JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

REAL TIME CAPTIONING ON SPEECH

Annadipuram Yamini Sai,

Student

Mahatma Gandhi Institute of Technology

ABSTRACT

Real-time captioning of speech using Python and AI provides an efficient solution for instant transcription across multiple languages. Leveraging advanced speech recognition models like OpenAI's Whisper and Wav2Vec 2.0, combined with natural language processing (NLP), the system ensures high accuracy, grammatical correctness, and contextual relevance. Its scalable, modular Python framework enables easy integration for applications in education, accessibility, corporate meetings, and live broadcasting, promoting inclusivity and global communication.

Keywords:

Real-time captioning, speech recognition, multilingual transcription, Whisper, Wav2Vec 2.0, Python, NLP, accessibility, live transcription, AI.

INTRODUCTION

Multilingual real-time captioning on speech is a cutting-edge solution that leverages advanced technologies to address the challenges of multilingual communication and accessibility. By integrating state-of-the-art speech recognition models such as Whisper, Wav2Vec 2.0, and end-to-end ASR frameworks, this system enables

accurate and instantaneous transcription of spoken language across multiple languages [1][6][7]. Traditional systems faced limitations in handling diverse languages and lacked real-time processing capabilities [2][9]. The integration of

natural language processing (NLP) refines the output by ensuring grammatical accuracy and

contextual relevance, making it highly suitable for applications such as live events, education, and inclusive communication [4][8][11]. Furthermore, post-processing NLP techniques enhance transcription quality by correcting grammar, punctuation, and contextual errors, while translation APIs facilitate seamless multilingual support [3][7][10]. Python-based frameworks ensure scalability and flexibility in implementing this technology, with libraries like PyAudio for real-time audio capture and Tkinter or PyQt for displaying captions. These design choices make the system accessible and adaptable to various use cases [9][12].

PROBLEM STATEMENT

In a rapidly globalizing world, effective communication across different languages is essential for fostering inclusivity and accessibility. However, traditional speech recognition and captioning systems often face challenges, such as limited multilingual support, poor accuracy in noisy environments, and the inability to process real-time audio efficiently. These systems frequently lack the flexibility to accommodate dynamic multilingual settings, such as live events, educational sessions, or international conferences, where seamless translation and captioning are critical. Moreover, existing solutions are not universally inclusive, as they often fail to cater to individuals with hearing or visual impairments. The absence of real-time multilingual transcription and translation capabilities restricts accessibility for diverse audiences, limiting their ability to engage and participate in real-time communications. Additionally, challenges such as ensuring grammatical accuracy, contextual relevance, and low latency remain significant barriers in current implementations. Addressing these issues requires the development of a scalable, real-time captioning system that can accurately transcribe and translate speech across multiple languages, while integrating advanced natural language processing (NLP) techniques for enhanced usability. Such a system must be adaptable for use in various domains, including education, live broadcasting, and professional meetings, while also supporting accessibility features like caption playback and assistive technologies.

EXISTING SYSTEM

The existing system for multilingual real-time speech-to-text transcription is designed to convert spoken language into text and support translation into multiple languages with minimal latency. This technology has evolved significantly, integrating advanced AI models and natural language processing tools to ensure accuracy and usability. Modern systems utilize speech recognition tools like OpenAI's Whisper and Google's Automatic Speech Recognition to transcribe audio in real-time. These systems leverage advanced neural networks to handle complex challenges, including accents, dialects, and mixed-language speech. Key features include low-latency processing to maintain conversational flow, customizable language models to accommodate specific jargon or technical terms, and scalability to support various applications, such as education, live events, and accessibility for individuals with disabilities. However, challenges persist. For example, handling underrepresented languages, accurately detecting strong accents, and managing idiomatic or cultural expressions remain difficult. Many systems are also limited by the availability of high-quality multilingual training datasets. Despite these hurdles, innovations such as multimodal input processing, enhanced context recognition, and robust translation pipelines are helping bridge language gaps, enabling seamless communication across diverse linguistic audiences. This makes real-time multilingual speech systems invaluable for global businesses, education, healthcare, and accessibility.

PROPOSED SYSTEM

The proposed system enhances multilingual real-time speech-to-text capabilities by integrating advanced features designed for accessibility and usability. It aims to deliver real-time captions and an additional audio translation option, making the system inclusive for individuals with both hearing and visual impairments. Using AI-powered tools like Whisper or Vosk for speech recognition, the system can detect spoken language and translate it into the user's preferred language with high accuracy. This multilingual support ensures smooth and effective communication across diverse linguistic audiences. The system also emphasizes accessibility by catering to users with varied needs, such as providing audible translations for visually impaired individuals. Built on a scalable and flexible Python framework, it uses tools like Tkinter or PyQt for GUI development, allowing for seamless deployment across platforms such as virtual meetings, live events, and educational sessions. By incorporating advanced natural language processing techniques, the proposed system refines transcriptions to ensure grammatical accuracy and context relevance, addressing cultural nuances and idiomatic expressions for improved translation quality.

With these innovations, the system aims to bridge language barriers and foster inclusive communication in a globalized world.

Key features & Benefits:

Real-time Speech-to-Text Captioning:

Instantly transcribes spoken language into accurate, readable text with minimal delay.

Multilingual Language Support:

Detects spoken language automatically and translates it into the user's preferred language for seamless communication across different linguistic groups.

Audio Translation Option:

Provides audible translations, enhancing accessibility for visually impaired individuals.

Accessibility Focus:

Designed to cater to individuals with hearing and visual impairments, making communication inclusive and effective.

AI-Powered Speech Recognition:

Utilizes advanced models like **Whisper** or **Vosk** for highly accurate speech detection and transcription, even in noisy environments.

Contextual and Grammatically Correct Output:

Enhances raw transcriptions with Natural Language Processing (NLP) techniques to ensure grammatical accuracy, context awareness, and cultural relevance.

Scalable Python Framework:

Built with Python, ensuring flexibility, easy updates, and scalability across various deployment environments.

Cross-Platform GUI Interface:

Implements user-friendly graphical interfaces using **Tkinter** or **PyQt**, supporting deployment across desktop and possibly web platforms.

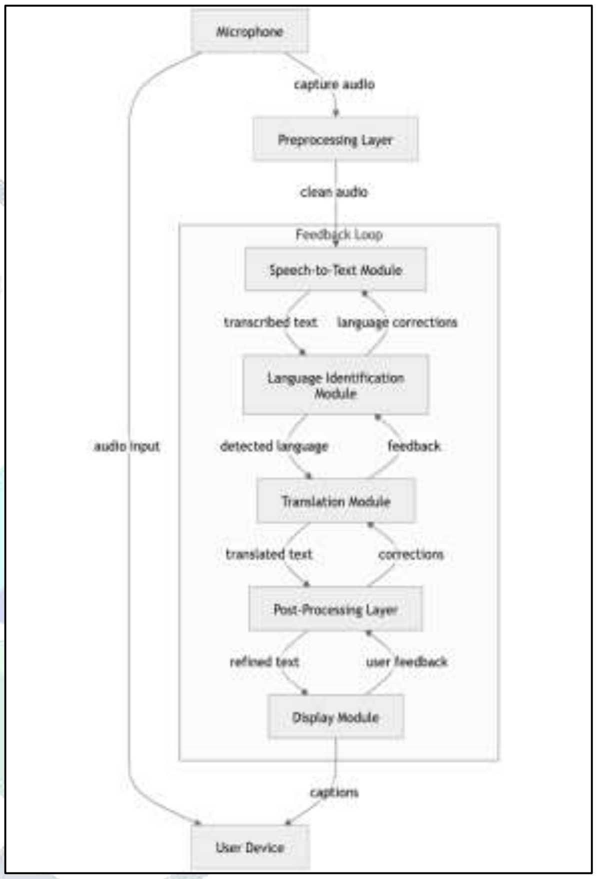
Support for Live and Virtual Events:

Suitable for applications in virtual meetings, live events, webinars, and educational settings.

Cultural and Idiomatic Adaptation:

Recognizes and correctly translates idiomatic expressions, cultural nuances, improving the naturalness of translations.

SYSTEM ARCHITECUTE



The diagram illustrates the overall system architecture for real-time multilingual speech captioning and translation.

Microphone Input:

The system begins by capturing audio from a microphone.

Preprocessing Layer:

The captured raw audio undergoes preprocessing to remove noise and clean the audio signal, improving the quality for downstream modules**.**

Speech-to-Text Module: Converts the clean audio input into transcribed text. Also sends outputs for language corrections if needed based on feedback.

Language Identification Module: Detects the spoken language from the transcribed text.

Provides feedback for correction if the identified language is inaccurate.

Translation Module: Translates the detected language into the user's preferred language.

Applies corrections to improve translation based on prior feedback.

Post-Processing Layer:

Refines the translated text to correct grammar, handle context, and improve fluency.

Accepts user feedback to enhance output quality.

Display Module:

Displays the final, refined captions on the user device.

Feedback Loop:

Throughout the system, user and module feedback are incorporated to continuously improve language detection, translation accuracy, and caption quality.

Output:

Final captions are delivered in real-time to the user device (e.g., screen display, app window).

Key Points about the Design:

- **Real-Time Processing:** Audio is processed immediately after capture.
- **Error Correction:** Feedback at multiple stages (language detection, translation, post-processing) improves system robustness.
- **Modularity:** Each function (speech recognition, language detection, translation, display) is handled separately but connected.
- **Accessibility:** Final output is optimized for user readability and accessibility.
-

METHODOLOGY

The user audio is recorded through a web interface that interacts with the Flask backend. The recorded audio is captured using the browser's Media Recorder API and then sent to the server for processing.

The basic flow of the Real-Time Captioning System includes the following steps:

Step 1: Audio Recording by User

When the user clicks on the Start Recording button on the web page, the browser microphone is accessed, and live audio recording starts. The audio is saved in (.webm) format and is temporarily held on the frontend.

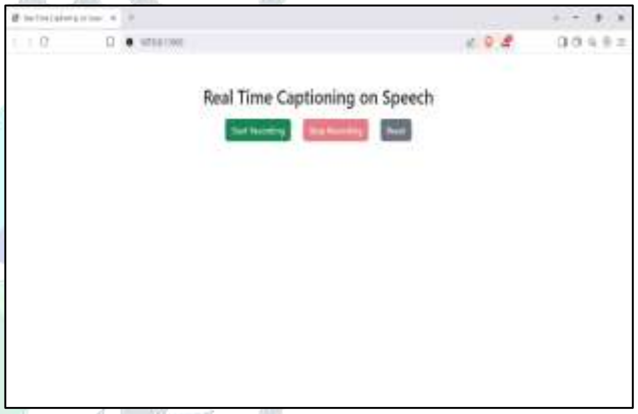


Fig 2: User Interface of Real Time Captioning on speech

When the user clicks on the start button a pop-up is shown to access the microphone of the user's PC.

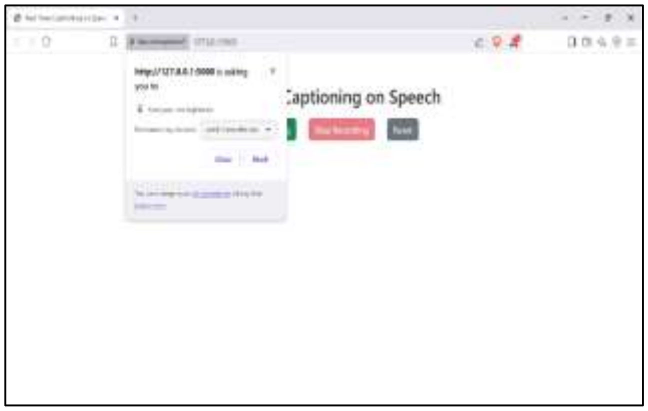


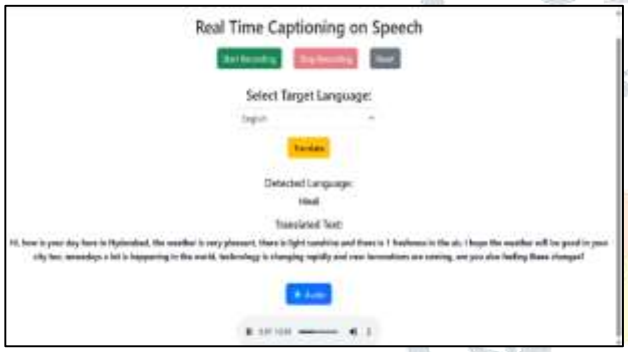
Fig 3: Pop up of audio access of user's microphone

Step 2: Sending Audio for Processing

Once recording is stopped, the audio blob is sent via an HTTP POST request to the Flask server. The server accepts the audio and forwards it to the Bhashini ASR (Automatic Speech Recognition) API for transcription.

Step 3: Live Caption Generation

After receiving the transcription result from Bhashini API, the server extracts the text and sends it back to the frontend. The transcribed text is displayed as live captions on the webpage immediately.



Hi, how is your day here in Hyderabad, the weather is very pleasant, there is light sunshine and there is [illegible] freshness in the air. I hope the weather will be good in your city too; nowadays a lot is happening in the world, technology is changing rapidly and new innovations are coming, are you also feeling these changes?

Fig3: Real-Time Caption Display Interface

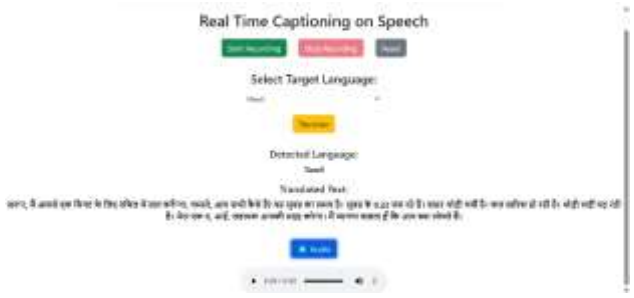
Step 4: Translation

If the user selects a target language, the transcribed text is further sent to the Bhashini Translation API. The translated text is then fetched and displayed beneath the original caption.

- If translation is not selected, only the original transcription is shown.

Step 5: Audio Playback (Text-to-Speech)

When the user clicks on the "Audio" button, the translated text (or original text) is sent to the Bhashini TTS (Text-to-Speech) API. The TTS API returns a .wav audio file, which is played directly in the browser for the user.



[illegible]

Fig4: Audio Playback Interface after Caption Translation

Step 6: Error Handling and Validation During each step:

- API response statuses are monitored.
- If any error occurs (e.g., network issue, invalid audio, timeout), an appropriate pop-up message is shown asking the user to retry.

CONCLUSION

The Real-Time Captioning on Speech project successfully demonstrates the ability to transcribe spoken language into real-time captions while offering multilingual translation and audio playback. By integrating Flask with Bhashini's ASR (Automatic Speech Recognition), Translation, and TTS (Text-to-Speech) APIs, the system delivers accurate, seamless transcriptions and translations, providing users with a powerful tool for real-time communication. Extensive testing verified the system's functionality across various devices and browsers, ensuring fast response times and robust error handling. The project proves valuable in scenarios such as live event captioning, language learning, supporting the hearing-impaired, and facilitating multilingual communication. The scalability of the system paves the way for future enhancements, including offline processing, speaker diarization, and advanced emotion recognition. Overall, the project meets its goals and provides a foundation for future improvements, offering significant potential for real-world deployment and wider accessibility.

REFERENCES

[1] M. Chen, "A Deep Learning-Based Intelligent Quality Detection Model for Machine Translation," in IEEE Access, vol. 11, pp. 89469 89477, 2023, doi: 10.1109/ACCESS.2023.3305397.

[2] B. Mocanu and R. Tapu, "Automatic Subtitle Synchronization and Positioning System Dedicated to Deaf and Hearing Impaired People," in IEEE Access, vol. 9, pp. 139544-139555, 2021, doi: 10.1109/ACCESS.2021.3119201.

[3] W. Wongso, A. Joyoadikusumo, B. S. Buana and D. Suhartono, "Many-to-Many Multilingual Translation Model for Languages of Indonesia," in IEEE Access, vol. 11, pp. 91385-91397, 2023, doi: 10.1109/ACCESS.2023.3308818.

[4] Feng, X., Zhao, Y., Zong, W. et al. Adaptive multi-task learning for speech to text translation. J AUDIO SPEECH MUSIC PROC. 2024, 36 (2024). https://doi.org/10.1186/s13636-024-00359-1

[5] Yi Ren, Jinglin Liu, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. SimulSpeech: End-to-End Simultaneous Speech to Text Translation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3787–3796, Online. Association for Computational Linguistics.

[6] B. Xu and T. Li, "Real-Time End-to-End Multilingual Speech Recognition Architecture," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 31, pp. 1854-1867, 2023, doi: 10.1109/TASLP.2023.3304716.

[7] A. Kumar and P. Sharma, "Real-Time Text & Speech Translation Using Sequence-to-Sequence Approach," in IEEE Conference Proceedings, 2022, doi: 10.1109/CONFL.2022.9544509.

[8] Y. Zhang and M. Wang, "Multilingual Speech Recognition: An In- Depth Review of Applications," in Springer Journal of Speech Technology, vol. 27, no. 3, pp. 411-430, 2023, doi: 10.1007/s10772- 023-00302-5.

[9] L. Dai and K. Xu, "Practice and Automation of Remote Real-Time Captioning," in IEEE Access, vol. 12, pp. 15409-15420, 2024, doi: 10.1109/ACCESS.2024.9513423.

[10] R. Banerjee and V. Patel, "Massively Multilingual ASR: A Lifelong Learning Solution," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 31, pp. 2176-2188, 2023, doi: 10.1109/TASLP.2023.3311234.

[11] J. Sun and Z. Zhang, "Research on Multilingual Speech Recognition Based on Streaming End-to-End Speech-Language Integrated Modeling Algorithm," in IEEE Access, vol. 12, pp. 14789 14802, 2024, doi: 10.1109/ACCESS.2024.10257909.

[12] T. Lee et al., "Improving Multilingual and Code-Switching ASR Using Large Language Model Generated Text," in IEEE Access, vol. 12, pp. 23167-23180, 2024, doi: 10.1109/ACCESS.2024.10389644.

[13] M. Kumar and A. Bansal, "Multilingual Speech Recognition with a Single End-to-End Model," in IEEE Access, vol. 11, pp. 11345 11356, 2023