# THE DEVELOPMENT OF VOICE-ASSISTED CHATBOT FOR HEALTHCARE INSTITUTIONS USING TRANSFORMERS-BASED TECHNIQUES.

By

Lee Zong Hao

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS)
INFORMATION SYSTEMS ENGINEERING
Faculty of Information and Communication Technology
(Kampar Campus)

FEBRUARY 2025

# ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Ms. Nurul Syafidah binti Jamil who has given me this bright opportunity to engage in a development of voice-assisted chatbot for healthcare institutions using transformers-based techniques. It is my first step to establish a career in AI engineering and development field. A million thanks to you.

To a very special person in my study journey, Samantha Lok Tian Wen, for her inspiration, unconditional support, and for standing by my side during my exploration on this AI sector. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

# COPYRIGHT STATEMENT

# ABSTRACT

This research focuses on developing a voice-assisted chatbot tailored for Traditional Chinese Medicine (TCM), utilizing advanced transformer-based AI models and generative techniques. The chatbot aims to address accessibility challenges in healthcare services, particularly for elderly, disabled, or literacy-challenged individuals. By enabling voice input and output, it ensures inclusivity and broader access to essential healthcare information.

The chatbot's core features include speech recognition and Natural Language Processing (NLP), allowing it to understand various dialects and accents, a critical need in multicultural regions like Malaysia. Leveraging large language models, it provides human-like responses and personalized recommendations based on TCM principles, including herbal remedies, dietary advice, and lifestyle suggestions. It is designed to function effectively even in noisy environments and to understand accented English, ensuring accurate communication across diverse linguistic backgrounds.

Additionally, the chatbot bridges the gap in TCM-specific medical knowledge by using specialized datasets to deliver detailed insights into treatments and principles. It serves as an educational resource for patients and practitioners, continually improving its responses through dynamic learning from user interactions.

This voice-assisted TCM chatbot significantly enhances healthcare workflows by reducing the burden on medical professionals, automating routine consultations, and providing 24/7 access to medical advice. It is especially beneficial for users in remote areas, offering timely and accurate information. By personalizing care and empowering users to manage their health proactively, this project represents a major step forward in integrating AI into healthcare, creating an inclusive and patient-centric system.

Area of Study: Chatbot development, Voice Data Integration, Speech-To-Text, Healthcare Application, Accessibility and Inclusion

Keywords: Bilingual Textual Data, Natural Language Processing, Transformer-based Model

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| *TCM* | Traditional Chinese Medicine |
| *LLM* | Large Language Model |
| *LORA* | Low-Rank Adaptation of Lage Language model |
| *NLP* | Natural Language Pro |

# CHAPTER 1

## Introduction

In recent years, the integration of artificial intelligence (AI) in various sector has revolutionized the way services are delivered and experienced. Among these advancements, chatbots have emerged as powerful tools, providing users with instant, automated assistance across different domains, from customer service to personal assistants [1]. As healthcare institutions strive to enhance patient care and streamline operations, the potential of AI-driven chatbots becomes increasingly evident [2].

A significant breakthrough in the field of AI is the creation of generative AI, which incorporate models capable of generating new content, such as text, images, or even sound, based on the data they have been trained on. Generative AI leverages complex neural networks, particularly those based on transformers to understand and generate human-like text with high accuracy and coherence. This advancement has brought up new possibilities in automating complex tasks that requires a deep understanding of human languages [3].

In the field of healthcare sector, the need for efficient, accurate and responsive communication channels is paramount. Traditional methods often fall short in addressing the growing demand for real-time information and support, particularly in scenarios requiring quick responses or when handling large volume of patient queries [4]. During the COVID-19 pandemic, healthcare systems globally were overwhelmed by the sudden spike in patient queries, revealing the inefficiencies of traditional methods. Hospitals struggled to manage the surge in demand, resulting in delays and errors in response times [5].

Transformers-based techniques is a part of generative AI, offering a promising solution to be challenges faced by healthcare institution [6]. This technique enables chatbots to understand and generate human-like text with high accuracy, making them ideal for complex and [7]. By implementing voice-assisted AI chatbot, it can provide a 24/7 accessibility for immediate response to user queries [8]. This is very helpful for those who lives in remote areas as they can inquire information before travelling to the hospital. With the power of machine learning, AI chatbot learns how to express wordings and information in a logical, conversational way, avoiding using bombastic scientific terms which most people do not understand. Voice-assisted AI chatbot has help revolutionized how medical industry normally works [9]. It offers more inclusive to everyone especially to those who are disabled and older age person who is unable to type, spell, read or not prefer typing for prompt. Whether it is making appointments or asking questions, the chatbot serves as a convenient and supportive resources for a diverse range of individuals seeking medical assistance.

## 1.1 Problem Statement and Motivation

- **Traditional AI Chatbot Lacks the Option for Input and Output**

For traditional AI chatbot, most of them are text-based chatbot which means they only takes text as prompt and gives output in texts only. This method has brought up some problem for some categories of people for example, disabled people who does not have arms, loss of fingers, blind or those who suffers from dyslexia will not be able to type or to form words, this has made them loss the opportunity to use this new technology. Furthermore, senior citizen also faced problems when using the text-based chatbot as well. Some of the senior citizen who suffers from vision problem, Parkinson's disease, or just the lack of education will also have the problem in typing out texts or forming sentences properly [18] [19].

- **Problem in Identifying and Converting Accented English**

Malaysia is a multi-cultured country with 3 main races which is Malays, Chinese, and Indian. Each race has their own language and accents and with multiple variants of accented English, the usual model will have a hard time to properly predict the correct words. Current challenge for most speech recognition models is lack of training on different accented English. This will lead to miscommunication or misunderstanding when user trying to convey their problems and causes confusion or even mistreatment [18] [19]. In short, the models must have the ability to understand every word user clearly in order to make sure there will not be any misguidance or misunderstanding.

- **Lack of Chatbot Knowledge on Traditional Chinese Medicine**

One of the biggest problems in current medical chatbots is the lacking knowledge on Traditional Chinese medicine (TCM) Most of the existing datasets that are available is only related to Western medication and treatment. This resulted in the medical chatbot only have limited knowledge to understand the TCM terminology, principles and medications. What is more concerning is that it may provide inaccurate and unreliable information to the user which would cause catastrophic disaster as TCM concepts defers from Western medication concepts[18].

- **Capturing Voice Frequency and Phonetic Clearly in Various Situation**

Current voice recognition models still have problem of clearly capture and identifying the voice of human especially when there is background noise or user has different accent. Furthermore, the current model does not learn from user's interaction which means they are static and will be looking for certain accent or certain volume to trigger the voice recognition system. Not only voice but how user express their sentences, most voice chatbot could not function without being input certain keywords, which means they are unable to understand the nuances of natural language or the semantics[19].

## 1.2 Motivation

During the COVID- 19 pandemic has revealed significant weakness in traditional healthcare communication methods, as hospitals were overwhelmed with patient queries and unable to provide timely, accurate responses. This highlighted the inefficiencies of existing systems and the need for more robust and responsive solutions [10].

While AI chatbot have been implemented to help manage patient interactions, these systems are predominantly text-based, limiting accessibility for certain populations, including elderly and the disabled, who may struggle with typing or reading text-based outputs. Additionally, in multicultural regions like Malaysia, the diversity of accents poses a significant challenge to existing speech recognition technologies, leading to frequent misunderstandings and potential errors in patient care [11].

Furthermore, most current AI-driven chatbots lack integration with Traditional Chinese Medicine (TCM), a critical gap given the widespread use of TCM in many parts of the world, including Malaysia. This absence of TCM knowledge in AI system could lead to incomplete or incorrect information being provided to patients, potentially endangering their health [12]. Figure 1.1 shows that people are starting to take notice on TCM and the trend of visiting TCM is increasing by time for over a decade.

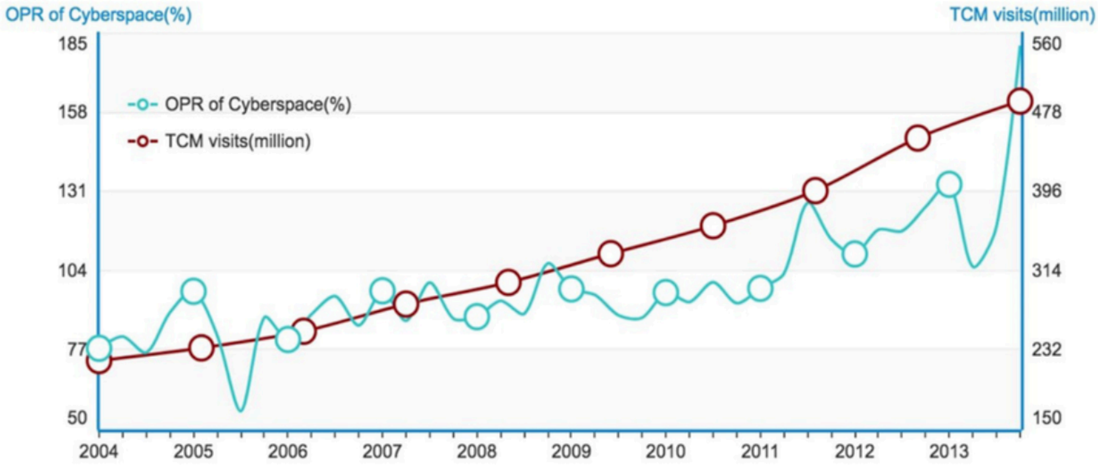



**Figure 1.1 OPR of Cyberspace vs TCM visits** [13]

By addressing these challenges, the research seeks to improve patient care, enhance communication efficiency, and ensure that healthcare services are inclusive and culturally sensitive.

## 1.3 Research Objectives

The main objective of this project is to develop a voice-assisted TCM chatbot for the general hospital website. The chatbot will provide reliable and accurate information related to Traditional Chinese Medicine (TCM) to the user through the use of speech which is voice data. The sub-objectives to accomplish the main objective are as follows:

- **Sub-Objective 1: To perform data preparation on both textual and voice data of Traditional Chinese Medicine**
  During the data preparation phase, text data is collected from Hugging face repository while voice data will be collected from various source including Hugging face, Github, or Kaggle. Text data in the datasets consists of 2 features which is "Query" and "Response", each query is a question about TCM and rules of what to response while each response is the answer to the query following the required rule. The data is in Chinese language and does not require

any translation as the pre-trained LLM model has the capability to understand Chinese language. However, the dataset requires a structure transformation as the current structure does not meet the requirement to be feed to the model for fine-tuning. Since the model needs to be trained in a conversational way and only about TCM, the datasets must be in a conversational method and talks only about TCM in which this text datasets suits the requirements. Voice data will undergo preprocessing to meet the requirement of the speech-to-text model and text-to-speech model. Speech-to-text model will have to be able to recognize voice and accurately output the text in various scenario, including loud background noise scenario. Text-to-speech model will have to properly pronouns every word of the text accurately to provide the most accurate information to the user. Speech-to-Text model will have to undergo accent fine-tune as well as in Malaysia consists of various ethnicity and all of them have unique English accents, hence it is important that the model will have to understand the accent to translate voice into the correct text.

- **Sub-Objective 2: To train a transformer-based model for voice data recognition and natural language understanding.**

  This project will be utilizing ChatGLM as the base LLM model for natural language understanding and generating output. The model will then be fine-tuned using the transformed datasets and LORA configuration settings. LORA configuration consists of various parameters which can then be explore and modify to achieve the best result with the lowest resource needed. The Speech-to-text and Text-to-Speech module will be utilizing open-source models to achieve further fine-tuning and have a more custom preference. Preprocessed voice data with mixture of background noise and various accent will be fed into the model for fine tuning.

- **Sub-Objective 3: To evaluate the performance of the chatbot in terms of losses, accuracy, response time, and output relevancy using both text and voice inputs.**

  Performance of the model used will be evaluated using various metrics. For LLM model, the performance evaluation is made during the fine-tuning phase where data is being validated and tested using validation set and testing sets.

The metrics used for the LLM model is losses, estimated time, and the output relevancy. For speech-to-text model will be evaluated using accuracy, which utilize the validation set and test set of the voice datasets, fine-tuning estimated time to ensure the resource used is optimize. Speech-to-Text model will be evaluated manually to ensure the text and speech is tally.

- **Sub-Objective 4: To integrate the trained chatbot into a web platform for seamless accessibility and usability in healthcare institutions.**
  After all the trained models has achieved optimal phase, it will be integrated into a web platform to be tested. Simple user-interface will be made to ensure user have a better experience using the chatbot.

- **Enhance the accessibility from text only to speech and text**
  This chatbot enables user to input through speech and response through both text and speech, which enables more people to be able to use the chatbot. With the use of voice input, miscommunication and misunderstanding will be greatly decrease as people will be able to convey more through speech than text.
- **Personalized user interaction**
  This chatbot will be using NLP technique which will be able to understand users' queries and preferences. The chatbot will be able to understand user's queries and provides accurate personalized recommendations. It can also provide treatment plans, relevant articles and other knowledge related to TCM tailored to users' needs.

## 1.4 Project Scope and Direction

This project will involve the development of a voice-assisted chatbot to provide users with information and nonclinical recommendations related to Traditional Chinese Medicine, including herbal remedies, acupuncture and dietary advise.

The key features of the project include:

I. The chatbot will allow user to use speech as input and output
II. The chatbot will be able to generate summarized response to user
III. The chatbot will response with natural language

IV. The chatbot allow users to retrieve information about Traditional Chinese Medicine
V. The chatbot will be able to provide nonclinical suggestions for user
VI. The scope of the data is text and voice data
VII. Language used for the model is English and Mandarin
VIII. Method used to train and finetune model is LORA, ChatGLM4-9b-chat

## 1.5 Contributions

The contribution is twofold, where the first is language model of text and voice for English and mandarin while the second is the website integration of the model. A voice-assisted TCM chatbot that accommodates users with disabilities, elderly populations, and individuals who may struggle with text-based interfaces. By providing a voice input/output mechanism, the chatbot **enhanced the accessibility** for a broader range of users, including those with physical impairments, vision problems or literacy challenges. The voice-assisted AI chatbot also revolutionize the traditional ways of consultation approach, patient no longer have to wait for long queue to enquire information or to make appointment, **increasing the workflow efficiency** Moreover, it also adapts **effective communication**, patient no longer have to use certain words or prompt to get the information they needed, with the implementation of NLP, natural language can easily be understand by the chatbot, making communication easier and less stressful.

Furthermore, this voice-assisted medical chatbot also serves as a knowledge banks for anyone who wish to learn more about traditional Chinese medication for **education purposes**. This voice assisted AI chatbot has been made through analyzing and learning previous data about traditional Chinese medication and hence it has the ability to offer detailed insights into TCM treatments, principles and practices, providing a more holistic understanding of healthcare. **Personalized health support** can also be provided by this chatbot in order to handle users' queries. With the utilization of transformer model, user inputs will be analyzed and a personalized recommendation will be provided based on TCM principles such as ratio of herbals remedies, lifestyle suggestions and even diets recommendations. With the tailored approach, a more **patient-centric healthcare experience** is achieved which will soon replace the traditional ways of consulting a medical professional.

In summary, the voice-assisted TCM chatbot leverages technology to make traditional Chinese medicine more accessible to a wider audience while introducing a novel approach to medical consultation. By focusing on patient-centric care, the chatbot enhances healthcare efficiency without overburdening the system. It reduces the need for manpower by addressing routine patient queries through preliminary consultations, ultimately improving the overall efficiency of medical field. This project creates a valuable resource that not only facilitates communication but also empowers individuals to take an active role in their health and well-being.

## 1.6 Report Organization

The details of this development are shown in the following chapters. In Chapter 2, some previous system or project is being reviewed and studied. Chapter 3 proposed a detailed solution for improvising the existing problem from existing project or system in which it contains the flowchart of this project in details. In Chapter 4, shows the preliminary action that has been taken, where the details and images is shown and explained. Chapter 5 consist of the conclusion for this current report.

# CHAPTER 2

## Literature Reviews

The increasing diversity in global populations has highlighted the need for inclusive healthcare systems capable of providing 24/7 assistance. Traditional healthcare systems often struggle to address the linguistic [14] and cultural diversity of patients leading to gaps in care [15]. In healthcare, chatbots can address a wide range of healthcare related questions, offering personalized advice and recommendations, which can lead to improved patient outcomes [16]. However, the existing chatbots lacked advanced contextual understanding and domain specificity, resulting in frequent inaccuracies in their responses. Their limited language comprehension capabilities hindered their ability to engage in natural, human-like interactions, leading to user experiences that felt robotic and disconnected [17]. Hence, the chatbots that need to respond empathetically to user inputs in order to improve user satisfaction [18].Voice-assisted chatbots have emerged as a promising solution to bridge these gaps by offering real-time, multilingual support. However, the accuracy of these systems is heavily dependent on the quality of speech-to-text (STT) transcription and intent recognition [19].Misrecognition of speech due to accents, background noise, or homophones remains a significant barrier to effective implementation[20].

Transformer-based models, such as XLM-RoBERTa, have revolutionized natural language processing (NLP) by enabling cross-lingual understanding and transfer learning. For example, BERT and ELECTRA are utilized to improve the understanding and generation of responses in healthcare chatbots [21] These models excel in processing natural language, allowing chatbots to engage in complex dialogues and manage user queries effectively. XLM-RoBERTa, in particular, has been widely adopted for tasks requiring multilingual text analysis due to its ability to handle multiple languages with high accuracy [22]. XLM-RoBERTa achieves an impressive 98.57% accuracy in intent detection, demonstrating its ability to discern user intentions effectively [23].The model's contextual understanding allows it to adapt to different languages and datasets, enhancing user interaction in diverse environments. However, its application in healthcare-specific contexts, such as Traditional Chinese Medicine

(TCM), remains underexplored. Fine-tuning transformer models for domain-specific tasks has been shown to enhance their performance in specialized fields [24].

Recent advancements in STT models have significantly improved the accuracy of speech transcription. Deep learning-based models, such as Google's Wave2Vec 2.0 [25] and OpenAI's Whisper, have demonstrated state-of-the-art performance in multilingual speech recognition. However, these models still face challenges in handling noisy environments [26] and diverse accents [27] particularly in low-resource languages. Integrating STT models with contextual understanding systems, such as transformer-based models, has shown promise in improving intent recognition accuracy[28].

Combining STT models with transformer-based text models offers a multimodal approach to intent recognition, addressing the limitations of standalone systems. Recent studies have demonstrated the effectiveness of such integrations in improving the accuracy of intent detection, particularly in noisy or multilingual environments [29]. For instance, the integration of Whisper with BERT-based models has shown significant improvements in intent recognition for healthcare applications. However, challenges remain in handling phonology errors and misrecognized words in STT outputs, which can lead to misinterpretation of user intents [30]

The development of bilingual chatbots for healthcare applications has gained traction in recent years. These systems aim to provide seamless communication in multiple languages, addressing the needs of diverse patient populations Dadich et al., 2024. For example, a bilingual chatbot developed for diabetes management demonstrated improved patient engagement and satisfaction [31]. However, existing systems often lack domain-specific knowledge, particularly in specialized fields like TCM limiting their effectiveness [32].

While significant progress has been made in STT and transformer-based models, several gaps remain. First, the integration of these models for bilingual healthcare applications, particularly in TCM, is underexplored. Second, the impact of phonology errors in STT outputs on intent recognition accuracy requires further investigation. Finally, there is a need for robust evaluation frameworks to assess the performance of multimodal chatbots in real-world healthcare settings [33].

This research aims to address these gaps by developing a bilingual voice chatbot for UTAR Hospital that integrates a state-of-the-art STT model with a fine-tuned XLM-RoBERTa model for accurate intent recognition in TCM-related queries. By leveraging a multimodal approach, the proposed system will enhance the accuracy of speech transcription and intent detection, providing a scalable solution for diverse patient populations.

Speech-to-Text Framework

The standard Speech-to-Text framework, on which this research is based, will be thoroughly investigated. It consists of four key stages: Speech Input Acquisition, Feature Extraction, Decoder, and Word Output.

1. Speech input acquisition: This process captures spoken audio through a microphone or recording device [34]. The analog signal is converted into a digital format through sampling and quantization [35], ensuring clarity and reducing noise. This step is critical because the quality of the input directly impacts the accuracy of the entire system.

2. Feature extraction: This process transforms the raw audio signal into meaningful features, such as Mel-Frequency Cepstral Coefficients (MFCCs) or spectral features, or others. These features represent the unique characteristics of the speech, such as phonemes, and reduce the complexity of the data while preserving essential information for recognition. This step is crucial for enabling the system to analyze and interpret the audio effectively. Following are common feature extraction techniques used for feature extraction task:

    1. Mel-Frequency Cepstral Coefficients (MFCCs): MFCCs are one of the most widely used features in speech recognition. They mimic the human auditory system by converting the frequency spectrum of the audio into a Mel scale and then applying a discrete cosine transform to extract coefficients [36].

    2. Spectral Features: Features like the Short-Time Fourier Transform (STFT) and spectrograms are used to analyze the frequency content of the audio over time. These features provide a visual representation of

the speech signal, which is useful for both traditional and deep learning-based systems

3. Perceptual Linear Prediction: PLP is another feature extraction technique that models human hearing more accurately by applying psychophysical concepts like loudness and critical band analysis [37].

4. Deep Learning-Based Features: Modern systems often use deep learning models, such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), to automatically learn features from raw audio. For example, WaveNet and Wav2Vec [38] directly process raw waveforms to extract high-level features without manual engineering. Another approach, based on self-supervised learning models like Wav2Vec 2.0 [39] have revolutionized feature extraction by learning representations from large amounts of unlabeled audio data.

5. End-to-End Feature Learning: In end-to-end systems, feature extraction and recognition are combined into a single model. For instance, Transformer-based architectures (e.g., Whisper by OpenAI) directly map raw audio to text, eliminating the need for explicit feature extraction.

3. Decoder: This is the core process that matches the extracted features to the most likely sequence of words. It uses acoustic models to map features to phonemes or words and language models to predict word sequences based on context and grammar. Modern systems often employ advanced machine learning techniques, such as deep neural networks or Transformer-based architectures, to improve accuracy and efficiency. The decoder balances precision and computational speed, especially in real-time applications. Some of the techniques used in modern decoders are such as follows:

    1. Deep Learning-Based Decoders: Use deep learning architectures, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, or Gated Recurrent Units (GRUs), to model temporal dependencies in speech [40].

2. Attention Mechanisms: Models like Listen, Attend, and Spell (LAS) [41,42] and Transformer-based systems (e.g., OpenAI's Whisper) use attention mechanisms to directly map audio features to text [43].

4. Word output: This process produces the recognized text as the final result. This step often includes post-processing to refine the output, such as adding punctuation, correcting grammar, and formatting the text for readability. The accuracy of the output depends on the performance of the previous stages, and advancements in technology continue to enhance the overall quality of speech-to-text systems. This final text can be used in various applications, such as transcription services, voice assistants, and real-time communication tools[44].

# CHAPTER 3

# System Methodology/Approach

### 3.1 Methods

This project will be implementing Agile methodology, as it allows the development to be iterated from design phase after testing it. The purpose of choosing agile methodology is because of the flexibility for test and refine before deploying, ensuring the workability and quality of the development. The testing phase is to make tests, benchmarks, and analyze the system for further improvement, in order to create a better version of the system before deploying it. If the system does not met the threshold requirement, it will loop back to design phase and make modification.

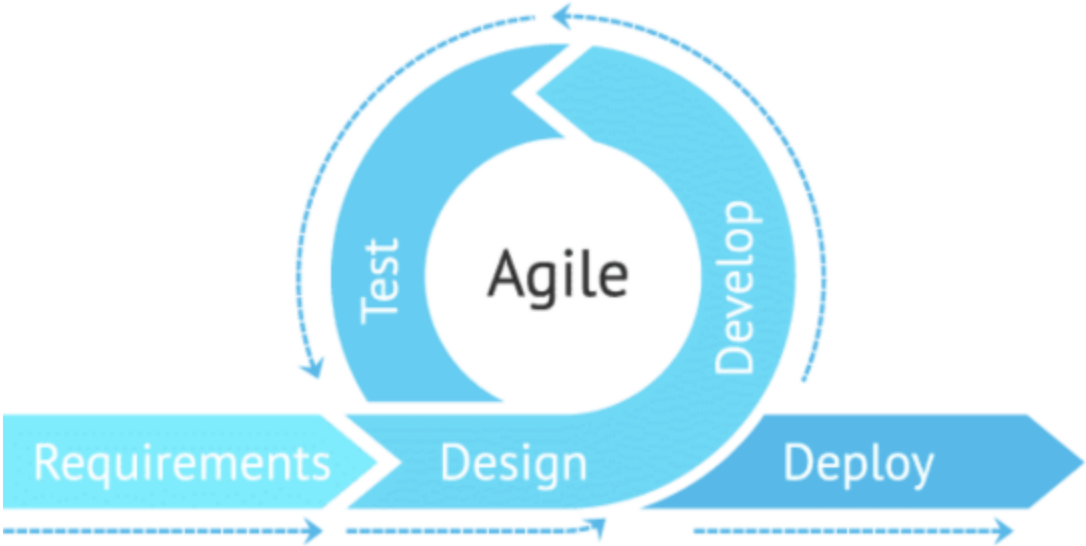



**Figure 4.1 Agile Methodology** [45]

- **Requirement phase**

The main purpose of this phase is gathering and defining the requirement needed to build this system including the datasets about TCM needed, core functionality such as voice interaction, healthcare information retrieval, and integration with hospital system.

- **Design Phase**

In this phase, system architecture, system design and prototype will be defined based on the identified requirement. The prototype will be developed referencing to the blueprint to ensure the user-friendly and accessibility of the system.

- **Develop Phase**

Develop phase is the phase that real working is getting done. Actual coding and development of the chatbot is started here following Sprint planning to ensure the development process is well managed.

- **Test Phase**

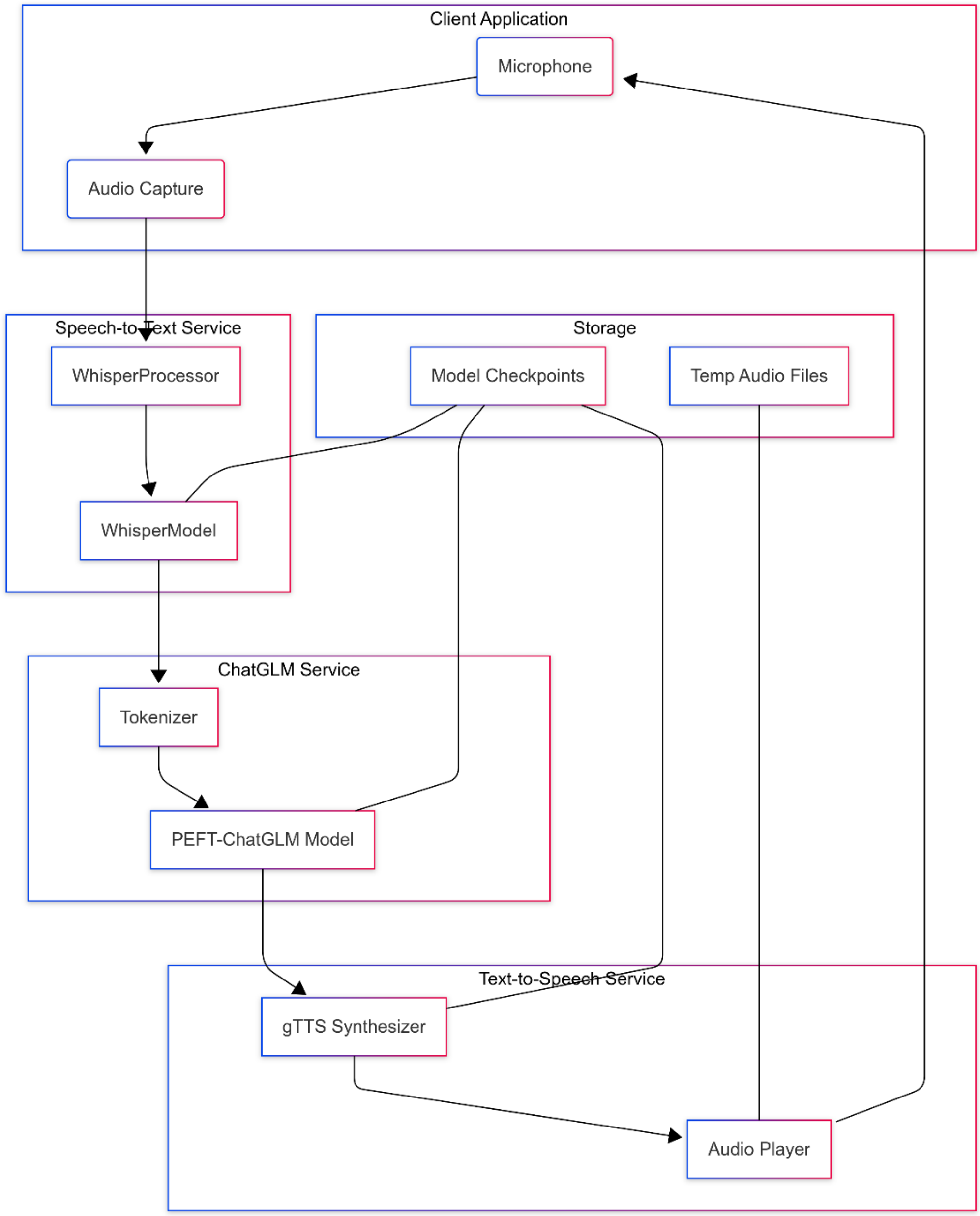
Testing phase, is the phase to validate the chatbot's functionality and quality. Few kind of testing will be done such as unit testing, integration testing and user testing to ensure each part of the chatbot will work properly and if the chatbot meets the client's requirements. Any feedback and error will be recorded to be revaluate again in the design phase.

- **Deploy phase**

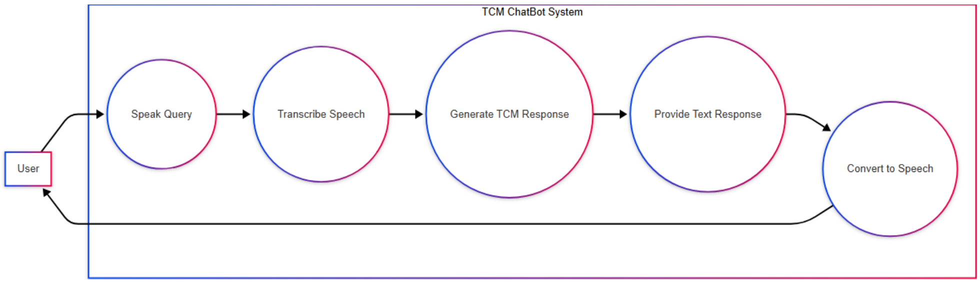
Once every part of chatbot is validated, the chatbot is now ready to be deploy and serve the user.

## 3.2 System Design Diagram/Equation

### 3.2.1 System Architecture Diagram



### 3.2.2 Use Case Diagram




**Actor**

User

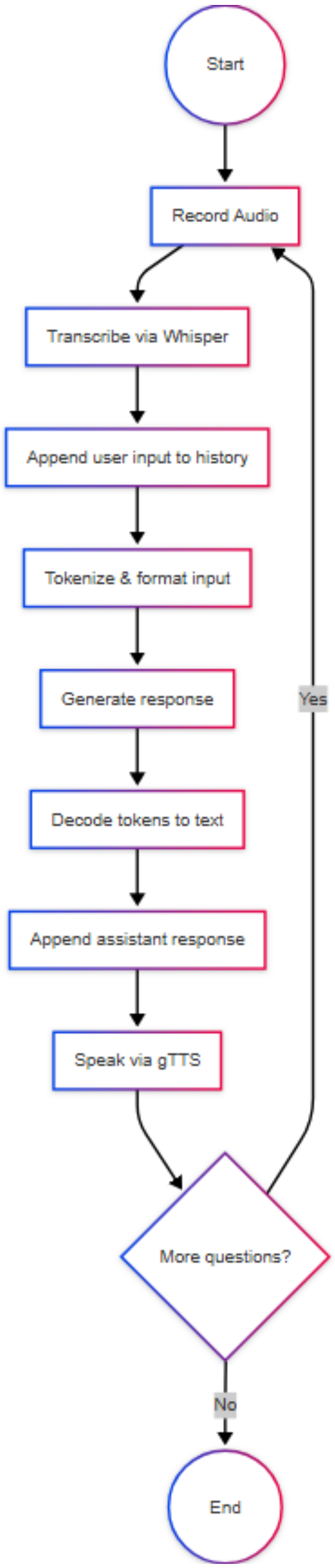
An end-user who speaks Traditional Chinese Medicine (TCM) questions into their device and listens to the spoken answers.

**Use Cases**

1. Speak Query
   - Goal: Capture the user's spoken question about TCM.
   - Preconditions: The user has the application open and microphone access granted.
   - Main Flow:
     1. User clicks "Start" or otherwise signals they're ready.
     2. The system records a short audio snippet.
   - Postconditions: Audio data is available for transcription.
2. Transcribe Speech
   - Goal: Convert the recorded audio into text.
   - Preconditions: A valid audio snippet exists.
   - Main Flow:
     1. The audio buffer is sent to the Whisper STT module.
     2. Whisper processes the audio and returns a text transcript.
   - Postconditions: A text string representing the user's query is produced.

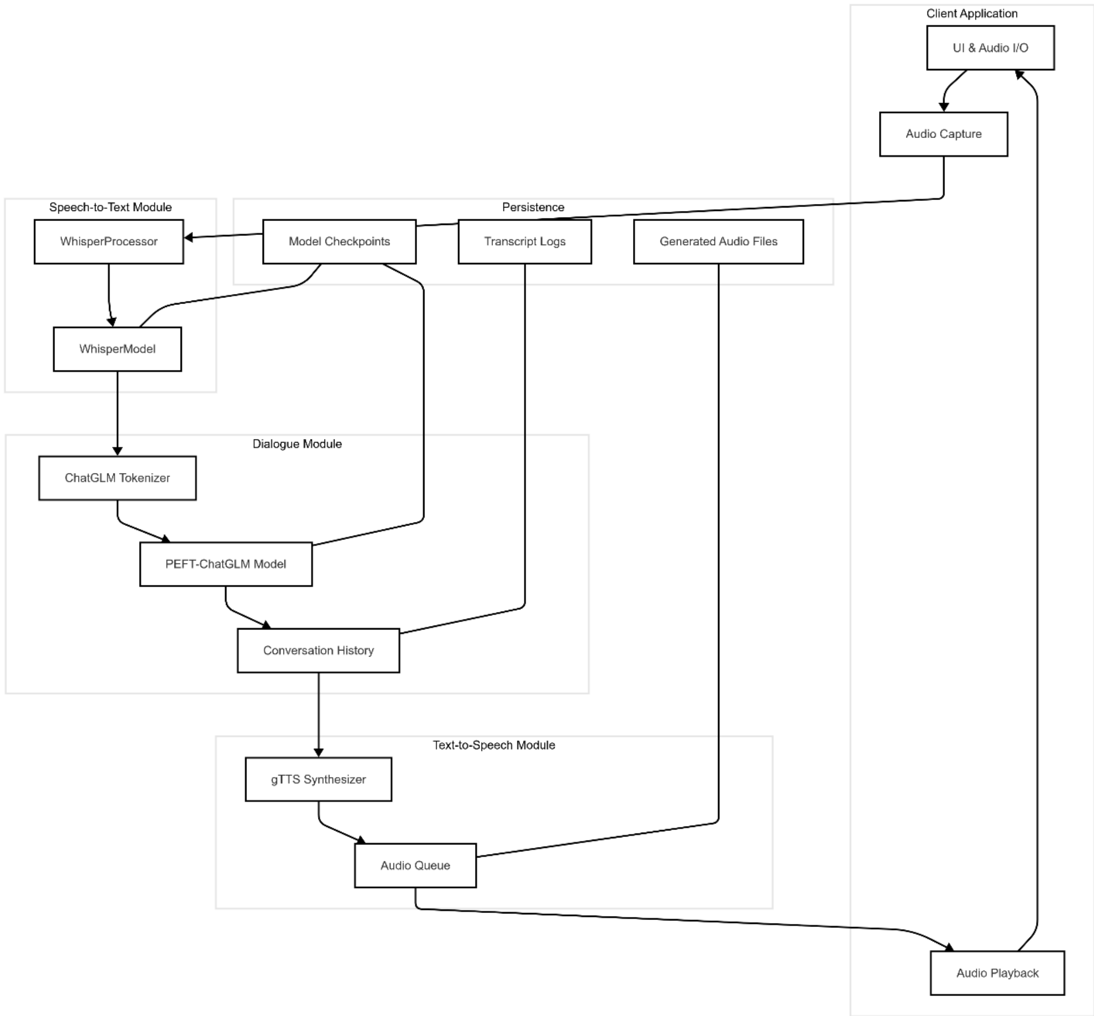
3. Generate TCM Response
   - Goal: Use the fine-tuned ChatGLM model to craft an expert reply.
   - Preconditions: A valid text query is available.
   - Main Flow:
     1. The transcript is tokenized and passed into the PEFT-fine-tuned ChatGLM.
     2. ChatGLM generates a contextually appropriate TCM answer.
   - Postconditions: A text response is ready for formatting.
4. Provide Text Response
   - Goal: Format and prepare the model's reply for synthesis.
   - Preconditions: The model has returned a raw text string.
   - Main Flow:
     1. The system wraps the reply in any UI templates, logs it to history, and queues it for TTS.
   - Postconditions: The response text is stored and ready for speech conversion.
5. Convert to Speech
   - Goal: Produce audible output so the user can "hear" the answer.
   - Preconditions: A queued response text is available.
   - Main Flow:
     1. The text is sent to the gTTS module (with language/TLD fallback).
     2. gTTS returns an MP3 (or WAV) file.
     3. The system plays the audio back through the device speakers.
   - Postconditions: The user hears the spoken TCM advice.

### 3.2.3 Activity Diagram



# Chapter 4 System Design

## 4.1 System Block Diagram




## 4.2 System Components Specifications

| Component | Responsibility | Key Technologies / Versions |
|---|---|---|
| UI & Audio I/O | Manages UI, microphone & speaker I/O | PyAudio |

| **Component** | **Responsibility** | **Key Technologies / Versions** |
|---|---|---|
| WhisperProcessor | Pre-/post-processing for Whisper input/output | transformers v4.x, torchaudio |
| WhisperModel | Transcribes audio → text | Finetuned whisper checkpoint |
| ChatGLM Tokenizer | Tokenizes/chat-formats conversational text | ChatGLM-Tokenizer library |
| LORA-ChatGLM Model | Generates TCM-expert responses | Finetuned ChatGLM-4B + LORA adapters |
| gTTS Synthesizer | Converts text → audio | gTTS v2.x, fallback TLD logic |
| Audio Queue | Buffers synthesized audio for playback | Python queue.Queue |
| Persistence Layer | Reads/writes model checkpoints and audio files | File system |

## 4.3 NLP framework – User Interaction

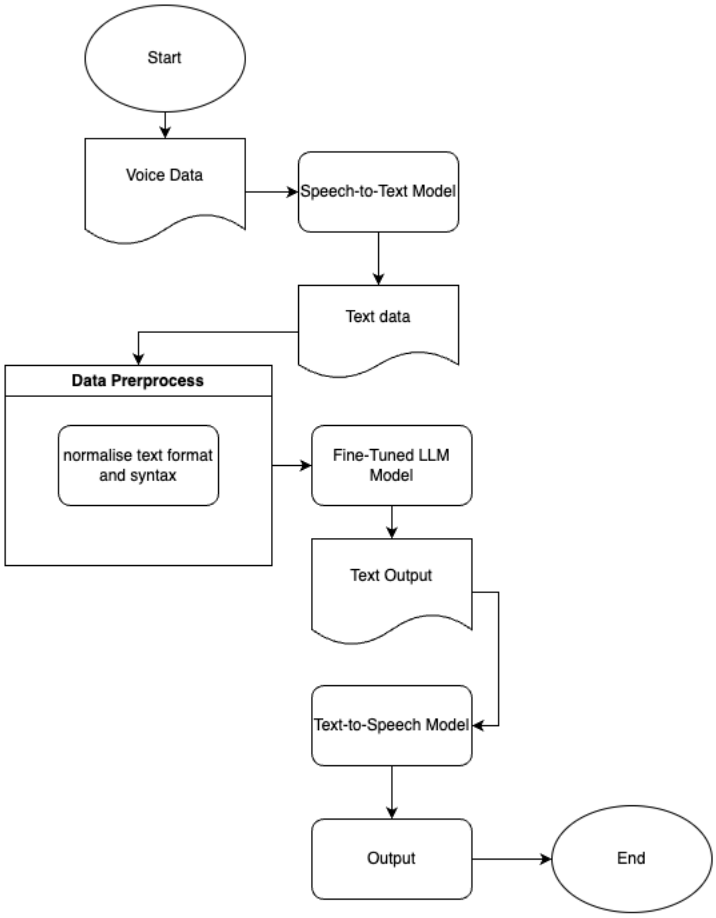



**Figure 4.3.1 Voice Assisted Chatbot User Interaction Flow**

Figure 4.3.1 illustrate the user interaction flow of the voice-assisted TCM chatbot. The chatbot will first capture voice input such as user request for information, a question, illness symptoms or any form of queries related to TCM. The chatbot will then convert the voice into text using Speech-to-text-model. After the conversion, the text input is then put into preprocessing phase which is normalizing the text format and syntax to be fed into the LLM Model. This step is important as it will improve the accuracy of the chatbot understanding the user's query by removing the data noise.

The processed data will be processed by the tuned model. The tuned model will analyze the data and perform searches from the processed data storage to find the most suited solution for user's queries. It will then generate the response and send to the text-to-speech model for another transform. The final output will be in voice which is sent out to the user.

### 4.4 Speech-to-Speech Framework – Model Building

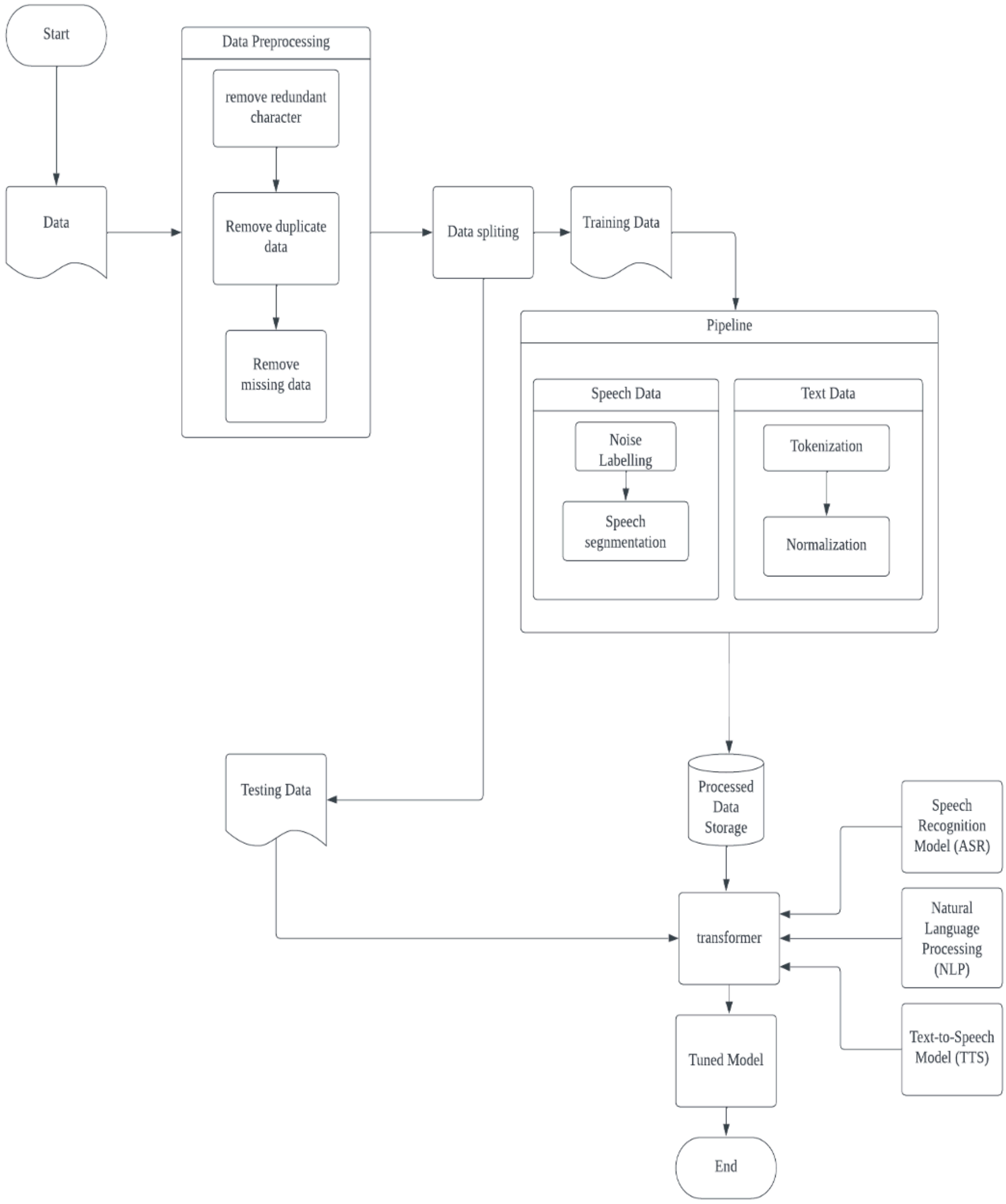



**Figure 4.4.1 Chatbot Development Flow**

Figure 4.4.1 depicts the systematic process of developing a voice-assisted Traditional Chinese Medicine (TCM) chatbot. The development framework begins with data preparation, followed by data preprocessing, model training and finally model evaluation and deployment.

The project initiates with the collection and preparation of relevant data, focusing on TCM-specific knowledge, including herbs, symptoms, and treatments. This data forms the foundation for training the chatbot. Data sources include repositories like Kaggle and UCI Machine Learning Repository, Hugging Face, and Github, which offer comprehensive datasets. In addition to text data, voice data is also collected for training the speech recognition and text-to-speech components of the chatbot.

During the data preprocessing, the voice data will undergo noise labelling, to ensure chatbot is robust in real-world scenarios where background noise is present. The voice data is then segmented into smaller, manageable parts to improve the accuracy of subsequent speech recognition process.

When data is done preprocessing, it will be split into train and test dataset with the ratio of 80:20. Training data is used for model development, while the testing data is reserved for evaluating the model's performance, ensuring it can generalize to new, unseen data.

Training data will then be fed into a pipeline after splitting where it undergoes further processing. This prepares the data for speech recognition model, which is trained to recognize speech even in noisy conditions. While text data is tokenized and normalized, converting it into a format that the model can efficiently process. The transformer model, with a large language model at its core, is configured to handle both text and voice data. The text data is vectorized and stored in vector storage, linked to the transformer for efficient similarity searching. The model also incorporates speech recognition (ASR) and text-to-speech (TTS) components, enabling voice interaction with the chatbot.

# Chapter 5 System Implementation

## 5.1 Setting up

### 5.1.1 Software

Before starting to develop the Voice Assisted AI chatbot, there are 2 main software needed to be installed:

1. PyCharm CE version 2024.1.4
2. Python 3.12

### 5.1.2 Libraries

This project also requires some important libraries to be installed in the IDEs, which in this case is PyCharm:

1. NLTK
   A library for NLP tasks which provide tools for tokenization, lemmatization, and other linguistic data analysis.
2. Transformers
   Library by Hugging face that provides pretrained models. It facilitates model fine-tuning and deployment for task like question answering.
3. Jieba
   A library for Chinese text segmentation, which allow splitting of Chinese text into meaningful words or phrase.
4. Numpy
   Library for numerical computing in Python such as operations on array, mathematical computations and data manipulations.
5. Torch
   A deep learning framework used for building, training , and deploying machine learning models.
6. PEFT
   Library to fine-tune LLM without modifying the entire model. It reduces the resources consumption during training by adapting only a subset of parameters.
7. Os

os.path.isdir, os.makedirs, os.path.isfile: filesystem checks for existence of directories/files.Used to clear stale offload folders, create a “manifests” directory, and verify that audio paths exist before processing.

8. Shutil
   shutil.rmtree: remove a directory tree.
   Cleans up the ./offload folder if it exists to avoid stale model shards.
9. Json
   json.load /json.dumps / open(...).write: read and write JSON manifests of audio transcripts or dialogue examples
10. Argparse
    argparse.ArgumentParser, add_argument, parse_args: parse command-line flags such as --checkpoint_dir, --audio_manifest, --build_manifests, etc., to control which evaluation routines to run.
11. Soundfile
    sf.read(path): load waveform and sample rate from various audio file formats.
    Used in load_audio to read input audio before any resampling.
12. Librosa
    librosa.core.resample(y, orig_sr, target_sr): convert audio from its original sampling rate to the model’s required rate (SAMPLING_RATE).
13. Jiwer
    wer(refs, hyps): compute Word Error Rate to evaluate ASR performance.
    cer(refs, hyps): compute Character Error Rate.Both metrics are printed in evaluate_asr
14. Evaluate (Hugging Face’s evaluate library)
    evaluate.load("rouge"): load the ROUGE metric for summarization-style evaluation.
    evaluate.load("bertscore"): load BERTScore for semantic

similarity.

Used in evaluate_llm to assess LLM-generated TCM responses against references.

15. Datasets

    load_dataset(...): fetches datasets from the Hugging Face Hub. In build_manifests, loads (1) Common Voice Chinese speech for ASR examples, and (2) the ShenNong_TCM_Dataset for dialogue examples, then writes them to JSON manifest files.

## 5.2 Settings and Configurations

### 5.2.1 Data Preparation and Preprocessing for Text data

The dataset used in this project is found from hugging face. This dataset contains query and responses related to TCM. The datasets contains 135 rows and only 2 columns which is query and response. Query column consists of the question, inquiry and request about TCM is conversational formats while responses column consists of the respond of the query in sentences form as well.

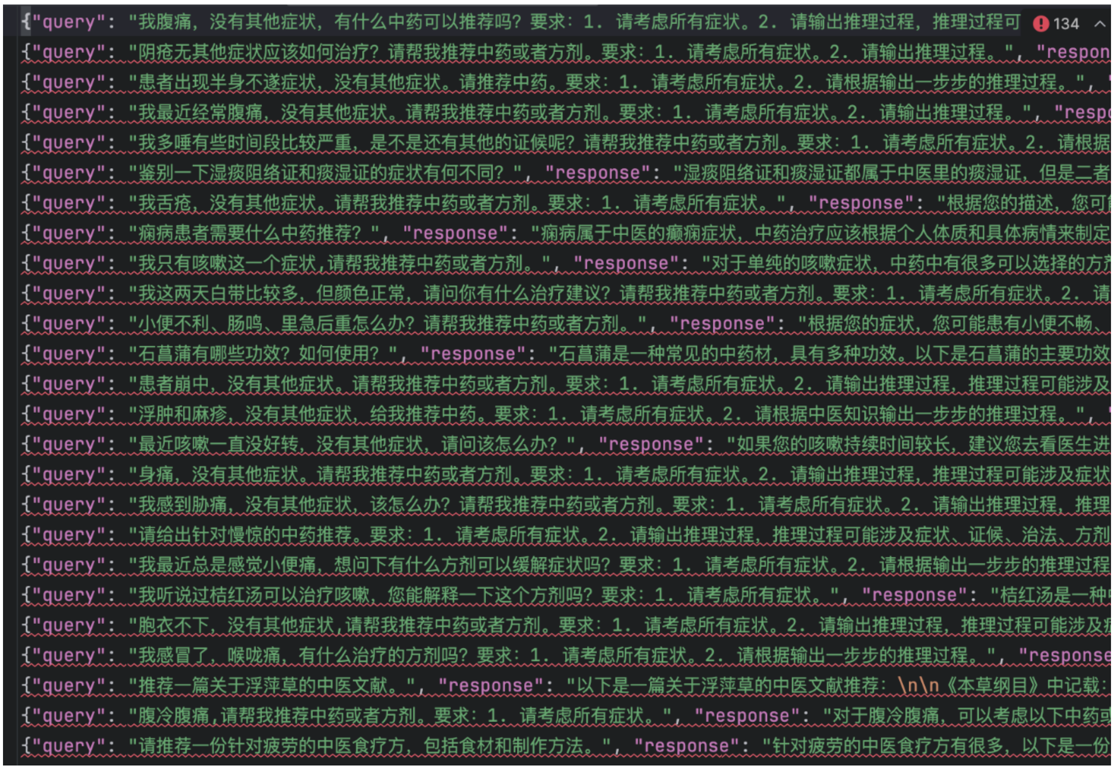



**Figure5.2.1.1 Original Datasets format**

Figure 5.2.1.1 shows the format and syntax from the original datasets which is in .jsonl format. The syntax for the datasets is “query:” then proceeding with “response”.

“Query” refers to the inputs in this datasets and the target variable is the “Response”. However, the LLM model used in this project does not fine tune with such syntax, hence normalization of datasets syntax is required.

```
{"messages": [{"role": "user", "content": "我腹痛，没有其他症状，有什么中药可以推荐吗？要求：1. 请考虑所有症状。2. 请输出推理过程，推理过程
{"messages": [{"role": "user", "content": "阴疮无其他症状应该如何治疗？请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理过程。"}
{"messages": [{"role": "user", "content": "患者出现半身不遂症状，没有其他症状。请推荐中药。要求：1. 请考虑所有症状。2. 请根据输出一步步的推理
{"messages": [{"role": "user", "content": "我最近经常腹痛，没有其他症状。请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理过程。"
{"messages": [{"role": "user", "content": "我多唾有些时间段比较严重，是不是还有其他的证候呢？请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。
{"messages": [{"role": "user", "content": "鉴别一下湿痰阻络证和痰湿证的症状有何不同？"}, {"role": "assistant", "content": "湿痰阻络证和
{"messages": [{"role": "user", "content": "我舌疮，没有其他症状。请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。"}, {"role": "assistant"
{"messages": [{"role": "user", "content": "痫病患者需要什么中药推荐？"}, {"role": "assistant", "content": "痫病属于中医的癫痫症状，中药
{"messages": [{"role": "user", "content": "我只有咳嗽这一个症状,请帮我推荐中药或者方剂。"}, {"role": "assistant", "content": "对于单纯的
{"messages": [{"role": "user", "content": "我这两天白带比较多，但颜色正常，请问你有什么治疗建议？请帮我推荐中药或者方剂。要求：1. 请考虑所有症
{"messages": [{"role": "user", "content": "小便不利、肠鸣、里急后重怎么办？请帮我推荐中药或者方剂。"}, {"role": "assistant", "content":
{"messages": [{"role": "user", "content": "石菖蒲有哪些功效？如何使用？"}, {"role": "assistant", "content": "石菖蒲是一种常见的中药材，
{"messages": [{"role": "user", "content": "患者崩中，没有其他症状。请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理过程，推理过
{"messages": [{"role": "user", "content": "浮肿和麻疹，没有其他症状，给我推荐中药。要求：1. 请考虑所有症状。2. 请根据中医知识输出一步步的推理
{"messages": [{"role": "user", "content": "最近咳嗽一直没好转，没有其他症状，请问该怎么办？"}, {"role": "assistant", "content": "如果您
{"messages": [{"role": "user", "content": "身痛，没有其他症状。请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理过程，推理过程可
{"messages": [{"role": "user", "content": "我感到胁痛，没有其他症状，该怎么办？请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理
{"messages": [{"role": "user", "content": "请给出针对慢惊的中药推荐。要求：1. 请考虑所有症状。2. 请输出推理过程，推理过程可能涉及症状、证候、
{"messages": [{"role": "user", "content": "我最近总是感觉小便痛，想问下有什么方剂可以缓解症状吗？要求：1. 请考虑所有症状。2. 请根据输出一步步的
{"messages": [{"role": "user", "content": "我听说过桔红汤可以治疗咳嗽，您能解释一下这个方剂吗？要求：1. 请考虑所有症状。"}, {"role": "assis
{"messages": [{"role": "user", "content": "胞衣不下，没有其他症状,请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。2. 请输出推理过程，推理过程
{"messages": [{"role": "user", "content": "我感冒了，喉咙痛，有什么治疗的方剂吗？要求：1. 请考虑所有症状。2. 请根据输出一步步的推理过程。"},
{"messages": [{"role": "user", "content": "推荐一篇关于浮萍草的中医文献。"}, {"role": "assistant", "content": "以下是一篇关于浮萍草的中
{"messages": [{"role": "user", "content": "腹冷腹痛,请帮我推荐中药或者方剂。要求：1. 请考虑所有症状。"}, {"role": "assistant", "content"
{"messages": [{"role": "user", "content": "请推荐一份针对疲劳的中医食疗方，包括食材和制作方法。"}, {"role": "assistant", "content": "针
{"messages": [{"role": "user", "content": "头痛已经持续了两天，怎么治疗比较好？请帮我推荐中药或者方剂。"}, {"role": "assistant", "content
{"messages": [{"role": "user", "content": "便秘严重的时候会有口干口渴的症状，这是为什么？"}, {"role": "assistant", "content": "便秘严重
{"messages": [{"role": "user", "content": "请问冬瓜皮有什么功效？"}, {"role": "assistant", "content": "冬瓜皮具有清热解毒、利尿消肿、降
{"messages": [{"role": "user", "content": "问诊：我最近经常头疼怎么办？"}, {"role": "assistant", "content": "您好，头疼有很多种可能的原
```

**Figure 5.2.1.2 Transformed syntax of dataset**

In figure 5.2.1.2 is a perfectly transformed and normalized syntax of the datasets. “messages” is stated as an indicator of a new line of data. The format proceeded with the “roles”, “content”, and “roles”, “content” again. As the chatbot in this project only do single respond to single queries, the “role” will firstly be the user and the “content” will be the messages of the user. For the respond, the role will be the “assistant” in which depicts the model itself, and the proceeding “content” refers to the sentence the chatbot will respond. After the transformation is done, the datasets will then be exported in .jsonl format.

### 5.2.2 Data Splitting

The processed data is then split into training sets and test sets with the ratio of 8:2. The splitted datasets will then be fed into the base models together with a LORA config files to undergo a fine-tuning process. LORA config file is needed because the fine-tuning process used in this project is LORA(Low-Rank-Adaptation of Large Language Models)

### 5.2.3 LORA config parameters testing

```yaml
training_args:
  # see `transformers.Seq2SeqTrainingArg
  output_dir: ./output
  max_steps: 3000
  # needed to be fit for the dataset
  learning_rate: 5e-4
  # settings for data loading
  per_device_train_batch_size: 1
  dataloader_num_workers: 16
  remove_unused_columns: false
  # settings for saving checkpoints
  save_strategy: steps
  save_steps: 500
  # settings for logging
  log_level: info
  logging_strategy: steps
  logging_steps: 10
  # settings for evaluation
  per_device_eval_batch_size: 4
  eval_strategy: steps
  eval_steps: 500
```

**Figure 5.2.3 LORA Configuration Parameters**

Figure 5.2.3 shows parts of the parameters in the LORA configuration files. The **"max_steps"** indicated the maximum iteration of learning in which is sets to 3000 in this project. **"learning_rate"** is the amount of fluctuation for each step which is 5e-4.

**“remove_unused_columns”** is set to false such that when one of the columns is empty, it will still takes the other input to learn as it will also be related to TCM.

**5.2.4 Data Preparation and Preprocessing for Voice data**

The Mozilla Common Voice 13 dataset is also available via the Hugging Face Hub. This dataset contains paired audio recordings and their corresponding text transcriptions for use in automatic speech recognition:

**Figure 5.2.4.1 Common Voice Dataset**

Figure 5.2.4.1 shows the columns and some rows of the datasets. It comprises over one million examples (i.e. audio–text pairs) and exposes only two primary columns when loaded via the dataset’s library: Audio, Sentence. Audio column **c**ontains short speech utterances (MP3 or WAV) contributed by volunteers around the world. Sentence column provides the exact text transcript of each utterance, normalized and cleaned for ASR training. Beyond these two fields, the full release also includes optional metadata such as speaker age, gender, and accent, which can be leveraged for speaker-conditional modeling or data filtering.

```python
def flatten_ids(seq):
    """
    Recursively flatten nested lists/tuples/arrays/tensors into a flat list.
    """
    for el in seq:
        if isinstance(el, torch.Tensor):
            yield from flatten_ids(el.tolist())
        elif isinstance(el, np.ndarray):
            yield from flatten_ids(el.tolist())
        elif isinstance(el, (list, tuple)):
            yield from flatten_ids(el)
        else:
            yield el


def main():
    # 1) Choose the CV version that has Malay (ms)
    configs = get_dataset_config_names("mozilla-foundation/common_voice_13_0")
    cv_version = "common_voice_13_0" if "ml" in configs else "common_voice_12_0"

    # 2) Languages
    langs = {"en": "English", "zh-CN": "Chinese", "ml": "Malay"}

    # 3) Load, slice, and tag each language
    parts = []
    for code in langs:
        split_arg = f"train[:{SPLIT_PCT}%]" if SPLIT_PCT else "train"
        ds = load_dataset(
            path: f"mozilla-foundation/{cv_version}",
            code,
            split=split_arg,
            trust_remote_code=True,
        ).cast_column( column: "audio", Audio(sampling_rate=16000))

        if MAX_SAMPLES:
            ds = ds.shuffle(seed=42).select(range(min(len(ds), MAX_SAMPLES)))

        ds = ds.map(lambda x: {"lang": code}, num_proc=4)
        parts.append(ds)

    dataset = concatenate_datasets(parts)
```

**Figure 5.2.4.2 Voice Data Loading and Preparation**

```python
# 4) Filter invalid examples
def is_valid(ex):
    audio_arr = ex["audio"]["array"]
    txt = ex.get("sentence", "")
    return hasattr(audio_arr, "__len__") and len(audio_arr) > 0 and isinstance(txt, str) and txt.strip()

dataset = dataset.filter(is_valid, num_proc=4)
```

**Figure 5.2.4.3 Filter for invalid examples**

Figure 5.2.4.2 shows the process of loading and preparing voice data. It starts by deciding how much of each language's recordings to use. Two settings at the top: **SPLIT_PCT** and **MAX_SAMPLES** control this: **SPLIT_PCT** (e.g. 10) means "take 10% of the data for each language," while **MAX_SAMPLES** (e.g. 5000) caps that slice so you never load more than 5,000 clips. Internally, it builds a "split argument" called **split_arg** (e.g. "**train [:10%]**") which tells the loader to grab just that fraction. It then calls **load_dataset(..., split=split_arg)** for English, Chinese, and Malay, resamples every audio file to a uniform 16 kHz quality, and shuffles and truncates if needed. After loading, each clip is tagged with its language code via a small **.map(lambda x: {"lang": code})** step. Finally, the three language-specific lists are merged into one big "multilingual" list and **.filter(is_valid)** drops any entries with missing audio or missing text as shown in figure 5.2.4.3.

```
# 5) Load Whisper
processor = WhisperProcessor.from_pretrained("openai/whisper-base")
model     = WhisperForConditionalGeneration.from_pretrained("openai/whisper-base")

# 6) PyTorch Dataset wrapper with safe flattening
class WhisperDataset(TorchDataset):
    def __init__(self, hf_ds, processor, lang_map):
        self.ds        = hf_ds
        self.processor = processor
        self.lang_map  = lang_map

    def __len__(self):
        return len(self.ds)

    def __getitem__(self, idx):
        ex = self.ds[int(idx)]

        # a) audio → input_features tensor
        feats = self.processor(
            audio=ex["audio"]["array"],
            sampling_rate=16000,
            return_tensors="pt"
        ).input_features[0]

        # b) get_decoder_prompt_ids may return list/tuple/dict/tensor/ndarray
        raw = self.processor.get_decoder_prompt_ids(
            language=self.lang_map[ex["lang"]],
            task="transcribe"
        )
        # if dict form, extract input_ids
        if isinstance(raw, dict) and "input_ids" in raw:
            raw = raw["input_ids"]

        # flatten into pure Python ints
        prompt_ids = [int(i) for i in flatten_ids(raw)]
```

**Figure 5.2.4.4 Preprocessing voice data**

Before teaching the model, every example must be turned into numbers. The code wraps the data in a custom **WhisperDataset** class, so it behaves like a ready-made batch provider. Inside its **__getitem__** method, each audio clip is passed to **processor (audio=…, sampling_rate=16000, return_tensors="pt")** to extract a log-mel "fingerprint"—think of it like a unique numeric signature of the sound. The text transcript is tokenized into a list of numeric IDs **(processor.tokenizer(...).input_ids)**. There's also a small "prompt" made by **processor.get_decoder_prompt_ids (language =…, task ="transcribe")** that tells the model which language it's dealing with. Because these prompts can come in nested lists or tensors, a helper function called **flatten_ids** walks through and flattens them into a simple list of integers.

```
# 7) Collator
def data_collator(batch):
    feats = torch.stack([item["input_features"] for item in batch])
    labs  = [item["labels"] for item in batch]
    padded = torch.nn.utils.rnn.pad_sequence(
        labs, batch_first=True,
        padding_value=processor.tokenizer.pad_token_id
    )
    return {"input_features": feats, "labels": padded}
```

**Figure 5.2.4.5 Collator**

Figure 5.2.4.5 shows prompt IDs and transcript IDs are stitched together into a single label sequence. A separate **data_collator** then batches these by stacking feature tensors and padding every label sequence to the same length, so each training batch is a neat rectangle of numbers.

### 5.2.5 Training of voice data

```
    # 8) Training arguments
    training_args = Seq2SeqTrainingArguments(
        output_dir="./whisper-multilang-finetuned",
        per_device_train_batch_size=4,
        num_train_epochs=3,
        save_steps=500,
        logging_steps=10,
        learning_rate=1e-5,
        fp16=torch.cuda.is_available(),
        report_to="none",
        remove_unused_columns=False,  # keep input_features & labels intact
    )

    # 9) Trainer
    trainer = Seq2SeqTrainer(
        model=model,
        args=training_args,
        train_dataset=torch_dataset,
        data_collator=data_collator,
        tokenizer=processor,  # suppress warning until v5
    )

    # 10) Train & save
    trainer.train()
    trainer.save_model("./whisper-multilang-finetuned")


if __name__ == "__main__":
    freeze_support()
    main()
```

**Figure 5.2.5.1 Training whisper model**

Figure 5.2.5.1 shows the loading of Whisper model and its matching processor via from_**pretrained("openai/whisper-base")**. It sets up training options in a **Seq2SeqTrainingArguments** object—such as where to save outputs **(output_dir)**, batch size **(per_device_train_batch_size)**, number of passes over the data **(num_train_epochs)**, and learning rate. All of this, plus the **WhisperDataset** and **data_collator**, goes into a **Seq2SeqTrainer**, which manages the actual training loop. Calling **trainer.train()** then shows the model each batch of audio features and correct labels, letting it slowly adjust its internal settings to minimize its transcription errors. Once training completes**, trainer.save_model()** writes out the fine-tuned model, ready to transcribe new speech in English, Chinese, or Malay.

```
99%|██████████| 5930/5994 [14:50<00:08,  7.47it/s]{'loss': 0.239, 'grad_norm': 5.634618282318115, 'learning_rate': 1.1511511511511513e-07, 'epoch': 2.97}
99%|██████████| 5940/5994 [14:51<00:07,  7.48it/s]{'loss': 0.2915, 'grad_norm': 5.950321674346924, 'learning_rate': 9.843176509843178e-08, 'epoch': 2.97}
99%|██████████| 5950/5994 [14:52<00:06,  7.23it/s]{'loss': 0.2587, 'grad_norm': 7.307726860046387, 'learning_rate': 8.174841508174842e-08, 'epoch': 2.98}
99%|██████████| 5960/5994 [14:54<00:04,  7.56it/s]{'loss': 0.3166, 'grad_norm': 9.118072509765625, 'learning_rate': 6.506506506506507e-08, 'epoch': 2.98}
00%|██████████| 5970/5994 [14:55<00:03,  7.62it/s]{'loss': 0.2102, 'grad_norm': 6.704226970672607, 'learning_rate': 4.838171504838172e-08, 'epoch': 2.99}
00%|██████████| 5980/5994 [14:56<00:01,  7.36it/s]{'loss': 0.26, 'grad_norm': 6.182464599609375, 'learning_rate': 3.1698365031698366e-08, 'epoch': 2.99}
00%|██████████| 5990/5994 [14:58<00:00,  7.25it/s]{'loss': 0.1282, 'grad_norm': 4.741860866546631, 'learning_rate': 1.5015015015015016e-08, 'epoch': 3.0}
00%|██████████| 5994/5994 [15:02<00:00,  6.64it/s]
'train_runtime': 902.607, 'train_samples_per_second': 26.553, 'train_steps_per_second': 6.641, 'train_loss': 0.3527146198469518, 'epoch': 3.0}
```

**Figure 5.2.5.2 Details during model training**

Figure 5.2.5.2 shows some details during model training. In which consists of metrics like **Loss, Grad_Norm, Learning Rate** and **Epoch**.

### 5.2.6 Preparing for Evaluation

```
1 usage
def build_manifests(common_n, shennong_n):
    os.makedirs( name: "manifests", exist_ok=True)

    cv = load_dataset(
         path: "mozilla-foundation/common_voice_11_0", name: "zh-CN",
         split="test", trust_remote_code=True
    ).shuffle(seed=0).select(range(common_n))
    asr = [{"audio_path":e["audio"]["path"], "transcript":e["sentence"]} for e in cv]
    open("manifests/asr_commonvoice.json","w",encoding="utf-8").write(json.dumps(asr,ensure_ascii=False,indent=2))

    sn = load_dataset( path: "michaelwzhu/ShenNong_TCM_Dataset", name: "train").shuffle(seed=1).select(range(shennong_n))
    dlg = [{"text":e["question"], "reference":e["response"]} for e in sn]
    open("manifests/dialogue_shennong.json","w",encoding="utf-8").write(json.dumps(dlg,ensure_ascii=False,indent=2))

    print(f"Built {len(asr)} ASR + {len(dlg)} dialogue examples.")
```

**Figure 5.2.6.1 Building Manifest**

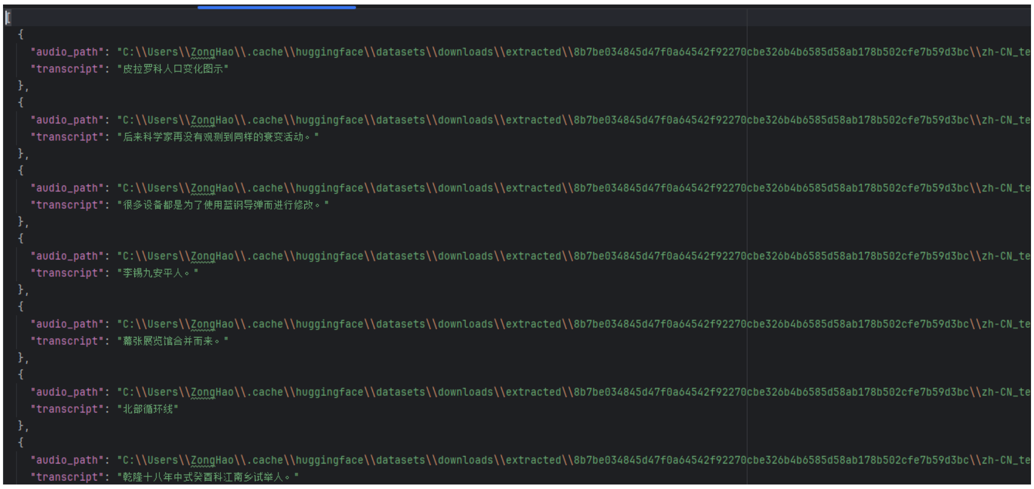


**Figure 5.2.6.2 asr_commonvoice.json**

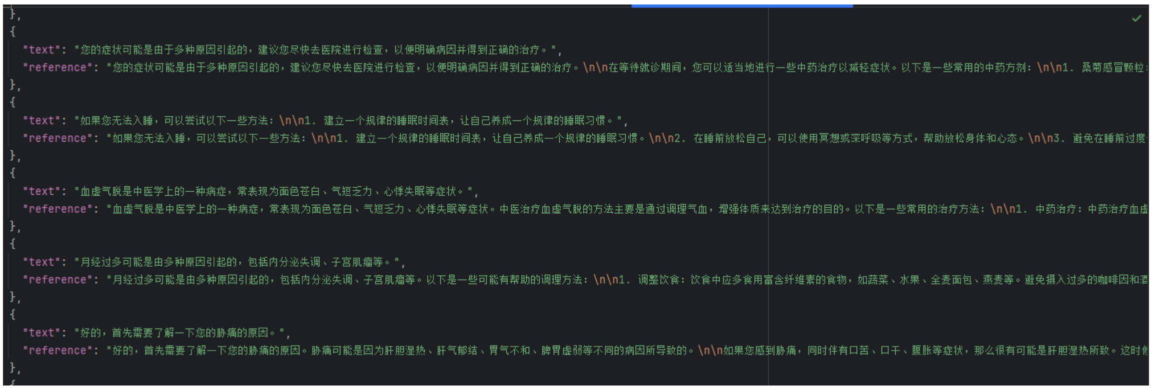


**Figure 5.2.6.3 dialogue_shennong.json.**

Figure 5.2.6.1 shows code to build two manifest files. It loaded Chinese speech clips from Mozilla Common Voice, selected a subset, and wrote their file paths and transcripts into **asr_commonvoice.json** as shown in figure 5.2.6.2. It also loads Traditional Chinese Medicine dialogues from the ShenNong dataset, selected examples, and wrote questions and expected answers into **dialogue_shennong.json** as shown in figure 5.2.6.3

### 5.2.7 Full Implementation

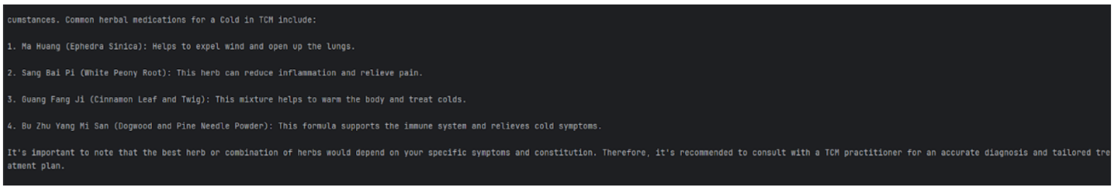



**Figure 5.2.7.1 Treatment for Cold in English**

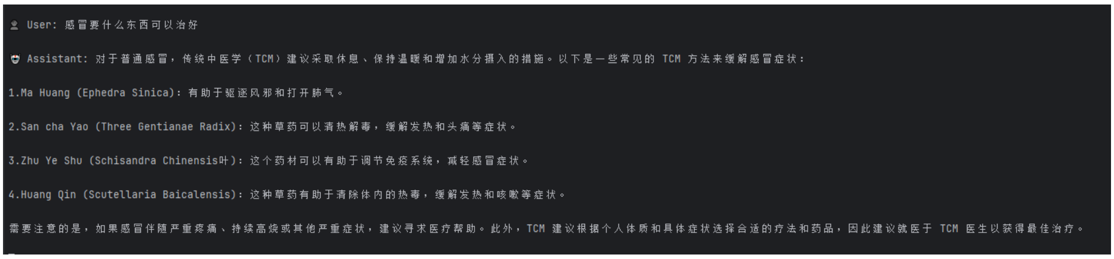



**Figure 5.2.7.2 Treatment for Cold in Chinese**

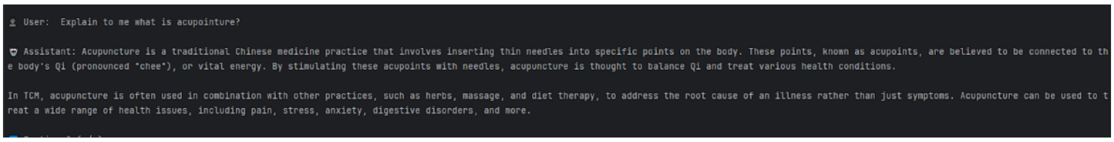



**Figure 5.2.7.3 What is acupuncture in English**

Figure 5.2.7.1, 5.2.7.2, 5.2.7.3 is showing a voice-powered chatbot for Traditional Chinese Medicine (TCM) consultation operates through a continuous cycle of audio capture, transcription, response generation, and speech synthesis. Five seconds of audio are recorded at a standard sampling rate and processed by a Whisper speech-recognition model to produce a text transcription. That transcription is appended to a conversation history, which includes a constant system instruction to restrict responses exclusively to TCM topics, before being submitted to a fine-tuned ChatGLM model enhanced with parameter-efficient fine-tuning (PEFT). The model's generated reply is converted into spoken output by Google's Text-to-Speech engine, which automatically detects language and applies an appropriate accent mapping. This loop—record, transcribe, generate, speak—continues until termination, enabling a seamless, locally hosted spoken-language consultation with a virtual TCM expert.

## 5.3 System Operation

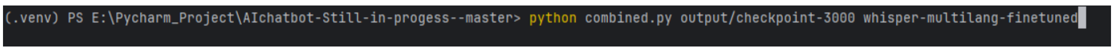



**Figure 5.3.1 Launching the Application**

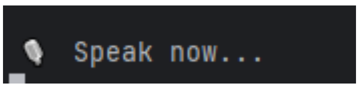



**Figure 5.3.2 Instruct user to give audio prompt**

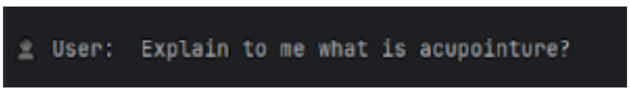



**Figure 5.3.3 Transcribe User Audio Prompt to text**

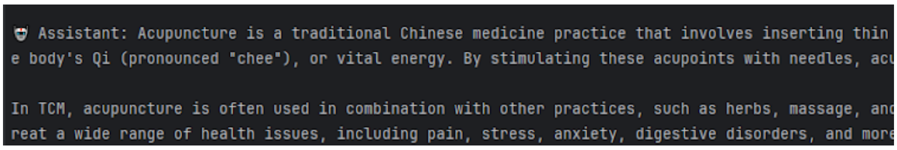



**Figure 5.3.4 Generate Output and read out Output**

## 5.4 Implementation Issues and Challenges

During the implementation phase of the system, several technical and functional challenges were encountered that impacted the performance, usability, and reliability of the system.

### 1. Model Size and Hardware Limitations

The fine-tuned ChatGLM and Whisper models used in this project are computationally intensive. Due to limited hardware resources, especially in terms of GPU memory and processing power, generating output from the models can take a significantly long time (sometimes up to 30 minutes or more depending on the complexity of the input). These delays negatively affect real-time responsiveness, which is critical in a voice-based interaction system. Additionally, while larger and more accurate models exist, they could not be fully utilized due to these hardware constraints.

### 2. Audio Capture Sensitivity and Silence Detection Issues

One major challenge was with audio input capture. The system lacks proper silence detection. When the system prompts the user to speak and no voice input is given (e.g., when the user stays silent), the Whisper model still attempts to transcribe non-existent audio and outputs incorrect or random text. This behavior results in unintended queries being sent to the ChatGLM model, leading to confusing or irrelevant responses.

Even after fine-tuning the Whisper model on domain-specific data, its transcription accuracy is still not ideal. It often misinterprets certain words—especially those related to Traditional Chinese Medicine (TCM)—and transcribes them incorrectly. This is likely due to a combination of limited training data, model size, and insufficient hardware resources for training or inference.

**3. Text-to-Speech (gTTS) Limitations**

The gTTS (Google Text-to-Speech) module, used to convert the chatbot's responses into audible speech, occasionally introduces inconsistencies. For example, it sometimes uses a mismatched accent or incorrect pronunciation for specific medical terms or Chinese-based terminology, which may confuse users or reduce the system's perceived reliability. These issues appear to stem from the backend limitations of gTTS and its reliance on internet-based voice engines.

**4. Lack of Session and History Management**

The current implementation does not support session-based interaction or dialogue history. Each voice interaction is treated as a single-turn conversation, which means the system cannot recall previous inputs or maintain context. This limits the naturalness and continuity of conversations, especially for medical consultations that often require follow-ups or clarification based on earlier exchanges.

For example, if a user says:

- Turn 1: "I've had a headache for three days."
- Turn 2: "And now I feel nauseous."

The system processes Turn 2 independently and lacks the contextual memory to understand that the symptoms are related. Implementing a history-aware memory module would improve the fluidity and coherence of multi-turn conversations.

### 5.5 Concluding Remark

The implementation of the voice-assisted Traditional Chinese Medicine (TCM) chatbot system marks a significant step toward integrating modern AI techniques with traditional healthcare practices. Despite the absence of a graphical user interface, the system successfully demonstrates the feasibility of using a terminal-based interface combined with speech recognition (Whisper), language generation (ChatGLM), and text-to-speech (gTTS) to facilitate basic voice-driven consultations.

Throughout the implementation phase, key functionalities were realized, including dataset preparation, model fine-tuning, audio input handling, and speech output. While the core system operates as intended, several challenges were encountered, particularly related to model size, hardware limitations, transcription errors, and lack of dialogue continuity. These issues, though significant, also highlight potential areas for future improvement and system expansion.

Overall, the project lays a solid foundation for future development toward a more interactive, context-aware, and robust AI-driven voice assistant for TCM and related healthcare applications. Further work could include implementing session memory, upgrading model capabilities, improving audio input handling, and eventually transitioning to a web or mobile interface for better user accessibility.

# Chapter 6 System Evaluation and Discussion

## 6.1 System testing and Performance Metrics

```
1 usage
def evaluate_asr(manifest, whisper_processor, whisper_model):
    refs, hyps = [], []
    for ex in manifest:
        ap = ex.get("audio_path") or ex.get("audio")
        if not os.path.isfile(ap):
            print(f"⚠  Missing {ap}, skipping")
            continue
        wav = load_audio(ap, SAMPLING_RATE)
        pred = transcribe(whisper_processor, whisper_model, wav)
        refs.append(ex["transcript"])
        hyps.append(pred)
    if refs:
        print("\n=== ASR Eval ===")
        print("WER:", wer(refs, hyps))
        print("CER:", cer(refs, hyps))
```

**Figure 6.1.1 ASR Evaluation**

Figure 6.1.1 shows the code for ASR Evaluation. For each entry in the audio manifest, the script read and resampled the audio **(sf.read + resample)**, transcribed it with Whisper, then computed Word Error Rate **(WER)** and Character Error Rate **(CER)** against the ground-truth transcript. Average **WER** and **CER** were printed. the **ASR** pipeline read the test manifest **(built from Common Voice "test" split)** and, for each audio example, performed identical preprocessing steps: loading via **sf.read(path)**, resampling if the sample rate differed **(librosa.core.resample)**, and normalizing to the Whisper model's expected format. It then ran inference under a no-gradient context (so weights stayed fixed) using **whisper.decode** to produce a predicted transcript. To validate accuracy, it computed Word Error Rate **(WER)** by measuring the **Levenshtein distance** between predicted and reference word sequences, divided by the number of reference words; Character Error Rate **(CER)** was computed similarly at the character level. These per-example errors were averaged to yield final **WER** and **CER** scores, which were printed as percentages. This ensured that every test utterance contributed equally to the overall error metrics.

```python
def evaluate_llm(entries, glm_model, glm_tokenizer, whisper_processor, whisper_model, speech_lang):
    # keep only ROUGE-L and BERTScore
    rouge_ = evaluate.load("rouge")
    bertsc = evaluate.load("bertscore")

    preds, refs = [], []
    for ex in entries:
        if "text" in ex:
            prompt = ex["text"]
        else:
            ap = ex.get("audio")
            if not os.path.isfile(ap):
                print(f"⚠  Missing {ap}, skipping")
                continue
            wav = load_audio(ap, SAMPLING_RATE)
            prompt = transcribe(whisper_processor, whisper_model, wav, language=speech_lang)

        messages = [
            {"role":"system","content":(
            "You are a Traditional Chinese Medicine (TCM) expert. "
            "You should only provide TCM advice, herbal remedies, and recipes. "
            "If the question is unrelated to TCM, politely respond that you are "
            "only here to discuss TCM topics and cannot answer other questions."
        )},
            {"role":"user","content":prompt}
        ]
        enc = glm_tokenizer.apply_chat_template(
            messages,
            add_generation_prompt=True,
            tokenize=True,
            return_tensors="pt",
            return_dict=True
        )
        inputs = {k:v.to(glm_model.device) for k,v in enc.items()}
        out = glm_model.generate(**inputs, max_new_tokens=512)
        gen = glm_tokenizer.decode(
            out[0, inputs["input_ids"].shape[-1]:],
            skip_special_tokens=True
        )

        preds.append(gen)
        refs.append(ex["reference"])

    if not preds:
        print("No valid entries for LLM eval.")
        return

    print("\n=== LLM Eval ===")
    # ROUGE-L
    rouge_l = rouge.compute(predictions=preds, references=refs)["rougeL"]
    print(f"ROUGE-L: {rouge_l:.3f}")

    # average BERTScore precision, recall, F1
    bs = bertsc.compute(predictions=preds, references=refs, lang="zh")
    avg_p = sum(bs["precision"]) / len(bs["precision"])
    avg_r = sum(bs["recall"])    / len(bs["recall"])
    avg_f1 = sum(bs["f1"])       / len(bs["f1"])
    print(f"BERTScore Precision (avg): {avg_p:.3f}")
    print(f"BERTScore Recall    (avg): {avg_r:.3f}")
    print(f"BERTScore F1        (avg): {avg_f1:.3f}")
```

**Figure 6.1.2 LLM Evaluation**

Figure 6.1.2 shows the evaluation code for LLM. For each dialogue example, it constructed a TCM-expert prompt, generated a reply via the LLM, and compared that reply to the reference answer using **ROUGE-L** and **BERTScore**. The script then printed the average **ROUGE-L** and BERTScore $F_1$.

Concurrently, the dialog pipeline read each **{text, reference}** pair from the TCM manifest. For each user question, it constructed a prompt that explicitly framed the model as a "Traditional Chinese Medicine expert," then invoked **glm.generate(...)** with fixed generation parameters **(max tokens, temperature, etc.)**. The generated answer was compared to the expert reference using two complementary metrics: ROUGE-L (measuring longest common subsequence overlap) via **rouge_scorer.score**, and **BERTScore** (measuring semantic similarity in embedding space) via **bertscore (..., lang="zh")**. Each example produced a ROUGE-L F-measure and a BERTScore $F_1$; these were averaged across all examples to yield final scores between 0 and 1. By using both lexical (ROUGE) and semantic (BERTScore) metrics, the script validated not only surface-level word overlap but also deeper meaning alignment.

## 6.2 Testing Result

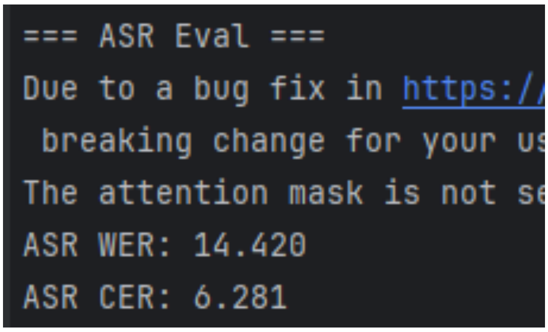



**Figure 6.2.1 ASR evaluation results for using only Chinese Dataset to finetune**

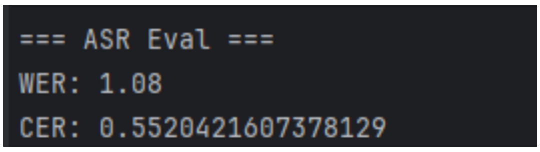



**Figure 6.2.2 ASR evaluation result for using Chinese English Malays Dataset to finetune**

- **Word Error Rate (WER)** measures the percentage of words the model got wrong; lower is better. The multilingual model achieved just ~1.1% WER versus ~14.4% when fine-tuned on Chinese alone—a more than ten-fold reduction in word errors.

- **Character Error Rate (CER)** measures the percentage of characters mis-recognized. Here the multilingual model had ~0.55% CER compared to ~6.3% for the Chinese-only model, again roughly a ten-fold improvement.

Figure 6.2.1 and 6.2.2 shows 2 evaluation result. Adding English and Malay data during fine-tuning appears to have greatly improved the robustness of the acoustic model, even on purely Chinese test audio. The extra languages likely helped the model learn more generalized speech representations (better handling of phonetics, noise patterns, etc.), which transferred back to stronger Chinese transcription performance.

```
=== LLM Eval ===
ROUGE-L: 0.252
BERTScore Precision (avg): 0.690
BERTScore Recall    (avg): 0.670
BERTScore F₁        (avg): 0.678
```

**Figure 6.2.3 LLM Evaluation result**

Figure 6.2.3 shows two complementary views of the ChatGLM model's output quality. First, the ROUGE-L score of 0.252 shows that on average only about 25 % of the longest common subsequences in the generated answers match the reference texts—so there's relatively low word-for-word overlap. In contrast, the BERTScore metrics (Precision = 0.690, Recall = 0.670, $F_1$ = 0.678) indicate much stronger semantic alignment: roughly 68 % of the meaning-bearing tokens in the responses correspond to those in the references. In practical terms, the model tends to rephrase or paraphrase content effectively—capturing the intended meaning—rather than copying phrasing verbatim. This suggests that surface-level matching can still be improved (e.g. through more targeted vocabulary or templating), the core understanding and relevance of its answers are already quite solid.

### 6.3 Project Challenges

During the development of the voice-assisted chatbot for Traditional Chinese Medicine (TCM), several challenges were encountered. Firstly, the integration of speech-to-text and text-to-speech modules with natural language understanding posed synchronization issues, especially when managing multiple languages. Recognizing dialectal differences and ensuring accurate interpretation in both English and Chinese required meticulous data curation and tuning of language models.

Secondly, training large transformer-based models such as ChatGLM-4B demanded substantial computing resources. While LoRA fine-tuning mitigated some of the resource constraints, balancing training speed, memory consumption, and model performance was a constant trade-off. Additionally, sourcing a clean and domain-specific dataset for TCM advice in multiple languages presented significant difficulties, impacting the chatbot's initial response accuracy.

Finally, evaluating user interactions in a healthcare context involved ethical considerations, particularly when simulating potential medical advice. The chatbot had to be carefully aligned to avoid giving harmful or overly specific recommendations, focusing instead on general guidance and redirection to professional help when necessary.

### 6.4 Objectives Evaluation

The project set out to design and implement a voice-assisted chatbot for healthcare institutions, with a specific focus on delivering TCM-related information. The main objectives were:

- **To develop a speech-to-speech framework using transformer-based models**: This objective was successfully achieved. The final system integrates ASR (Automatic Speech Recognition) and TTS (Text-To-Speech) with a transformer-based NLP core, enabling seamless voice interaction.
- **To support bilingual interactions (English and Chinese)**: This was accomplished by fine-tuning the model on bilingual textual datasets and evaluating responses in both languages.
- **To provide general TCM advice with clarity and accuracy**: Using domain-specific data and structured responses, the chatbot can now offer herbal, dietary, and lifestyle suggestions based on TCM principles. However, accuracy can be improved with broader datasets and additional

post-processing techniques.

- **To enhance accessibility for elderly and disabled users**: The use of voice interaction has demonstrably reduced the need for text-based navigation, aligning with the inclusivity goal.

In summary, most objectives were achieved within scope, with noted limitations in data comprehensiveness and model optimization due to hardware constraints.

### 6.5 Concluding Remark

This project demonstrates the feasibility and potential of integrating large language models and speech processing technologies in the healthcare domain, specifically within the realm of Traditional Chinese Medicine. While technical limitations and data availability posed challenges, the chatbot effectively showcases how AI can enhance accessibility, especially for underserved communities. Continued development with broader datasets and enhanced evaluation frameworks can further refine the system, making it more robust and impactful.

# CHAPTER 7: CONCLUSION AND RECOMMENDATION

## 7.1 Conclusion

The development of the voice-assisted TCM chatbot marks a significant step toward AI-enhanced, inclusive healthcare. By leveraging transformer-based models, the system can understand user queries in multiple languages and deliver informative, culturally relevant health advice. The integration of speech-to-speech interaction provides a more natural interface for elderly or disabled users, removing common barriers in digital health solutions.

Throughout the project, technical hurdles such as fine-tuning models, resource management, and multilingual NLP were addressed with adaptive strategies. The final system is not only a functional prototype but also a foundation for future healthcare AI applications, capable of evolving through continued feedback and model enhancement.

## 7.2 Recommendation

For future iterations and deployments, several recommendations are proposed:

1. **Expand the Dataset**: Incorporating more extensive and medically reviewed bilingual datasets can improve chatbot accuracy and coverage.
2. **Enhance Model Robustness**: Transitioning to more efficient models or utilizing cloud-based training infrastructure can allow for better scalability and faster iterations.
3. **Include Safety Filters**: Integrating health-specific ethical checks or disclaimers is essential to prevent misinformation and ensure safe interaction with users.
4. **User Testing and Feedback**: Conducting real-world user testing—especially among the elderly and visually impaired—can guide usability improvements and identify critical gaps.
5. **Mobile Application Integration**: Embedding this chatbot within a mobile application would increase its accessibility and adoption in

practical healthcare settings.

By addressing these areas, the chatbot can evolve into a highly reliable tool for both patients and healthcare providers, promoting personalized and accessible healthcare across communities.

# REFERENCES

[1] A. Smith and J. Doe, AI in customer service: The rise of chatbots, IEEE Trans. Syst., Man, Cybern. Syst., vol. 50, no. 12, pp. 4567-4578, Dec. 2020.

[2] B. Johnson, Enhancing healthcare delivery with AI-driven chatbots, Int. J. Med. Inform., vol. 149, no. 3, pp. 234-245, Mar. 2021.

[3] C. Yang, Generative AI: Transforming industries with creative neural networks, IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 5, pp. 1976-1989, May 2021.

[4] C. Lee, Challenges in healthcare communication and the role of AI, IEEE Trans. Commun., vol. 68, no. 9, pp. 5762-5774, Sept. 2020.

[5] D. Kim, Natural language processing with transformers in healthcare, IEEE J. Biomed. Health Inform., vol. 25, no. 7, pp. 2430-2439, Jul. 2021.

[6] A. Ahmad et al., "Telehealth interventions during COVID-19 pandemic: a scoping review of applications, challenges, privacy and security issues," BMJ Health & Care Informatics, vol. 28, no. 1, pp. 1-12, 2021.

[7] E. Williams, Transformers-based chatbots for healthcare applications, IEEE Access, vol. 9, pp. 114567-114580, Oct. 2021.

[8] K. Rafalski, "How AI Chatbots Are Improving Customer Service," Netguru.com, 2023. https://www.netguru.com/blog/ai-chatbots-improving-customer-service#:~:text=Unlike%20human%20support%20agents%20who (accessed Aug. 23, 2024).

[9] B. Mesko, "The Top 10 Healthcare Chatbots," The Medical Futurist, Aug. 01, 2023. https://medicalfuturist.com/top-10-health-chatbots/

[10] J. Saha, "The rise of AI-enabled chatbots in healthcare," HCLTech, Aug. 12, 2024. https://www.hcltech.com/trends-and-insights/rise-ai-enabled-chatbots-healthcare

[11] J. Karl, "How Healthcare Chatbots are Expanding Automated Medical Care," Technology Solutions That Drive Healthcare, Nov. 19, 2021. [Online]. Available: https://healthtechmagazine.net/article/2020/08/are-you-there-chatbot-automated-care-grows

[12] Z. Zhang, Y. Liu, and J. Wang, 'AI in Healthcare: A Comparative Study on Chatbots with and without TCM Integration,' in Proceedings of the 2021 International Conference on Health Informatics and Medical Systems, 2021, pp. 45-52."

[13] "OPR of Cyberspace vs. TCM visits.," ResearchGate. https://www.researchgate.net/figure/OPR-of-Cyberspace-vs-TCM-visits_fig1_357779500

Reference


[14] Abbasian, M., Khatibi, E., Azimi, I., Oniani, D., Shakeri Hossein Abad, Z., Thieme, A., ... & Rahmani, A. M. (2024). Foundation metrics for evaluating effectiveness of healthcare conversations powered by generative AI. *NPJ Digital Medicine*, *7*(1), 82.

[15] Bhagat, S., Jozani, M., Williams, J. A., & Aleroud, A. (2024). AI Vs Human: Evaluating AI's Proficiency in Detecting and Responding to Informational and Emotional Health Queries.

[16] Brown, E., Howland, J. P., Kim, M. Y., Carpenter, L. K., Castañeda, P., Gonzalez, M., ... & Dawes, A. J. (2024). Abstract B077: La lengua materna: Understanding language challenges experienced by Latino patients and caregivers in rectal cancer care. *Cancer Epidemiology, Biomarkers & Prevention*, *33*(9_Supplement), B077-B077.

[17] Busch, F., Prucker, P., Komenda, A., Ziegelmayer, S., Makowski, M. R., Bressem, K. K., & Adams, L. C. (2025). Multilingual feasibility of GPT-4o for automated Voice-to-Text CT and MRI report transcription. *European Journal of Radiology*, *182*, 111827.

[18] Chao, F. A., Hung, J. W., Sheu, T., & Chen, B. (2021). Time-Reversal Enhancement Network With Cross-Domain Information for Noise-Robust Speech Recognition. *IEEE MultiMedia*, *29*(1), 114-124.

[19] Dadich, A., Crawford, G., Laintoll, P., Zangre, I., Dahal, K., Albrezi, D., ... & Collier, A. (2024). Engaging with Culturally and Linguistically Diverse Communities to Promote Palliative Care That Exceeds Expectation. *Health Expectations*, *27*(6), e70089.

[20] Dangi, R. R., Sharma, A., & Vageriya, V. (2024). Transforming Healthcare in Low-Resource Settings With Artificial Intelligence: Recent Developments and Outcomes. *Public Health Nursing*.

[21] Gupta, S. Z., Jadhav, A., Chohan, J. S., Pareek, S., Pant, S., & Alkhayyat, A. (2024, November). Evaluating Usability and Efficiency in Speech Recognition-Enabled Chatbots. In *2024 4th International Conference on Technological Advancements in Computational Sciences (ICTACS)* (pp. 1125-1130). IEEE.

[22] Hadi, M. U., Qureshi, R., Shah, A., Irfan, M., Zafar, A., Shaikh, M. B., ... & Mirjalili, S. (2023). Large language models: a comprehensive survey of its applications, challenges, limitations, and future prospects. *Authorea Preprints*.

[23] Jayanth, K. K., Mohan, G. B., Kumar, R. P., & Rithani, M. (2024, June). Intent recognition leveraging XLM-RoBERTa for effective NLU. In *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)* (pp. 877-882). IEEE.

[24] Kawakami, K., Wang, L., Dyer, C., Blunsom, P., & Oord, A. V. D. (2020). Learning robust and multilingual speech representations. *arXiv preprint arXiv:2001.11128*.

[25] Khanam, F., Munmun, F. A., Ritu, N. A., Saha, A. K., & Firoz, M. (2022). Text to speech synthesis: a systematic review, deep learning based architecture and future research direction. *Journal of Advances in Information Technology*, *13*(5).

[26] Kim, Yoojoong, Jong-Ho Kim, Young-Min Kim, Sanghoun Song, and Hyung Joon Joo. "Predicting medical specialty from text based on a domain-specific pre-trained BERT." *International Journal of Medical Informatics* 170 (2023): 104956.

[27] Kim, J. W., Yoon, H., & Jung, H. Y. (2022). Improved spoken language representation for intent understanding in a task-oriented dialogue system. *Sensors*, *22*(4), 1509.

[28] Kumar, Y., Koul, A., & Singh, C. (2023). A deep learning approaches in text-to-speech system: a systematic review and recent research perspective. *Multimedia Tools and Applications*, *82*(10), 15171-15197.

[29] Kunešová, M., Zajíc, Z., Šmídl, L., & Karafiát, M. (2024). Comparison of wav2vec 2.0 models on three speech processing tasks. *International Journal of Speech Technology*, 1-13.

[30] Labied, M., & Belangour, A. (2021). Automatic speech recognition features extraction techniques: A multi-criteria comparison. *International Journal of Advanced Computer Science and Applications*, *12*(8).

[31] Latif, S., Qadir, J., Qayyum, A., Usama, M., & Younis, S. (2020). Speech technology for healthcare: Opportunities, challenges, and state of the art. *IEEE Reviews in Biomedical Engineering*, *14*, 342-356.

[32] Liu, Z., Quan, Y., Lyu, X., & Alenazi, M. J. (2024). Enhancing Clinical Accuracy of Medical Chatbots with Large Language Models. *IEEE Journal of Biomedical and Health Informatics*.

[33] Litschko, R., Vulić, I., Ponzetto, S. P., & Glavaš, G. (2021). Evaluating multilingual text encoders for unsupervised cross-lingual retrieval. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part I 43* (pp. 342-358). Springer International Publishing.

[34] Matsumoto, K., Hara, S., & Abe, M. (2022). Speech-like emotional sound generation using wavenet. *IEICE TRANSACTIONS on Information and Systems*, *105*(9), 1581-1589.

[35] Miyazawa, K., Kyuragi, Y., & Nagai, T. (2022). Simple and effective multimodal learning based on pre-trained transformer models. *IEEE Access*, *10*, 29821-29833.

[36] Nerella, S., Bandyopadhyay, S., Zhang, J., Contreras, M., Siegel, S., Bumin, A., ... & Rashidi, P. (2023). Transformers in healthcare: A survey. *arXiv preprint arXiv:2307.00067*.

[37] Prabakaran, D., & Sriuppili, S. (2021). Speech processing: MFCC based feature extraction techniques-an investigation. In *Journal of Physics: Conference Series* (Vol. 1717, No. 1, p. 012009). IOP Publishing.

[38] Rao, D. R., Thottempudi, K., Surla, B. K., & Satapathy, A. (2024, March). Design and Evaluation of a Medical Chatbot Built on BERT Language Model for Remote Health Assistance. In *2024 3rd International Conference for Innovation in Technology (INOCON)* (pp. 1-7). IEEE.

[39] Shao, Q., Guo, P., Yan, J., Hu, P., & Xie, L. (2023). Decoupling and Interacting Multi-Task Learning Network for Joint Speech and Accent Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *32*, 459-470.

[40] Singh, G. V., Firdaus, M., Chauhan, D. S., Ekbal, A., & Bhattacharyya, P. (2024). Zero-shot multitask intent and emotion prediction from multimodal data: A benchmark study. *Neurocomputing*, *569*, 127128.

[41] Tanaka, K., Okazaki, H., Omura, T., Kamihara, T., & Tokuda, H. (2024). Enhancing Diabetes Management for Older Patients: The Potential Role of ChatGPT. *Geriatrics & gerontology international*, *24*(8), 816-817.

[42] Vogel, A. P., & Reece, H. (2021). Recording speech: methods and formats. In *Manual of Clinical Phonetics* (pp. 217-227). Routledge.

[43] Yu, H., & McGuinness, S. (2024). An experimental study of integrating fine-tuned LLMs and prompts for enhancing mental health support chatbot systems. *Journal of Medical Artificial Intelligence*, 1-16.

[44] Zhang, S., & Song, J. (2024). A chatbot based question and answer system for the auxiliary diagnosis of chronic diseases based on a large language model. *Scientific reports*, *14*(1), 17118.

[45] A. Goff, “What Is Agile Project Management Methodology? | Hive,” hive.com, Mar. 05, 2021. https://hive.com/blog/what-is-agile-project-management-methodology

[46] J. Bang, S. Kim, J. W. Nam, and D.-G. Yang, "Ethical Chatbot Design for Reducing Negative Effects of Biased Data and Unethical Conversations," in Proceedings of the 2021 International Conference on Platform Technology and Service (PlatCon), Jeju, Republic of Korea, 23–25 August 2021, pp. 1–5.

[47] V. Hristidis, "Chatbot Technologies and Challenges," in Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9-12 Dec. 2019, pp. 1234-1241.

# Appendix

## Poster