Enhancing Multilingual Communication through NLP and Speech Recognition Integration

Pranav Dalvi pranav.dalvi932@gmail.com
Kirti M. Doongursee College

Abstract

This research paper presents the development of a multilingual speech translation system that leverages Natural Language Processing (NLP) and Speech Recognition technologies. The system is designed to recognize spoken language, detect the target language for translation, and deliver the translated text in both written and spoken forms. This paper outlines the methodologies employed, including the use of spaCy for NLP tasks and Google Speech Recognition and Translation APIs, and discusses the system's performance and potential applications.

Introduction

In the era of globalization, effective communication across linguistic barriers is essential for personal and professional interactions. The rapid advancements in Natural Language Processing (NLP) and Speech Recognition technologies have paved the way for innovative solutions to facilitate multilingual communication. This paper presents a comprehensive system that integrates NLP and Speech Recognition to provide real-time speech translation services. The system aims to bridge the gap between speakers of different languages by detecting the target language for translation, and delivering the translated text in both written and spoken forms.

The core components of the system include the language detection, translation command extraction, speech recognition and translation. By leveraging the capabilities of spaCy for NLP tasks and Google APIs for Speech Recognition and Translation, the system offers a robust and user-friendly solution for multilingual communication.

By developing a system that seamlessly integrates NLP and Speech Recognition technologies, we aim to enhance multilingual communication and contribute to the growing body of research in this field. The successful implementation of this system demonstrates the potential for further innovation and improvement in the realm of speech translation.

Keywords

Natural Language Processing (NLP), Speech Recognition, Machine Translation, Multilingual Systems, SpaCy, Google Speech Recognition API, Google Translation API

Literature Review

The field of Natural Language Processing (NLP) has seen significant advancements with the development of sophisticated language models such as BERT, GPT, and spaCy. These models enable computers to understand and process human language more effectively. In the realm of speech recognition, technologies like Google Speech Recognition have revolutionized how we interact with machines. Combining these

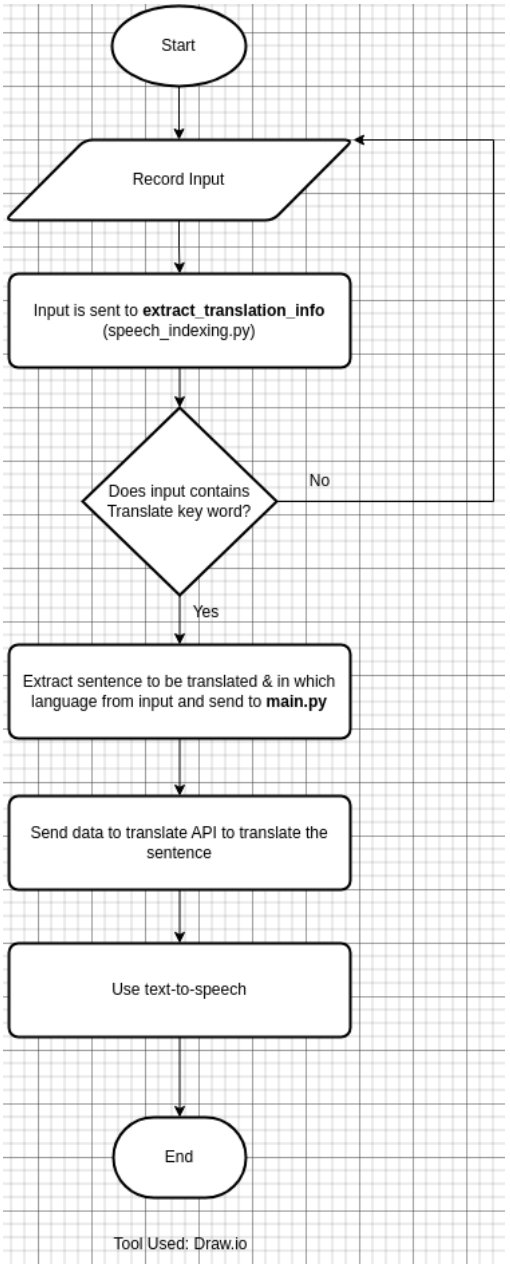
technologies with machine translation APIs, such as Google Translate, provides powerful tools for building multilingual communication systems. Previous research has demonstrated the effectiveness of combining NLP with speech recognition for various applications, including automated customer service, language learning, and real-time translation services.

ATR began its S2ST research in order to overcome the language barrier problem in 1986. The drastic increase in demand for translingual conversations, triggered by IT technologies such as the Internet and an expansion of borderless communities as seen in the increase in the number of EU countries, has boosted research activities on S2ST technology, Many research projects have addressed speech-to-speech translation technology, such as VERB-MOBIL, C-STAR,1 NESPOLE!,2, and BABYLON.[5]

Methodology

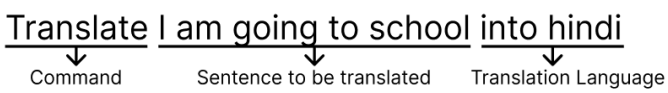
Algorithms Used

- **Language Detection and Translation Command Extraction**
  - Utilizes spaCy for tokenizing and analyzing the structure of input sentences.
  - Detects language keywords using regular expressions and pre-defined lists.
  - Cleans the text to remove unnecessary command phrases for accurate translation.
- **Speech Recognition and Translation**
  - Uses the SpeechRecognition library to capture and transcribe spoken input.
  - Integrates Google Translate API for translating transcribed text.
  - Converts translated text to speech using Google Text-to-Speech (gTTS).



Implementation

1. Input is recorded using the speech_recognition module and then converted into text sentence format.
2. The sentence is sent to the extract_translation_info function where it is segmented and extracted for further processing using NLP and regex.



3. If the sentence contains the command word "Translate," then the sentence will be sent for translation.
4. The sentence is translated using the Google API for translation.
5. The sentence is converted back into speech using the gTTS API (Google Text to Speech).

Code

speech_indexing.py

```
def detect_language(text, supported_languages):
    for key, value  in supported_languages.items():
        print(f"Key: {key}")
        if key in text.lower():
            return key.lower()
    return None
```

- Detecting Language extracts information about in which language the sentence should be translated in.

```
def clean_text(text, language):
    text = re.sub(rf"\b(?:translate|convert|translation)\b.*\b(into|to|in)\s+{language}\b", "", text, flags=re.IGNORECASE)
    text = re.sub(rf"\b(into|to|in)\s+{language}\b", "", text, flags=re.IGNORECASE)
    text = re.sub(r"\b(this sentence|of following sentence)\b", "", text, flags=re.IGNORECASE)
    return text
```

- Clean text function extracts sentence to translate by removing or cleaning the Translate and into followed by language (eg. Marathi, Hindi, etc)

```
def extract_translation_info(sentence, supported_languages):
    doc = nlp(sentence)
    translate_index = None
    for token in doc:
        if token.lemma_ == "translate" or token.lemma_ == "convert" or token.lemma_ == "translation":
            translate_index = token.i
            break

    language = detect_language(sentence, supported_languages)
    if translate_index is not None and translate_index + 1 < len(doc):
        text_to_translate = ""
        for token in doc[translate_index + 1:]:
            text_to_translate += token.text + " "
    else:
        text_to_translate = ""

    text_to_translate = clean_text(text_to_translate, language)

    return {
        "translate_index" : translate_index,
        "language": language,
        "text_to_translate": text_to_translate.strip()
    }
```

- extract translation info function converts sentence into nlp tokens and then sends sentence to detect_language and clean_text functions and return the extracted values for translation.

main.py

```python
def recordInput(src_lang, supported_languages):
    r = sr.Recognizer()
    with sr.Microphone() as source:
        print('listening')
        r.pause_threshold = 1
        r.adjust_for_ambient_noise(source)
        playsound('./audio_files/listening.mp3')
        audio = r.listen(source)

    try:
        text = r.recognize_google(audio, language = src_lang)
        print(f"Input: {text}")
        nlp_op = extract_translation_info(text, supported_languages)
        return nlp_op

    except Exception as e:
        print(f"Error: {e}")
        return "None"
```

- Record Input function records user voice and processes the input to convert it into text and send to extract_translation_info function.

```python
def translateText(text, tgt_lang):
    tgt_out = translator.translate(text, dest=tgt_lang)
    print(f"Output: {tgt_out.text}")

    return tgt_out.text
```

- Translate Text function translates the text into specified language.

```python
def speak(text, tgt_lang):
    tts = gTTS(text=text, lang=tgt_lang)
    tts.save('./speech.mp3')
    time.sleep(4) # Depends on the system's performance
    playsound('./speech.mp3')
```

- speak function simply converts text into speech mp3 file and playsound is used to play these mp3 files.

```python
if __name__ == "__main__":

        supported_languages = {"hindi":"hi", "marathi":"mr", "gujarati":"gu", "kannada": "kn", "malayalam":"ml", "spanish":"es", }

        nlp_op = recordInput("en-in", supported_languages)
        print (nlp_op)
        if nlp_op["translate_index"] != None:
            if(nlp_op["language"] in supported_languages):
                print("en-in", supported_languages[nlp_op["language"]])
                tgt_out = translateText(nlp_op["text_to_translate"], supported_languages[nlp_op["language"]])
                speak(tgt_out, supported_languages[nlp_op["language"]])
            else:
                print('Try Saying "Translate I am going to school in Hindi"')
                playsound('./audio_files/error1.mp3')
                playsound('./audio_files/suggestion1.mp3')

        else:
            print('Try Saying "Translate I am going to school in Hindi"')
            playsound('./audio_files/error1.mp3')
            playsound('./audio_files/suggestion1.mp3')
```

- This function combines all the functions given above and helps generate output with appropriate error handling.

Aims and Objectives

- To develop a robust system that can accurately recognize and transcribe spoken language.
- To implement a language detection mechanism that identifies the target language for translation.
- To evaluate the system's performance across different languages and use cases.

Future Work

- To develop an Android app for the user interface.
- Increase accuracy by applying noise reduction techniques.
- Develop APIs for seamless application integrations.
- Testing with offline models for translation and speech recognition.

Conclusion

This research demonstrates the integration of NLP and Speech Recognition technologies to build a multilingual speech translation system. The system successfully identifies the target language, transcribes spoken input, translates it, and outputs the translated text in both written and spoken forms. The combination of spaCy for NLP tasks and Google APIs for speech recognition and translation proves to be effective. Future work includes extending the system to support more languages and improving the accuracy of language detection and translation commands

Reference

1. The ATR Multilingual Speech-to-Speech Translation System: https://www.researchgate.net/publication/3457553_The_ATR_Multilingual_Speech-to-Speech_Translation_System
2. SpaCy: https://spacy.io/usage
3. Google Text-to-Speech: https://gtts.readthedocs.io/en/latest/
4. Google Translation: https://py-googletrans.readthedocs.io/en/latest/
5. The ATR Multilingual Speech-to-Speech Translation System: https://www.researchgate.net/publication/3457553_The_ATR_Multilingual_Speech-to-Speech_Translation_System