SAE1.01-02
Implémentation d'un besoin client et comparaison d'approches algorithmiques

F.Martins K.Rufflet-Ejiri

January 17, 2024

Contents

Intro :

The objective of this project is to sort telegrams from a "test" database (text file) into five different categories (sciences & environement / culture / economy / politics / sports), depending on the words that constitute them, using a java program.

The program runs thanks to glossaries (one for each category), composed by keywords and scores from one to three, corresponding to the importance of these words in each category (for example : a telegram containing the words "game", "opponent", or "referee" will be more likely to belong to the sports category than those containing the words "currency", "market", or "bank", which belong to the economy vocabulary ; the scores for the words "game", "opponent", or "referee" will be close to 3 in the sports cat and close to 1 in the economy cat.).

For each telegram, one score for each category will be saved, and the highest one will be considered as the category the telegram belongs to. At the end of the project, we want to have at least 95% accuracy.

Point :

(all the programs we coded are available in the src/ file)

Firstly, for each category, we manually wrote a man_lex_category.txt file containing around fifteen words appearing to belong to this category. The scores for each telegram were all close to 0, because the words "databases" were too small.

Secondly, we computerized the lexicon files generation. To achieve this, we first listed all the existing words in the telegrams in a dictionary thanks to a java function (initDico(d)). Then we checked how many times each one of these word (in the dictionary) appeared in each telegram, while knowing which category the telegram is supposed to belong to, and again we assigned a score from one to three to these words. As a result of this process, we obtained five new lex_category.txt files, but this time, all of them contained every existing word.

Despite this, the result was still not good enough : we still had accuracies of 60 70%. Therefore we adjusted the criterion of number of ocurences of words for each score and finally, we managed to get more than 75% accuracy. We considered that this is good enough.

Results :

At the end, we obtained an average of 75% of accuracy, which is good, but not perfect. Some telegrams may be detected in the wrong category

Complexity :

Here are the number of comparisons for the two score functions that we are using, with sequential algorithms :

calculations done in: 860ms

By sorting the lexicons alphabetically and by switching to a dichotimous algorithm, we can reduce the number of comparisons and run the program faster :

calculations done in: 649ms

The number of comparisons is the same for the two versions : comparisons fo score() : 91015 comparisons for calculscore() : 11255

Conclusion :

In order to upgrade this program, there are some points we could improve to save some execution time. First : the word detection : currently, the program checks the whole words each time, but instead of that it could only check the prefix of the word ("gymnast", "gymnasium", and "gymnastics" have the same prefix "gym". Instead of comparing the whole word to the whole lexicon file, we could imediately detect that it starts with "gym", and deduce that the word has good odds of belonging to the sports lexicon). Second : in the telegrams, some words are very common for every category, (such as "le", "la", or "à" ...) and shouldn't be taken into account, because they alter the results. Another improvement of the program could be to detect and Signore them. Third : The current score interval for each word is [1;3]. By widening it we could get a better score precision and therefore a better accuracy.