Evolution in Electrical and Electronic Engineering Vol. 3 No. 1 (2022) 114-124

EEEE

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/eeee
e-ISSN : 2756-8458

Development of Malay Speech Translation System using Python

Saw Zi Jian[1], Chua King Lee[1]*

[1]Departmant of Electronic Engineering, Faculty of Electrical and Electronic Engineering,
Universiti Tun Hussein Onn Malaysia, Batu Pahat, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/eeee.2022.03.01.014
Received 27 January 2022; Accepted 24 February 2022; Available online 30 June 2022

Abstract: Translator is a useful tool which allows a person to communicate with other people using different languages that are not familiar with when visiting a foreign place. The existing voice translators experiencing slower process in translating spoken word or sentences from one language to another. The paper presents the development of translation system for translating spoken Malay word or phrases into Chinese and English or vice versa using Python. Mel Frequency Cepstral Coefficient (MFCC) and Hidden Markov Model (HMM) algorithms are proposed in the system to improve the system performance while converting speech to text or text to speech. The reliability of translation and speed performance of the system have been conducted with 15 people, 8 males and 7 females. The reliability test for sample rate 8000 bps, 11025 bps, 32000 bps and 48000 bps indicating the corresponding translation quality are 53%, 73%, 100% and 100% respectively. This suggested that the sample rate 32000 bps and 48000 bps is more reliable and the translation quality is better. The translation speed performance for sample rate 8000 bps, 11025 bps, 32000 bps and 48000 bps are 5.73s, 5.81s, 5.87s and 5.97s respectively. Although sample rate 8000 bps has resulted fastest translation speed, but its corresponding system quality is poor. As conclusion, a Malay speech translation system is successfully developed and tested. The system can accurately translate the Malay words in relatively short time. Thus, the result suggests that the most suitable system operational sample rate is 32000 bps, because it result in faster and reliable translation.

Keywords: Speech Translation System ,Voice Translator, Python

1. Introduction

The translation is one of the keys not only for professionals in the fields of linguistics and education but also for social, political and economic zones [1]. The translator has helped lots of people when contacting people of different language or visit a foreign place whether travelling or for business [2]. Some people may face difficulties in communication at this time, a translator is able to help people to

Corresponding author: chua@uthm.edu.my

translate the word. In this new modern and technology generation, there has a lot of translator available online such as Google Translate, Bing Translator, Systran, PROMT, Babylon, WorldLingo, Yandex, and Reverso.

Language is considered a tool to facilitate and allow human communication. Whether it is a term that we are familiar with, we can pick it up in a matter of seconds. But whether it is a language we do not know, we will need to use dictionaries, manual parsers, translators, or other translation software. All of these solutions disrupt the flow of any conversation that someone could have with another person of a different dialect, because the pause is needed for translation [3].

Malaysia is a country in Southeast Asia where Malay language is the official language in Malaysia. It is a multi-ethnic nation with different languages spoken by different ethnic groups. Some of people may face a problem to communicate with other people. Therefore, translator is an effective and efficient tool to overcome communication barrier between different races due to different languages, or dialects.

The existing voice translators typically handle the process of translating word or sentences from one language to another slower [3]. Thus, this project proposes to use Python to develop a reliable and faster translator that not only can translate Malay speech to English and Chinese word but also can listen to the text-speech.

The aims to describe the development of Malay to Chinese and English translator system using Python, design a graphical user interface (GUI) for the system as a standalone executable application and evaluate the reliability and speed performance of the system.

2. Scope

Figure 1 illustrates the block diagram for the Speech translator system which is developed using Python. The spoken speech is captured using a microphone and the conversions are realized using Mel frequency Cepstral coefficients (MFCC) and Hidden Markov Model (HMM) algorithms. The MFCC mainly used to convert recorded speech into text while the HMM responsible for converting the text into speech. In between, the Google Translate is utilized for translating the text from Malay word to Chinese or English word. The translation result is displayed on screen, meanwhile it also possible to listen the translation spoken through speaker. Since there are lot of Malay dialect so the language that chosen for developing this system is focusing on common Malay language so called ©Bahasa Melayu Baku©.

For testing purposes, the duration to record speech from microphone has been fixed to five seconds, therefore the recording could be used to perform testing on isolated and connected word. The isolated word selected for testing are ‘semalam’(yesterday) and ‘makan’(eat) while ‘makan roti’(eat bread) and ‘nasi lemak’(nasi lemak) are the connected words chosen for the test. The test wes conducted by 15 people in a quiet and noisy rooms. To imitate or create noisy place for testing, white sound is purposely playing around microphone to add a noise to the quiet room. Each person has to speak all selected isolated and connected words and uttering the words for two times. The respective test was conducted on sample rates 8000 bps, 11025 bps, 32000 bps and 48000 bps.

In addition, an executable application is built, so it is convenience to be used by people. This application is developed by using software PyCharm version 2021.1 which is easier to used and capable to build applications.

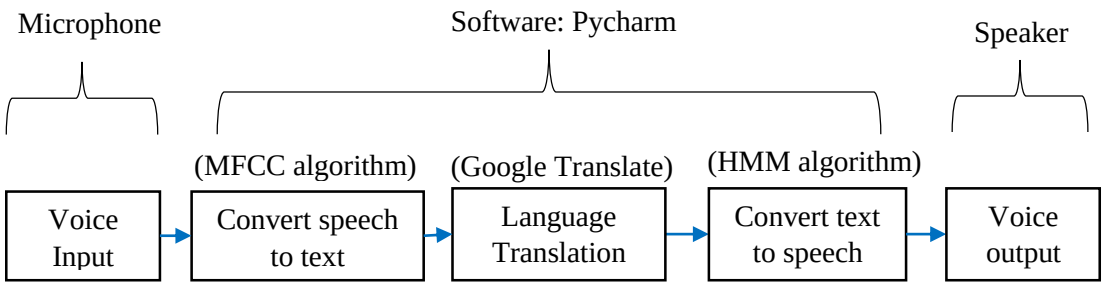

Figure 1: Block diagram of the translator

3. Methodology

PyCharm version 2021.1 is used to develop this system. The reason is it has an intelligent code editor. Furthermore, it can compose high-quality codes and uses different colour schemes for keywords, groups, and functions to make the code more readable and understandable [4]. Additionally, it can code navigation where it saves time and effort by quickly navigating to a function, class, or file while editing and enhancing code. Indeed, symbol, element, or variable can also find quickly in the source code.

Figure 2 shows the process of translation. First the voice signal is to balance the speech sound spectrum, which has a sharp high-frequency roll-off. Then frequency Domain function to make audio continue on every frame. Mel-frequency wrapping applied in the frequency domain to get a better resolution at low frequencies. Discrete Cosine Transform (DCT) use to extract high frequency and low frequency changes in the signal. Google translate is to translate text from Malay to English and Chinese. After this, take input text and analysis. It converts to context-dependent labels for example CV, V or CVC. Then find the sound from dataset for CV and CVC for example “se,ma,lam” CV/CV/CVC and lastly it will combine together.

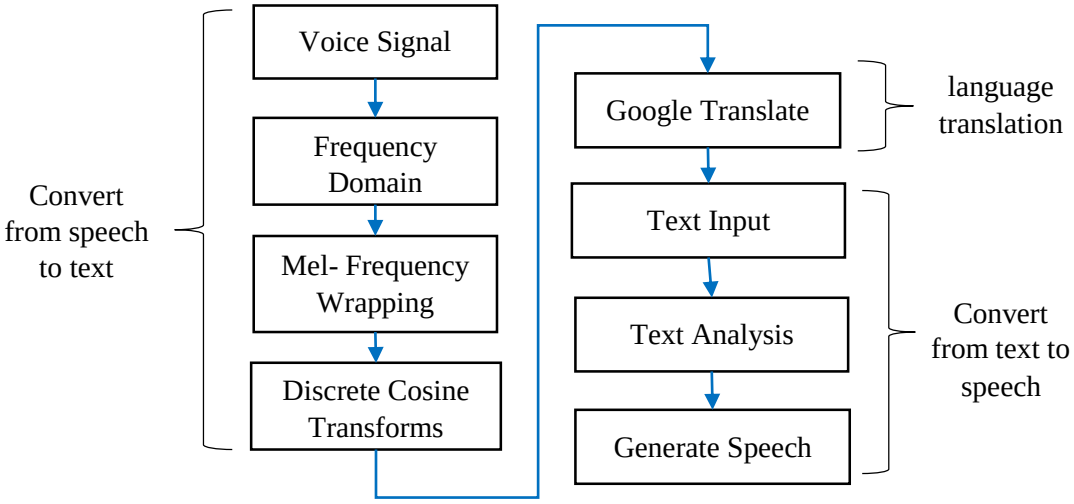

Figure 2: Process of translation

3.1 Speech to text

The system records the voice input from the microphone. MFCC use the speech recognition system based on Python. Speech recognition requires audio input, which makes it quite simple. It is versatility and simplicity of use make it a great choice for any Python project. Then, it is open-source and is free to use. The language is ms-MY means Malay-Malaysia so only the Malay language is recognised. It takes input from the microphone using the Recognizer class record (). The recording method has two arguments. Both of these settings are set to None by default. In default mode, the recording method

records audio data from the beginning until it runs out however by modifying and adding float values, it can get a duration for 5 seconds. For example, if need 5 seconds of an audio record, it needs to specify the duration parameter to 5.0. Figure 3 shows the code for speech recognition.

```
import speech_recognition as sr
r = sr.Recognizer()
r.recognize_google(audio,language='ms-MY')
def GetAudio(self):
    r = sr.Recognizer()
    m = sr.Microphone()

    with m as source:
 audio = r.record(source, duration=5.0)
```

Figure 3: Coding for Speech Recognition

For the MFCC process, firstly is a voice signal. Voice signal is to balance the speech sound spectrum, which has a sharp high-frequency roll-off. Figure 4 shows the code for the voice signal.

```
audio = normalize_audio(audio)
```

Figure 4: Coding for voice signal

Next frequency domain function to make audio continue on every frame. Fast Fourier Transform (FFT) will be used to transform it into the frequency domain. The Discrete Fourier Transform (DFT) over a discrete signal y(n) of N samples, transforms each frame of N samples into the frequency domain. Basically, the output DFT and FFT is same but different calculation but this system, it uses FFT. Figure 5 shows the code for frequency domain that can be defined as Eq. 1.

$$Frequency\ Domain = \sum_{n=0}^{N-1} y_n\, e^{-j2\pi kn/N}, k = 0,1,2 \ldots \qquad Eq.\ 1$$

Discrete Cosine Transform (DCT) use to extract the high frequency and low-frequency changes in the signal [5] – [9]. DCT is applied to produce a set of cepstral coefficients. Before this, Mel-frequency wrapping is usually represented on a log scale. This results in a signal in the cepstral domain with a frequency peak corresponding to the pitch of the signal and a number of formants representing low frequency peaks.

```
audio_winT = np.transpose(audio_win)

audio_fft = np.empty((int(1 + FFT_size // 2), audio_winT.shape[1]), dtype=np.complex64, order='F')

for n in range(audio_fft.shape[1]):
    audio_fft[:, n] = fft.fft(audio_winT[:, n], axis=0)[:audio_fft.shape[0]]

audio_fft = np.transpose(audio_fft)
```

Figure 5: Coding for Frequency Domain

Mel-frequency wrapping applied in the frequency domain to get a better resolution at low frequencies. It can be defined as equation below. Mel-frequency wrapping or can be call as Mel-filter bank. Basically, the filter has the space on centre frequencies on the frequency axis. The filter used to solve is Hanning filter shape triangle. Figure 6 show coding Mel-frequency wrapping. It can convert between original frequency and Mel-frequency as stated in Eq. 2 and Eq. 3 to easily estimate energies spot and once this energy are estimated then the log of these energies also known as Mel spectrum can be used for calculating DCT.

$$Mel\ Frequency, m = 2595 * log_{10}\left(1 + \frac{f}{700}\right) \quad Eq.\ 2$$

$$Original\ frequency, f = 700\left(10^{\frac{m}{2595}} - 1\right) \quad Eq.\ 3$$

```
def freq_to_mel(freq):
    return 2595.0 * np.log10(1.0 + freq / 700.0)

def met_to_freq(mels):
    return 700.0 * (10.0 ** (mels / 2595.0) - 1.0)
```

Figure 6: Coding for Mel-frequency wrapping

Figure 7 show coding discrete cosine transform. M is total number of triangular filter. Formula DCT usually represented as Eq. 4.

$$c(n) = \sum_{m=0}^{M-1} \log_{10}(s(m))\cos(\frac{\pi n(m-0.5)}{M}) \quad n = 0,1,2 \ldots \quad Eq.\ 4$$

```
def dct(dct_filter_num, filter_len):
    basis = np.empty((dct_filter_num, filter_len))
    basis[0, :] = 1.0 / np.sqrt(filter_len)

    samples = np.arange(1, 2 * filter_len, 2) * np.pi / (2.0 * filter_len)

    for i in range(1, dct_filter_num):
        basis[i, :] = np.cos(i * samples) * np.sqrt(2.0 / filter_len)

    return basis
```

Figure 7: Coding for discrete cosine transform

3.2 Language translation

Google Translate is the most convenient and smoothest online text translation tool available. In this system, it uses the Google Translate module which is a freely accessible library that can be built and used in raw Python code. Language translation use google translate text from Malay to English and Chinese. Figure 8 shows the coding for google translate. It is using googletrans Python import Translator and specific the language for example ‘en’ English and ‘Zh-CN’ Chinese.

```
def translate(self, *args):
    translator = Translator()
    translatedms = translator.translate(*args, dest='en')
    self.ids.lblmly.text = translatedms.text

    translatedcn = translator.translate(*args, dest='zh-CN')
    self.ids.lblcn.text = translatedcn.text
```

Figure 8: Coding for google translate

3.3 Text to speech

Text-to-Speech module can accept a maximum of 100 characters. The dataset is an open source database where it has around 80000 iterations. It supports voices for the Malay language, Chinese language and English.

HMM use text to speech system based on Python. It converts the text to speech and can listen the sound. It takes input text and analysis. Convert to context-dependent labels for example CV, V or CVC. Then find the sound from dataset for CV and CVC for example “se,ma,lam” CV+CV+ CVC and lastly it will combine together [10] – [13]. For word “makan” CV+CVC is disyllabic syllable while for the connected first word is “makan roti” CV+CVC, CV+CV is disyllabic syllable and second word “nasi lemak” CV+CV, CV+CVC [14]. Figure 9 shows coding for text-to-speech. It is using gtts Python import gTTS. It will take the translate text English (ttext2) and Chinese (ttext3) convert to speech.

```
def texttospeed1(self):
    ttext1 = self.ids.final_text.text
    self.speech1 = gTTS(text=ttext1)
def texttospeed2(self):

    ttext2 = self.ids.final_text2.text
    self.speech2 = gTTS(text=ttext2, lang='en')
def texttospeed3(self):
    from playsound import playsound
    ttext3 = self.ids.final_text3.text
    self.speech3 = gTTS(text=ttext3, lang='zh-CN')
```

Figure 9: Coding for text-to-speech

3.4 GUI design

Figure 10 shows the coding of GUI design. This application is design using the kivy library Python. Basically, all the button has set the font size which is size 25. The font in the application is use “champagne – limousines” while for the Chinese text use font “jizijingdian” sound [15].

```
BoxLayout:
    orientation: 'vertical'
    Button:
        id: lblcn, lblmly, lblMensaje
        text:'Start Record'
        font_name: 'bm.ttf'
        font_size: 25
        on_press: root.Btninputmic()
        on_release: root.translate(lblMensaje.text)
```

Figure 10: Coding for GUI design

Basically, to use this application, first need to enter the sample rate at the text box for example sample rate 8000 bps, 11025 bps, 32000 bps and 48000 bps. The text box is using TextInput. The window is used Window Manager. It specific each window name for example first window, and second window to switch to another window by using on_release. The background colour for each window is using rgba 0.17,0.17,0.17 black colour while the button colour is using colour 1,1,1,1 grey colour. Besides, the button for all the window is using Button and each window has set BoxLayout to keep the widgets in their designated places on each window. The word shows after click button translate to

English and Chinese is using Label. The sound is the icon button which is using background_normal by set the name of image to use.

4. Results and Discussion

The project has chosen 'semalam'(yesterday), 'makan'(eat) as isolated words and 'makan roti'(eat bread) , 'nasi lemak'(nasi lemak) as connected word. It has been tested with 15 people, 8 males and 7 females in noisy and quiet places. To create a noisy place, "white noise" is playing around microphone while doing the recording. "White noise" is a sound like noise when open TV without channel. Besides that, different sample rates were chosen for a testing sample rate of 8000 bps, 11025 bps, 32000 bps and 48000 bps. The reason for choosing the sample rate 8000 bps, 11025 bps, 32000 bps and 48000 bps is because these sample rates are basically being used on many applications for example sample rate 48000 is used for DVDs suitable for creating DVD audio discs, 32000 bps is suitable for cassette recordings from ferric stock, speech or FM radio but a little loss of quality. For sample rate, 11025 bps were used for AM broadcasting and the sample rate of 8000 bps was used in the telephone system.

This test has used an earphone's microphone as input, the person who speaks has to stay around 50cm from the microphone for a quiet place. While for the noisy place, the "white sound" was playing about 15cm away from the microphone.

The first and second isolated words used for testing are "semalam" and "makan", respectively. However, the first and second connected words selected for the test are "makan roti" and "nasi lemak", respectively.

4.1 Reliability test of translation

The reliability test of translation is to test whether the translated word is identified as the speaker is saying. Formula Eq. 5 was used to calculate the reliability test in percentage. For example, if the first isolated word, "semalam" is spoken in a quiet place, the system only recognizes 8 out of 15 correct translation for the word "semalam", thus the calculation is 8 divided with total number of people which is 15 and multiply 100 per cent.

$$Reliability\ formula = \frac{Number\ of\ speaker\ who\ get\ word\ same\ as\ said}{Total\ number\ of\ speaker} \times 100\% \qquad Eq.\ 5$$

Table 1 shows reliability test results in quiet and noisy places using isolated and connected words. Notice that the system generally well performs in a quiet place as compared to noisy place. The accuracy of translation has achieved 100% for sample rates of 32000 bps and 48000 bps in quiet room. This indicates sample rate is significant in speech translation, a higher sample rate would result higher accuracy in translation.

Table 1: Reliability test for Isolated and Connected words

Environment	Word	sample rate 8000 bps (correct translation)	sample rate 11025 bps (correct translation)	sample rate 32000 bps (correct translation)	sample rate 48000 bps (correct translation)
Quiet place	First word	53%(8)	73%(11)	100%(15)	100%(15)
(Isolated word)	Second word	67% (10)	87%(13)	100%(15)	100%(15)
Noise place	First word	20%(3)	33%(5)	33%(5)	20%(3)
(Isolated word)	Second word	27% (4)	33%(5)	33%(5)	20%(3)
Quiet place	First word	53%(8)	80%(12)	100%(15)	100%(15)
(connected word)	Second word	67% (10)	87%(13)	100%(15)	100%(15)
Noise place	First word	20%(3)	27%(4)	33%(5)	20%(3)
(connected word)	Second word	13%(2)	20%(3)	20%(3)	13%(2)

Besides, the test demonstrates that the translation is not affected by isolated and connected words. In fact, it is affected by surrounding conditions while using the system.

In a summary, the reliability test proves that the higher the sample rate, the better translation quality, especially in a quiet place. A higher sample rate technically has collected more samples per second which presents a closer recreation of the original audio. In other words, when more samples are taken, more detail about the sound waves' rise and fall is recorded.

4.2 Speed test of translation

This section conducts analysis of the time taken to recognize speed words starting from input voice signal until the end of translation. The test was performed using the selected isolated and connected words at sample rate 48000 bps, 32000 bps, 11025 bps and 8000 bps. The time is manually captured by using a stopwatch. In general, the stopwatch is started when the button record is activated and it is stopped when the recognition is done. The test has repeated for 15 times.

Table 2 and Table 3 show the average time collected in quiet and noisy places using isolated and connected words. The initial expectation is shorter translation if the sample rate is high. However, surprisingly the test results indicate that the higher the sample rate, the slower the speech recognition. This is probably because a higher sample rate would collect more samples, therefore the longer time needed to process these data. Table 2 also clearly indicates that speech recognition in a quiet place is faster than in a noisy place. When comparing the recognition time between isolated and connected words, it is found that the recognition time for the connected word generally is longer than for the isolated word. Therefore, it could be summarized that the speed of translation is affected by the sample rate and the total number of words. The translation speed is inversely proportional with the sample rate and a total number of words.

Table 2: Duration of speech recognize for Isolated word

Environment	Word	sample rate 8000 bps(s)	sample rate 11025 bps(s)	sample rate 32000 bps(s)	sample rate 48000 bps(s)
Quiet place	First word	5.74	5.81	5.87	5.97
	Second word	5.70	5.81	5.92	5.97
	average	5.72	5.81	5.90	5.97
Noise place	First word	5.93	5.95	5.98	6.01
	Second word	5.95	5.94	5.99	6.01
	average	5.94	5.95	5.99	6.01

Table 3: Duration of speech recognition for Connected word

Environment	Word	sample rate 8000 bps(s)	sample rate 11025 bps(s)	sample rate 32000 bps(s)	sample rate 48000 bps(s)
Quiet place	First word	5.85	5.91	5.98	6.02
	Second word	5.87	5.91	5.99	6.01
	average	5.86	5.91	5.99	6.01
Noise place	First word	5.95	6.01	6.01	6.00
	Second word	5.95	6.02	5.99	6.01
	average	5.95	6.02	6.00	6.01

4.3 Graphical User Interface (GUI) for application

The application is name as MalaySpeechTranslation. It is used PyInstaller to create in executable files. The total size of this application is about 300MB. It requires internet connection to used.

Figure 11 shows the main GUI window for supplication of the translation system. The window contains pushbutton “start”, “exit” and “-”. The translation is started using “start” button, ended using “exit” button and there is no function for button “-”.

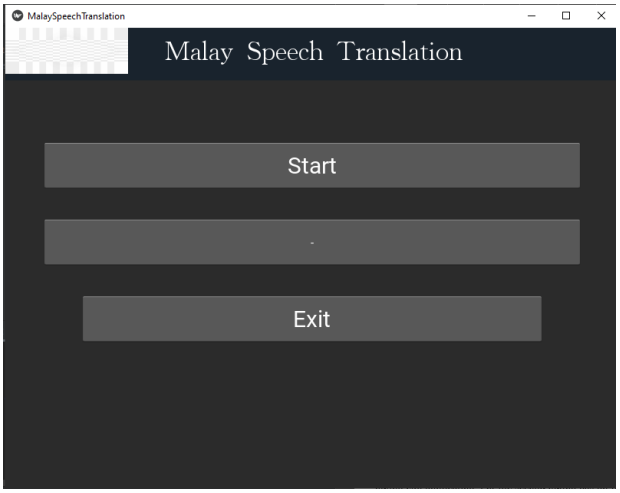

Figure 11: Main Window GUI

Once the translation process in Figure 11 is activated, Figure 12 would appear. This GUI window has a text box to define desired sample rate, a button “Record” to initiate the beginning of recording, a button “back” to go back to the main Window, button “translate to English” to convert from Malay text to English, button “translate to Chinese” to convert Malay text to Chinese, icon speakers to pronounce the translated text and button “Graph” to move to another new window to display the waveform of the input signal. Once the pushbutton “record” is pressed, the system would delay for 5 seconds waiting user to speak. Successful message would be shown if the spoken word is successfully record, else error message would be given. There are two options of translation, from Malay to English or Chinese. The option is manually selected using button “Translate to English” and “Translate to Chinese”. Once the option is selected, the system translates the converted text and temporarily save the result for subsequence process. A sound icon would pronounce the word if it is being pressed.

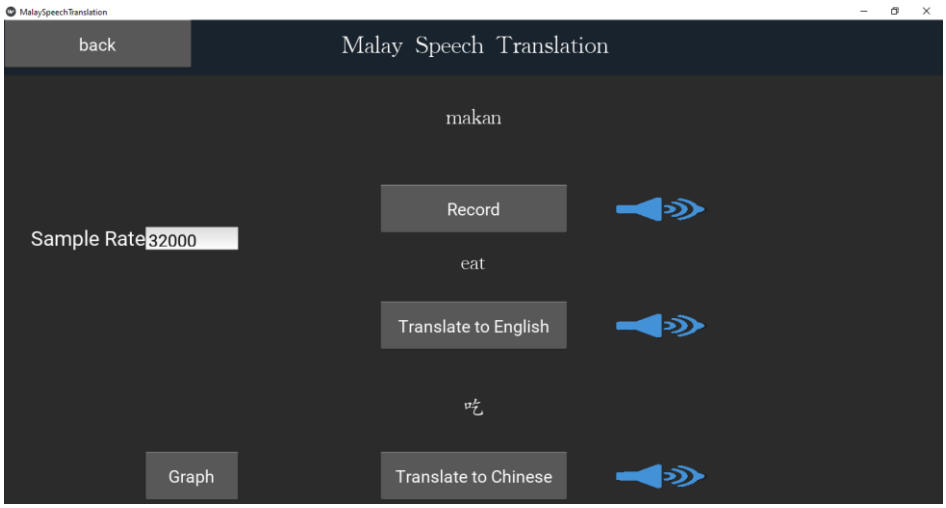

Figure 12: Second Window GUI

Figure 13 shows an example of voice signal, frequency domain display, cepstral coefficient display and details information for the voice signal. The button “save” is utilized to save the graph in .jpg format. The button “detail” would display all cepstral coefficient value, maximum frequency, sample rate and audio duration. Button “Clear” is used to clear the graph information. User can back to previous window by using button “back”.

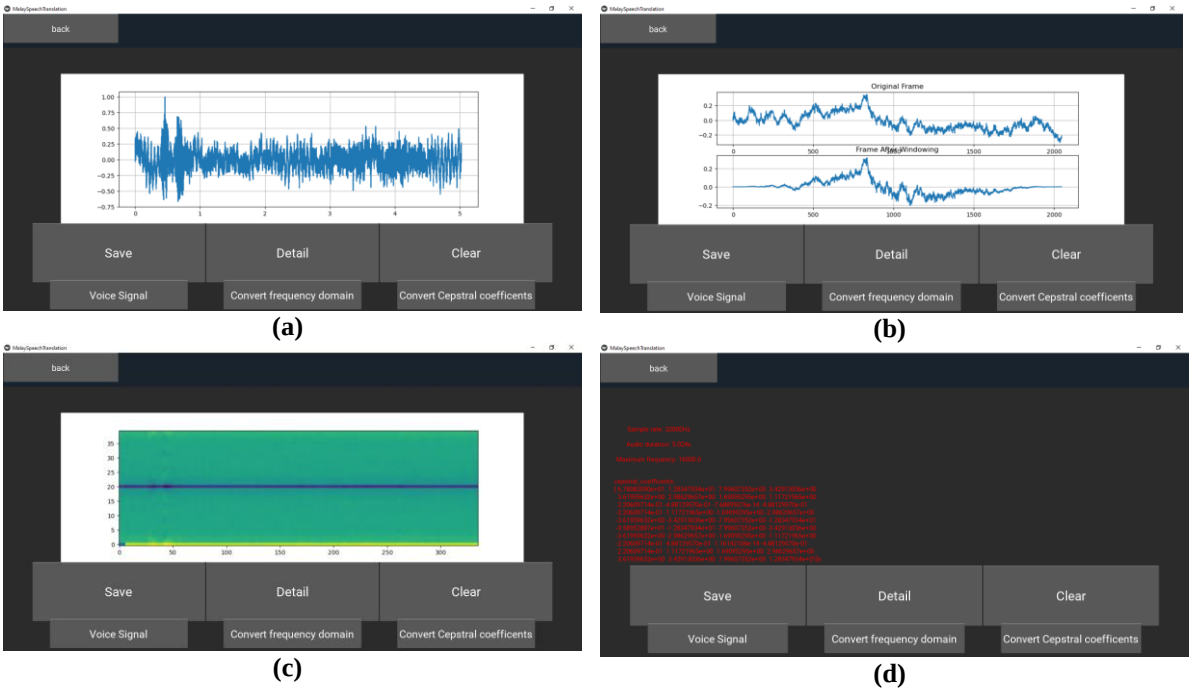

Figure 13: (a) Voice signal; (b) Voice signal in frequency domain; (c) Cepstral coefficient of voice signal; (d) Detail of voice signal

5. Conclusion

As conclusion, a Malay speech translation system has successfully developed. This system is well performed at sample rate of 32000 bps and 48000 bps. However, the investigation found that sample rate would affect speed of translation. Therefore, the optimum performance of the translation system requires a balance between sample rate and speed of translation. The investigations suggest that the suitable sample rate for this system is 32000 bps. Future work may consider to add other feature such as voice command to translate word by speak without click pushbutton.

Acknowledgement

The authors would like to thank the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia for its support.

References

[1] M. Vanjani, “A Comparison of Free Online Machine Language Translators”, Journal of Management Science and Business Intelligence, pp. 5-1, 2020

[2] O. Koksal and N. Yürük, “The Role of Translator in Intercultural Communication”, International Journal of Curriculum and Instruction, vol.12, pp. 327–338, 2020

[3] H.Krupakar and K. Rajvel, “A Survey Of Voice Translation Methodologies - Acoustic Dialect Decoder”, International Conference on Information Communication and Embedded System (ICICES), pp. 1-9, 2017

[4] Ilyamich, “MFCC's Made Easy”, April 22, 2018. [Online]. Available: Kaggle, https://www.kaggle.com/ilyamich/mfcc-implementation-and-tutorial/notebook. [Accessed June 18, 2019]

[5] Sonney, K, “PyCharm Features”, January 18, 2020. [Online]. Available: https://opensource.com/article/20/1/python-journal. [Accessed January. 18, 2020]

[6] D. Anggraeni, W. S. M. Sanjaya, M. Y. S. Nurasyidiek and M. Munawwarohi, "The Implementation of Speech Recognition using Mel-Frequency Cepstrum Coefficients (MFCC) and Support Vector Machine (SVM) method based on Python to Control Robot Arm", Journal of Management Science and Business Intelligence, pp. 1-10, 2018

[7] S. Hizlisoy and R. S. Arslan, "Text independent speaker recognition based on MFCC and machine learning", Selcuk University Journal of Engineering Sciences, pp. 73-78, 2021

[8] Arslan, R. S and Barisci, N, "A detailed survey of Turkish automatic speech recognition". Turkish journal of electrical engineering and computer sciences", pp. 3253-3269, 2020

[9] T. Singh, "MFCC implementation and tutorial", April 22, 2018. [Online]. Available: https://medium.com/@tanveer9812/mfccs-made-easy-7ef383006040. [Accessed April 22, 2018]

[10] S. Naziya and R.R. Deshmukh, "Speech Recognition System – A Review", Journal of Computer Engineering, Vol. 18, pp. 1-9, 2016

[11] V. Gupta, R. S. Shankar, H. D. Kotha and J. Raghaveni, "Voice Identification in Python Using Hidden Markov Model", Vol. 29, pp.8000-8112, 2020

[12] K. Shubham, V. Singh, S. Kumar, S. Raj, S. Ran, and N. S. Patil, "Acoustic Speech Recognition", International Journal of Advance Research and Innovative Ideas in Education, pp. 2156-3161, 2017

[13] A. F. Jalin and J. Jayakumari, "Text to speech synthesis system for tamil using HMM", 2017 IEEE International Conference on Circuits and Systems (ICCS), pp. 447-451, 2017

[14] R. M. Hanifa, K. Isa and S. Mohamad, "Malay speech recognition for different ethnic speakers : An exploratory study", IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), pp. 91-96, 2017

[15] M. Driscoll, "Python: Build a Mobile Application With the Kivy Python Framework", September 26. 2020. [Online]. Available: https://realpython.com/mobile-app-kivy-python/. [Accessed September 26. 2020]