Final Experimental Report

1. **Overview**

This is the final report for the project of the Python course held 2016/2017 at the Jiao Tong University, Shanghai, by Professor Tianfang Yao. And this report will explain the task in an overall view.

**a. System Function**

The function of this project is Chinese word segmentation. A paragraph or a sentence is input, and the project will divide it into parts according to Chinese grammars. Besides, you can input the address of a file directly and a new file divided will be generated. As for the updates of the dictionary, new words can be added manually. The program can also automatically recognize those words that may are new words, and inquire the uses that whether add the word or not.

**b. Running Environment**

Python is the running environment of our project. The version is 3.23.

**c. Development Environment**

We develop it on Microsoft Windows 10 and Window8.The development software is PyScripter 3.3.

1. **Task Arrangement**

The task mainly contains three parts, as follow:

1. Word segmentation: Long Quanyu

He did the algorithm that can cuts the Chinese sentences, based on the HMM algorithm. The specific method will be introduced later.

1. Lexicon: Zhao Pengzhen

He finished the establishment of the lexicon, using two ways, one is list, the other is tree. The first and second step of segment algorithm is based on the establishment of ‘FREQ’, which contains the most of commonly used words and the frequency of each word. Higher the corresponding frequency is, there is more possibility that the word is used in a sentence.

And he also finished the part that users can add new word, can change the frequency, that is, the possibility can be changed.

1. Use Interface: Fang Yihang

The component of use interface was finished by him, and he also wrote the copyright and the manuals

The following part of report will be divided into three parts .

1. **Problem Analysis**
2. **Segmentation**

There are many methods to finish the segmentation, eventually, we choose the method of string and frequency, and using HMM. Although knowing the principle of HMM( Viterbi, Forward, Backward), it’s not easy to write the program, since we need to consider all the different occasions, and it’s difficult to debug. And how to get the frequency of ‘start-probability’, ‘transformation-probability’ and ‘emit-probability’? And how to accurate the frequency?

1. **Lexicon**

Forming a lexicon including the frequency of Chinese words to be applied to the segmentation, and use a function to finish this task; Forming the frequency using a formula quickly; Open different files encoded in different ways; Forming a string of a sentence that has been segmented without any punctuation and mark every character with ‘B,M,E,S’ according to its position in a word; Judge if an input word is a Chinese word, as we know, Chinese words can have English letter in them; Satisfy the demand that users can operate the lexicon, including adding new words, setting up general service word, searching related words, etc.

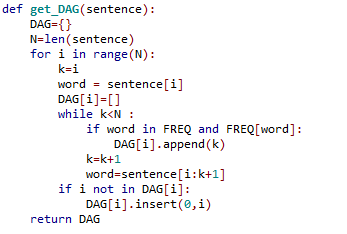
1. Interface

Arrange the widgets properly (including their positions, commands, sizes, etc.), deal with UI and its sub UIs.

1. **Solution Design**
2. **segmentation**

use ‘昨晚我在秋明山输给一辆五菱宏光’ as an example. How to segment this sentence? As we can see, this sentence contains two new words, ‘秋明山’ and ‘五菱宏光’.

**First step**: Get DAG(有向无环图directed acyclic)

based on the dictionary, to find all the possible path from first word to last word.

Left graph is the code of getting DAG.

Here are the DAG of the example sentence ‘昨晚我在秋明山输给一辆五菱宏光’

{0: [0, 1], (This means ‘昨晚’ is in the dictionary, and so on)

1: [1],

2: [2],

3: [3],

4: [4],

5: [5, 6],

6: [6],

7: [7, 8],

8: [8],

9: [9, 10],

10: [10],

11: [11],

12: [12],

13: [13],

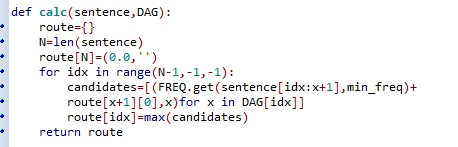
14: [14]}

**Second step:** Dynamic programming

In the from\_gen\_trie function, we get FREQ and two important parameter, one is the dictionary contain most of the words that are commonly used in our life, the other is the frequency corresponding each word. The calculation of frequency will be introduced in the Dictionary part. The higher the frequency is, the more corresponding word is been used.

Use dynamic programming, we find the path where the frequency becomes maximum. To implement dynamic programming, we have to count from the end of the sentence. Then we will get a dictionary which records the maximum frequency and the best path in each step.

The following code is to calculate the ‘score’ of each path, the find the maximum score and the corresponding route.



Here are the route of the example sentence ‘昨晚我在秋明山输给一辆五菱宏光’

{0: (-109.53110721540244, 1),

1: (-107.73366444490432, 1),

2: (-98.49168218898622, 2),

3: (-93.28345869148502, 3),

4: (-88.86984512585009, 4),

5: (-78.91511912018498, 5),

6: (-70.69138200849038, 6),

7: (-62.84624277976349, 8),

8: (-57.57798154580915, 8),

9: (-50.815222638408976, 10),

10: (-50.029588145127406, 10),

11: (-39.96070649235165, 11),

12: (-31.708337567982717, 12),

13: (-19.851223786496867, 13),

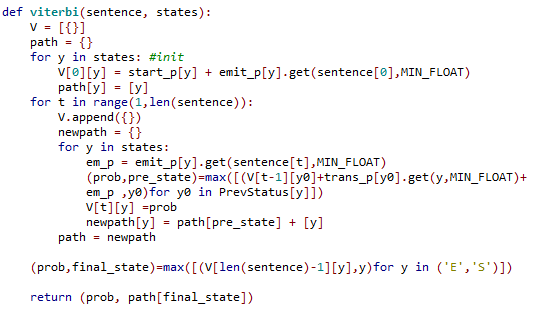
14: (-8.615493880376498, 14),

15: (0.0, '')}

**Third step:** Viterbi of HMM

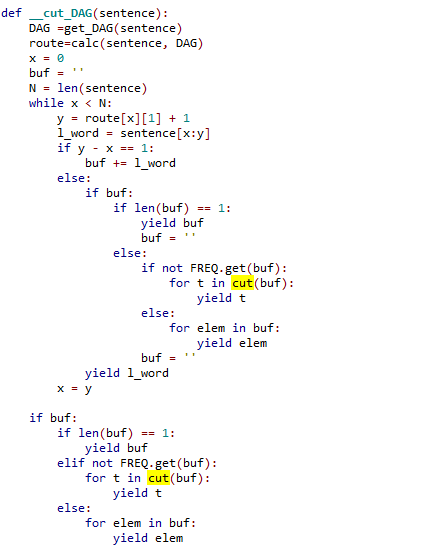
Although we have get the result of segment, but for those new words that isn’t in the Dictionary, so we have to find the new word automatically.

The principle of Viterbi will be omitted.

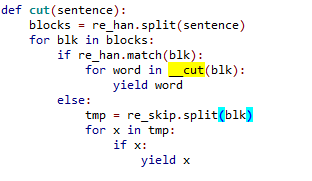
Here are the code of Viterbi

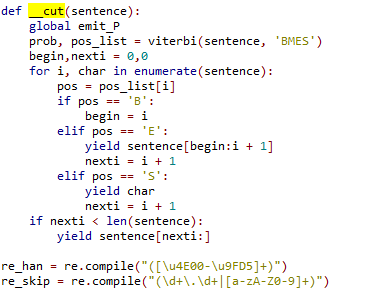
With the help of this function, we can find the hidden sequence of a Chinese sentence. But how to use this function to find the new words?

Firstly, we collect a string of single word based on the best route that have demonstrated. For Example, ‘我在秋明山’ and ‘五菱宏光’ will be collected in the variable named ‘buf’, then use the Viterbi to find the hidden sequence of ‘我在秋明山’ and ‘五菱宏光’, like ‘SSBME’ and ‘BMME’. That are the hidden sequences we expect to get. Lastly, we begin to cut the sentence, for ‘S’, we cut it on its left and right; for ‘BME’, we cut on the left side of ‘B’, and cut on the right side of ‘E’. This is the method of HMM, here are the code we write, using a strong key word in python named ‘yield’.



And the ‘cut’ function is defined in the ‘Finalseg’ file. Here are the code of ‘cut’ function.





but the most convenient way is adding the new word to the dictionary, because frequency-based algorithm can’t ensure the accuracy. For instance, after our own programming, our ’Viterbi’ function return the ‘SSBME’ and ‘SBME’ corresponding to ‘我在秋明山’ and ‘五菱宏光’, so we get the result of the segmentation.

昨晚|我|在|秋明山|输给|一辆|五|菱宏光

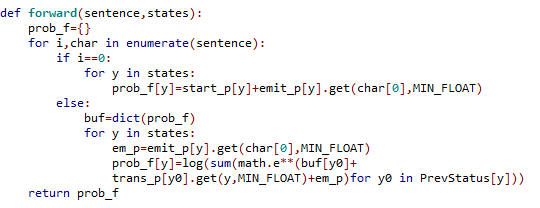
We can see that the segment of ‘五菱宏光’ exists error. The best way is to add ‘五菱宏光’ to Dictionary.

But can we solve this problem by using HMM?

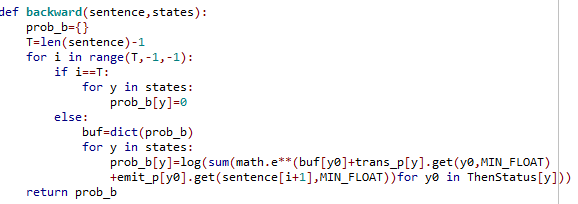
We use Forward and Backward algorithm to calculate the frequency of ‘start-probability’, ‘transformation-probability’ and ‘emit-probability’, and the algorithm also can upload the new frequency that is more accurate. We also can use the algorithm to train our lexicon.

Here is our procedure to implement forward-backward algorithm.

Here, the sentence is a fraction of the full sentence, the forward algorithm based on the former part.)



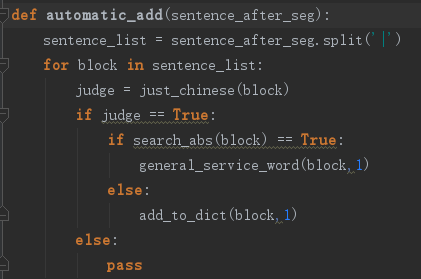
And this is the code of backward algorithm, calculation from the last word.



Finally, we do the EM algorithm, which can upload the three kind of frequency.

1. **Lexicon**
2. **Forming the lexicon**

At first, we found a basic lexicon via the Internet. And then, we defined a function automatic\_add(sentence\_after\_seg):



Using this function, we need input a text that has been segmented already. Then the function will segment the text to block which can be Chinese word, number, English letter and so on, after which the function will use function just\_chinese(word) to judge if one block is a Chinese word. Next, if the block is a word just combined of Chinese characters( For we cannot judge if a word has both English and Chinese in it is a Chinese word we use, so we just add the pure Chinese word to the lexicon automatically.), the function will check if the word is in the lexicon. If the word is in the lexicon, then the function will add 1 frequency to the word. Or the function will add the word to the lexicon.

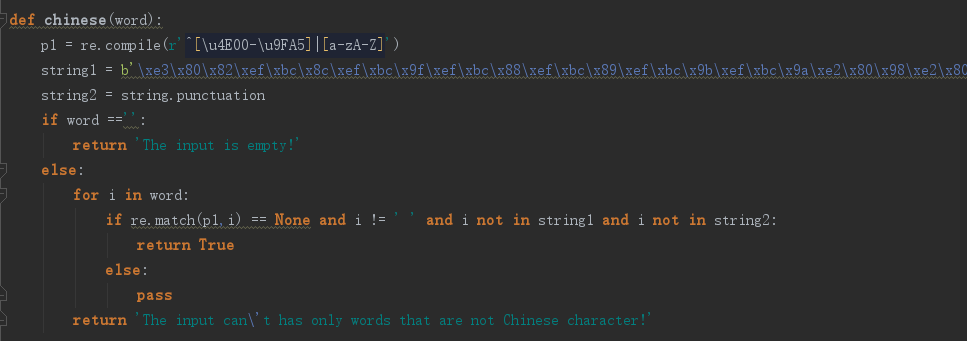
We downloaded many texts in the Internet and assigned them to the function to improve the lexicon.

At last, we use the text in the website’ [www.g20chn.cn’](http://www.g20chn.cn’) segmented by ourselves to improve the lexicon.

1. **Judge if a word is a Chinese word**

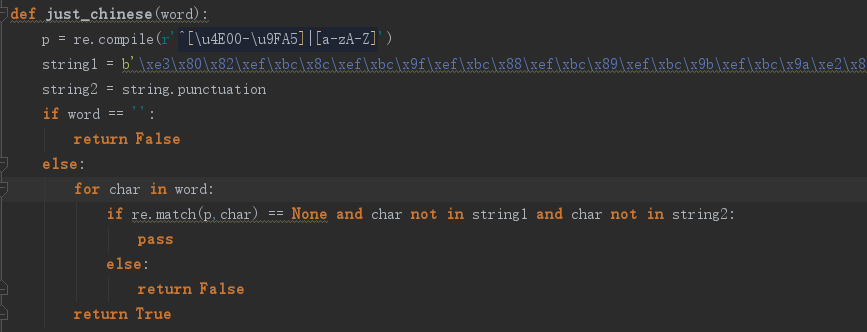
There are two functions in this part.

First, the function chinese(word):



In this function, firstly it defines a Regular expression standing for all Chinese characters. Next it judges if the word is empty. Then it use a for loop to find if there is Chinese characters in the word. If there is, return True, which stands for its maybe a Chinese word. Or it returns False to tell the user this is not a Chinese word.

The second function is just\_chinese(word):

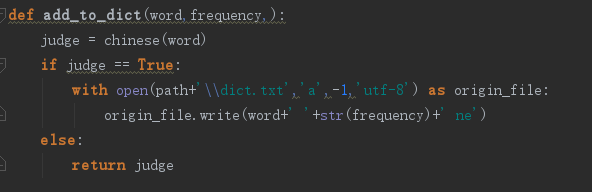


In this function, it also defines a Regular expression standing for all Chinese characters and it defines 2 strings standing for most of the punctuation both in Chinese and English. Next it judges if the word is empty. Then it use a for loop and a if statement to find if the word is just combined of Chinese characters. If there is a character is not, return False. Or when the loop run over, it returns True.

1. **Operation to lexicon**

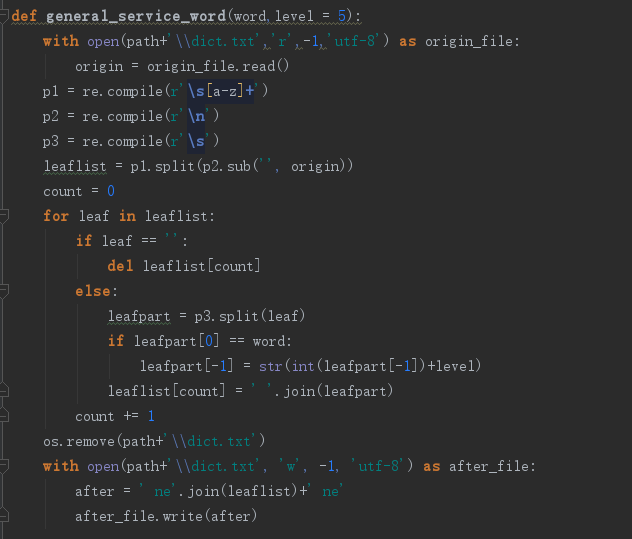
There are four operations to the lexicon.

1. Add new words to the lexicon. We write a function add\_to\_dict(word,frequency):



In this function, it transfers the function chinese(word) at first to judge if the input word maybe a Chinese word. Then, if the word is, the function will add this word to lexicon using open function with the ‘a’ model.

1. Set up the general service word. We write the function general\_service\_word(word,level = 5):

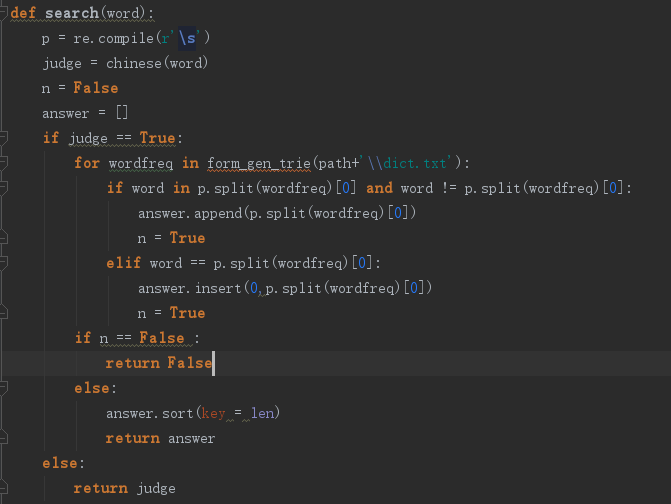


Firstly, this function open the lexicon with the ‘r’ model. Secondly, the function define 3 Regular expressions as p1, p2 and p3. Then use p2 to remove all the ‘\n’ in the lexicon and use p1 to split the string of the lexicon to a list with word and their frequency. Thirdly, remove the empty units of the list and judge the unit of the list which is equal to the variable word and increase its frequency according to the level. At last store the new lexicon to the file.

1. Search word

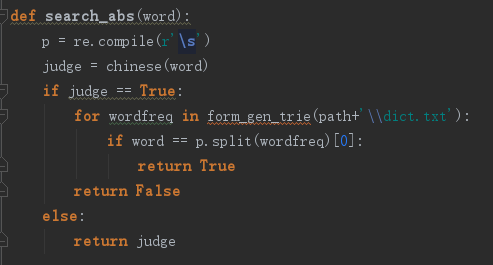
There are two function in this part:

The first is search(word):



This function can find all the words in the lexicon including the input word. It will judge if the word is a Chinese word at first. Then there is a loop. The loop will add all words in the lexicon including the input word to a list and assign True to n if there is related word. At last, if the n is True, return the list including the related words. Or it will returns False.

The second is search\_abs(word):



This function is just same with function search, but this just search the word same with the input word.

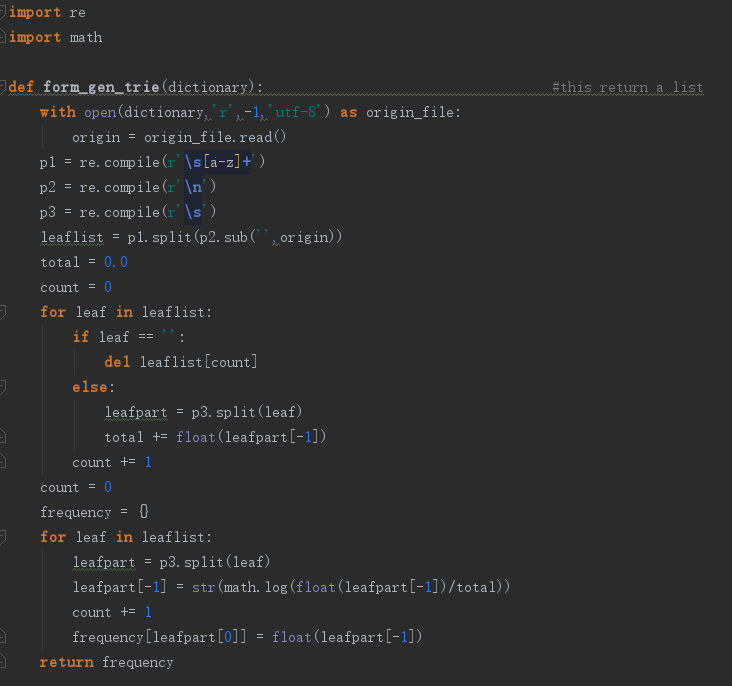
1. Automatically add word to lexicon.

We have introduce this function above.

Function: automatic\_add(sentence\_after\_seg)

1. **Forming the list of the words and their frequency**

In this part, we write a function form\_gen\_trie(dictionary):



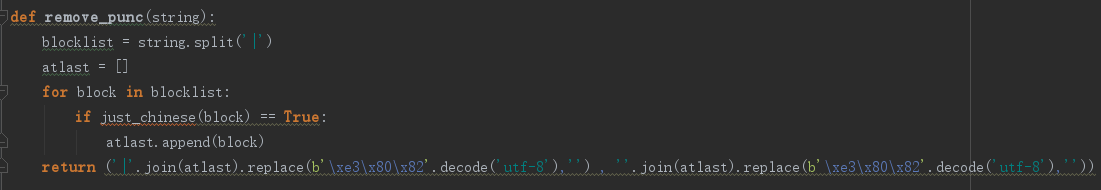
In this function, we write to for loop. The first loop is used to count the sum of all the words in the lexicon. The second is used to calculate the mark of every word in the lexicon. Then store the word and its mark to a list. Using the list in the main segmentation program.

The function define 3 Regular expressions as p1, p2 and p3. Then use p2 to remove all the ‘\n’ in the lexicon and use p1 to split the string of the lexicon to a list with word and their frequency.

1. **Forming a string of a sentence that has been segmented without any punctuation and mark every character with ‘B,M,E,S’ according to its position in a word.**

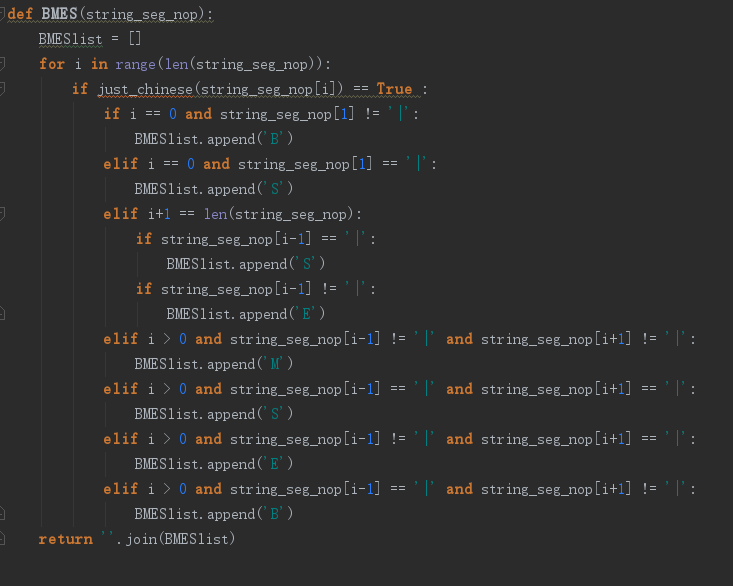
This part is the preparatory work of the generation of the probability of every Chinese character’s position(B,M,E,S), which is combined of two functions.

The first function is defined to form a word list without any punctuation:



This function use function just\_chinese(word) to pick out all the words only combined of Chinese characters and add them to a list. Then join the list with ‘’ or ‘|’ to form two string. The string with ’|’ is used to form the string of the ‘BMES’ and the string without it is used to be compared with the string of ‘BMES’.

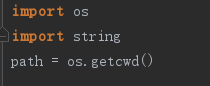
The second function is defined to form the string of ‘BMES’:

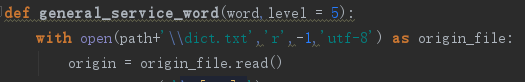


This function transforms the texts having been segmented to a string of every character’s position. B stands for begin, M stands for middle, E stands for end and S stands for isolation. This function marked every character according to its relative location to the separator ‘|’.

1. **The problem of path**

We use the function os.getcwd() to get the absolute path of the main program and use the relative path between the main program and its files to get the path of these files, such as:





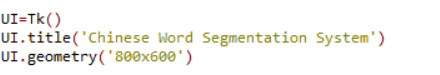
1. **User Interface**
2. Solution Design:

Create the main UI and other ones.

Add widgets including labels, buttons, menus, textboxes, etc.

Add commands to the widgets to make the system work.

1. Solution Implementation:
2. The main UI:



1. Some sub UIs:





1. Some of the widgets:











1. Some of the commands:

