English to Cantonese Speech Translator

Creator: oscarhelmet@github

Table of Content

Problem definition

The project aims to develop an audio-to-audio translation tool specifically for the Cantonese language, which is prevalent in regions of Southern China, including Hong Kong and Macau, and also spoken in diaspora communities in Malaysia, Singapore, and other parts of the world. Unlike existing text-based translation services, this tool emphasises ease of use and accessibility by allowing users to input their speech directly. The translator will capture spoken phrases or sentences in Cantonese and output the translated audio in the desired target language.

This technology is primarily designed for travellers and expatriates who may encounter language barriers in Cantonese-speaking regions. It is also valuable for students, linguists, business professionals, and anyone interested in cross-cultural communication. The tool's focus on audio input and output makes it particularly useful for individuals who may find it challenging to read or write in Cantonese due to its complex script. The significance of the Cantonese Audio-to-Audio Translator project can be summarized in several key points:

- **Ease of Communication:** The tool will facilitate smoother and more efficient communication between Cantonese speakers and non-speakers, reducing misunderstandings and fostering better interactions.

- **Travel and Hospitality:** The translator can greatly enhance the travel experience for tourists in Cantonese-speaking regions, allowing them to navigate with greater confidence and independence.

- **Accessibility:** By using audio as both input and output, the tool is more accessible to users who may have difficulty with reading or typing, including the visually impaired or those unfamiliar with the written form of the language.

System architecture

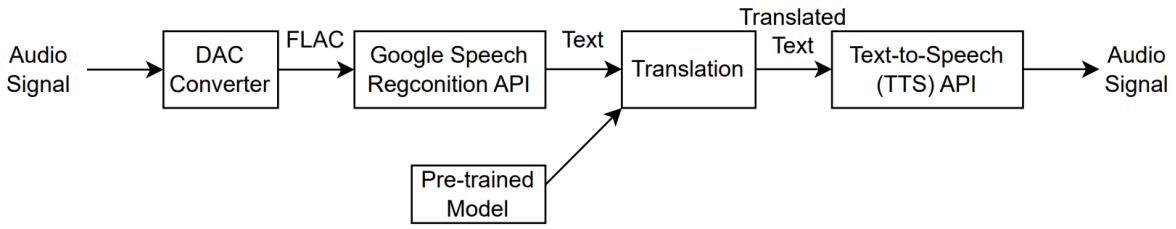

Figure 1 System Diagram of the Proposed

The process begins with the user speaking in Cantonese. This audio signal is an analog wave that needs to be converted to a digital format by the analog-to-digital converter (DAC), which converts it into a digital signal that can be processed by computer systems. The digital audio is then encoded into the FLAC (Free Lossless Audio Codec) format. FLAC is used here for its lossless compression qualities, ensuring high fidelity without taking up as much space as uncompressed formats.

The FLAC file is sent to Google Speech Recognition API, which is a pre-trained model capable of recognizing and transcribing speech. This step converts the spoken English audio into text. The transcribed English text is then input into a pre-trained translation model. This model translates the text from English to the Cantonese. Once the text is translated, it is fed into a Text-to-Speech (TTS) API that synthesizes speech from the translated text. This step converts the translated text back into an audio signal in the target language. The final output is an audio signal of the translated text, which can be played back to the user. This audio signal is digital and can be output through speakers or headphones.

Training and Retraining Components:
Google Speech Recognition API: This component is a pre-trained model developed by Google. It is not re-trained as part of this project; instead, it's utilized as a service.

Translation (Pre-trained Model): Like the speech recognition API, this is also a pre-trained model. In this project mT5 (multilingual Text-to-Text Transfer Transformer) translation model is employed [1]. There may be options to re-train or fine-tune this model on specific Cantonese datasets to improve accuracy for this language, which is less commonly supported than many others.

Text-to-Speech (TTS) API: This component typically uses a pre-trained model. Most TTS services provide a range of voices and languages but do not usually allow for retraining.

Libraries Used: torch, speech_recognition, google.cloud texttospeech, pygame, flask, numpy, matplotlib, time, pandas, datasets, transformers, tqdm, seaborn

Data source

Speech Recognition Component

- **Library Used**: Python Speech Recognition library
- **Recognition Service**: Google Speech Recognition
- **Purpose**: To transcribe spoken words into written text for further processing.

Translation Model (Self-trained)

- **Model**: mT5 (multilingual T5)
- **Dataset**: ALT (Asian Language Treebank) Dataset
- **Training Data**: 18,089 samples
- **Test Data**: 1,020 samples
- **Languages**: Simplified Chinese and English (out of the 13 available in the dataset)
- **Dataset Source**: ALT Dataset on Hugging Face
- **Features**:
    - **Input**: Text in English
    - **Output**: Corresponding translation in Simplified Chinese

Chinese to Cantonese Translation (Pre-trained)

- **Model**: mT5 (multilingual T5)
- **Dataset**: botisan-ai/cantonese-mandarin-translations
- **Training Data:** 24.2k smaples
- **Purpose**: To translate text from Simplified Chinese to Cantonese.
- **Model Source**: Hugging Face Model Repository
- **Dataset Source:** Cantonese-Mandarin-Translations
- **Features**:
    - **Input**: Text in Simplified Chinese
    - **Output**: Corresponding translation in Cantonese

Text-to-Speech (TTS) Engine

- **Provider**: Google Cloud Text-to-Speech API
- **Purpose**: To convert translated text into spoken audio.
- **Features**:
    - Supports multiple languages and dialects.
    - Offers various voices and speaking styles

Data preprocessing

During the data preprocessing phase, a multilingual dataset is fetched and subsequently loaded into a DataFrame for manipulation and analysis. This dataset is characterized by the presence of several key columns: 'SNT.URLID', 'SNT.URLID.SNTID', 'url', and 'translation'. Notably, the 'translation' column is of particular importance for the training process, as it encompasses translations of each entry across 13 distinct languages.

The dataset is partitioned into two subsets: one for training and the other for testing. The training subset consists of 18,089 entries, while the test subset comprises 1,020 entries. To facilitate ease of access and future processing, these subsets are exported into two separate files. The training data is saved in a file named 'train_dataset.tsv', and the test data in 'test_dataset.tsv'.

Both files are formatted as TSV (Tab-Separated Values), a simple text format for storing tabular data. Each row in the TSV files corresponds to an entry in the dataset, with individual fields (SNT.URLID, SNT.URLID.SNTID, url, and translation) separated by tabs. TSV is often preferred in data processing tasks due to its compatibility with many data analysis tools and its human-readable format, which is straightforward to edit and inspect.

The export process involves writing the DataFrames to disk using a delimiter of a single tab character to distinguish between columns. This ensures that each cell in the DataFrame corresponds to a distinct field in the TSV file. The index is typically excluded from this export to prevent the addition of an unwanted column of row identifiers in the TSV files, which should only contain the actual data from the DataFrame.

By following these detailed steps, the data preprocessing phase sets a solid foundation for the model training phase, ensuring that the input data is of high quality and in the correct format. The dataset is initially downloaded from a huggingface [2]. Once the dataset is downloaded, it is loaded into a pandas DataFrame. After loading the data, it's crucial to perform an initial inspection to understand the dataset's structure, identify any missing values, and spot potential inconsistencies. If any issues are identified during the inspection phase, data cleaning is performed. With a clean and structured DataFrame, the data is then written to TSV files. The pandas library's to_csv function is employed for this purpose, setting the sep parameter to '\t' to indicate tab separation and index parameter to False to exclude row indices from the file. The TSV files are saved to the local file system with appropriate names ('train_dataset.tsv' for the training set, 'test_dataset.tsv' for the test set) to ensure they are easily distinguishable and accessible for subsequent training and testing phases.

{'bg': 'সিডনির র\u200dযান্ডউইক রেসের মাঠের আটটি থরোব্রেড রেসের ঘোড়ার এক্যুইন ইনফ্লুয়েঞ্জাতে আক্রান্ত হওয়ার খবরটি নিশ্চিত করা হয়েছে।', 'en': 'It has been confirmed that eight thoroughbred race horses at Randwick Racecourse in Sydney have been infected with equine influenza.', 'en_tok': 'It has been confirmed that eight thoroughbred race horses at Randwick Racecourse in Sydney have been infected with equine influenza .', 'fil': 'Kumpirmado na walong magandang lahing pangkarerang kabayo sa Randwick Racecourse sa Sydney ang nagkaroon ng trangkaso.', 'hi': 'यह पुष्टि की गई है कि सिडनी में रैंडविक रेसकोर्स के आठ घनी नस्ल के घोड़े इक्वाइन इन्फ्लूएंजा से संक्रमित हैं।', 'id': 'Telah dikonfirmasi bahwa 8 kuda pacu thoroughbred di pacuan kuda Randwick di Sydney telah terinfeksi oleh flu kuda.', 'ja': 'シドニーのランドウィック競馬場の 8 頭のサラブレッド競走馬が馬インフルエンザに感染していることが確認された。', 'khm': 'គេបានបញ្ជាក់ថាសេះពូជប្រណាំងទាំង៨នៅទីលានប្រណាំងសេះរង់វីក \u200bទីក្រុងស៊ីដនីបានឆ្លងរោគផ្តាសាយធំរបស់សត្វសេះ។', 'lo':

'ມີການຢັ້ງຢືນແລ້ວວ່າ ມ້າແຂ່ງພັນແທ້ແປດໂຕ ທີ່ ແຂນວີກ ເລັ່ສຄອສ ໃນ ຊິດນີ ໄດ້ຕິດເຊື້ອໄຂ້ຫວັດໃຫຍ່ມ້າ.', 'ms': 'Telah disahkan bahawa lapan kuda lumba thoroughbred di Tapak Lumba Randwick di Sydney telah dijangkiti influenza kuda.', 'my': 'ဆစ်ဒနီ က ရန့်ဝစ်(ခ်) မြင်းပြိုင်ကွင်း မှ မျိုးသန့် ပြိုင်မြင်း ရှစ်ကောင် ဟာ မြင်းတုတ်ကွေးရောဂါ ကူးစက်ခံခဲ့ရတယ် ဆိုတာ အတည်ပြုခဲ့ပါတယ် ။', 'th': 'ได้เป็นที่ยืนยันแน่นอนแล้วว่าม้าแข่งพันธุ์ดีแปดตัวที่สนามแข่งแรนด์วิค ในเมืองซิดนีย์ได้ติดเชื้อไข้หวัดใหญ่ม้า', 'vi': 'Đã có thông tin khẳng định rằng tám chú ngựa đua thuần chủng tại Trường đua Randwick ở Sydney đã bị nhiễm cúm ngựa.', 'zh': '已经证实，悉尼兰德威克赛马场的 8 匹纯种赛马已经感染了马流感。'}

Table 1. Example of one of the *translation* Column

Data analysis

Specifically, the project leverage the mT5-base model from Google, which is a part of the T5 (Text-to-Text Transfer Transformer) family, fine-tuned for multilingual translation tasks. This model is capable of understanding and generating text in multiple languages, making it ideal for translation tasks.

The mT5-base model is accessed through the Hugging Face transformers library, which provides a convenient API for both the tokenizer and the model itself. The tokenizer is responsible for converting text into a format that the model can process (i.e., token ids), and it is extended to include special tokens for different languages to give the model context about the input and target languages.

The model is trained using PyTorch, a popular deep learning framework that provides automatic differentiation capabilities and a wide array of tools for building and training neural networks. The training process is performed on a GPU, as indicated by the .cuda() calls, to accelerate the computation.

Training Procedure

The training procedure involves several steps:
Encoding the Input Text: The input text is prepended with a special token indicating the target language and then tokenized.
Encoding the Target Text: The target text is tokenized without a target language token since the model is expected to generate this text.
Batch Transformation: Batches of data are transformed and loaded onto the GPU for efficient training.
Model Training: The model undergoes a series of epochs where it learns to translate from the input language to the target language. During training, the AdamW optimizer is used, which is a variant of the Adam optimizer with improved regularization. Learning rate scheduling is also applied to adjust the learning rate throughout training, starting with a warm-up period.
Evaluation: After certain steps, the model is evaluated on a test dataset to gauge its performance on unseen data. The loss on this dataset provides an indication of the model's generalization capabilities.

Metrics Used for Assessment

The primary metric used for assessing the model is the loss function provided by the model's forward pass, which in the case of Seq2Seq models is typically a form of cross-entropy loss between the predicted sequence and the actual target sequence. This loss is a direct measure of how well the model's predictions match the expected output.
During training, we record the loss after each batch and report the average loss at regular intervals to monitor the model's progress. Additionally, the model is evaluated on a test set, and the loss is computed to assess the model's generalization.

Visualizations

To visualize the model's performance over time, we use the matplotlib library to create a plot of the smoothed losses. The smoothing is performed by taking a moving average of the losses, which reduces the noise and allows us to see the trends more clearly.
The loss is plotted against the number of epochs to show how the model's performance improves as it learns from the training data. This visualization is crucial for understanding the training dynamics and for making decisions regarding training duration, early stopping, or adjusting hyperparameters.

Deployment and results

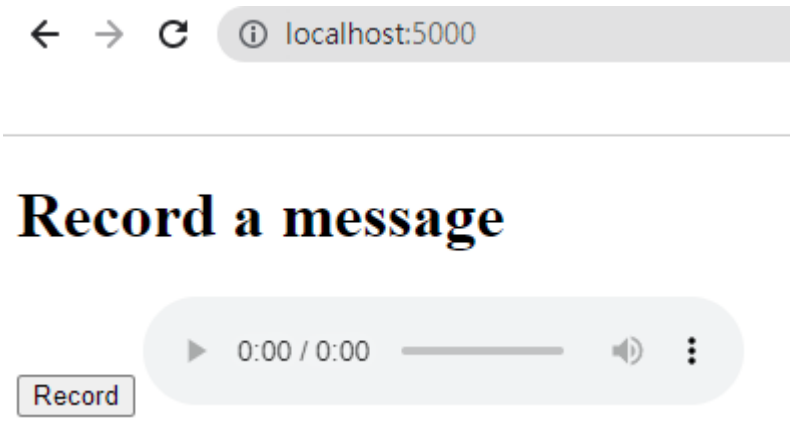

Figure 2 Screenshot of the Application

Example Deployment and Results Document:

User Interface of the Product:
To showcase the user interface of the Flask application, you would include screenshots demonstrating the key features. For instance:

1. **Voice Recognition Feature:** A screenshot of the page or section where users can record their voice for recognition after the Record button is pressed.

2. **Translation Feature:** A screenshot displaying the interface for translating text, with fields for input text and the translation result.

3. **Text-to-Speech (TTS):** Feature: A screenshot showing the TTS interface where users can type text to be converted into speech.

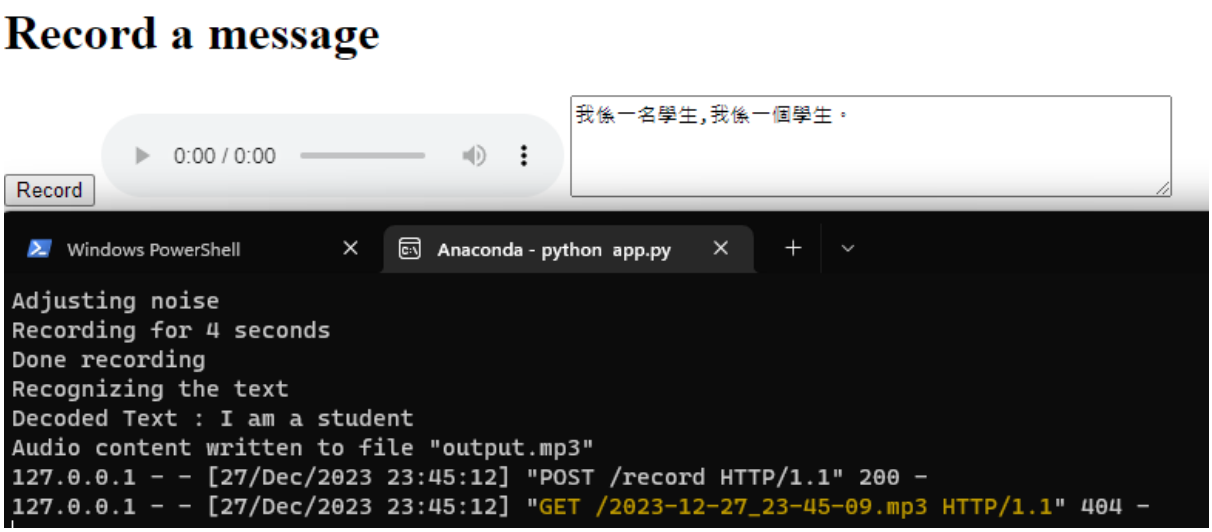

Figure 3 Test Result of the Application

How to Execute the Product:

Execute the app.py in Anaconda Terminal and wait until the initialisation of all models. After loading the models, the web application is listening on localhost port 5000.

Highlighted Outcomes:
1. Real-time Voice Recognition: From Figure 3, the real-time translation will be both shown and playback in audio.

2. Multiple Translation: To address different ambiguity in semantics of a sentence, two or three sentences will be outputted.

References

[1] L. Xue *et al.*, ‘mT5: A massively multilingual pre-trained text-to-text transformer’, *CoRR*, vol. abs/2010.11934, 2020.

[2] H. Riza *et al.*, "Introduction of the Asian Language Treebank," *2016 Conference of The Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment Techniques (O-COCOSDA)*, Bali, Indonesia, 2016, pp. 1-6, doi: 10.1109/ICSDA.2016.7918974.