esposas. Ta
mbién contó la esclava á la hija del rey moro, que los nazarenos nunca quedan huérfanos de madre, porque cuando pierden á la que los concibió, les queda otra, llamada M
aría, que es una madre inmortal. Tout le monde aime les centres commerciaux. En été comme en hiver, les gens se bousculent dans ces endroits fantastiques remplis de bou
tiques variées et d'animations diverses. En effet, même si vous ne souhaitez pas acheter quelque chose, il est toujours intéressant de passer du temps dans les centres
commerciaux. Dann besuchen sie das Gebäude des Reichstags am Ufer der Spree. Hier wählen die Deutschen ihren Präsidenten. Außerdem trifft sich dort das Parlament und ma
cht die Gesetze für Deutschland. Deutschland ist ein demokratisches Land: Alle Bürger Deutschlands dürfen das Parlament wählen. Davanti al mio letto si trova la mia tel
evisione e una poltrona su un tappeto. Spesso mi siedo e guardo la televisione per ore. La mia stanza ha anche una scrivania dove si trova un computer, che uso quando d
evo studiare. --------------------

Language locations:
 [(english, 0, 23), (english, 27, 60), (english, 104, 104), (english, 109, 112), (english, 130, 130), (english, 132, 159), (english, 163, 181), (english, 196, 215), (en
glish, 222, 287), (english, 296, 317), (english, 347, 347), (english, 349, 356), (english, 373, 393), (english, 415, 419), (english, 425, 451), (english, 455, 523), (en
glish, 535, 551), (english, 559, 612), (english, 623, 633), (english, 659, 667), (english, 669, 706), (english, 711, 726), (portuguese, 747, 750), (portuguese, 802, 803
), (portuguese, 821, 827), (portuguese, 872, 872), (portuguese, 874, 875), (portuguese, 934, 936), (italian, 937, 940), (portuguese, 943, 944), (portuguese, 980, 985),
(portuguese, 1085, 1085), (spanish, 1126, 1133), (spanish, 1210, 1228), (spanish, 1230, 1241), (spanish, 1257, 1270), (spanish, 1279, 1342), (spanish, 1357, 1372), (spa
nish, 1381, 1389), (spanish, 1425, 1445), (spanish, 1456, 1467), (spanish, 1502, 1517), (french, 1533, 1610), (french, 1620, 1665), (french, 1675, 1675), (french, 1677,
 1847), (german, 1854, 1888), (german, 1892, 1905), (german, 1920, 1959), (german, 1964, 2125), (italian, 2135, 2188), (italian, 2233, 2256), (italian, 2260, 2288), (it
alian, 2295, 2313), (italian, 2329, 2344)]
accuracy: 57.97%
```

Figure 13: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 9$ and $\alpha = 0.1$

Two different cases have then been presented. In both cases, two FCM models are used,

with one of them being the same. However, in the first case the accuracy increases and in the second case it decreases greatly. This shows that the first implementation of the normalization of the values was not entirely successful, and the reason for this is due to the fact that a bigger weight was being attributed to models with bigger k values, and it was seen previously in this section that values of k bigger than 6 tend to start decreasing the accuracy. In Figure 12 the results are much more inaccurate because the FCM with order 9 is being more valued than the FCM with order 5 for the calculations of the final results of the number of bits, and this should not happen because this value of k is too big to generate approximately accurate results. The FCM with order of 5 should be more valued in this case, and this logic was followed when implementing the second approach to the normalization (section 2.1.1) and when collecting the results that follow.

The results in Figures 14 and 15 were obtained with the same FCM models representing the languages that were used to obtain the results in Figures 12 and 13. However, these new results were obtained with the second approach to the normalization of the final values. It is possible to check that the number of bits throughout the graph are much lower in this new approach, and the accuracy increased from 57.97% to 88.92%. This is due to the fact that the weight attributed to the FCM with order 5 is bigger than the one attributed to the model with order 9 in the calculations, so the information that the first FCM model provides will be more relevant than the information provided by the second FCM model. This contributes to improving the accuracy because, as seen before, the second model has a higher k value and generates results with less accuracy. The accuracy obtained with the two models for each language is also very close to the accuracy obtained with just the FCM model of order 5, with the first being 88.92% and the second being 89.26%.

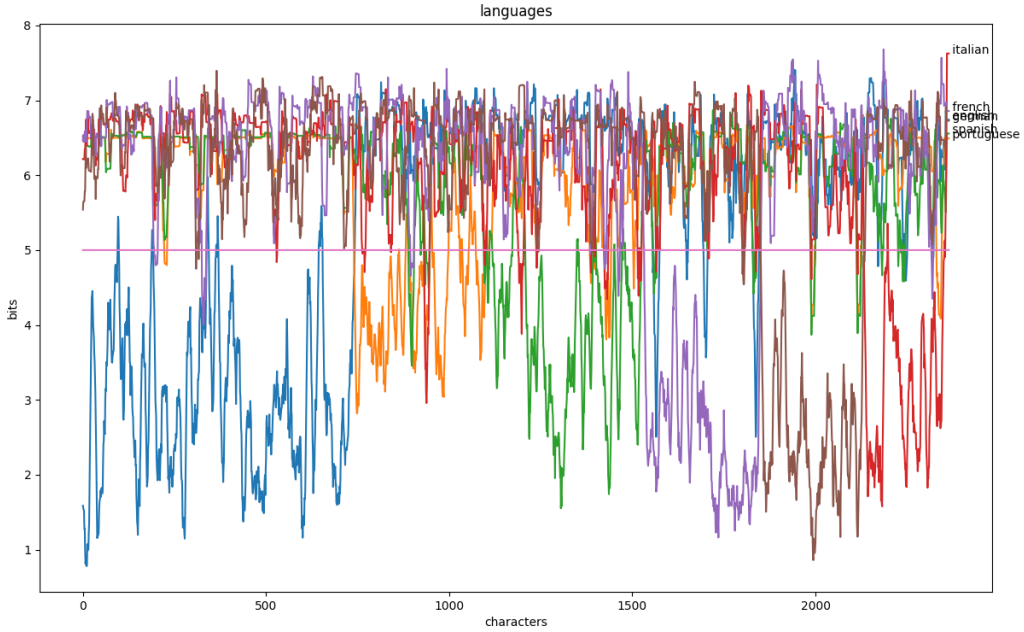

Figure 14: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 9$ and $\alpha = 0.1$ and the new approach to normalization

Figure 15: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$ and one FCM with $k = 9$ and $\alpha = 0.1$ and the new approach to normalization

Lastly, to further experiment with the new approach to the normalization of the final

values, a final experiment was conducted using three FCM models to represent each language. In Figures 16 and 17 it is possible to observe the results that were obtained with the old approach to the normalization and in Figures 18 and 19 it is possible to observe the results that were obtained with the new approach to the normalization. It can be verified that the estimations of number of bits are lower in the second case, and the accuracy is higher. This proves that the new normalization provides a better implementation for using multiple FCM models to represent each language that is being used for measuring the similarity with the sample text.

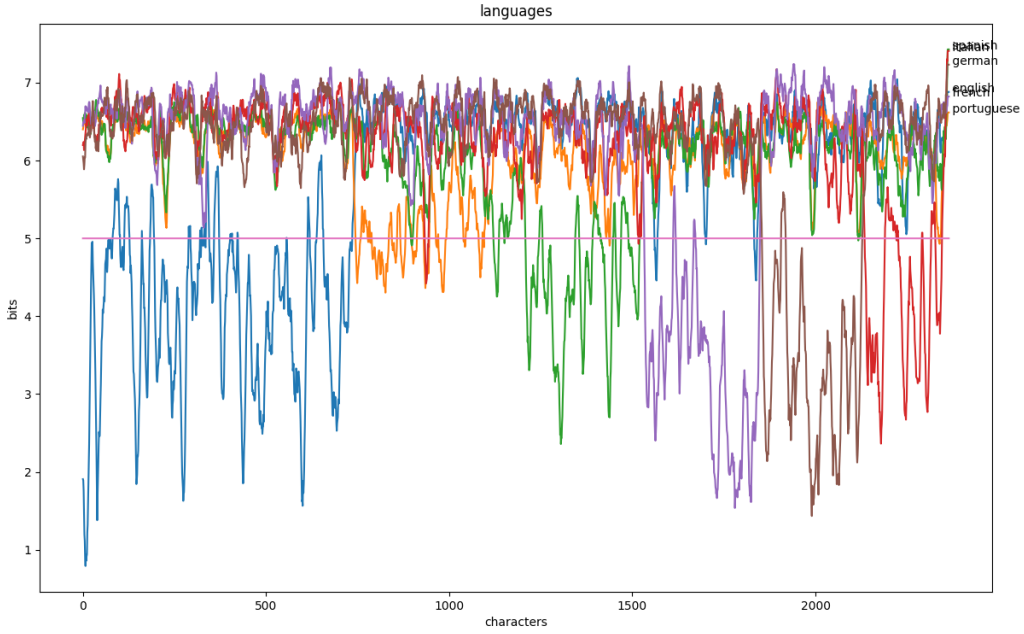

Figure 16: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$, one FCM with $k = 3$ and $\alpha = 0.1$ and one FCM with $k = 9$ and $\alpha = 0.1$

```
(venv) → TAI_Lab2 git:(main) ✗ python3 src/locatelang.py --file test_files/multiple2.txt --show True --languages english portuguese spanish italian french german --k 5
 3 9 --alpha 0.3 0.1 0.1 --threshold 5

____________________
Straight, in the middle of the room, cramped in the freedom of its growth by no encircling walls or soon-reached ceiling, a shadowy tree arises; and, looking up into th
e dreamy brightness of its top—for I observe in this tree the singular property that it appears to grow downward towards the earth—I look into my youngest Christmas rec
ollections! All toys at first, I find. Up yonder, among the green holly and red berries, is the Tumbler with his hands in his pockets, who wouldn't lie down, but whenev
er he was put upon the floor, persisted in rolling his fat body about, until he rolled himself still, and brought those lobster eyes of his to bear upon me—when I affec
ted to laugh very much, but in my heart of hearts was extremely doubtful of him. Era uma manhã muito fresca, toda azul e branca, sem uma nuvem, com um lindo sol que não
 aquecia, e punha nas ruas, nas fachadas das casas, barras alegres de claridade dourada. Lisboa acordava lentamente: as saloias ainda andavam pelas portas com os seirões
s das hortaliças; varria-se devagar a testada das lojas; no ar macio morria à distância um toque fino de missa. Este rey moro tenía una hija muy hermosa y compasiva, ll
amada Casilda. Una esclava castellana contó á la hija del rey moro que los nazarenos amaban á su Dios, y á su rey, y á sus padres, y á sus hermanos, y á sus esposas. Ta
mbién contó la esclava á la hija del rey moro, que los nazarenos nunca quedan huérfanos de madre, porque cuando pierden á la que los concibió, les queda otra, llamada M
aria, que es una madre inmortal. Tout le monde aime les centres commerciaux. En été comme en hiver, les gens se bousculent dans ces endroits fantastiques remplis de bou
tiques variées et d'animations diverses. En effet, même si vous ne souhaitez pas acheter quelque chose, il est toujours intéressant de passer du temps dans les centres
 commerciaux. Dann besuchen sie das Gebäude des Reichstags am Ufer der Spree. Hier wählen die Deutschen ihren Präsidenten. Außerdem trifft sich dort das Parlament und ma
cht die Gesetze für Deutschland. Deutschland ist ein demokratisches Land: Alle Bürger Deutschlands dürfen das Parlament wählen. Davanti al mio letto si trova la mia tel
evisione e una poltrona su un tappeto. Spesso mi siedo e guardo la televisione per ore. La mia stanza ha anche una scrivania dove si trova un computer, che uso quando d
evo studiare. ____________________

Language locations:
 [(english, 0, 80), (english, 102, 112), (english, 128, 160), (english, 162, 183), (english, 195, 288), (english, 294, 319), (english, 321, 323), (english, 325, 329), (
english, 333, 335), (english, 345, 357), (english, 372, 401), (english, 405, 405), (english, 408, 421), (english, 424, 555), (english, 557, 613), (english, 620, 637), (
english, 657, 735), (portuguese, 745, 758), (portuguese, 778, 779), (portuguese, 785, 785), (portuguese, 789, 835), (portuguese, 844, 857), (portuguese, 867, 880), (spa
nish, 896, 897), (portuguese, 898, 911), (portuguese, 931, 936), (italian, 937, 940), (portuguese, 941, 945), (portuguese, 964, 968), (portuguese, 971, 992), (portugues
e, 1039, 1044), (portuguese, 1081, 1090), (portuguese, 1092, 1095), (spanish, 1121, 1135), (spanish, 1151, 1151), (spanish, 1209, 1245), (spanish, 1255, 1344), (spanish
, 1355, 1391), (spanish, 1414, 1446), (spanish, 1455, 1469), (spanish, 1490, 1519), (french, 1533, 1611), (french, 1619, 1667), (french, 1673, 1848), (german, 1854, 190
5), (german, 1919, 2127), (italian, 2135, 2189), (italian, 2221, 2222), (italian, 2226, 2290), (italian, 2293, 2315), (italian, 2328, 2345)]
accuracy: 73.4%
```

Figure 17: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$, one FCM with $k = 3$ and $\alpha = 0.1$ and one FCM with $k = 9$ and $\alpha = 0.1$

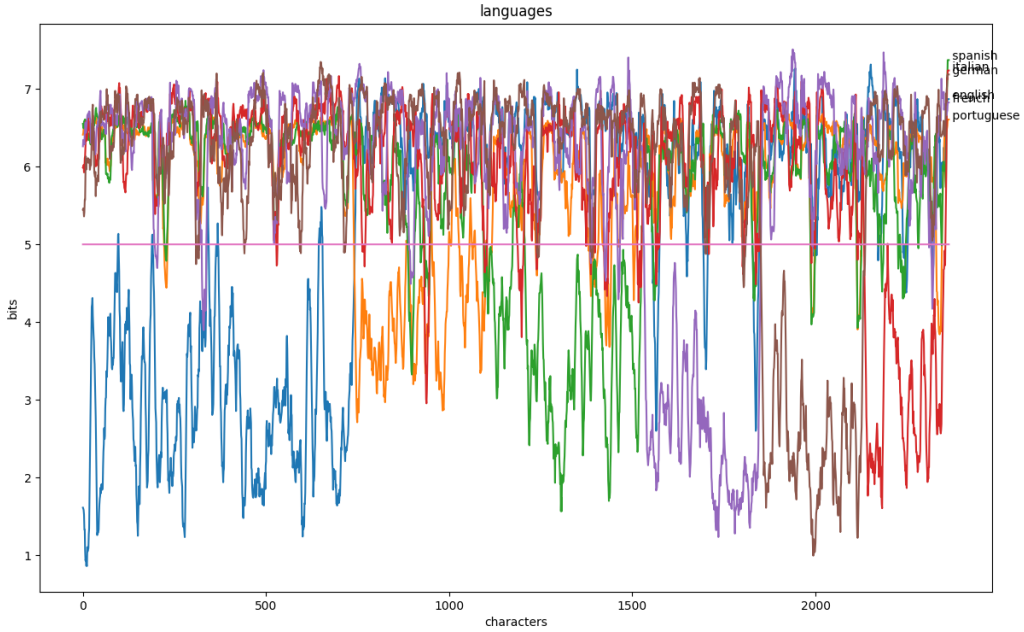

Figure 18: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$, one FCM with $k = 3$ and $\alpha = 0.1$ and one FCM with $k = 9$ and $\alpha = 0.1$ and the new approach to normalization

Figure 19: Plot of calculated amount of information for each language, considering one FCM with $k = 5$ and $\alpha = 0.3$, one FCM with $k = 3$ and $\alpha = 0.1$ and one FCM with $k = 9$ and $\alpha = 0.1$ and the new approach to normalization

4 Conclusion

The goal of this project was achieved with success, and three modules were produced that allow the detection of languages in sample texts, including where each language segments starts and ends.

In relation to the *findlang* module, all languages were identified correctly, and this module even allows distinguishing dialects of the same language. It was verified that as the k order of the FCM model decreases, the more generic the k-sized contexts will be, and thus they will be more frequent in the text to analyze, which means that the estimated number of bits will be smaller, and the similarity will be higher between the text and the model of the language in which it is written. However, if the value of k is too larger, the context used to build the FCM will be too specific and harder to find in the text, which will translate in requiring more bits to represent the text. Considering this, the value of k should not be neither too small nor too big, with the best values located between 3 and 6, as it was verified in the experiments.

The value of α also influences the calculation of similarity. As the α decreases, the FCM models rely more on empirical observations and less on random information. This is useful for measuring similarities, since we want to base the calculations on real observations to obtain the best possible results.

Each language that serves as a model for comparisons can be represented by one or more FCM models. When using more than one FCM model, a normalization function is used in the calculations of the number of bits. This function considers that FCM models that have an order between 3 and 6 should have a bigger weight in the calculations, and models with an order outside that range should have a smaller weight. When comparing experiments where multiple FCM models are used and this function is not with experiments where it is used, it is possible to verify that it greatly increases the accuracy of the detection of languages in the sample texts.

Lastly, it was observed that languages with the same origins are more similar, and therefore it is more difficult to distinguish between them when analyzing a text. The solution adopted to avoid this problem was to train languages that belonged to the same family with more extensive data sets, in order to build richer and more variate FCM models to represent each language.