Data: From Patient to Health Record

Thesis submitted to the University of Nottingham for the degree of
BSc Computer Science, Apr 2021.

Yiming LI

20031525

Supervised by Dr. Boon Giin LEE

School of Computer Science University of Nottingham, Ningbo, China

Abstract

Artificial Intelligence (AI) has a huge potential to improve the efficiency of clinical tasks. Nowadays, doctors are overwhelmed by typing records into computers using traditional keyboard and mouse and they usually spend more time on typewriting than diagnosing the patients. Voice recognition method can transcribing doctors' speech into plain text instead of typing in all the information manually. This can help doctor reduce clerical work and reduce the time required to diagnose each patient. Together with Natural Language Processing (NLP) methods, which can extract key information from the text recognized by voice recognition method, this project provides a feasible solution for doctors to fill in the EHR form and diagnose the patient simultaneously without distraction. Therefore, doctors can focus on interactions with patients and avoid manual mistakes. However, in Chinese clinical field, there were currently no effective automatic filled Electronic Health Record (EHR) systems focusing on speech recognition that can be used. This project aimed to create an automated report generation system for doctors, which was capable of transcribing real time voice input or audio files into plain text and processing and extracting the key information to fill in the EHR form. The challenge was to achieve high accuracy in voice recognition and key information extraction to minimize the manual inspections. A simple and basic auto-diagnose function was included in this project as additional work, towards a potential smart-based EHR system for future prospects.

Acknowledgements

This project could not have been completed without the help from my supervisor - Dr. Boon Giin LEE, to whom I express my highest gratitude for helping me on every stage of the way and always supporting me during the hard times in this project.

I also want to express my appreciation to Tianlang Tan, Chengtao Luo and Jingwen Zhou, who also cooperate their Final Year Projects with Dr. Boon Giin LEE. I learned a lot from them as peer experiences.

Contents

Abstract i

Acknowledgements ii

List of Tables v

List of Figures vi

Abbreviations vii

Chapter 1 Introduction 1

1.1 Background 1

1.2 Motivation 3

1.3 Aims and Objectives 4

1.4 Report Outline 5

Chapter 2 Related Work 6

2.1 Overview of Voice Recognition 6

2.2 Overview of Nature Language Processing 9

2.3 Overview of Voice Assistant for EHR 11

Chapter 3 Project Specification 13

3.1 Functional Requirement 13

3.2 Non-Functional Requirement 14

3.3 Software Specification 15

Chapter 4 Methods 16

4.1 System Design 16

4.2 UI Design and Implementation 28

4.3 Additional Design 29

Chapter 5 Result and Discussion 31

5.1 Result 31
5.2 Additional Result 33
5.3 Discussion 34
Chapter 6 Conclusion **38**
6.1 Conclusion 38
6.2 Future work 39
Chapter 7 Reflection **40**
7.1 Project Management 40
7.2 Contribution 42
7.3 Reflections 43
References **45**
Appendices **48**
Appendix A Research Ethics **49**
A.1 Ethics Form 50
Appendix B Pseudo code **51**
B.1 Crawler Algorithm for dataset collection. 51
B.2 Baidu Voice Recognition Method. 53
B.3 Google Voice Recognition Method. 54
B.4 Jieba Language Process Method. 55
B.5 Save CSV File to Local Directory. 56
B.6 LSTM for disease prediciton. 57

List of Tables

2.1 Top 5 information extraction frameworks/tools included in publications from 2009-16. 10

3.1 Software Specification. 15

4.1 Trigger words. 27

5.1 Comparison between Baidu and Google voice recognition methods. 32

5.2 Comparison between different NLP modes. 33

List of Figures

2.1 Voice Recognition Workflow 7
2.2 Baidu AI server Main Page 8
2.3 Google AI server Main Page 9
2.4 Features and Usage Documentation for Jieba. 11
2.5 Iflytek Main Page 12

4.1 Activity Flow Diagram. 17
4.2 Crawler Algorithm Working Flow. 18
4.3 Sample Dataset. 20
4.4 Labeled cases in Dataset. 21
4.5 Baidu voice recognition Working Flow. 22
4.6 Google voice recognition Working Flow. 23
4.7 Jieba Language Process Working Flow. 25
4.8 Example UI for EHR. 28
4.9 User Interface of the Application. 29

5.1 Accuracy and Loss of LSTM network. 34
5.2 Result of Perfect Speech Input. 35
5.3 Result of Speech Input with noise. 36

7.1 Original Timetable. 41
7.2 New Timetable. 41

Abbreviations

AI Artificial Intelligence.

API Application Programming Interface.

CEHRS Chinese EHR System.

EHR Electronic Health Record.

LSTM Long Short-Term Memory.

MoH Ministry of Health.

NLP Natural Language Processing.

PoS Part of Speech.

WPM Words Per Minute.

Chapter 1

Introduction

The Electronic Health Record (EHR) serves as a collection of patients' health record and health status throughout their whole life for clinical purposes. It is a computer information system which can support data collection, storage and access in both hospitals and healthcare centers to uniform the medical record format [1]. The utilization of Artificial Intelligence (AI) into EHR form auto-filling system is worth to be investigated for improving the interaction between doctors and patients, and subsequently reduce doctors paperwork workload. This chapter will introduce the background and motivation of this project and outlines the aims and objectives of the work.

1.1 Background

Comparing to traditional paper records, EHRs offered advantages such as remote data access, unified data standard, searchable digital database and integrated patient records including medical history [2]. The Chinese government, as well as Ministry of Health (MoH) of China, had regarded EHR as an efficient tool to improve the safety and quality of Chinese health care service and set a goal to ensure the universal usage of EHR among the whole population in most of the hospitals and clinics by the end of 2020 [3]. A research

taken by Jennifer King demonstrated that over 75% of EHR adopters identified that EHR enhanced the health care service [4].

EHRs offered more efficient entry and retrieval of relevant patient information. However, a potential weakness of EHR was the discommodious input interaction using traditional keyboard and mouse. The maximum number of words per minutes (WPM) was 80, when concentrating on typing [4]. A 2016 study estimated that doctors spent between 37% and 49% of their working hours on clerical tasks [5]. Doctors were overwhelmed by this clerical work and had a great possibility to make serious mistakes by typing manually under this circumstance [6]. All that paperwork contributed to the high level of burnout and depression in the profession, according to a 2018 study [2].

To solve the inefficiency of current situation of traditional input of EHR system, the potential of AI in clinical workflow has been explored in many studies [7]. It showed its influence on clinicians, health systems and patients since it produced accurate image interpretation and high accurate health diagnosis while reduced manual errors [8]. The technical behind involved Voice Recognition, Natural Language Processing (NLP) and Deep Learning. Such a system can serve as clinical stenographers that transcribe doctors' observations and instructions and insert them into patient's EHRs [2]. After all relative information were extracted, possible diseases analyzed by a deep learning model could be listed to reduce the probability of misdiagnose. With the assist of this automated report generation system, it would liberate doctors from tedious clerical work and improve the accuracy and quality of EHRs.

As the practice of voice recognition in the past 10 years, the accuracy of the outputs was not ideal. A research in 2010 implemented a voice recognition method and compared result with the manually translated content. The average accuracy was less than 82%, with 6.1% of incorrect recognition and 11.2% of rejected voice [9]. Even though speech averages about 110-150 Words Per Minute (WPM) and typing was only about 40 WPM, due to the poor performance of voice recognition system, 70% of extra time was required to correct errors [2].with the development of the machine learning, plenty of voice recognition

methods or pretrained models were provided with a higher accuracy, such as Google voice recognition Application Programming Interface (API) [10] and Baidu voice recognition API [11]. Combined with noise reduction algorithm to obtain a clear voice input, the accuracy would achieve near 99%, if the speech was made clearly.

Apart from voice recognition methods, an approach was necessarily required to allow machine to derive meaning from human languages, as well as decompose a sentence into independent words. Many Natural Language Processing (NLP) studies had been conducted and developed to analyze the Part of Speech (PoS) and the meaning of languages. Tested by Che et al. [12], a sufficient accuracy and speed have been attained in some of Chinese processing modules, including WordSeg (97.4% of accuracy, 185KB/s of speed), POSTag (97.80% of accuracy, 56.3KB/s of speed), NER (92.25% of accuracy, 7.2KB/s of speed) and so on. With a further training on an additional dictionary containing specific disease names and medical drug names, it could be perfectly adapted to medical segmentation analysis.

However, the number of research conducted on EHR in China was much lower than researches in the USA. During 2008-2017, there were 1031 publications on EHR in the USA while there were only 173 publications in China [13]. Research on EHR was a relatively new emerging and promising field in China. This project mainly aimed to explore the feasibility of implementing voice recognition methods (speech to text) and NLP methods (text analysis) in Chinese EHR System (CEHRS) to contribute to EHR in medical field. A simple and basic auto-diagnose function was included in this project as well.

1.2 Motivation

EHR system was widely used in Chinese hospitals and clinics [3]. However, a potential weakness of EHR was the discommodious input interaction using traditional keyboard and mouse, which was time-consuming and distracting. Doctors were overwhelmed by clerical

work and had a great possibility to make serious mistakes when filling in EHR manually [6]. To solve this problem, it was worthwhile to find out the feasibility of an automated report generation system, which could translate real time voice input or audio files into plain text and process and extract key information to fill in the EHR form automatically. It would liberate doctors from tedious clerical work and reduce the likelihood of human errors occurring.

1.3 Aims and Objectives

The main aim of this project was to create an auto-filling system using voice recognition and natural language processing to reduce the working load of doctors. Using this system, the speech of doctors should be recognized and analyzed to fill into the EHR form at the same time when doctors are diagnosing the disease of patients. After that, the system should provide a preliminary prediction of possible diseases from extracted information. In this way, doctors can save those time spent on writing clerical documents or typing in digital records and be more focused on diagnosis. The system needed to be capable of translating real time voice input or audio files into plain text and process and extract the key information to fill in the electronic health record (EHR) form. In other words, the system was an automated report generation system, which can serve as clinical stenographers that transcribe doctors' observations and instructions into text and insert them into a patient's EHR.

The key objectives of this project include:

1. Collect disease classification used as labels for model training, validation and testing..
2. Collect doctor's diagnose prescription in voice format for models training, validation and testing.
3. Implement voice recognition methods for transcribing doctors' speech into text.

4. Implement Chinese language processing methods for analyzing the text and fill in the EHR.

5. Exploring different voice recognition methods and different Chinese language processing methods for performance analysis.

6. Develop a prototype to demonstrate the proposed work for simulating the realistic situation.

The extension objectives and future work of this project include:

1. Collect symptom descriptions for training, validating and testing the disease prediction model.

2. Predict possible diseases using deep learning methods.

1.4 Report Outline

1. This chapter explained the definition of EHR, described the background and motivation of this project, as well as provided the aims and objectives of the work.
2. Chapter 2 described related work in the field of voice recognition, natural language processing and Chinese EHR auto-filling system.
3. Chapter 3 listed both functional requirements and non-functional requirements of this system.
4. Chapter 4 outlined system design including all the methods implemented in this project and user interface design.
5. Chapter 5 showed the results of implementation steps and had a discussion on discovered problems.
6. Chapter 6 presented conclusion of this project and gave an overview on future work.
7. Finally, Chapter 7 expressed project management and reflections on completed work.

Chapter 2

Related Work

This technical aspects of this project involved AI approaches, including voice recognition and natural language processing (NLP). Voice recognition was required to transcribe doctors' speech into text and NLP was required to extract key information from this text. With these two methods, the AI-based EHR system could automatically fill in the EHR forms when doctor was diagnosing the patients. This chaper will introduce relative studies on voice recognition, state-of-the-art of NLP, and current status of AI-based EHR in healthcare industry in China.

2.1 Overview of Voice Recognition

Voice recognition is a process to transform audio input into plain text output. Voice recognition system captures and converts speech via a microphone. Some voice recognition applications can transcribe recordings from a number of formats, including wav, mp3 and wma. This project focused on Chinese language voice recognition. Unlike English, Chinese has several different words with the same pronunciation and a wide range of accents.. Google voice recognition API [10], Iflytek voice recognition API [14] and Baidu voice recognition API [11] are the main developers that provide Chinese language recognition

service. All these APIs were tested in this project to examine the accuracy of each API.

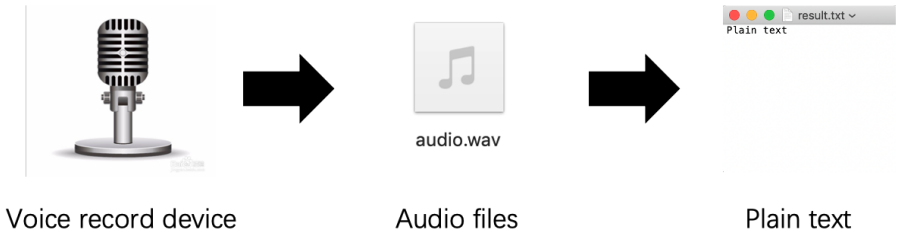


Figure 2.1: Voice Recognition Workflow

2.1.1 Baidu Voice Recognition

Baidu, as the leading internet-searching company in China, puts great effort on AI field in recent decades, including voice recognition. This technology relies on deep learning, which involves training a very large multilayered virtual network of neurons to recognize patterns in vast quantities of data. Baidu uses this technology on its own application to let users perform the search operation by voice, and it also provides a voice-controlled personal assistant named Duer. Baidu voice recognition API is a pre-trained model and supports the self-training model on the voice self-training platform. Self-training can be completed by uploading the vocabulary text as a dictionary. It can accurately improve the vocabulary recognition rate of the specific domain by 5-20%. The developer needed to register for an account where an access key will be allocated for authentication purpose to allow the use of Baidu voice recognition API in the self-developed application. Every user can use this API freely for 40,000 times [11].

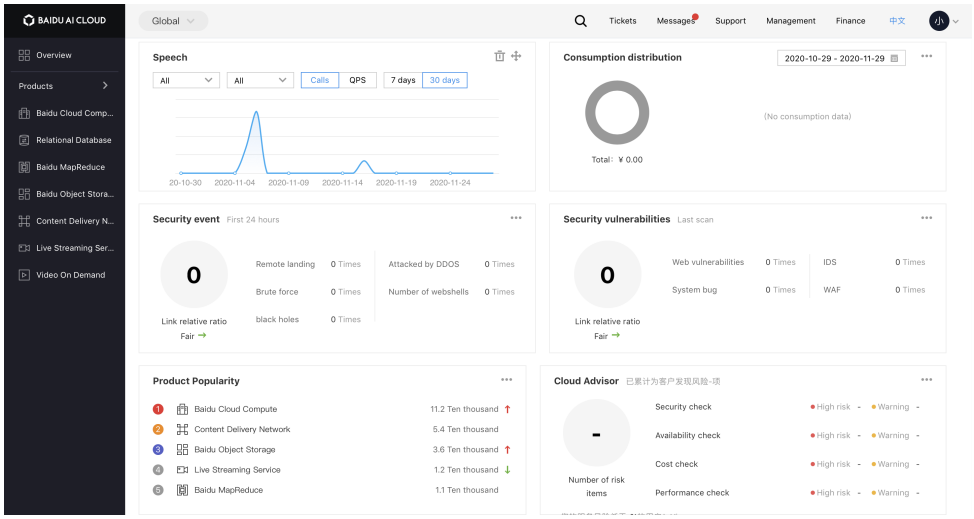


Figure 2.2: Baidu AI server Main Page

2.1.2 Google Voice Recognition

Google with voice recognition API supports voice recognition that supports more than 125 languages and variants for free, without the restriction of access key. There is a selection of trained models for voice control, phone call and video transcription optimized for domain-specific quality requirements. Real-time speech recognition can be achieved as the API processes the audio input streamed from application's microphone or sent from a prerecorded audio file. The same as Baidu voice recognition API, it can customize speech recognition to transcribe domain-specific terms and rare words and boost transcription accuracy of specific words or phrases. In addition, it can automatically convert spoken numbers into addresses, years, currencies, and more using cases. Moreover, Google voice recognition API for python is simple to use, by simply importing a library called 'speech_recognition' and invoke a function, without any authentication and time limitation [10].

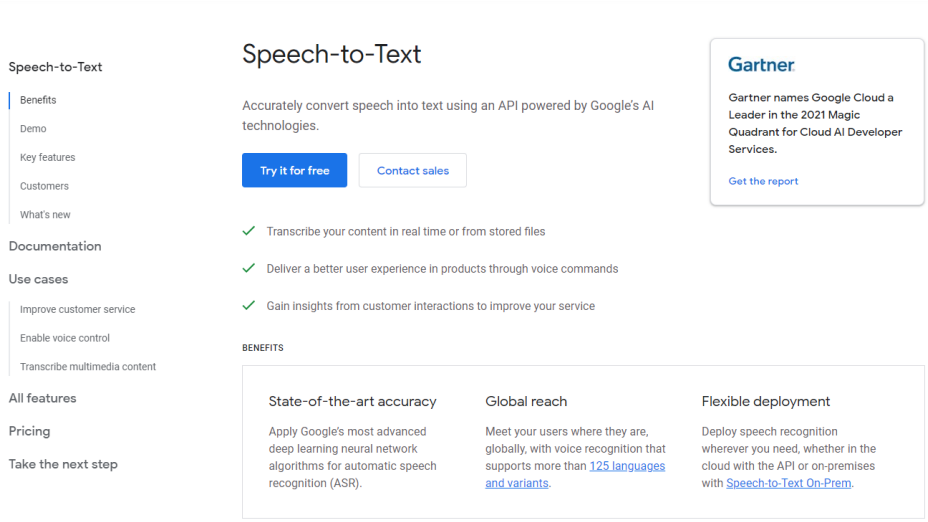


Figure 2.3: Google AI server Main Page

2.1.3 Iflytek Voice Recognition

Iflytek Voice Recognition API for python is almost the same as Baidu's API, which requires registration for an APIkey. However, it requires one more authentication, which is adding whitelist for users' IP addresses. This will become an obstacle for users to use because it requires further manual settings before running the code. In this case, Iflytek Voice Recognition API will not be implemented and tested in this project [14].

2.2 Overview of Nature Language Processing

NLP is in the field of linguistics and artificial intelligence concerned with the interactions between computers and human language. The main idea is how to program computers to process and extract key information. With NLP, a computer will be capable of extracting information and insights contained in the documents. An NLP system includes two parts: syntactic processing modules and semantic processing modules [15]. Several different NLP systems based on English have been utilized to extract key information from clinical contents, such as MedLEE, MetaMap, KnowledgeMap, cTAKES, HiTEX and MedTagger [15]. 65% of existing NLP systems use rule-based methodology while the

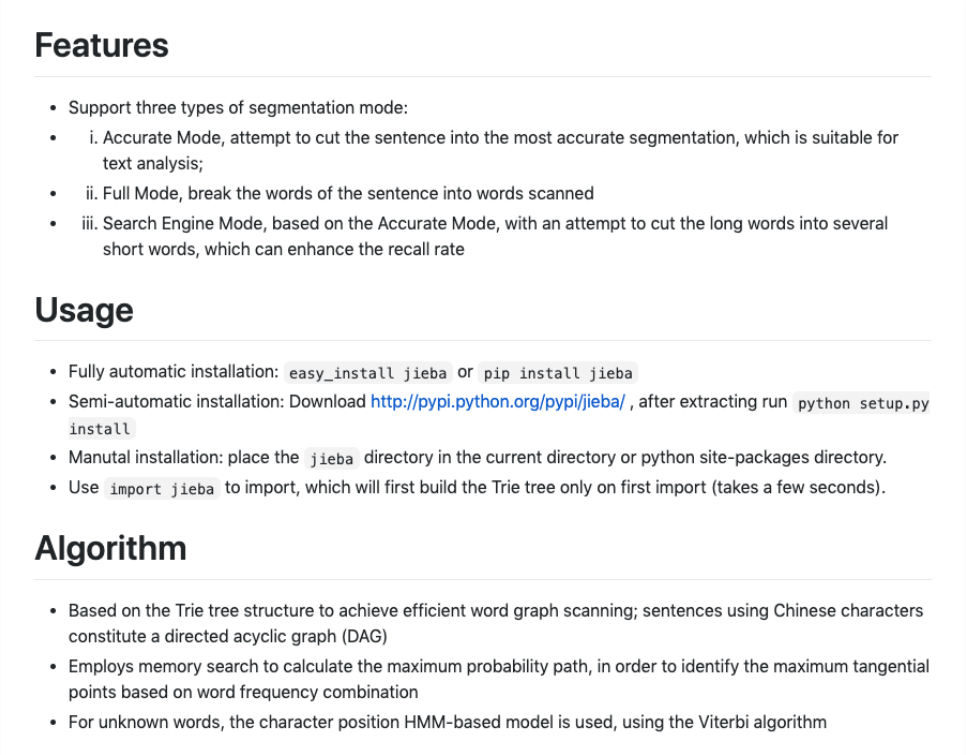
rest use machine learning based methodology to do the information extraction. The table below listed five frameworks or tools on clinical information extraction, which were the most cited in publications from 2009 to 2016.

Table 2.1
Top 5 information extraction frameworks/tools included in publications from 2009-16.

Name	**Description**	**No. of Papers**
UIMA	software framework for analysis of unstructured contents	31
cTAKES	Open-source NLP system based on UIMA framework	26
MetaMap	National Institutes of Health developed NLP tool	12
MedLEE	NLP systems for narrative clinical notes	10
GATE	Java-based open-source software	5

There are many English language processing frameworks and tools such as NLTK, Stanfordnlp, and CoreNLP. However, similar tools based on Chinese language are relatively too little to find any research on utilizing these tools on EHR. Chinese sentence syntax and word characteristic components are quite different from English grammar, so a Chinese language processing tool is required in this project.

"Jieba" (Chinese for "to stutter") [16] is one of Chinese text segmentation tools, which is built to be the best Python Chinese word segmentation module. It implemented almost all the functions similar to other English language processing frameworks and tools. Keyword Extraction, Part of Speech Tagging, Tokenize are the main functions which used in this project. It supports three types of segmentation mode, Accurate Mode, Full Mode and Search Engine Mode. For unrecognizable words, a HMM-based model is used with the Viterbi algorithm.



Features

- Support three types of segmentation mode:
- i. Accurate Mode, attempt to cut the sentence into the most accurate segmentation, which is suitable for text analysis;
- ii. Full Mode, break the words of the sentence into words scanned
- iii. Search Engine Mode, based on the Accurate Mode, with an attempt to cut the long words into several short words, which can enhance the recall rate

Usage

- Fully automatic installation: `easy_install jieba` or `pip install jieba`
- Semi-automatic installation: Download http://pypi.python.org/pypi/jieba/ , after extracting run `python setup.py install`
- Manutal installation: place the `jieba` directory in the current directory or python site-packages directory.
- Use `import jieba` to import, which will first build the Trie tree only on first import (takes a few seconds).

Algorithm

- Based on the Trie tree structure to achieve efficient word graph scanning; sentences using Chinese characters constitute a directed acyclic graph (DAG)
- Employs memory search to calculate the maximum probability path, in order to identify the maximum tangential points based on word frequency combination
- For unknown words, the character position HMM-based model is used, using the Viterbi algorithm

Figure 2.4: Features and Usage Documentation for Jieba.

2.3 Overview of Voice Assistant for EHR

Several voice recognition and language processing systems had been developed which only supported English. Alexa voice assistant of Amazon [17], Saykara [18] and Suki [19] are famous and mature applications of AI assistant for doctors on either mobile platform or computer platform. Unfortunately, there has been no similar research or application in Chinese hospitals and clinics except IFLYTEK CO.LTD. [14] started implementing a voice EHR system in 2017.

2.3.1 Iflytek EHR

System of Iflytek [14] is designed to synchronously records the voice into patients' medical record, when doctors communicate with the patient. Iflytek EHR system will explore feasible approach and method to implement voice recognition technology as well as NLP

on automatically filling keywords into EHR to improve the efficiency of doctor's diagnose based on Chinese. This is quite similar to the aim and object of this project. However, there is only description of such system on the Chinese official website of Iflytek and Iflytek even did not mention such a system on English version of website. In this case, this project regards it as an unfinished system, which is still under development. Conclusively, there does not exist any developed AI-based EHR systems in China.

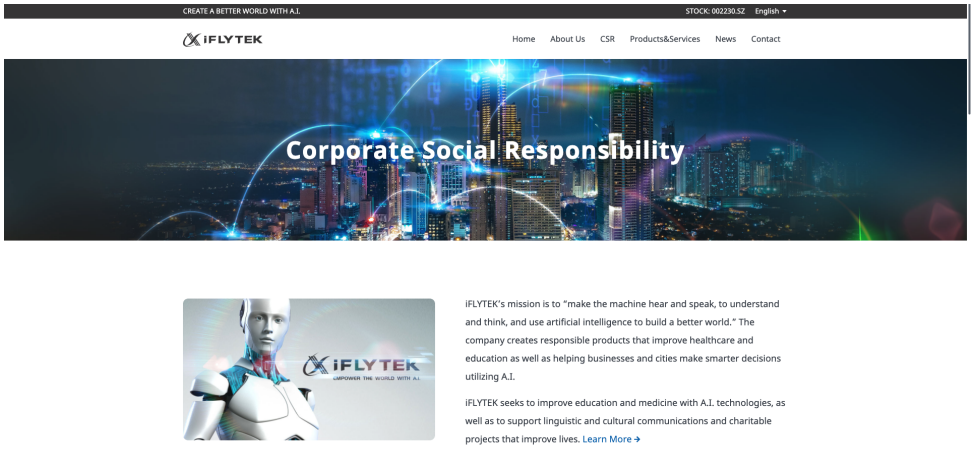


Figure 2.5: Iflytek Main Page

Chapter 3

Project Specification

This section states the main requirements for the project including functional and non-functional requirements. Functional requirements define basic system behaviour, while non-functional requirements show constraints or restrictions on the design of the system.

3.1 Functional Requirement

1. The application can be started and used by doctors.
2. The application must be able to record real time speech.
3. The application must include option to load audio files from local computer.
4. The application must be able to convert voice inputs into text outputs.
5. The application must be able to analyze raw text and fill the corresponding information into EHR form.
6. The application must be available for users to manually modify the EHR form.
7. The application should include option to save the generated EHR form to local directory.
8. The application can be closed and ended by doctors.

3.2 Non-Functional Requirement

3.2.1 Usability

The user interface should be user-friendly, which means it should be easy to use. The buttons and text fields should be clear and readable in a neat format. Any user should understand the meaning of each button or text field that are used in the application without any training. Efficiency and satisfaction are two main categories to measure the usability to figure out whether the user can achieve their goals quickly and is pleasant to the UI design. Beyond that, error rate of any user operations should be under 5%.

3.2.2 Performance

The application should load within 5 seconds to run the program. After audio recording or audio loading, the application should convert speech to text within 5 seconds, process text within 5 seconds and predict disease within another 5 seconds. Maximum of 15 seconds are acceptable to give the final result in EHR form. There should be no frozen scenes or stuck moments and there should be a status bar to inform users the current status of application.

3.2.3 Operating Environment

The application should work on computer-based environment with python installed. Microphone or speaker is required if users want to use real-time speech as input.

3.2.4 Maintainability

The application should be able to recover from critical failures in a few minutes. Any run time error should be recorded in a log file for further maintenance. In addition, warning

or error message should pop up.

3.3 Software Specification

This chapter presents the software Specification of this project. Python is a very versatile programming language that can be used across a variety of different fields, including user interface design, number of available API as well as many mature AI methods. In this case, Python is selected to use in this project. Because the application has not been encapsulated now, PyQt5 is also required to run this application. In addition, several libraries are also required. After completing the development, it is possible to encapsulate the application so that there will be no need for users to configure the environment and install those libraries.

Table 3.1
Software Specification.

Detail	**Requirement**
Platform	Desktop-based
Operating System	Any operating system with Python installed
Configure of Environment	Python3, PyQt5
Libraries	SpeechRecognition 3.8.1, jieba 0.42.1, requests 2.24.0, json-minify 0.3.0 , bs4 0.0.1, OpenCV 3.4.2.16, keras 1.0.8, tensorflow 1.13.1, openpyxl 3.0.5
Language	Chinese

Chapter 4

Methods

In this chapter, system design and all the methods implemented in this system will be presented. In system design, the whole activity flow of this system will be listed and more detailed information for each part will be introduced in the following parts. For each method, there will be a working flow diagram and pseudo-codes. All the pseudo-codes will be included in Appendix.

4.1 System Design

As mentioned in Chapter 1.3 - aims and objectives, main tasks in this project was to utilize all the required methods, including data collection, voice recognition and Chinese natural language processing. Apart from that, user interface was required for a better demonstration as long as it was clear for users to understand the meanings of each part.

Apart from the data collection, the whole activity flow is shown in Figure 4.1. The input of the application was an audio file or real-time speech recording on symptom and treatment description using voice record device. Then, voice recognition method was used to recognize audio or speech into plain text. After that, NLP method was used to separate the words from the sentence. Then, trigger words were detected to filter all the

information into corresponding cell of the EHR form. Finally, users had the option to save the EHR form to local csv files.

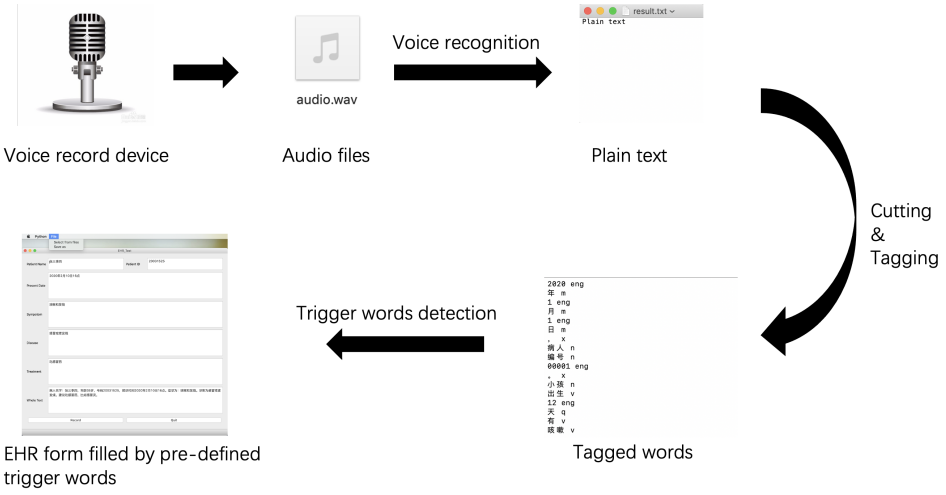


Figure 4.1: Activity Flow Diagram.

4.1.1 Dataset Collection

There was no public clinical diagnosis dataset available for Chinese language, so the best way to prove the feasibility of this system was to collect its own dataset and use these data to test on the whole system. All the data sources came from an online doctor diagnose website (https://www.haodf.com/). On this website, diseases were classified into different classes within different departments in the hospital. Patients could choose the specific disease classification to ask questions in it. Patients typed in the description of their symptom based on their own observation and doctors could ask further questions to diagnose the disease for the patient. The data in the website was in both text-based and voice-based.

To save the time for data collection, a crawler algorithm was implemented to grab useful information from the websites. As shown in Figure 4.2, after setting the request headers and URLs, bs4 library could be used to analyze the HTML file to extract desired information according to HTML tags. The output of this algorithm was an excel file with

disease categories, which was used as a label dictionary for model training. Pseudo code was attached below.

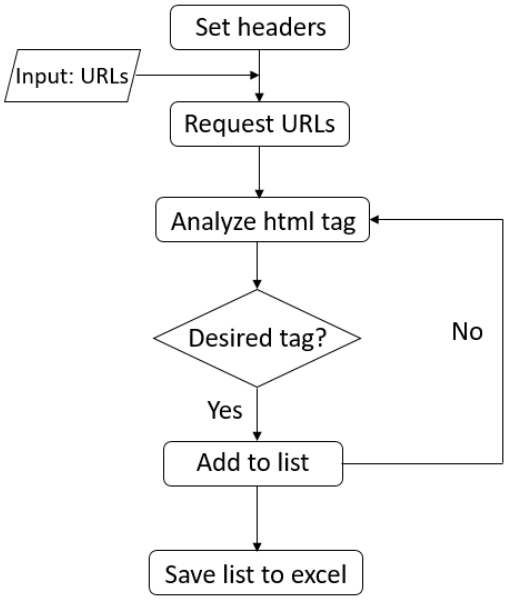


Figure 4.2: Crawler Algorithm Working Flow.

Algorithm 1 Crawler Algorithm.

Input: Useful URLs for grabbing $URLs$;

Output: Excel files with different disease categories;

```
1: Set headers of URLs;
2: for each URL in URL list do
3:     request.get(URL,headers);
4:     transform into utf-8 and save to local file;
5: end for
6: for each html file do
7:     if class of <div> tag ==“ct” then
8:         for each <div> do
9:             if class of <div> tag with class ==“m_little_green” then
10:                main_category_list.append(m_little_green.span.text);
11:            end if
12:        end for
```

```
    end if
    if class of <div> tag ==“m_ctt_green” then
        for each <div> tag do
            if class of <li> tag with class ==“topli” then
                sub_category_list.append(li.a.text);
            end if
        end for
    end if
end for
excel = xl.load_workbook(path);
for each category in main_category_list do
    Remove duplicate categories;
    Write into excel file;
end for
for each sub-category in sub_category_list do
    Remove duplicate categories;
    Write into excel file;
end for
excel.save(path);
```

This dataset was used as labels for model training, testing and validation. The structure of this dataset was designed as demonstrated in Figure 4.3. One table for each main category. Inside this main category, there were several sub-categories, which was labeled as one alphabet combined with one integer number. Then, there were detailed disease name, here called “sub-sub-category”, with further labels, such as A1-1. In this dataset, there were 29 main categories in total and more than 200 sub-categories. Almost all the disease types that classified by a normal hospital were mentioned in this dataset.

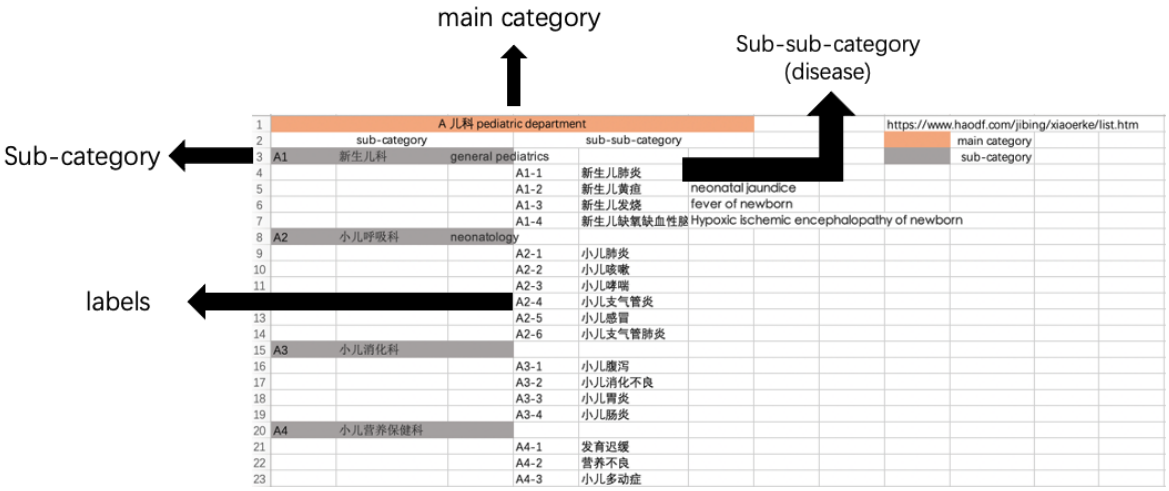


Figure 4.3: Sample Dataset.

4.1.2 Data Pre-processing

Next step was to collect doctor's diagnose based on the symptom of patients. After collecting the raw data, data pre-processing was required to manually clean the data and label all the cases to make the data follow the classification in order to do the following training or testing. Figure 4.4 shows the samples of manually labeled cases. First column was unique case number. Second column was symptom description while third column added doctors' diagnose. Rest columns were labels linked to first dataset. "Symptom description" was from the observation and description of disease from the patients. Doctors would repeat these symptoms and combine them with their diagnosis together to be the "Doctor's diagnosis". Key information included in the diagnose should contain patient name, patient ID, patient present date, symptom, disease type and current treatment. This information was later extracted to fill into the EHR forms. These doctor's diagnosis were recorded as voice inputs.

num	symptom description from patient/病人症状描述	time & date &symptom description from doctor & diagnose/时间+ID+医生症状复述+诊断意见	sub-category	disease
KS0001	出生12天有咳嗽，拍片说是肺炎，医院要求住新生儿科打抗生素，住院一周出院继续有咳嗽，医生让服用进口阿奇霉素，服用2天宝宝晚上哭闹不睡，肚子拉稀。还是继续咳嗽，每天5声左右。在住院期间给宝宝吃了进口阿奇霉素5天 然后停了几天 出院宝宝还咳嗽 又让我们继续喂5天 现在已经喂3天了。医院让重新拍了个片对比，比第一次阴影稍微吸收了一点。然后说如果咳嗽就让再次住院。	2020年1月1日，病人编号00001。小孩出生12天有咳嗽，拍片说是肺炎，医院要求住新生儿科打抗生素，住院一周出院继续有咳嗽，医生让服用进口阿奇霉素，服用2天患者晚上哭闹不睡，肚子拉稀。还是继续咳嗽，每天5声左右。在住院期间患者吃了进口阿奇霉素5天 然后停了几天 出院宝宝还咳嗽 又让继续喂5天 现在已经喂3天了。医院让重新拍了个片对比，比第一次阴影稍微吸收了一点。然后说如果咳嗽就让再次住院。有口服的阿莫西林克拉维酸钾，商品名是得酶或者棒林，可以买来继续给孩子使用3天左右。同时可给孩子吃些止咳化痰的中成药，例如：小儿咳嗽糖浆、小儿止咳糖浆等等。如果5天后情况仍无好转，建议去复诊，或者去更上级的医院复诊。	A1	A1-1
KS0002	一个月前偶尔喘长气，没有咳嗽，没有喘，后来好了，近一个星期又出现这种情况，总喘长气没有其他症状，他这个季节有些过敏性鼻炎，但不太严重，近期也没有鼻塞，心肌酶正常，然后孩子最近喘不过气这种情况的同时，鼻子是通气的状态没有鼻塞的症状。还做了下面的这两个检查，想知道这种情况喘长气是什么原因导致的呢，跟鼻炎有关系吗，还是属于哮喘呢，还是介于两者之间的什么原因，很担心，迫切希望得到您的建议。想知道是什么原因导致的孩子喘不过气这种症状。	2020年1月1日，病人编号00002。病人一个月前偶尔喘长气，没有咳嗽，没有喘，后来好了，近一个星期又出现这种情况，总喘长气没有其他症状，他这个季节有些过敏性鼻炎，但不太严重，近期也没有鼻塞，心肌酶正常。然后孩子最近喘不过气这种情况的同时，鼻子是通气的状态没有鼻塞的症状。气短要注意喘的情况，顺尔宁需要吃三周。	A2	A2-3
KS0003	主任您好！我们宝宝37周+1出生，一直纯奶粉喂养，每天6次，每次140ml左右，之前每天一次大便。从4个月开始一直吃lifespace益生菌。上周六开始突然一天大便两次，有奶瓣和粘液，周日拉了5次。周一白天正常，本以为好了。可是晚上开始到今天差不多一天5次左右，一喂奶没一会就放屁大便。每次不是很多，这两天我们自己给她吧奶浓度稀释了，基本上120ml的水才给1勺奶粉，喂奶频次增加。但还是拉奶瓣。想问问主任需要如何处理？今天已经第五天了。	2020年1月1日，病人编号00003。宝宝37周+1出生，一直纯奶粉喂养，每天6次，每次140ml左右，之前每天一次大便。从4个月开始一直吃lifespace益生菌。上周六开始突然一天大便两次，有奶瓣和粘液，周日拉了5次。周一白天正常，可是晚上开始到今天差不多一天5次左右，一喂奶没一会就放屁大便。每次不是很多，这两天我们自己给她吧奶浓度稀释了，基本上120ml的水才给1勺奶粉，喂奶频次增加。但还是拉奶瓣。可能是乳糖不耐受，可以吃蒙脱粉和复方胃蛋白酶，奶粉暂时停吃。	A3	A3-1
KS0004	第一次抽动症发病时间是2020年3月，症状是频繁眨眼。无用药。5月下旬左右痊愈。大半个月前再次发病。	2020年1月1日，病人编号00004。第一次抽动症发病时间是2020年3月，症状是频繁眨眼。无用药。5月下旬左右痊愈。大半个月前再次发病。建议规律治疗，包括药物治疗，心理干预，定期复查，避免各种诱发因素，预防发作。可以看专科医院，专科门诊，找专科医生，一般我们这看精神科，也可以看中医。	A4	A4-3
KS0005	出生40多天抽搐，诊断为癫痫，一直服左乙拉西坦，后来加重，去西安儿童医院住院，添加奥卡西平口服液，控制情况较好，后来复发，去检查，脑电图结果初步诊断为婴儿痉挛症，然后停止吃奥卡西平。后续一直治疗，最后一次住院去西安儿童医院打ACTH，目前在家口服泼尼松，妥泰，丙戊酸，开浦兰。回家后较以前发作较为减轻，但每天还有抽搐！发作时孩子伴有哭声。每天约两次，一次五六下！为进一步治疗，特请主任帮忙看一下！万分感谢！ 以上为最后一次住院的记录和检查项目 附出生后四次脑电图报告！ 既往史:50多天，查出有巨细胞病毒感染肝炎！！后来治愈 出院时: 左乙拉浓度:19.06ug/ml 丙戊酸浓度:69.72ug/ml 肝功肾功等检查一切正常 现在: 丙戊酸浓度:59.22ug/ml 肝功肾功等检查一切正常 8:14日，妥泰加到下午1片又四分之一	2020年1月1日，病人编号00005。出生40多天抽搐，诊断为癫痫，一直服左乙拉西坦，后来加重，去西安儿童医院住院，添加奥卡西平口服液，控制情况较好，后来复发，去检查，脑电图结果初步诊断为婴儿痉挛症，然后停止吃奥卡西平。后续一直治疗，最后一次住院去西安儿童医院打ACTH，目前在家口服泼尼松，妥泰，丙戊酸，开浦兰。回家后较以前发作较为减轻，但每天还有抽搐！发作时孩子伴有哭声。每天约两次，一次五六下！既往史:50多天，查出有巨细胞病毒感染肝炎！！后来治愈。出院时: 左乙拉浓度:19.06ug/ml，丙戊酸浓度:69.72ug/ml，肝功肾功等检查一切正常。现在: 丙戊酸浓度:59.22ug/ml，肝功肾功等检查一切正常。8/14日，妥泰加到下午1片又四分之一。目前口服强的松，赛宝宁目标剂量一片/2次。（先从1/4片二次，一周后2/4片二次，5/8片二次，6/8片二次，7/8片二次，一片二次，其它药物不变。	A5	A5-1
KS0006	孩子有一周了，玩着玩着说哭就哭特别是晚上睡觉之前总是哭闹，昨天去中医院做的检查说是心肌炎	2020年1月1日，病人编号00006。孩子有一周了，玩着玩着说哭就哭特别是晚上睡觉之前总是哭闹，昨天去中医院做的检查说是心肌炎。建议做心脏彩超+心功能，心电图，动态心电图。可以再查一下甲状腺功能，肝功生化，EB病毒，巨细胞病毒，柯萨奇病毒等。如果都正常，可以口服果糖二磷酸钠口服液，一次一支，一天两次。	A6	A6-1

No. Symptom description Doctor' s diagnose labels

Figure 4.4: Labeled cases in Dataset.

4.1.3 Voice Recognition

The first step was to implement voice recognition methods. Two voice recognition methods were implemented in this project: Baidu voice recognition method and Google voice recognition method. Both of them could recognize speech in Chinese and have API for Python. Baidu voice recognition method required an access key to pull requests from the Baidu AI server while Google voice recognition simply required importing a library and invoking in-build functions.

4.1.3.1 Baidu Voice Recognition

For Baidu voice recognition, a request with client ID and private key to the Baidu AI server was done first to do authentication. Headers, languages and other information were required to pull this request. After that, the server received an audio file in the format of wav as input. If the speech in this file was recognized successfully, the speech would be transformed into plain text and the result would be included in return value. Otherwise,

it would return an error representative code.

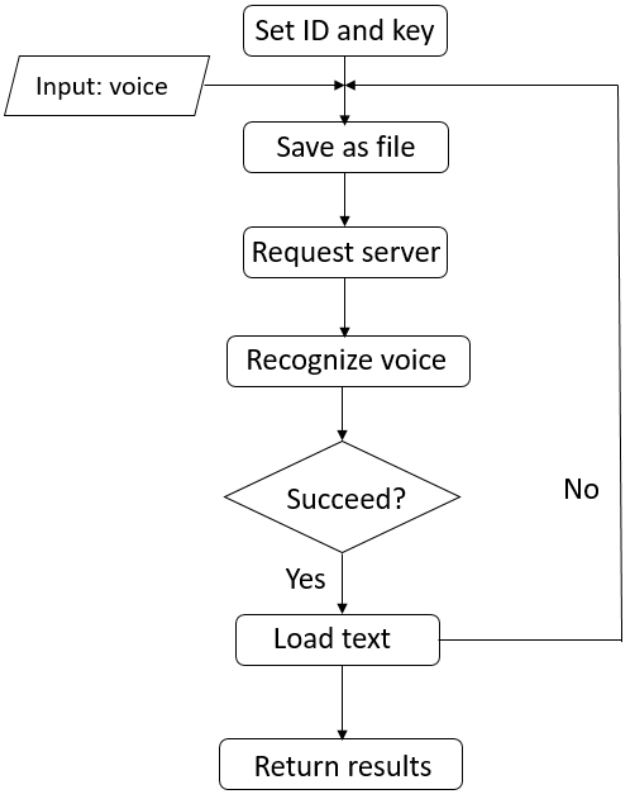


Figure 4.5: Baidu voice recognition Working Flow.

Algorithm 2 Baidu Voice Recognition Method.

Input: Audio file or real-time voice;

Output: Recognized plain text;

1: headers ⟵ header settings;

2: clientID ⟵ client ID; clientKey ⟵ client key;

3: req = request(server,audio_file,header);

4: result = json.load(req);

5: **if** result[“err_msg”] == ”success.” **then**

6: print(result[“result”]);

7: **end if**

8: **return** 0;

4.1.3.2 Google Voice Recognition

For Google voice recognition, a library called "speech_recognition" was provided. By initialize a recognizer, a function called "recognize_google" can be invoked with two parameters. First one was the audio file and the second one was target language. If recognized successfully, it would return the recognized text. Otherwise, it would return either an unknown error or a request error. Since it did not require any settings or authentication, Google voice recognition API for python was far more precise and simpler than Baidu's.

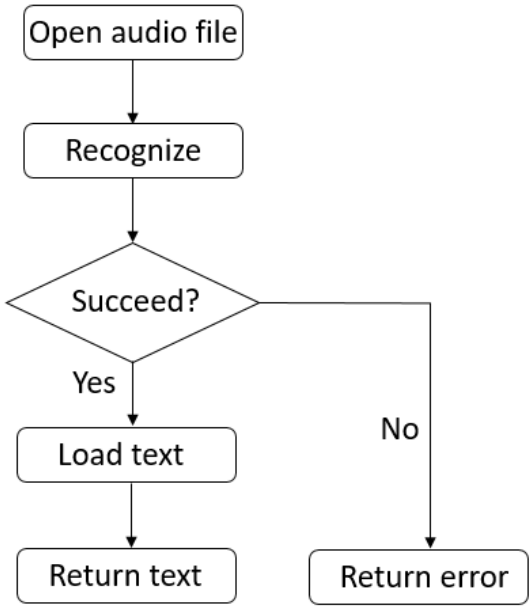


Figure 4.6: Google voice recognition Working Flow.

Algorithm 3 Google Voice Recognition Method.

Input: Audio file or real-time voice;

Output: Recognized plain text;

```
r = speech_recognizer();
if speech_recognition.error == null then
    print(r.recognize_google(audio,language));
end if
return speech_recognition.error;
```

4.1.4 NLP - Information Extraction

Since there wasn't much choices for Chinese language processing methods and Jieba library contains all the functions that were needed, Jieba library was used to do the information extraction in this project. Inside the Jieba library, there were two kinds of Part of Speech Tagging Function offered. The first one was default mode which allowd developers to modify dictionary by their own and the other one was pre-trained mode called "paddle mode" with abundant lexicon. The idea was to combine these two kinds of mode to obtain a better information extraction. Incorporate the more accurate result of "paddle mode" with the definition of unfamiliar terminology with dictionary of default mode, it was believed that the result will be more accurate, which can avoid doctors to proofread contents or modify mistakes.

Algorithm 4 Part of Speech Tagging in Jieba.

```
//Tags the POS of each word after segmentation, using labels compatible with ictclas.
//Example:
import jieba.posseg as pseg
words = pseg.cut("我爱北京天安门")
for w in words do
    print('%s %s' % (w.word, w.flag))
end for
OUTPUT:
— 我 - r
— 爱 - v
— 北京 - ns
— 天安门 - ns
```

In this project, both Part of Speech Tagging Functions were used. The normal mode allowed users to add their own dictionary for further training. The paddle mode was a mature pre-trained model. These two different mode generated results in different formats.

After that, the results were stored as lists for further comparison. Then, trigger words were used to detect the key information. If trigger words were found, the system would collect the following word phrases until meet the next trigger words or full stop. Word phrases recorded for the same trigger word from two different model will be compared. Any same part between two models would become the final result and be filled into the corresponding cell in EHR forms by intersection. If there was no same part, the system leaved the cell blank. Doctors could manually extract the information from the whole paragraph of recognized text or manually fill in the cells. After the form was filled, doctors could save the file to local directory.

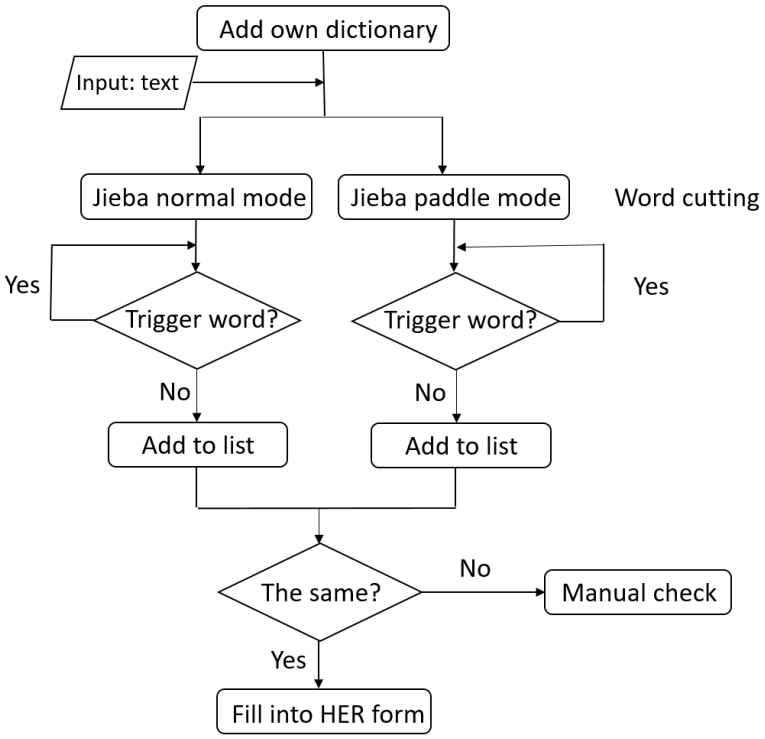


Figure 4.7: Jieba Language Process Working Flow.

Algorithm 5 Jieba Language Process Method.

Input: Plain text;

Output: Part of speech - separated word phrases;

```
    //first approach with normal mode
1:  normal_words_list = jieba.posseg.cut(text,HMM=False);
2:  for word in normal_words_list do
3:      if word is trigger words then
4:          key_info_list ⟵ the following information until next trigger word;
5:      end if
6:  end for
    //second approach with paddle mode
7:  jieba.enable_paddle();
8:  paddlemode_words_list = jieba.posseg.cut(text,use_paddle=True);
9:  for word in paddlemode_words_list do
10:     if word is trigger words then
11:         paddle_key_info_list ⟵ the following information until next trigger word;
12:     end if
13: end for
    //compare the result of two approaches and record the duplicated parts
14: for info in key_info_list do
15:     if info == paddle_key_info_list.info then
16:         result[“info”] = info;
17:         fill the corresponding cell in EHR form
18:     end if
19: end for
```

Trigger words are listed in Table 4.1. All the trigger words were coming from Chinese synonym dictionary. Using trigger words was the simplest way to separate desired information into corresponding parts. To use this system, doctors should first mention the trigger words and then speak the related information. By given synonym words, there still was free rein for doctors in case of any habit phrases.

Table 4.1
Trigger words.

Trigger words	In Chinese
Name	姓名，名字
Age	年龄，年纪
Id	编号，号码，id
Date	日期，时间
Symptom	诊断，疾病
Treatment	建议，方法，治疗，意见

4.1.5 Save EHR Form to Local Directory

After the EHR form was filled in, doctors could save the form to a local csv file. After selecting the directory and naming the file, all the information was structured in order. With this method, the form was saved for future use such as subsequent visit. A better way to prevent the privacy leakage was to save the file to cloud, which was more secure.

Algorithm 6 Save File to Local Directory.

Input: path, file_name;

Output: Csv file storing EHR form;

```
directory = QFileDialog.getSaveFileName(path, file_name);
if directory != null then
    header = ['Property','Information'];
    rows ⟵ records;
end if
open(file,'w') as f:
writer = csv.writer(f);
writer.writerows(header);
```

```
9: writer.writerows(rows);
```

4.2 UI Design and Implementation

Figure 4.9 is an example of EHR form. In this project, a user interface was designed and implemented shown as Figure 4.10. General and basic medical information such as patient names, IDs, dates, symptoms, disease, treatments were included. Also, a container for the whole recognized content was reserved for any possible manual error check. Record button and quit button were on the main page to allow doctors record real-time speech to fill into the forms and quit the system. Apart from that, in the file option, it allowed doctors to use a local audio file as input and save the form into local csv files.

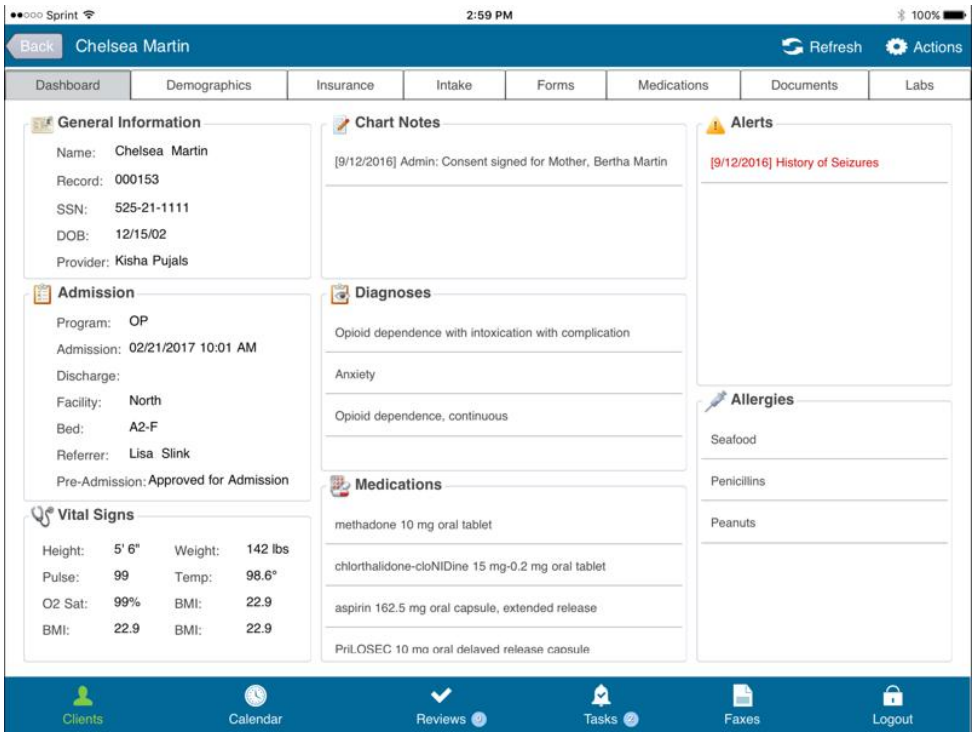


Figure 4.8: Example UI for EHR.

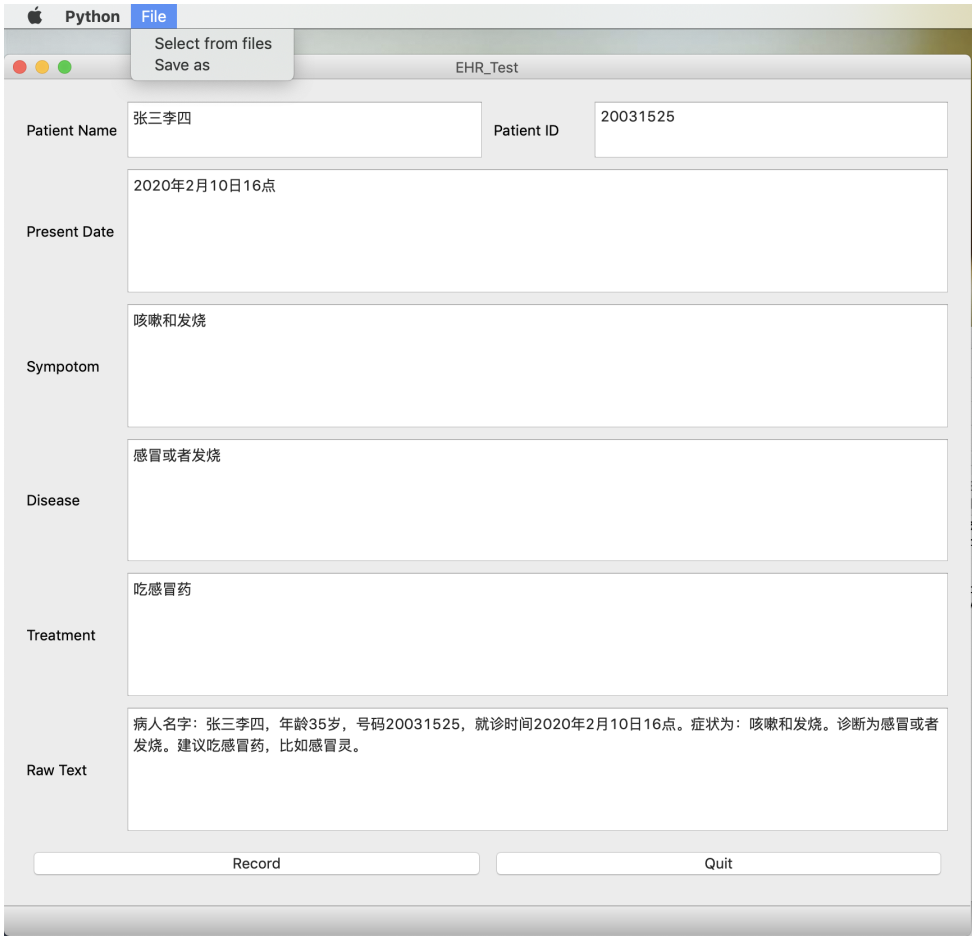


Figure 4.9: User Interface of the Application.

4.3 Additional Design

After finishing all the original aims and objects, additional work was designed and implemented. A deep learning network was required to predict the possible disease base on the symptom description. The input of the network was a sentence, which was a sequence of words. In this case, Long Short-Term Memory (LSTM) was recommended to train the model [20]. With maximum length defined, the part exceeded this length was ignored and the sentence less than maximum length was padded with blank space. After that, the model was set up before training. Dense, dropout, activation function, loss function, optimizer, batch size and epoch number were the main parameter to tune for the accuracy of the training. After training, the accuracy and loss were shown in a tendency chart.

Algorithm 7 LSTM.

Input: train_X, train_y, val_X, val_y;

Output: Prediction model;

```
language ⟵ Chinese / English;
//initialize a model with maximum length
model = Model(max_len);
//translate sentences into embeddedness
data = sst_binary(path);
x,y,w = model.transform(data);
//model setting
model = Sequential();
model.add(Embedding);
model.add(LSTM(128));
model.add(Dropout);
model.add(Dense);
model.add(activation_function);
model.compile(loss_function,optimizer,metrics);
//train and evaluate
model.fit();
score = model.evaluate();
plt.plot(epoch,score)
```

Chapter 5

Result and Discussion

5.1 Result

In this chapter, the result of main methods will be presented. The first section is the accuracy comparison between Baidu voice recognition method and Google voice recognition method. It demonstrates why Baidu voice recognition method is preferred in this project. The second section will be the accuracy for NLP, which is considered as the correctness of filling the EHR forms. And the last section will be the preliminary result of LSTM.

5.1.1 Accuracy for voice recognition methods

Table 5.1 demonstrated the result of Baidu voice recognition method and Google voice recognition method for both Chinese and English language speech. It could be clearly observed that Baidu voice recognition method showed better result on mandarin and Google voice recognition method showed better result on English. In the case of mixture of mandarin and English, Baidu voice recognition method could still recognize most of the contents in Chinese mode. The English part could be correctly recognized while Google voice recognition method had great probability to recognize it as a Chinese word. In

addition, Baidu voice recognition method could correctly recognize most of the medical terminologies from the speech without adding any dictionary. If any word in one sentence was recognized incorrectly, the sentence would be regard as incorrect. Notice that punctuation marks were ignored.

Table 5.1
Comparison between Baidu and Google voice recognition methods.

Description	Mode	Baidu voice recognition	Google voice recognition
All mandarin	Chinese	80.0%	76.3%
All English	English	86.5%	89.1%
Mixture of mandarin and English	Chinese	81.3%	70.8%
mandarin with terminology	Chinese	80.4%	63.2%

The results was calculated by average percentage of difflib, Levenshtein and fuzzywuzzy comparison methods and kept one decimal places. Although Baidu voice recognition had usage limitation on the number of times, its processing time was relatively faster than Google AI server. Baidu voice recognition method offered better performance for mandarin recognition as well as the quicker response. After considering the advantage and disadvantage of both voice recognition methods, Baidu voice recognition method was selected.

5.1.2 Accuracy for NLP

After the implementation NLP methods in two different modes, the results were counted as whether the cells in the form was filled in with correct information. Any blank cell was regard as False-Negative, any cell contains relevant information was regard as True-Positive and any cell contained irrelevant information was regard as False-Positive. For incorrect results, doctors had the option to manually edit the form to solve the problem. The first mode in Jieba NLP is normal mode, which contained punctuation mark after

separating the part of speeches, while the paddle mode had blurred boundary between sentences. So, the accuracy of normal mode was much higher than the paddle mode. Finally, the results between these two mode were compared and the intersection of these two results was left and filled in the EHR form.

Twenty cases were test to get the accuracy of NLP method, which meant 120 cells should be filled in accordingly. After testing, it was found that normal mode could extract almost all the information correctly while paddle mode had 15% incorrect cases. Most of the incorrect cases was that the key information belong to one cell was not identified because of the unclear boundary between sentences. The final result was gained as the same parts of these two modes, these incorrect cases happened in paddle mode would also lead to the wrong result in final result. The key idea to use the same parts of these two modes as result was because the extracted information were filtered to remove some of the irrelevant information.

Table 5.2
Comparison between different NLP modes.

Mode	**result(percentage)**
Normal Mode	119/120(99.2%)
Paddle Mode	102/120(85.0%)
Comination	102/120(85.0%)

5.2 Additional Result

5.2.1 Preliminary result of LSTM

After finishing the original aims and objectives, additional work was included to use LSTM to predict the disease type based on the symptom description. Because that there was not any public symptom description dataset in China, the dataset used in this project was manually selected from different websites. Also, symptom descriptions for common

disease from different patient had a great probability to be almost the same. In this situation, data collection was time-consuming.

Finally, 100 samples were selected for 5 different diseases. With this relatively small dataset, the accuracy was nearly 20.1%, which was quite low and need further improvement on both quantity of dataset and quality of algorithms. The current size of dataset was insufficiently to train the network at this stage. Possible improvement approach would be included in future work.

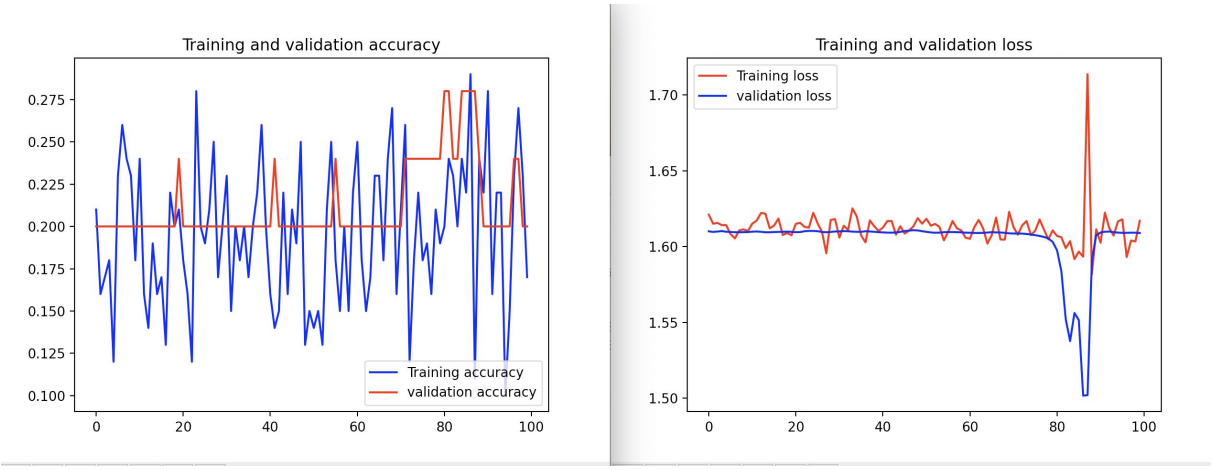


Figure 5.1: Accuracy and Loss of LSTM network.

5.3 Discussion

Baidu voice recognition method showed better result on mandarin and mixture of mandarin and English. Baidu voice recognition method could still recognize most of the contents if some of the English terminology was involved in Chinese speech. These two cases were the usage scenario that this project was targeted to, so Baidu voice recognition method was used in this automatic EHR generation system.

The average accuracy of voice recognition method was not high enough due to several reasons other than recognition error: a slip of the tongue, accent, noises, and the speed of speech. If the speakers could speak with no oral mistakes in a proper speed, together with a noise reduction algorithm, the result was believed to be much better. As shown

in Figure 5.2 and Figure 5.3, the result of speech input with a slip of the tongue and oral mistakes was not accurate enough and some blanks were filled in with wrong information. For instance, patient ID in Figure 5.3 was incorrect. In this case, users should manually check and correct the form. In general, current implemented automatic EHR generation system was served as support mechanism, which could help doctors to some extent and still need improvements.

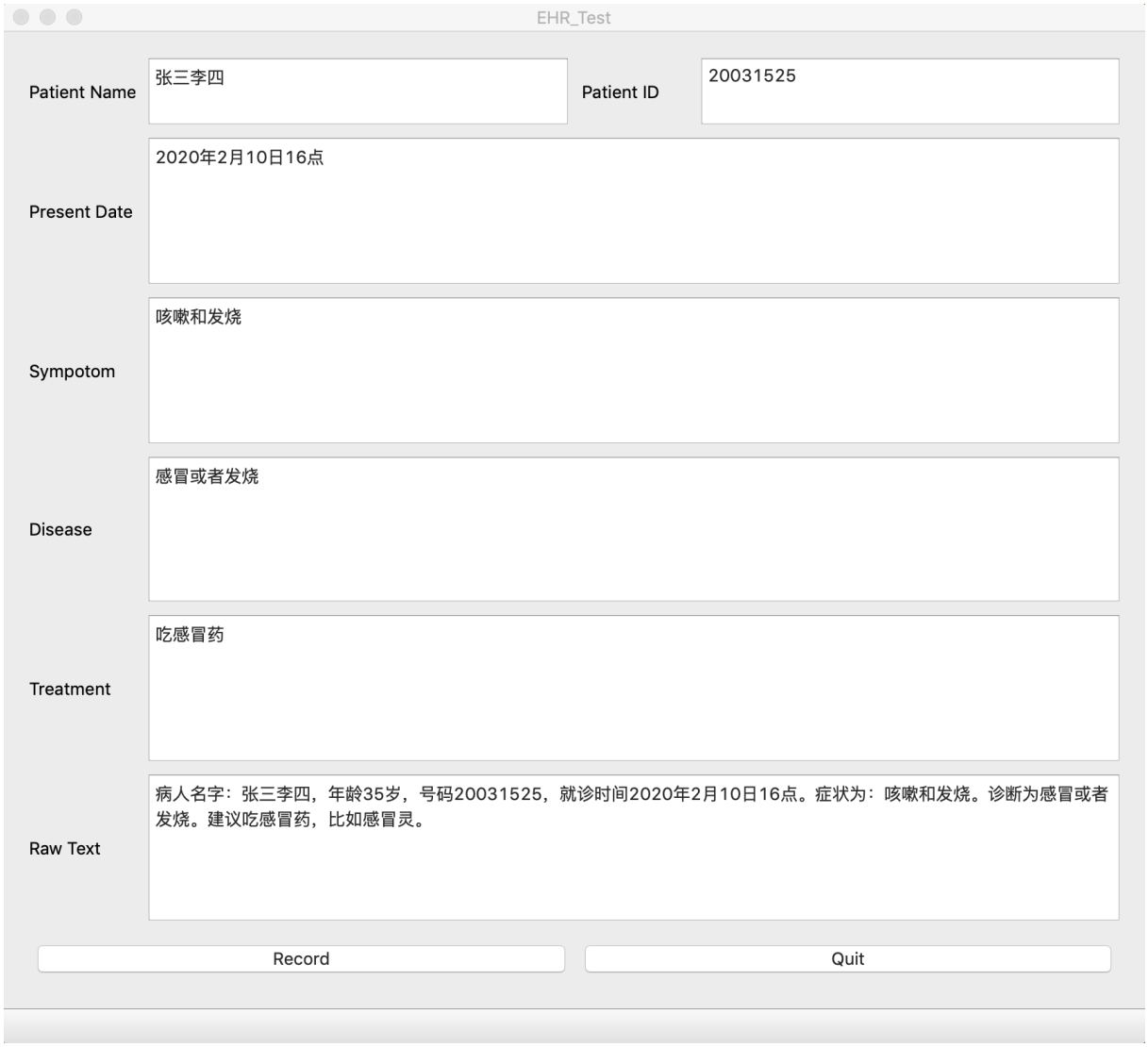


Figure 5.2: Result of Perfect Speech Input.

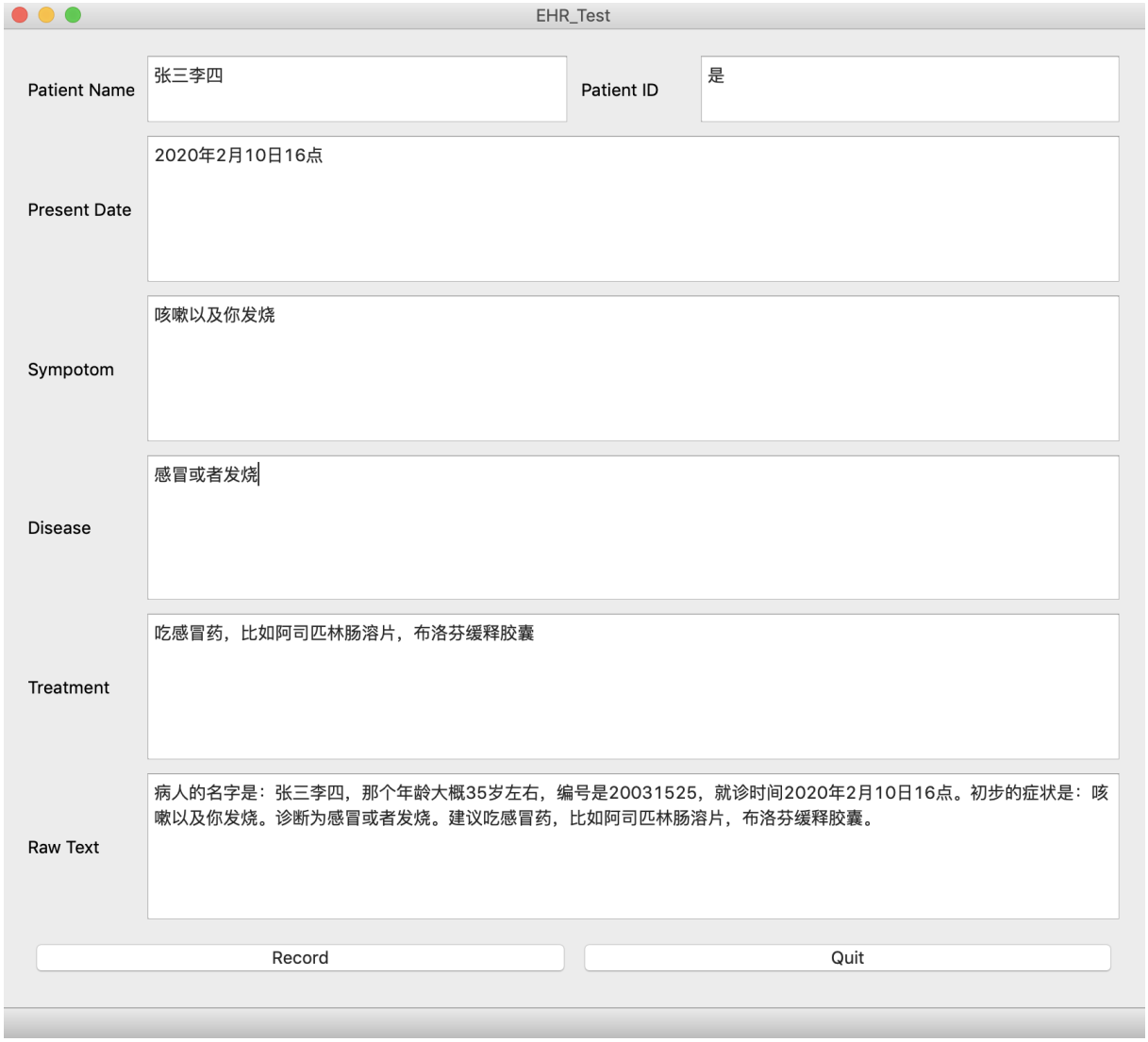


Figure 5.3: Result of Speech Input with noise.

In the NLP method, the final result was gained as the same parts of the results from two modes, those incorrect cases happened in paddle mode would also lead to the wrong result in final result. Although the final result might be affected by unstable performance of paddle mode, the paddle mode offered better recognition on terminologies such as disease names and drug names. So, the result would be better if consider both of the normal mode and paddle mode. The decision was made between intersection or union set of two modes. As union set would keep all the information in both modes and it might cause problems when combine two results. For instance, A is abcd, B is bfde, A and B have

the same part b and d. Then combination can be abcdef, abfcde, abfcfe... In this case, intersection was better to remain the key information. Doctors could also manually check the final result to ensure the correctness.

Chapter 6

Conclusion

6.1 Conclusion

This project aimed to explore the feasibility of creating a system capable of translating real time voice input or audio files into plain text and process and extract the key information to fill in the EHR automatically. By using this system, it could support doctors and improve the efficiency and quality on processing clerical works. Doctors did not need to manually fill in the EHRs and could concentrate on diagnosing patients. In this system, Baidu voice recognition method was used to transform real-time speech or audio file to plain text because of its high accuracy on recognizing Chinese and jieba language processing method was used to extract the key information from plain text as well as fill these information into the EHR form. The accuracy of Baidu voice recognition was higher than 80% and the accuracy of final result provided by the system was over 85%. To further save the time for doctors manually modify the wrong result, possible improvement of accuracy would be focused on in the future work.

6.2 Future work

6.2.1 Improvement on Current System

1. For the voice recognition methods, a noise reduction method can be implemented to process the speech before recognition. It may make contribution to a better result of the voice recognition.
2. The challenge is that not everyone can speak fluently, any slip of the tongue, accent, noises, and the speed of speech could affect the accuracy. Methods other than trigger word based extract can be tested.
3. The final content filled into the form is simply the intersection of result from two different NLP methods. A better design can be proposed to avoid the final result is affected by failure in one of the methods.

6.2.2 Extension of the Current System

1. More symptom description dataset should be collected to train the neural network.
2. LSTM network need further parameter tuning on learning rate, optimizer, batch size and number of epoches to achieve an accurate prediction on possible diseases. Doctors can refer to this prediction and avoid mis-diagnose.

Chapter 7

Reflection

7.1 Project Management

This project was divided into four stages: Stage 1. Planning and Feasibility Analysis, Stage 2. Project Design, Stage 3. Project Implementation and Integration and Stage 4. Test and Maintenance. As shown in Aims & Objects and Related Work, Stage 1 has been completed at the beginning of this project. After that, Stage 2 was done before any substantial implementation to have an overview of the project. Then, Stage 3 was the most important part of the project because all the methods in Project Design were implemented to achieve the goal of this project. Finally, Stage 4 were produced to check the correctness of codes, examine the usability of the system and receive the result of different methods. Improvement and additional work were also implemented in Stage 4.

The new Gantt chart in Figure 7.2 demonstrated the actual project plan using Waterfall methodology. The sequence and time period were decided according to the importance levels and risk management. Some of the tasks were implemented in parallel. Different from the original timetable in Figure 7.1, most of the research part was in the first half of the first semester and most of the heavy work as challenges in completing this FYP were changed to complete in the second half of the first semester. Then, in the second

semester, tests and improvements were provided to obtain a better performance of the whole system. After achieving all the basic requirements, extended research and work have been focused on in week 20 - week 24. In other words, the actual rate of progress was faster than the original plan, so additional work is involved in the project.

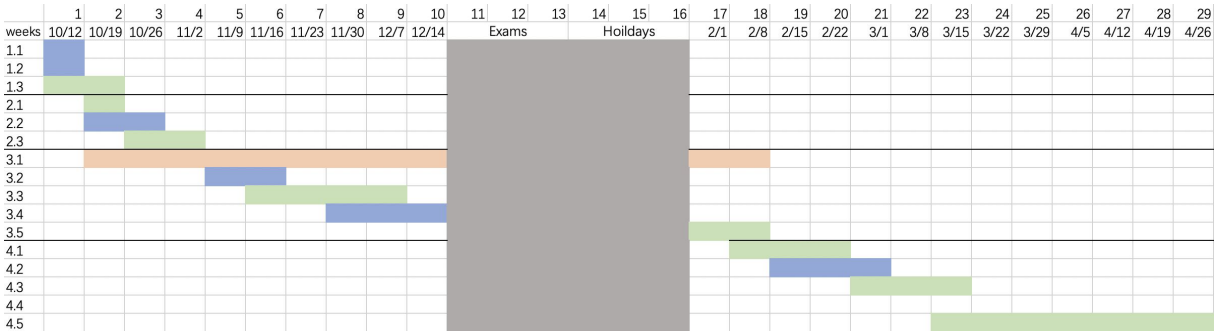


Figure 7.1: Original Timetable.

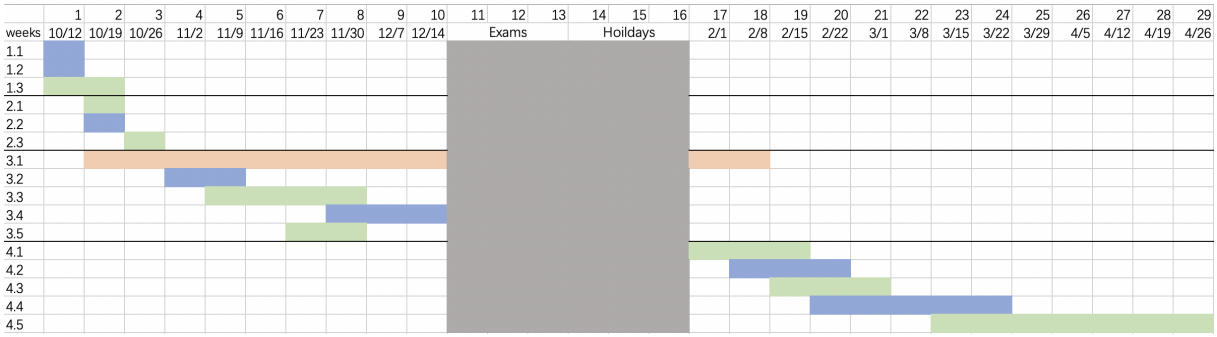


Figure 7.2: New Timetable.

Stage 1 Planning and Feasibility Analysis

1.1 Ethics form signed and approval.

1.2 Requirements identification and specification of this project.

1.3 Articles reading and accomplishment of proposal.

Stage 2 Project Design

2.1 Decision of the format and content of database.

2.2 Further articles and websites reading to find useful voice recognition and nlp libraries or API.

2.3 User interface design for the software.

Stage 3 Data Collection and Implementation

3.1 Collection of doctor diagnose prescription and/or conversation contents online for creation of audio type dataset.

3.2 Development of user interface of EHR form with all basic functions provided.

3.3 Implementation and combination of speech recording, voice recognition, and Chinese language processing.

*3.4 Interim report writing and validation.

3.5 Use trigger words to extract key information from the result of Chinese language processing.

Stage 4 Test and Maintenance

4.1 Test on speech recording, voice recognition, and Chinese language processing functions.

4.2 Test on the performance of keyword classification.

4.3 Test on the application interaction.

4.4 Implementation of LSTM to predict the diseases.

*4.4 Final report writing and validation.

7.2 Contribution

More than twenty articles have been read before starting the implementation. Three databases were designed and collected in this project. The first one was the detailed

disease category dataset used as labels in future model training. The second one was doctor diagnose description used to test the system whether it can extract key information from doctor's speech and fill into the EHR form. Doctor diagnose description included patient name, patient ID, diagnose date, symptom description, treatment and disease. The third dataset was symptom description dataset used to train the network to predict diseases. Apart from dataset collection, several methods were implemented, including speech recording, Baidu voice recognition, Google voice recogniton and Chinese language processing in both normal and paddle mode. Also, a user interface has been created to give a brief demonstration of the results from voice recognition, and Chinese language processing methods. After completing all these parts, test were done and an LSTM model was trained to predict the diseases.

7.3 Reflections

In this project, Python was used to develop the whole system and lots of libraries in Python were imported. A lot of learning sources were look on the internet to implement all the methods. However, there still remain some trouble to control the version of libraries to solve any possible conflict. For instance, tensorflow and keras have strict version requirements to work together. Tensorflow version 1.13.1 and keras version 2.2.4 were selected at last to satisfy the requirements. Otherwise, there would be conflicts and compile errors. Also, Pyqt5 was used to implement the user interface. It took some time to get familiar with it and divide the coding pattern into MVC(Model-View-Control) pattern. It is quite useful to following MVC pattern because the UI needed to be improved and changed overtime. In this way, changing the model of the UI will not affect the view and functional part.

During the development, Gitlab was used to control the version of the system and record the progresses for each week. The speed of the development was faster than expected and the development subsequence was stick to the plan and had additional work involved, in which the working flow is designed and implemented clearly. User manuals were written

in README.md under each directory to demonstrate the library used in the code, the usage of the demo code and references of learning sources when writing the code. Meeting minutes were also included to record process and To-Do list for every week. Meetings for every week were extremely useful for me to check the progress and stick to the plan.
After implemented the methods, these methods were bound with the user interface. The user interface was user-friendly and could be used by users without any user manual. Lots of time was spent to collect the user cases to test on the system and gave a basic accuracy of all the methods. Because medical records were private information, it was really harder to find these information on the internet than expected. So, the additional work was not quite successful because the limitation on the amount of data samples. Apart from that, all the progress was under control.

In this project, Baidu voice recognition method and Google voice recognition method were implemented and tested successfully. Jieba NLP method was implemented for Chinese language process and standford NLP method was implemented for English language process. As this project aimed to target to the field of Chinese EHR system, Jieba was used in this system. To calculate the accuracy, three types of sentence matching algorithms were used: difflib, levenshtein and fuzzywuzzy. Also, to reduce the noise in voice recognition part, 5 complex noise reduction algorithms were tried but failed. At last, a simple noise reduction method was invoked from SpeechRecognizer library to achieve a basic noise reduction requirement. Apart from that, web grabbing algorithm was implemented to collect dataset automatically from website to save the time. Finally, a lstm model was trained but not quite successful due to the insufficient dataset samples. In gernal, all the effort tried were related to the project and contributed to the system development. Personally speaking, the final accomplishment achieved the desired effect.

References

[1] T. Shu, H. Liu, F. R. Goss, W. Yang, L. Zhou, D. W. Bates, and M. Liang, "Ehr adoption across china's tertiary hospitals: A cross-sectional observational study," *International Journal of Medical Informatics*, vol. 83, no. 2, pp. 113 – 121, 2014.

[2] Y. A. Kumah-Crystal, C. J. Pirtle, H. M. Whyte, E. S. Goode, S. H. Anders, and C. U. Lehmann, "Electronic health record interactions through voice: a review," *Applied clinical informatics*, vol. 9, no. 3, p. 541, 2018.

[3] J. Owusu-Marfo, Z. Lulin, H. A. Antwi, and M. O. Antwi, "Electronic health records adoption in china's hospitals: A narrative review," 2019.

[4] J. King, V. Patel, E. W. Jamoom, and M. F. Furukawa, "Clinical benefits of electronic health record use: national findings," *Health services research*, vol. 49, no. 1pt2, pp. 392–404, 2014.

[5] T. D. Shanafelt, L. N. Dyrbye, C. Sinsky, O. Hasan, D. Satele, J. Sloan, and C. P. West, "Relationship between clerical burden and characteristics of the electronic environment with physician burnout and professional satisfaction," in *Mayo Clinic Proceedings*, vol. 91, no. 7. Elsevier, 2016, pp. 836–848.

[6] G. Li, Y. Qian, Y. Huang, J. Chen, and X. Bai, "Building electronic health record using voice recognition and big data techniques," in *2nd Symposium on Health and Education 2019 (SOHE 2019)*. Atlantis Press, 2019.

[7] J. He, S. Baxter, J. Xu, J. Xu, X. Zhou, and K. Zhang, "The practical implementation of artificial intelligence technologies in medicine," *Nature Medicine*, vol. 25, 01 2019.

[8] E. Topol, "High-performance medicine: the convergence of human and artificial intelligence," *Nature Medicine*, vol. 25, pp. 44–56, 2019.

[9] S. Doyle, "Determining voice recognition accuracy in a voice recognition system," Feb. 23 2010, uS Patent 7,668,710.

[10] PyPI.org, "Google voice recognition api," https://pypi.org/project/SpeechRecognition/, accessed Dec 10, 2020.

[11] Baidu.com, "Baidu voice recognition api," https://ai.baidu.com/tech/speech, accessed Dec 10, 2020.

[12] W. Che, Z. Li, and T. Liu, "Ltp: A chinese language technology platform," in *Coling 2010: Demonstrations*, 2010, pp. 13–16.

[13] X. Chen, Z. Liu, L. Wei, J. Yan, T. Hao, and R. Ding, "A comparative quantitative study of utilizing artificial intelligence on electronic health records in the usa and china during 2008–2017," *BMC Medical Informatics and Decision Making*, vol. 18, 12 2018.

[14] Iflytek.com, "Iflytek voice ehr system," https://www.iflytek.com/health/zlwyl, accessed Dec 10, 2020.

[15] Y. Wang, L. Wang, M. Rastegar-Mojarad, S. Moon, F. Shen, N. Afzal, S. Liu, Y. Zeng, S. Mehrabi, S. Sohn, and H. Liu, "Clinical information extraction applications: A literature review," *Journal of biomedical informatics*, vol. 77, pp. 34–49, 2018.

[16] fxsjy, "Jieba python api," https://github.com/fxsjy/jieba, accessed Dec 10, 2020.

[17] Amazon.com, "Alexa voice assistant of amazon," https://www.digitaltrends.com/home/what-is-amazons-alexa-and-what-can-it-do/#:~:text=While%20Alexa%

20is%20the%20official%20name%20for%20Amazon\OT1\textquoterights,to%20be%20Alexa%20or%20something%20that%20sounds%20similar, accessed Dec 10, 2020.

[18] Saykara.com, "Saykara mobile ai assistant for physicians," https://www.saykara.com, accessed Dec 10, 2020.

[19] Suki.ai, "Suki ai-powered, voice-enabled digital assistant for doctors," https://www.suki.ai, accessed Dec 10, 2020.

[20] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

Appendices

Appendix A

Research Ethics

A.1 Ethics Form

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

School of Computer Science
Preliminary UG & PGT Research Ethics Checklist

This form is required for all UG and Taught MSc research projects to determine whether the proposed research requires ethical review. The module convener or project supervisor is responsible for exercising appropriate professional judgement when completing this form.

SECTION I. Applicant Details

1. Name	Yiming Li
2. Status	**Undergraduate Student**
3. Email address	scyyl3@nottingham.edu.cn

SECTION II. Module Details

4. Module name/number or MA/MSc/MPhil course and department	CSAI
5. Supervisor's name	BOON GIIN LEE
6. Supervisor's email address	boon-giin.lee@nottingham.edu.cn

SECTION III. Project Details

Project title	**Data: From Patient to Health Record**
Proposed start date	**2020/09/21**

Please answer each question by ticking the appropriate box:		**Yes**	**No**
1.	Does the study involve human participants?	√	
2.	Does the study involve personal data, and/or include the processing of data in the public domain in such a way as to invade the privacy of the individuals concerned? (see Moodle for guidance on what constitutes personal data.)		√
3.	Does the study involve the use of biological materials?		√

I confirm that the information provided is an accurate description of the proposed project and I understand that if any elements of the project should change, I will need to submit a revised form.

Signature of Applicant: ……Yiming Li……..

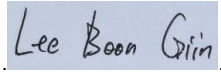
SECTION IV. For completion by the Supervisor

- The answer to all questions in Section III is **NO** and no further ethics approval is required ☐

OR

- The answer to at least one question in Section III is **YES** and a **UG & PGT Research Ethics Checklist** (available from link **here**) must be submitted to the Faculty Office for further clearance. ☒

Submit a scan of the completed and signed form to Moodle either by the student or supervisor.

Signed: ……  …… [Supervisor]

Name: ……Boon Giin Lee…….. Date: ……2020/9/25……

Appendix B

Pseudo code

B.1 Crawler Algorithm for dataset collection.

Algorithm 8 Crawler Algorithm.

Input: Useful URLs for grabbing $URLs$;

Output: Excel files with different disease categories;

```
1: Set headers of URLs;
2: for each URL in URL list do
3:     request.get(URL,headers);
4:     transform into utf-8 and save to local file;
5: end for
6: for each html file do
7:     if class of <div> tag ==“ct” then
8:         for each <div> do
9:             if class of <div> tag with class ==“m_little_green” then
10:                main_category_list.append(m_little_green.span.text);
11:            end if
12:        end for
13:    end if
```

```
14:     if class of <div> tag ==“m_ctt_green” then
15:         for each <div> do
16:             if class of <li> tag with class ==“topli” then
17:                 sub_category_list.append(li.a.text);
18:             end if
19:         end for
20:     end if
21: end for
22: excel = xl.load_workbook(path);
23: for each category in main_category_list do
24:     Remove duplicate categories;
25:     Write into excel file;
26: end for
27: for each sub-category in sub_category_list do
28:     Remove duplicate categories;
29:     Write into excel file;
30: end for
31: excel.save(path);
```

B.2 Baidu Voice Recognition Method.

Algorithm 9 Baidu Voice Recognition Method.

Input: Audio file or real-time voice;

Output: Recognized plain text;

```
1: headers ⟵ header settings;
2: clientID ⟵ client ID; clientKey ⟵ client key;
3: req = request(server,audio_file,header);
4: result = json.load(req);
5: if result[“err_msg”] == ”success.” then
6:     print(result[“result”]);
7: end if
8: return 0;
```

B.3 Google Voice Recognition Method.

Algorithm 10 Google Voice Recognition Method.

Input: Audio file or real-time voice;

Output: Recognized plain text;

```
1: r = speech_recognizer();
2: if speech_recognition.error == null then
3:     print(r.recognize_google(audio,language));
4: end if
5: return speech_recognition.error;
```

B.4 Jieba Language Process Method.

Algorithm 11 Jieba Language Process Method.

Input: Plain text;

Output: Part of speech - separated word phrases;

```
    //first approach with normal mode
1:  normal_words_list = jieba.posseg.cut(text,HMM=False);
2:  for word in normal_words_list do
3:      if word is trigger words then
4:          key_info_list ⟵ the following information until next trigger word;
5:      end if
6:  end for
    //second approach with paddle mode
7:  jieba.enable_paddle();
8:  paddlemode_words_list = jieba.posseg.cut(text,use_paddle=True);
9:  for word in paddlemode_words_list do
10:     if word is trigger words then
11:         paddle_key_info_list ⟵ the following information until next trigger word;
12:     end if
13: end for
    //compare the result of two approaches and record the duplicated parts
14: for info in key_info_list do
15:     if info == paddle_key_info_list.info then
16:         result[“info”] = info;
17:         fill the corresponding cell in EHR form
18:     end if
19: end for
```

B.5 Save CSV File to Local Directory.

Algorithm 12 Save File to Local Directory.

Input: path, file_name;

Output: Csv file storing EHR form;

```
directory = QFileDialog.getSaveFileName(path, file_name);
if directory != null then
    header = ['Property','Information'];
    rows ⟵ records;
end if
open(file,'w') as f:
writer = csv.writer(f);
writer.writerows(header);
writer.writerows(rows);
```

B.6 LSTM for disease prediciton.

Algorithm 13 LSTM.

Input: train_X, train_y, val_X, val_y;

Output: Prediction model;

```
language ⟵ Chinese / English;
//initialize a model with maximum length
model = Model(max_len);
//translate sentences into embeddedness
data = sst_binary(path);
x,y,w = model.transform(data);
//model setting
model = Sequential();
model.add(Embedding);
model.add(LSTM(128));
model.add(Dropout);
model.add(Dense);
model.add(activation_function);
model.compile(loss_function,optimizer,metrics);
//train and evaluate
model.fit();
score = model.evaluate();
plt.plot(epoch,score)
```