Volume 9, Issue 4, April – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24APR1907

Harnessing Open Innovation for Translating Global Languages into Indian Lanuages

Y V Nagesh Meesala[1]; V Sai Surya[2]; P Sai Kiran[3]; R Sree Vardhan[4]
Department of Computer Science Engineering, Raghu Engineering College, Visakhapatanam, India

Abstract:- Understanding foreign languages can be challenging for individuals living in India's diverse linguistic landscapes. We propose a new technology that utilizes machine translation to address this issue, specifically focusing on speech recognition and synthesis. It aims to convert online video resources into Indian languages by integrating open-source technologies like text-to-speech (TTS), speech-to-text (STT) systems, and FFmpeg library to separate or augment audio and video. We used the whisper model, the application that can read up to 60 different Languages in the form of audio as input, and it transcripts the audio into text with segments of sentences based on timestamps. The sentence-based transcription generated by whisper is then translated into the desired language using Google Cloud translate_v2. Later, Each timestamp was individually converted into audio using the Google Cloud text-to-speech service, ensuring the audio fits inside the length of its respective timestamp. The individual audio segments are then augmented to generate the final audio in the desired language. Finally, the audio is attached to the original video, ensuring video-audio synchronization. The accuracy of the translation was verified by comparing the naturalness of the audio with general spoken language standards. This application benefits visually impaired individuals and those who cannot read text, providing them with a means to acquire knowledge in their native languages.

Keywords:- *Open Innovation, Text-to-Speech Speech-to-Text, Machine Translation.*

I. INTRODUCTION

India is a multilingual country with over 19,500 languages, including 19,500 major and 1599 other languages. Even though many languages are spoken throughout India, only twenty-two have been officially recognized as scheduled languages. Unfortunately, most of the information on the Internet is in English and other Global Languages, which everyone needs help understanding. The manual translation of these online resources into regional languages is a time-consuming and tedious process. Fortunately, machine translation, a subfield of artificial intelligence, offers a solution to this problem. Machine translation allows for quick and efficient translation of online resources into regional languages with minimal human involvement. The main goal of machine translation is to translate text from one human-readable language to another. Machine translation is based on natural language processing, a well-established research area focusing on synthesizing inputs and extracting valuable insights.The Indian government strongly supports and encourages the use of machine translation since it helps people gain access to information in their native languages. Machine translation can translate content across various domains, including sports, history, business, health, literature, news, and social media. In this study, we explore how open innovation, which involves using existing software applications at each stage of the translation process, can make the translation process easier and more efficient. Text output from videos effectively translates video content into different languages using automated closed captions. This can help reach a large audience, including visually impaired individuals and those who cannot read a text. The text can be converted into an audio output using text-to-speech (TTS) software, which reads news, SMSs, email, etc.. TTS systems are categorized into three types: articulatory, formant, and concatenative synthesis, each with advantages and limitations. In this paper, various global language videos are machine translated into Indian regional languages using a pipeline of pre-existing software applications such as the FFmpeg library, the speech-to-text module of the whisper(Automatic et al. (ASR)), the Google Translate API, and the text-to-speech of the Google cloud. This project was initially developed by a team of four undergraduate students as a solution to one of the challenges in the Communications vertical of the "Open Innovation hackathon for Smart Cities" by Berkeley-Haas, University of California, Berkeley, in collaboration with the Andhra Pradesh State government organized at Koneru Lakshmaiah Education Foundation, Guntur, India, in July 2017.Their system achieved up to audio generation of the desired Indian regional language. Our project has the advancements and continuation to what they have achieved, where we generated timestamp-based audio generation to sync the translated audio to the original video for audio-video synchronization. This process eliminates language barriers and ensures a coherent and immersive audio-visual experience, ultimately enhancing viewer comprehension and enjoyment. Section 2 discusses previously developed related works, Section 3 discusses the proposed architecture, Section 4 shows the analysis of the results, and Section 5 concludes the paper, providing recommendations for future developments based on inputs from users when the application was shown to them.

II. LITERATURE SURVEY

"Machine translation can be done in three ways: Direct machine translation, Rule-based machine translation, and Corpus or Empirical machine translation" [3]. "The first effort of translating English to Indian regional languages was made in 1995 when a group of Professors, RMK Sinha et al., at IIT Kanpur, India, developed a software application named AnglaBharti [4]. This application was developed using transfer-based and interlingua methods. A pseudo target language is generated by the system for the English text passed through the application, and this pseudo language is converted into an Indian regional language with the help of a text generation module.

The purpose of post-editing requires human involvement to check and correct grammatical errors, if any. However, to reduce human involvement in the post-editing procedure of the output generated by this application, IIT Kanpur and IIIT Hyderabad, India, developed another application called "Anusaaraka" in the same year. This application "works on language, word, and statistical knowledge." [5].

A statistical framework for machine translating English documents to Malayalam was proposed by Mary Priya et al. This framework needs a bilingual dictionary and Malayalam corpus [6]. This framework uses "suffix separation and stop word elimination" [6] to reduce the complexity of the training dataset. However, this framework consumes much memory to store the Malayalam corpus. To overcome this space complexity, Remya Rajan et al. proposed a "rule-based machine translation technique" for translating English to Malayalam [14]. Transfer-link rules were developed to obtain a target tree. Similarly, morphological rules were derived from the input to assign morphological features. " Roman to Unicode file, word dictionary file, morph dictionary file, and transfer links" are the significant factors in rule-based machine translation [14].

While the above machine translation system involves Roman and Unicode files, a morphological analyzer for the Malayalam language was proposed by Rinju O.R. et al. [7]. This analyzer takes the input, processes the input, and retrieves grammatical information like morphemes. It also provides information about the speech, like the verb tense, speaker gender, numbers, and case information of the noun. For analysis accentuation, a list is derived from the information retrieved. It can be inferred from the results that a rule-based approach is better than the probabilistic Method.

Focusing on translation in the sports domain, mainly cricket, and using the transfer-based approach, Aasha V C and Amal Ganesh developed a translation method from English to Malayalam [8]. To improve the quality of translation, 44 rules are developed. "Each word in the given English input sentence is tokenized, and a parse tree is formed using them. Using the breadth-first search (BFS) order, rules are applied to the parse tree to obtain the target language. The lengthier sentences sometimes generate incorrect outputs, which can be handled by dividing the longer sentences into shorter ones" [8]. There are several similar machine translation proposals and applications for the Malayalam language.

Apart from the machine translation applications developed for Malayalam, there are systems for each Indian language. In 2014, Keerthi Lingam et al. proposed "Rule-based machine translation from Global to our preferred language with emphasis on prepositions." [12]. This Method mainly focuses on handling prepositions when translating documents from English to Telugu. It is because there are multiple meanings for the same preposition in Telugu. Factors like "time, gender, context, and many other features play an important role in translation" [12]. As the title suggests, this is developed on a rule-based approach. For this reason, a finite set of production rules and English, a Telugu dictionary based on these rules, are developed. "reformatting and reformatting" [12] are used to identify text in diagrams and flowcharts.

In 1799, a Danish scientist, Christian Kratzensteim, was the first to work on text-to-speech systems. His research marked the beginning of TTS evolution. Kratzensteim built a "model of the human vocal tract," which produces vowels. After that, with much research, many improvements were made. In 1981, P. Rubin and Thomas Baer initiated an articulatory synthesizer for perpetual research [9]. "In this approach, human vocal tracts are labeled with six articulators, such as tongue tip, tongue body, velum, lips, jaw, and hyoid bone position, which will outline the vocal tract." [10]. These parameters calculate "area and width functions" [10]. After calculating these functions, "speech is produced by digital filter representation of the given vocal tract transfer functions [10].

"In 2001, Alan Black et al. developed a small fast run-time synthesis machine named "Flite" (pronounced as "F" - lite standing for Festival Lite), which is designed as an alternative to the Festival platform. This is a TTS synthesizer application for many Indian languages [11]. As the name suggests, it is much faster and smaller than the Festival system [11]. This Festival speech system was also initially developed by Alan Black and some of his colleagues in the Centre for Speech Technology Research (CSTR) at the University of Edinburgh. These were combined and released as a Festvox project. Later, a Silicon Valley entrepreneur, Suresh Bazaj, in collaboration with Prof. Alan Black, empowered a non-profit initiative, Hear2read, which aids the visually impaired (VI) people to gain knowledge by converting the visual text into audio and reading the website contents. This initiative utilizes the flite TTS software [11].

Of the many TTS systems developed for Indian languages, some prominent ones include Sruthi, a TTS system developed at IIT Kharagpur for Hindi and Bengali languages. Vani is another TTS software application developed at IIT Bombay. Swaram is another TTS system that accepts Malayalam text and produces speech. A "multilingual newspaper reading system" has been developed to read the contents of newspapers, which supports Hindi, Malayalam, Tamil, and English [13]. In these systems, the user can select the required language, favorite topics, or domains in the newspapers. The main challenge in developing this system was maintaining a single database to handle memory and develop translation rules for each language.

Recently, TTS systems for optical character recognition (OCR) documents have been developed, which can be used to read OCR documents of many languages and generate speech from the text in OCR documents. The text is digitalized using OCR and then transformed into a grayscale image. These images are divided into segments to obtain relevant features [15]. A TTS system was developed for the Tamil language in 2014 that consists of "text analysis, syllabication, Letter-to-sound rules, and concatenation" [16]. Phonemes are mapped to words, and a database of these phonemes is developed. Then, a joining cost is computed to locate the best possible speech order. Akshara-to-font mapping, pronunciation of akharas (letters), and normalizing text are significant issues that arise during machine translation to Indian regional languages. The "TFIDF (term frequency-inverse document frequency)" is applied to identify different fonts. Similarly, in 2016, Ancy Anto and K.K. Nisha, students of the Computer Science department at Rajiv Gandhi Institute of Technology, developed a "Text-to-Speech Synthesis System for English to regional language translation.[10]"

Open innovation [2] is “a paradigm that assumes that firms can and should use external ideas as well as internal ideas, and internal and external paths to market, as the firms look to advance their technology” [2]. This term was first coined and promoted by Henry Chesbrough, a professor at the Haas School of Business at University of California, Berkeley [2].

There are certain advantages to open innovation. It reduces the cost of research, improves productivity, and enhances viral marketing, digital transformation, and scope for new business models. Open innovation also has certain disadvantages, such as losing confidentiality of information and losing competitive advantage for revealing intellectual property [2]. Open innovation is classified into specific models depending upon the circumstances of its usage. Examples are the Government model, product platforming model, etc." [2]. Open innovation is an ecosystem consisting of three parts, which describe "open innovation, innovation systems, and business ecosystems" [2].

Open source and open innovation are not similar and have conflicts in terms of patent rights. These two terms are not mutually exclusive. Some companies and open-source initiatives merge into these two concepts. An example is IBM, where the company presents its Eclipse platform as an open innovation case when companies are inside an open innovation network.

In the year 2019, Srikar Kashyap Pulipaka, Chaitanya Krishna Kasaraneni, and Surya Sai Mourya Kosaraju, Machine translated English Videos into Indian Regional Languages using an Open Innovation project aimed to facilitate the translation of English videos into various Indian regional languages through a comprehensive software application. The system, developed through open innovation principles, offers users the flexibility to select their preferred language among Bengali, Gujarati, Hindi, Kannada, Marathi, Punjabi, Tamil, or Telugu. In its basic version, the software operates on UNIX/Linux or MacOS platforms with pre-loaded Python 3 packages. Utilizing open-source tools alongside IBM Watson and Google Translate APIs, the system undergoes four distinct phases: audio separation, speech-to-text conversion, text translation, and text-to-speech conversion[1].

In the initial phase, the system separates audio from the input English videos using the ffmpeg library, a versatile multimedia framework capable of handling various formats. Following this, the audio segments undergo parallel processing for efficient speech-to-text conversion using IBM Watson's supercomputer capabilities. Leveraging parallel processing significantly reduces processing time compared to sequential methods. Subsequently, translated text in the desired regional language is generated through Google Translate API, ensuring high accuracy in translation. Finally, the Flite engine, an open-source tool developed by Carnegie Mellon University, synthesizes the translated text into audio output, completing the process of translating English videos into Indian regional languages[1].

III. PROPOSED SYSTEM

Our project presents an innovative solution for translating videos in around 100 global languages into a regional language spoken widely in India. Leveraging a combination of Google Cloud Platform (GCP) services and open-source libraries like Whisper, our proposed system seamlessly integrates speech-to-text (STT) transcription along with timestamps, translation, and text-to-speech (TTS) synthesis functionalities.

➢ *The System Operates Through a Series of Meticulously Designed Steps Outlined below:*

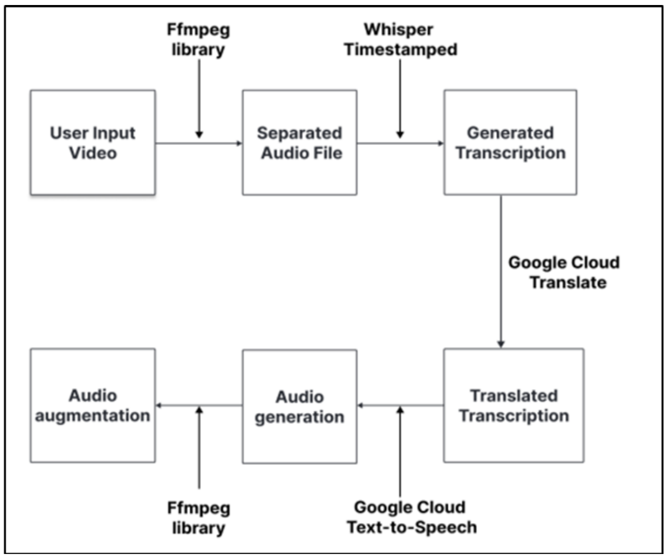


Fig 1 Proposed System Architecture

➢ *Requirements*

To run this project there are certain requirements.

- Load any operating system, including UNIX/Linux, MacOS, or Windows, with the Python 3 package pre-loaded.
- Install the FFmpeg library and set the appropriate path to the FFmpeg executable, enabling audio processing tasks such as extraction and concatenation of audio and video streams.
- Install the whisper-timestamped open-source model. This can be done by cloning the GitHub repository and following the given steps on the main page.
- Python packages should be pre-loaded: google-cloud-speech, google-cloud-translate, google-cloud-text-to-speech, pydub
- Google Cloud Platform (GCP) Credentials: Obtain a valid API key for Google Cloud Translate to access Google Cloud services. The API key is necessary for authenticating requests to the Google Cloud Translation API and facilitating language translation operations.

➢ *Phases of Machine Translation*

As seen in the architectural diagram, this project has six phases:

- User Input through Tkinter
- Audio separation
- Speech-to-text (STT) conversion
- Translation to the desired language
- Text-to-speech (TTS) conversion
- Audio Integration

- *User Input phase*

The user input phase is the initial step in the video translation process, where users choose the video file, input and output languages, and voice type for the translated audio. The user interface for this phase uses the Tkinter library to create a graphical window for user interaction. The window includes a file selection section, where users can browse and choose an MP4 video file for translation. The selected file's path is displayed within the window.

Users can choose the input language from a dropdown menu with various language options and select the output language, which is limited to Indian languages. They can also choose the voice type for the translated audio from a dropdown menu with two options: male or female. Once all selections are made, users can submit the form by clicking the 'Submit' button, which triggers backend processing. The user interface includes a message box that displays the status of the translation process and informs users when it is in progress and when it is completed successfully.

- *Audio Extraction*

Audio extraction is essential for isolating the audio component from a video file, enabling further processing such as transcription and translation. The FFmpeg library is a widely used multimedia framework renowned for its versatility in handling various multimedia tasks, including all the major components of the transcoding pipeline (demuxers, decoders, filters, encodes, muxers) run in parallel and manipulating both audio and video files. Improved throughput and CPU utilization decreased latency. The provided code harnesses the power of the FFmpeg library to execute commands that extract the audio track from the input video file (video.mp4) by utilizing Python's subprocess. Run () function, the code invokes FFmpeg to perform the extraction operation in a streamlined manner which ensures that only the audio component is retained, which is then saved as a separate MP3 file (temp_audio.mp3) for subsequent processing.

- *Speech-to-Text Conversion*

The integration of Whisper_timestamped into our project streamlines the process of speech-to-text conversion. The effectiveness of Whisper_timestamped is due to its key attributes, such as efficiency, accuracy, customizability, and modular design. The library's proficiency in handling real-time applications and large volumes of audio data is a testament to its efficiency. Its accuracy, inherited from the Whisper model, ensures reliable transcriptions, while the option to fine-tune pre-trained models for specific domains or languages enhances its adaptability.

Given an audio file as input, the library efficiently processes the audio and generates a transcription in the same language. Notably, each sentence within the generated transcript is accompanied by the corresponding starting and ending time stamps, fostering seamless synchronization with the original audio content. Code loads the "tiny" model using the whisper.load_model ("tiny") function. The audio file named "temp_audio.wav" is using model. Transcribe ('temp_audio.wav'). The result is a dictionary containing the transcribed text and associated timestamps. A new file is opened in write mode, and a loop is then used to iterate through the "segments" in the transcription result. The segment information is formatted as a string, combining the start and end times with the transcribed text, and then written to the text file using the file.write(f"{start_time}-{end_time}: {text}\n"). In summary, this code snippet uses the Whisper_timestamped library to transcribe an audio file, extract time-stamped transcriptions, and write them into a readable text file.

- *Language Translation Process*

In the translation phase, the project employs the Google Cloud Translate_v2 service using the Google Cloud Translation API to seamlessly convert text transcripts obtained from speech-to-text conversion into IL (Indian Language), the target audience's spoken language. This service is powered by advanced machine learning algorithms for precise and efficient translation across languages.

Notably, its broad language support, customizable features, and swift processing capabilities ensure that the translated content retains its original meaning and context, thus enhancing accessibility for IL-speaking viewers. The user is initially prompted to select their desired language, and the corresponding language object is selected from the languages list and voice gender for the translation.

The text is translated from the input language to the selected language using the Google Translation API with open('transcript. txt', 'w,' encoding='utf-8') as f:: The translated text is written to a file named 'transcript.txt.' This code performs translation files based on user input

- *Text-to-Speech Conversion*

The translated text is transformed into speech using the Google Cloud Text-to-Speech service, resulting in natural-sounding audio in the desired language. This process involves converting the input transcription text into audio segments, adjusting them to fit the target duration, and combining them into the final audio segment. The transcription text was read line by line from a file, with each line containing a timestamp and the corresponding text. The timestamp is then split into start and end times, allowing the calculation of the target duration for the audio segment generated from the text.

The text is converted to speech using the Google Cloud Text-to-Speech service, generating an audio segment from the input text. The generated audio segment is adjusted to fit the target duration using the adjust_audio_duration function. This function checks if the current duration of the audio segment is longer, shorter, or equal to the target duration. If it exceeds the target duration, the function speeds up the audio segment. If it's shorter, silence is added to compensate. When the durations match, the original audio segment is returned.

The adjusted audio segment is then appended to the final audio segment and exported as an MP3 file using the Audio Segment library. This method ensures precise audio-video synchronization by tailoring the audio to match each timestamp's exact duration. Google Cloud Text-to-Speech provides natural-sounding audio, while the adjust_audio_duration function maintains precise timing.

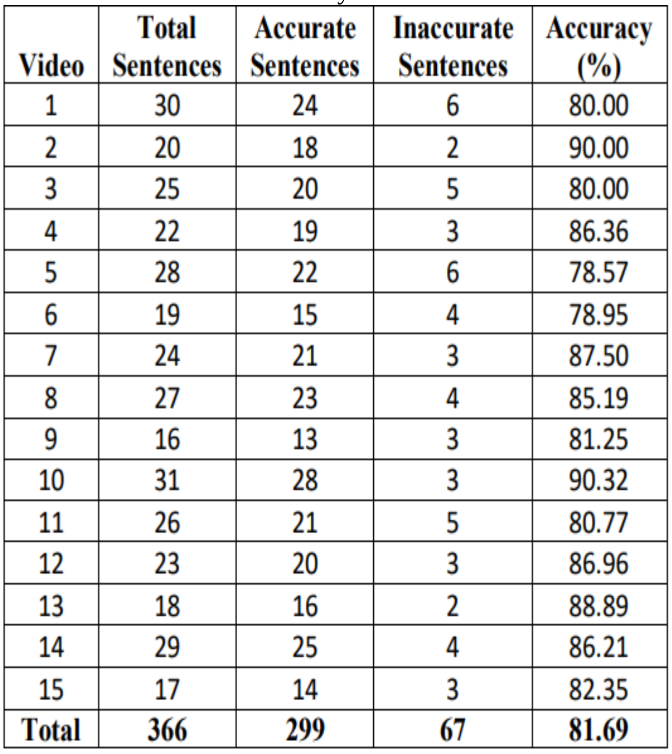
- *Audio Integration*

In the final phase, the synthesized audio is seamlessly integrated into the video output, replacing the original audio. This integration is achieved using the FFmpeg library, which was previously used to separate the audio from the video. By concatenating the input video ('input_video1.mp4') and the translated audio ('translated_audio.wav') with specified video (v=1) and audio (a=1) parameters, we ensure the accurate merging of the video and audio components. This precise concatenation creates a new video file ('finished_video.mp4') where the translated Telugu audio replaces the English audio. This process not only enhances accessibility for IL-speaking viewers but also guarantees a synchronized audio-visual experience, enriching the overall quality and impact of the video content.

IV. RESULT ANALYSIS

The Machine translation of International Videos to Indian Regional languages is implemented using the Open Source library, such as Whisper and Google Translation API, which were used in this process. To calculate the accuracy, we need to compare each sentence of the Telugu transcription with its corresponding sentence in English. If a sentence in Telugu conveys the same meaning as its English counterpart, it is considered accurate. Otherwise, we consider this to be inaccurate. Then, we calculated the percentage of accurate sentences. Then, in the speech part, we compared the synchronization of the translated audio with the video. In this case, the audio is mainly generated based on timestamps, so the length of the Input and Output videos is considered to test the accuracy.

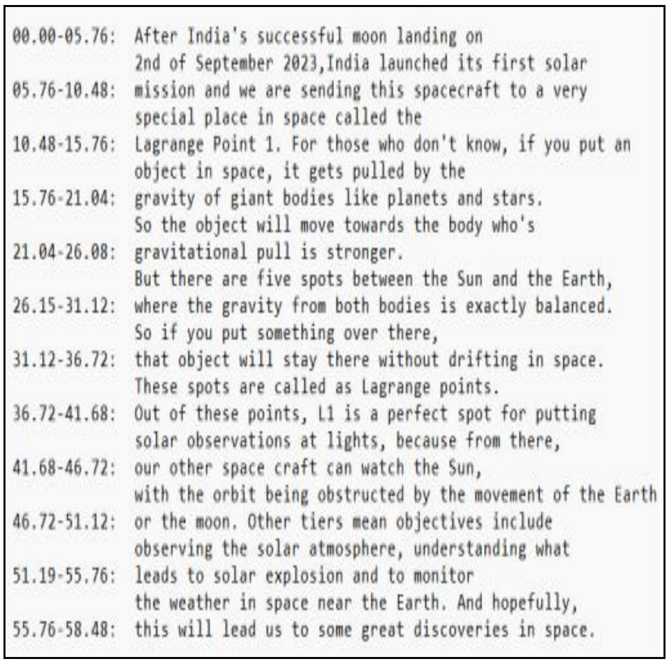
Table 1 Accuracy in Translation



Video	Total Sentences	Accurate Sentences	Inaccurate Sentences	Accuracy (%)
1	30	24	6	80.00
2	20	18	2	90.00
3	25	20	5	80.00
4	22	19	3	86.36
5	28	22	6	78.57
6	19	15	4	78.95
7	24	21	3	87.50
8	27	23	4	85.19
9	16	13	3	81.25
10	31	28	3	90.32
11	26	21	5	80.77
12	23	20	3	86.96
13	18	16	2	88.89
14	29	25	4	86.21
15	17	14	3	82.35
Total	**366**	**299**	**67**	**81.69**

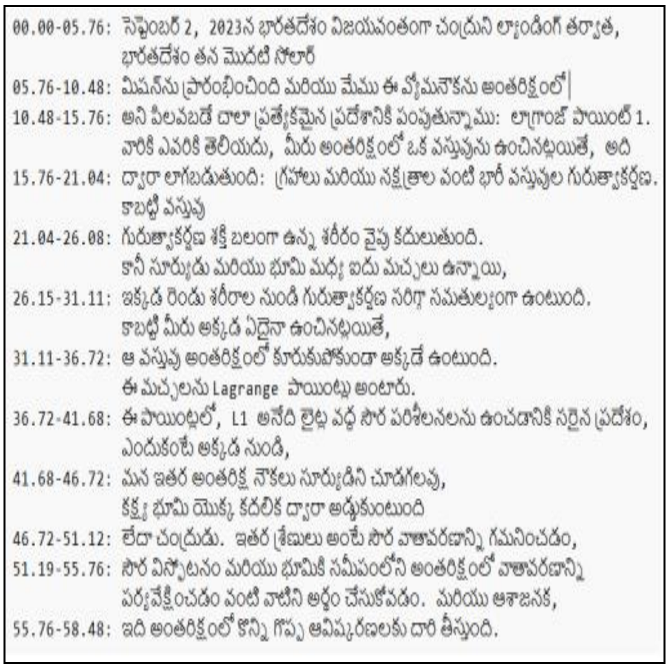
Table 1 shows the evaluation of the proposed system for machine translation of English videos to Telugu. For example, the video provides an insightful overview of India's inaugural solar exploration mission following its successful moon landing in September 2023. Delving into the mission objectives, the video highlights the strategic deployment of a spacecraft to Lagrange Point 1 (L1) for optimal solar observation. It introduces the Aditya spacecraft, equipped with advanced instruments for studying the solar atmosphere and monitoring space weather near the Earth. With a focus on scientific exploration and discovery, the video elucidates the significance of India's venture into solar exploration and its potential contributions to space science.

- The table displays the accuracy analysis for machine translation across 15 videos, each containing varying sentences.
- Video 2 achieved the highest accuracy rate of 90.00%, with 18 out of 20 sentences accurately translated. - Video 5 had the lowest accuracy rate of 78.57%, with 22 out of 28 sentences accurately translated.
- Overall, across all 15 videos, 299 out of 366 sentences were accurately translated, resulting in an overall accuracy rate of 81.69%.



```
00.00-05.76:  After India's successful moon landing on
              2nd of September 2023,India launched its first solar
05.76-10.48:  mission and we are sending this spacecraft to a very
              special place in space called the
10.48-15.76:  Lagrange Point 1. For those who don't know, if you put an
              object in space, it gets pulled by the
15.76-21.04:  gravity of giant bodies like planets and stars.
              So the object will move towards the body who's
21.04-26.08:  gravitational pull is stronger.
              But there are five spots between the Sun and the Earth,
26.15-31.12:  where the gravity from both bodies is exactly balanced.
              So if you put something over there,
31.12-36.72:  that object will stay there without drifting in space.
              These spots are called as Lagrange points.
36.72-41.68:  Out of these points, L1 is a perfect spot for putting
              solar observations at lights, because from there,
41.68-46.72:  our other space craft can watch the Sun,
              with the orbit being obstructed by the movement of the Earth
46.72-51.12:  or the moon. Other tiers mean objectives include
              observing the solar atmosphere, understanding what
51.19-55.76:  leads to solar explosion and to monitor
              the weather in space near the Earth. And hopefully,
55.76-58.48:  this will lead us to some great discoveries in space.
```

Fig 2 English Transcription with Timestamps



```
00.00-05.76:  సెప్టెంబర్ 2, 2023న భారతదేశం విజయవంతంగా చంద్రుని ల్యాండింగ్ తర్వాత,
              భారతదేశం తన మొదటి సోలార్
05.76-10.48:  మిషన్‌ను ప్రారంభించింది మరియు మేము ఈ వ్యోమనౌకను అంతరిక్షంలో|
10.48-15.76:  అని పిలవబడే చాలా ప్రత్యేకమైన ప్రదేశానికి పంపుతున్నాము:  లాగ్రాంజ్ పాయింట్ 1.
              వారికి ఎవరికి తెలియదు,  మీరు అంతరిక్షంలో ఒక వస్తువును ఉంచినట్లయితే,  అది
15.76-21.04:  ద్వారా లాగబడుతుంది:  గ్రహాలు మరియు నక్షత్రాల వంటి భారీ వస్తువుల గురుత్వాకర్షణ.
              కాబట్టి వస్తువు
21.04-26.08:  గురుత్వాకర్షణ శక్తి బలంగా ఉన్న శరీరం వైపు కదులుతుంది.
              కానీ సూర్యుడు మరియు భూమి మధ్య ఐదు మచ్చలు ఉన్నాయి,
26.15-31.11:  ఇక్కడ రెండు శరీరాల నుండి గురుత్వాకర్షణ సరిగ్గా సమతుల్యంగా ఉంటుంది.
              కాబట్టి మీరు అక్కడ ఏదైనా ఉంచినట్లయితే,
31.11-36.72:  ఆ వస్తువు అంతరిక్షంలో కూరుకుపోకుండా అక్కడే ఉంటుంది.
              ఈ మచ్చలను Lagrange  పాయింట్లు అంటారు.
36.72-41.68:  ఈ పాయింట్లలో,  L1  అనేది లైట్ల వద్ద సౌర పరిశీలనలను ఉంచడానికి సరైన ప్రదేశం,
              ఎందుకంటే అక్కడ నుండి,
41.68-46.72:  మన ఇతర అంతరిక్ష నౌకలు సూర్యుడిని చూడగలవు,
              కక్ష్య భూమి యొక్క కదలిక ద్వారా అడ్డుకుంటుంది
46.72-51.12:  లేదా చంద్రుడు.  ఇతర శ్రేణులు అంటే సౌర వాతావరణాన్ని గమనించడం,
51.19-55.76:  సౌర విస్ఫోటనం మరియు భూమికి సమీపంలోని అంతరిక్షంలో వాతావరణాన్ని
              పర్యవేక్షించడం వంటి వాటిని అర్థం చేసుకోవడం.  మరియు ఆశాజనక,
55.76-58.48:  ఇది అంతరిక్షంలో కొన్ని గొప్ప ఆవిష్కరణలకు దారి తీస్తుంది.
```

Fig 3 Telugu Translation with Timestamps

The script begins by utilizing the Whisper model and transcribing the provided audio file, "temp_audio.wav", into a textual format. The audio content is then segmented, capturing distinct portions along with their start and end times, which are carefully organized and stored in the script's memory.

After retrieving the transcribed text with detailed temporal information, the script employs the capabilities of the Google Cloud Translation service to translate it into Telugu. This ensures that the content remains coherent and retains its original meaning across languages, promoting effective communication and understanding.

Upon completion of the translation process, the script generates a new file, "translated_transcript.txt," containing the translated transcripts. This file makes the content accessible in the preferred language of the user. By integrating transcription, translation, and file generation functionalities, the script facilitates the creation of multilingual transcripts, enhancing accessibility and understanding of original audio content for diverse linguistic backgrounds.

V. CONCLUSION

In conclusion, our project offers a comprehensive solution to the issue of linguistic accessibility in India's diverse landscape. The exponential growth of online digital content has led to significant language barriers that impede knowledge dissemination and inclusivity. By utilizing cutting-edge technologies and open innovation principles, we have developed a strong framework that facilitates the seamless translation of online resources into Indian regional languages.

Our project showcases its success through its potential to benefit a wide range of users, including those who are visually impaired or have limited literacy skills. By providing access to knowledge in familiar languages, we empower marginalized communities to actively participate in education, culture, and socioeconomic activities

ACKNOWLEDGMENT

➤ *We Extend our Sincere Gratitude to Our Professors and Organizations whose Invaluable Contributions have Enriched this Research:*

First and foremost, we thank our project advisors, Assistant Professor Y V Nagesh Meesala, for their unwavering support, guidance, and encouragement throughout the project. Their expertise and experience have been instrumental in shaping the direction and outcomes of our research. We are also profoundly grateful to the team of undergraduate students who initially developed the project to solve the challenge in the Open Innovation Hackathon for Smart Cities in 2017. Their pioneering work laid the foundation for our advancements and continuation, and we are indebted to their innovative thinking and dedication. The author would like to thank OpenAI for developing Whisper AI (Speech Recognition System). We want to thank the organizers of the Open Innovation hackathon for Smart Cities, particularly the Berkeley-Haas, University of California, Berkeley, and the Andhra Pradesh State government, for providing the platform that inspired the project's inception. We are thankful to the FFmpeg library's developers, the whisper's speech-to-text module, the Google Translate API, and the text-to-speech of the Google Cloud for creating and maintaining the software applications that have been integral to our project. We would also like to acknowledge the contributions of the researchers and developers in machine translation, text-to-speech synthesis, and open innovation, whose work has been instrumental in advancing the technologies that have made our project possible. Lastly, we would like to express our gratitude to the visually impaired individuals and those who cannot read text, whose unique needs and challenges have motivated us to develop a solution that improves accessibility and inclusivity in the digital world. Completing this project was only possible with the collective efforts and support of all the individuals above and the organizations. We express profound gratitude for their contributions and unwavering dedication to propelling technology forward for the benefit of society..

REFERENCES

[1]. Srikar Kashyap Pulipaka, Chaitanya Krishna Kasaraneni, Surya Sai Mourya Kosaraju, Machine Translation of English Videos to Indian Regional Languages using Open Innovation Conference Paper · Dec 2019
[2]. Henry William Chesbrough, "Open Innovation – The New Imperative for Creating and Profiting from Technology," 2nd Edition, March 2003.
[3]. Sugata Sanyal, Rajdeep Borgohain, "Machine Translation System in India," ICSCN 2015.
[4]. R.M.K. Sinha, K. Sivaraman, Aditi Agrawal, Renu Jain, Rakesh Srivastava, Ajai Jain, "ANGLA BHARTI: A Multilingual Machine Aided Translation Project on Translation from English to Indian Languages," Proc. IEEE, 1995.
[5]. Akshar Bharati, Vineet Chaitanya, Amba P. Kulkarni, and Rajeev Sangal, " ANUSAARAKA: Machine Translation in Stages," A Quarterly in Artificial Intelligence, Vol. 10, No. 3, July 1997.
[6]. Mary Priya Sebastian, Sheena Kurian K, and G. Santhosh Kumar, "A Framework of Statistical Machine Translator from English to Malayalam," Proc. 1st Amrita ACMW Celebration on Women in Computing, September 2010.
[7]. Rinju O.R., Rajeev R. R., Reghu Raj P.C., Elizabeth Sherly, " Morphological Analyzer for Malayalam: Probabilistic Method Vs. Rule-Based Method," International Journal of Computational Linguistics and Natural Language Processing, ISSN: 2279- 0756, vol. 2, Issue 10, October 2013
[8]. Aasha V C, Amal Ganesh, "Machine Translation from English to Malayalam Using Transfer Approach," IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2015
[9]. Philip Rubin, Thomas Baer, "An Articulatory Synthesizer for Perceptual Research," Bell-Northern Research and INRS Telecommunications.
[10]. Ancy Anto, K. K. Nisha, "Text to speech synthesis system for English to Malayalam translation," International Conference on Emerging Technological Trends (ICETT), October 2016.
[11]. Alan W Black and Kevin A. Lenzo, Home Page- Dr. Alan W Black, 2001.
[12]. Keerthi Lingam, E. Rama Lakshmi and L. Ravi Teja, "Rule-based Machine Translation from English to Telugu with Emphasis on Prepositions," International Conference on Networks & Soft Computing, 2014.
[13]. Jose Stephen, Anjali.M, Bhadran V.K, "Voice Enabled Multilingual Newspaper Reading System," in Proc. IEEE, 2013.
[14]. Remya Rajan, Remya Sivan, Remya Ravindran, K.P Soman, "Rule-Based Machine Translation from English to Malayalam," International Conference on Advances in Computing, Control and Telecommunication Technologies, 2009
[15]. FFmpeg Retrieved from online
[16]. Whisper timestamped from online
[17]. Google Translate from online
[18]. Google Text-to-Speech from online.