Improved Chinese Language Processing for an Open Source Search Engine

CS 297 Project Report

Presented to

Dr. Christopher Pollett

Department of Computer Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Class

CS 297

By

Xianghong Sun

December 2019

Abstract

Yioop is an open source search engine. It supports many languages and has abilities to process and analyze text. However, the support for most languages in Yioop is limited. In order to get better processing on a specific language, some extensions are needed. My goal for this project is to extend the support for the Chinese language in Yioop. In this semester, I extended the Chinese language features of Yioop in these three areas: Chinese words Segmentation, Chinese words Part of Speech Tagging, and Chinese Question and Answering System. In order to accomplish the tasks, I have spent months getting familiar with the current Yioop system and starting to understand how Natural Language Processing works in real world.

Keywords: Yioop, Search Engine, Natural Language Processing, Chinese, Segmentation, Part of Speech Tagging, Question and Answering

Table of Contents

I. Introduction

Almost everyone uses search engines, such as Google and Bing. When users search something in a search engine, the search engine needs to fetch the related documents by looking for the keywords and return the results in relatively short time. Yioop is an open source search engine software written in PHP. It provides many features, such as search results, media services, social groups, blogs, and wikis. In order to process the input users are entering, many NLP techniques are used. For example, if the user is entering a question in Chinese, the engine needs to segment the sentence, generate the key words, fetch the database to find the results, and extra the answer, etc. How good the engine can process depends on how good each feature is implemented.

For Chinese part, term Segmentation, POS tagging and Question and Answering are three important NLP features. In this paper, I am going to introduce how I implemented those three features as well as a Chinese word suggestion system one after another.

The Chinese term segmentation is usually the first step in Chinese NLP, so it is overall the most important feature. If it does not work properly, such as having a low accuracy, the other features that depends on it will also have a low accuracy. The algorithm for Chinese Segmentation in current system is reverse maximum matching algorithm. It is one of the most efficient algorithms to segment Chinese, but it does not have a good accuracy compared to the latest research. So, I improved this feature by implementing a Stochastic Finite-State Word-Segmenter that segment Chinese based on the term weights. It significantly improves the current Chinese Segmentation System.

The Chinese Part-of-Speech tagging is also an important feature in Chinese NLP. It does not directly solve any NLP problem, but it can help other feature's accuracy. It would be used in my next part, which is the Question and Answering System. There was no POS tagger for Chinese in Yioop System. The POS tagger for English used an algorithm that assuming an unknown word is noun and changing it if the POS of the word did not fit the sentence. Currently, most of the researches of POS tagging for Chinese focuses on Machine Learning. However, Yioop System did not have the right ML libraries for me to do this way. So, my implementation focused on the terms around the one needed to be tagged.

The last part of my project is a Question and Answering System. The major process was introduced in my first paragraph. Currently, it is implemented partially. I will continue working on it in next semester.

In next several sections, I will explain every deliverable in detail.

DELIVERABLE 1: Build a Chinese word suggestion dictionary on Yioop

My first deliverable was to get familiar with Yioop by adding a Chinese word suggestion dictionary to the system. In this system, when a user searches something in the search bar, there will be some words or phrases listing out to make suggestions. The words are usually ranked by their search frequency. my task was to obtain a Chinese word dictionary, find the frequency of the words, use the Yioop to build the Chinese word frequency dictionary, and add the dictionary to the system.

The Chinese words I obtained was from CC-CEDICT. As stated on the website, CC-CEDICT is “a continuation of the CEDICT project started by Paul Denisowski in 1997 with the aim to provide a complete downloadable Chinese to English dictionary with pronunciation in pinyin for the Chinese characters.”

The second step was to get the word frequency people are searching. The best place to find it is Google. So, I made use of Google Trends to find it. The problem of Google Trends was that it did not show the frequency directly. It gave a number that showing the interest over time. A value of 100 was the peak popularity for the term. A value of 50 meant that the term was half as popular. A score of 0 meant there was not enough data for this term. Every word which had some data would always have a value of 100 in its peak. The way I calculated the score was based on stability. If a word was searched very frequent in one time but dropped down quickly, it meant it was not stable, and would have a low score. If a word was searched frequency most the time, it was stable, and would have a high score. After some days of requesting data from Google trends, I got the frequency of every word in the dictionary. There were some known problems of the

frequency dictionary. Not only Chinese search Chinese, Japanese characters also contains Chinese characters. Some words were more frequently searched by Japanese than Chinese.

Next step was to format the dictionary so Yioop can made use of it. Yioop had some build in tools to make the dictionary so I passed the file to it to generate the final file.

After I updated the file, I figured out the words are not ranked in frequency because the JavaScript did not rank it properly. So, I rewrote some JavaScript code to make it work.

Figure 1 is a screenshot of the result of my changes working.

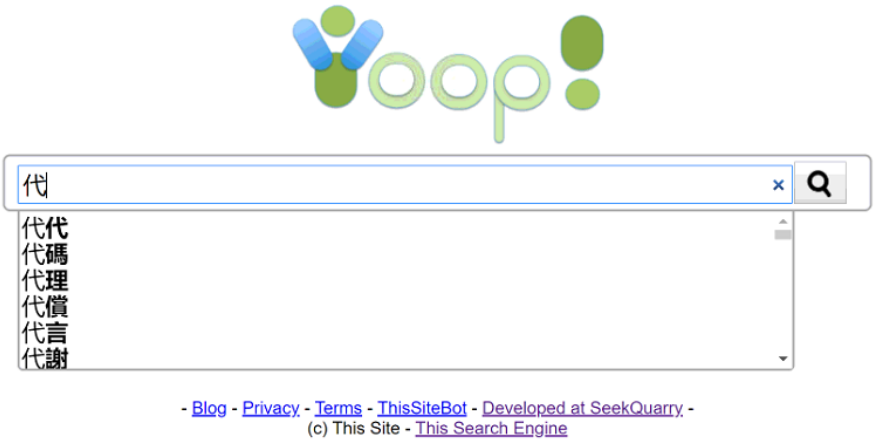

Figure 1: Chinese word ranked by frequency.

DELIVERABLE 2: Implement a Chinese Word segmentation Algorithm

For searching convenience, a search engine use crawls to index web pages to an inverted index. When a user searches for a word, the search engine searches the inverted index to get the web pages. For most Western languages, like English and Spanish, words are separated by space. So, it is easier to index the Western languages into an inverted index; however, many Asian languages do not have a delimiter, which means it is harder to segment the sentences into words and index the them. Currently, the Chinese segmentation method in Yioop uses backward maximum matching to segment Chinese text. The backward maximum matching algorithm is a very general algorithm for most languages that do not have delimiters between terms. However, compared to latest algorithms, the results are not very good. I built a Stochastic Finite-State Word-Segmentation Algorithm [1] to achieve this goal.

Sproat [1] describes the concept of this algorithm. First, we need to train some data to obtain the frequency of each terms. The term frequency here is different from Deliverable 1 since Deliverable 1 is the search frequency. We calculate a weight for each word based on this frequency by using this formula, where f is the frequency of the word and N is the corpus size:

$$C = -\log\left(\frac{\mathrm{f}}{\mathrm{N}}\right)$$

For each sentence, we start by segmenting them in all possible ways. For each way, we calculate the weight of the sentence by adding up all the weight of the words. The way with lowest weight of segmenting the sentence will be the answer.

The result seems good. Academia Sinica(AS) dataset has an accuracy of 96.5% and Peking University(PKU) dataset has an accuracy of 89.8%.

The reverse maximum matching method in Yioop has some bugs. It will mess up if there is English letter or Arabic numbers in the sentence. If these bugs are fixed, the accuracy will be better. The RMM method in current Yioop system has result of 80.42% for AS and 55.10% for PKU. PKU one has very low accuracy is because of the bugs I stated previously.

Figure 2 is the result of the implementation.

Figure 2: Chinese Term Segmentation

DELIVERABLE 3: Implement a Chinese Part-of-Speech tagging system

Part-of-Speech tagging, also known as POS tagging, is the process of annotating terms with part of a speech it corresponds to. POS tagging itself does not directly solve any particular NLP problem, but it is an important tool for us to help simplify many NLP problems. Yioop did not have any POS tagging system for Chinese, so this was a new feature I am adding to the system.

Most POS tagging problems are caused by the ambiguous words and unknown words. Compared to English and German, Chinese words are more ambiguous. Tseng [2] shown in his paper that "29.9% of tokens in CTB had more than one POS tag, while only 19.8% and 22.9% of tokens were ambiguous in English and German, respectively." Tseng [2] also stated that "40.6% of unknown words were proper nouns in English, while both Chinese and German had less than 15% of unknown words as proper nouns." Table 1 shown the unknown words of different languages. These features needed to be considered while tagging the words.

Language	*English%*	*German%*	*Chinese%*
Proper nouns	40.6	12.2	14.0
Other nouns	24.0	53.0	41.5
Verbs	6.8	11.4	19.0
ALL	100.0	100.0	100.0

Table 1 unknown words in English, German and Chinese

The model I was implementing was based on the words surrounded by the word to be tagged. Assuming the current word the model was tagging had index 0. The features to be considered in my model were the second word before it, which had index -2, the first word before it, which had index -1, the first word behind it, which had index 1, and the second word behind it, which had index 2. For each weighted feature, I should have implemented a better algorithm of how to calculate the weight for each of the features. Currently the weights were hardcoded. It

should be extended so that it is calculated to maximize the probability. Finally, I used the Viterbi algorithm to calculate the most likely POS of the word. Figure 3 is the result of POS tagging.

It had a very high accuracy on training set, 92.4%, as expected. But, when I used a test dataset, the accuracy dropped to 74.6%. I will put more work on this next semester to make it better.

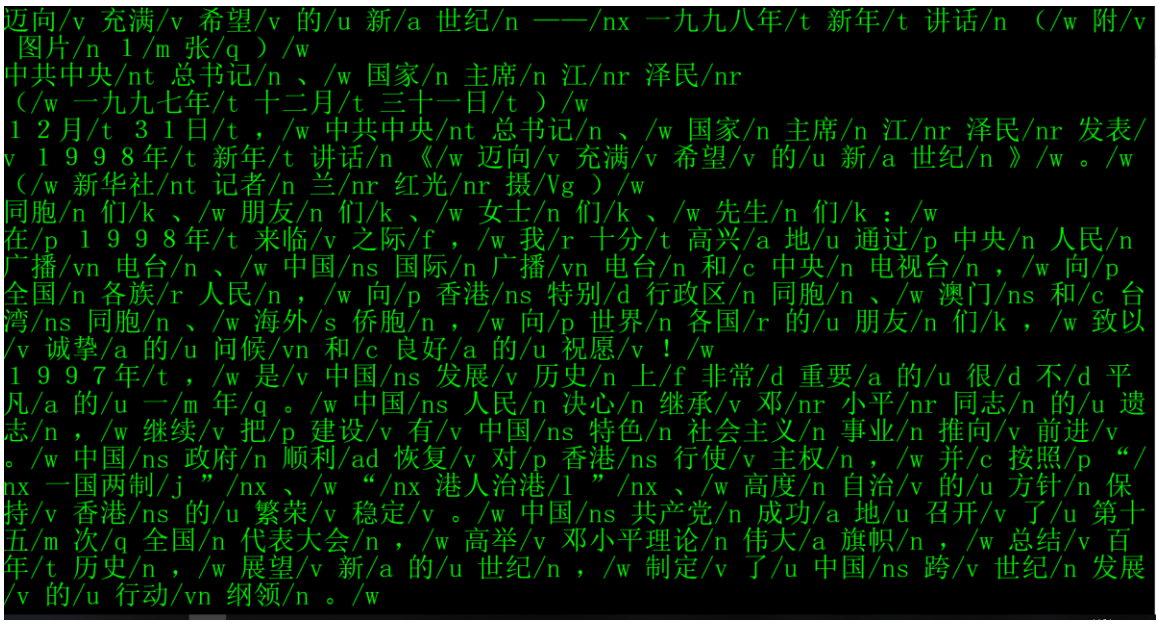

FIGURE 3: Chinese POS tagging

DELIVERABLE 4: Implement a Chinese question and answering system(partial)

Question answering (QA) is a system that can automatically answer questions posed by humans. Currently, QA system finds answers that are already known by somebody and have been written in some web pages, rather than generating new knowledges.

My QA system was still under development as I was writing the report. Now it can only extract keywords and define the answer type. Extracting answers part will be the tasks for next semester.

There are different types of questions and answers. My system design focuses on the answer type. My system will first extract key words of the question to see what type of answer it should generate. The answer types can be put into these classes:

- Person (from "Who . . . ")
- Place (from "Where . . . ")
- Date (from "When . . . ")
- Number (from "How many . . . ")
- Explanation (from "Why . . . ")
- Method (from "How . . . ")

Then I used POS tagging to tag the original question. Nouns and verbs are the primary key words in the question and other words are the secondary keywords. Stop words should also be removed. Now I have obtained the keywords for the question. The following steps are the task for me to do in the future implementation.

After that, I need to use the keywords to generate query for Yioop Query System to find search related pages for me to get candidate answers.

My system will go through the candidate answers and extract sentences from it. The Chinese sentences are different from the English sentences since one Chinese sentence might contain multiple English sentences. You can think about some English sentences separated by comma

instead of period. That is caused by Chinese gramma. In Chinese, if one sentence has some relationship with the sentence after it, you can use a comma instead of a period and you can even omit the subject of the sentence. That causes some problems when I was trying to extract answers. My next plan is to deal with these problems and complete the system. Figure 4 is the current process on the QA system.

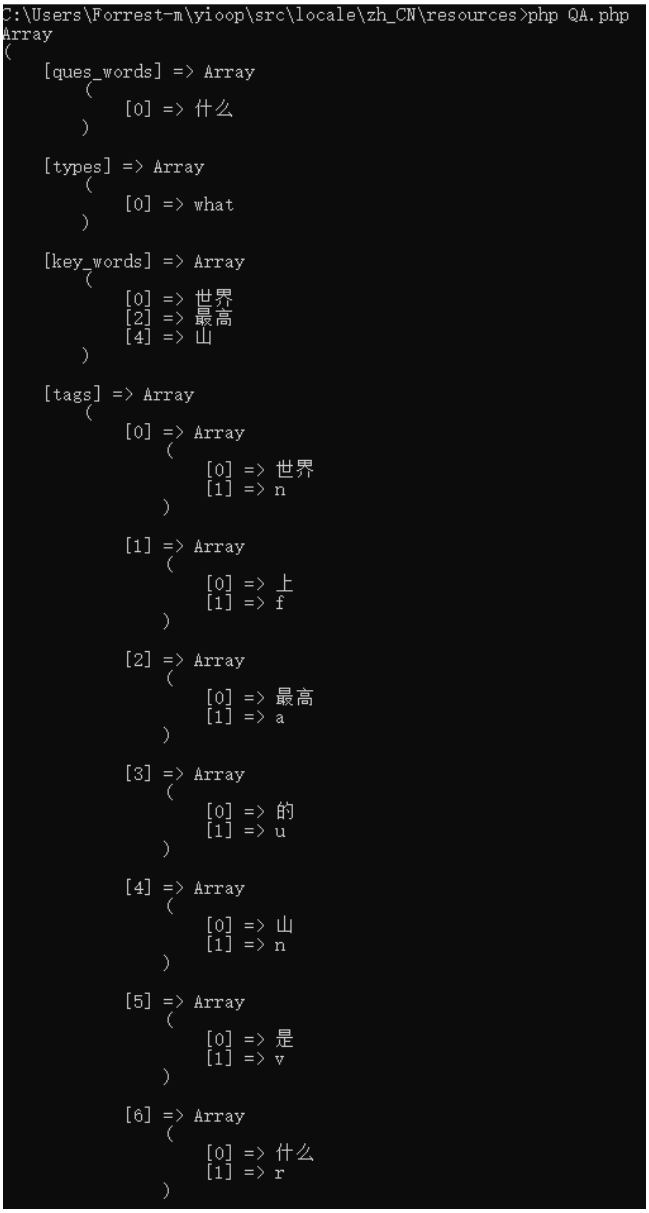

Figure 4: Chinese Question type and keywords extraction

Conclusion

The CS297 project gives me a starting point of knowing how NLP works in real world. I learned a lot of NLP techniques by reading implementing the tasks in this project. Although some of the programs I implemented do not have their expected accuracy, I can improve them in my CS298 project and make them better.

Overall, I think the goals for the CS297 project is to learn some basic knowledge for me to implement something more advanced in CS298 and I am looking forward to adding more and better Chinese language support in Yioop search engine.

In the CS298, part of my goals will focus on improving the current implementation on POS tagging and question and answering system. And, other part will be implementing more features such as build an API for work suggestion system based on the type of inputs.

REFERENCE

[1] Sproat, Richard & Shih, Chilin & Gale, William & Chang, Nancy. (2002). A Stochastic Finite-State Word-Segmentation Algorithm For Chinese. Computational Linguistics. 22. 10.3115/981732.981742.

[2] Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. 2005. Morphological features help pos tagging of unknown words across language varieties. In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing.

[3] J Prager - Information Retrieval, 2006 Open-Domain Question-Answering

[4] Pollett, C. "Open Source Search Engine Software!" Open Source Search Engine Software. https://www.seekquarry.com/