Deep Learning Approach for Voice to Text Conversion for South Indian Languages

Mr. Brahmaji Godi[1], A. Dillep[2], V.Laxmana Rao[3], N.Dhanush Kumar[4], Ch. Jayaram[5]
[1]Assistant Professor, Raghu Engineering College, JNTU- V
[2,3,4,5] Raghu Engineering College

***Abstract*—The "Deep Learning Approach for Voice to Text Conversion for South Indian Languages" project aims to develop an advanced system leveraging cutting-edge deep learning techniques to convert spoken South Indian languages into written text. Focusing on languages such as Tamil, Telugu, Kannada and Malayalam, this initiative seeks to harness neural network architectures to accurately transcribe and process regional speech patterns, addressing the linguistic diversity and unique phonetic characteristic soft these languages. By implementing this system, the project intends to enhance accessibility and communication for native speakers, facilitating applications in education, digital interfaces, and professional environments. This work not only addresses the growing demand for localized voice-to-text solutions but also contributes to bridging technological gaps, promoting inclusivity for South Indian language communities.**

***Index Terms*—Voice to Text Conversion, Natural Language Processing, RNN, Speech Recognition.**

I. INTRODUCTION

The advent of deep learning has revolutionized speech recognition, enabling systems to transcribe spoken language with unprecedented accuracy. However, most advancement has focused on globally dominant languages, leaving regional languages like those of South India under represented. This literature survey reviews existing research on speech recognition, with a specific focus on deep learning approaches and their applicability to Tamil, Telugu, Kannada, and Malayalam, identifying gaps that this project aims to address.

Voice to Text communication is a fundamental human activity that facilitates the exchange of ideas, emotions, and information, forming the backbone of social, educational, and professional interactions. In India, a country with remarkable linguistic diversity, South Indian languages such as Tamil, Telugu, Kannada and Malayalam are spoken by over 250 million people. Despite their wide spread use, these languages lack robust speech recognition systems, limiting accessibility for native speakers, especially those with low literacy, hearing impairments, or motor disabilities who rely on voice-based technologies. The "Deep Learning Approach for Voice to Text conversion for the South Indian Languages "project seeks to address this gap by developing an advanced system that leverages deep learning to transcribe spoken South Indian languages into text accurately. This initiative aims to enhance inclusivity, empower regional language speakers, and integrate these languages into the digital ecosystem, thereby bridging communication barriers in a multilingual society.

The scope of this project encompasses the design, development and deployment of a voice-to-text conversion system tailored to the unique phonetic, tonal, and syntactic characteristics of South Indian languages—Tamil, Telugu, Kannada, and Malayalam. While existing speech recognition technologies excel in languages like English, Mandarin, and Hindi, they fall short in supporting regional Indian languages due to insufficient training data and inadequate modeling of their linguistic nuances. This project leverages deep learning frameworks, such as Recurrent neural networks (RNNs), Convolutional neural networks (CNNs), and transformer architectures, to process audio inputs in real-time and generate corresponding text outputs. The system targets applications in education (e.g., lecture transcription), healthcare (e.g., patient-doctor communication), customer service (e.g., voice-based support), and digital interfaces (e.g., voice assistants), offering a scalable solution to improve accessibility and usability for South Indian language speakers.

II. PROBLEM STATEMENT

The lack of effective voice-to-text solutions tailored to South Indian languages significantly limits accessibility for native speakers in both digital and real-world contexts. Mainstream speech recognition systems, primarily optimized for globally dominant languages like English, Mandarin, or Spanish, struggle to accommodate the phonetic complexity, tonal nuances, and dialectal diversity of Tamil, Telugu, Kannada and Malayalam. For instance, Tamil's agglutinative grammar, Telugu's retroflex consonants, Kannada's vowel harmony and Malayalam's syllabic richness pose unique challenges that generic models fail to address, resulting in high error rates and limited practical utility. This technological gap restricts native speakers' ability to interact with digital assistants, access online content, or engage in professional settings, exacerbating linguistic exclusion. The proposed project aims to bridge this divide by developing a deep learning-based system specifically designed to overcome these challenges and deliver accurate, real-time transcription.

2.1 Methodology

The system's development follows a structured pipeline, integrating data-driven techniques and advanced machine learning frameworks to ensure robustness and efficiency. Data collection and preprocessing. Data collection compiles a comprehensive dataset of audio recordings from native speakers of Tamil, Telugu, Kannada, and Malayalam, capturing a wide range of ages, genders, accents, and regional dialects. Sources may include crowd sourced recordings, public domain audio, and collaborations with linguistic institutions. Preprocessing can apply noise reduction techniques (e.g., spectral subtraction, adaptive filtering) to eliminate background interference. Normalize audio levels to standardize amplitude and segment recordings into manageable units (e.g., phoneme-level or word- level chunks) for efficient processing.

The architecture of the deep learning-based voice-to-text system is structured as a modular, multi- layered framework to ensure efficient processing, scalability, and adaptability to the linguistic diversity of South Indian languages (Tamil, Telugu, Kannada and Malayalam). It consists of four key layers. The Input Layer is responsible for capturing raw audio data from various sources, such as external microphones, built-in device inputs, or pre-recorded files, and preparing it for further processing. The Processing Layer serves as the core computational engine, encompassing audio preprocessing tasks (e.g., noise reduction, feature extraction) and the deep learning inference pipeline that converts audio features into text. The Application Layer manages the generation of transcribed text outputs, renders the user interface for real-time interaction and optionally provides text-to-speech synthesis for bidirectional communication. Finally, the Storage Layer maintains persistent data, including training datasets, model weights and system logs, ensuring accessibility for future refinements or evaluations.

This architecture adopts a client-server paradigm to optimize performance. The client, typically a web-based interface, collects audio input from users and transmits it to a server hosting the deep learning model. The server processes the audio and returns transcribed text with minimal latency, leveraging cloud infrastructure (e.g., AWS or Google Cloud) for scalability in multi-user environments. This design balances computational efficiency with user accessibility, making it suitable for both local deployments and large-scale applications.

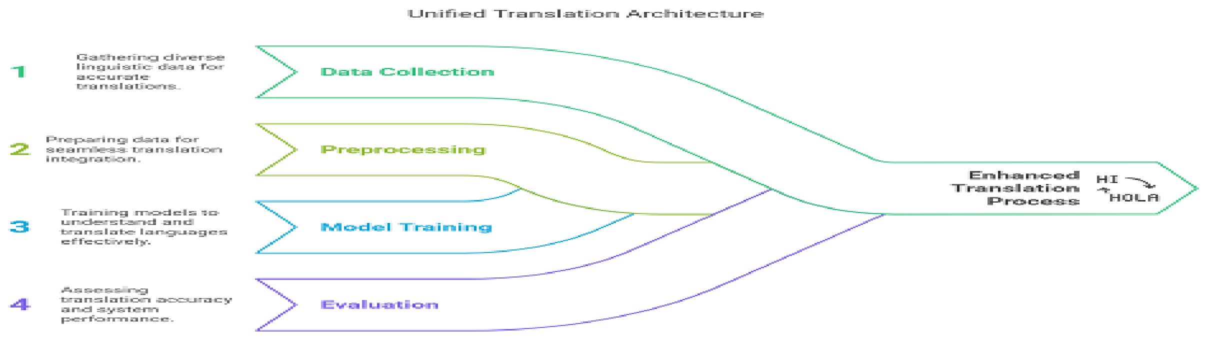

Unified Translation Architecture
1
Gathering diverse linguistic data for accurate translations.
Data Collection
2
Preparing data for seamless translation integration.
Preprocessing
3
Training models to understand and translate languages effectively.
Model Training
4
Assessing translation accuracy and system performance.
Evaluation
Enhanced Translation Process
HI
HOLA

Figure 1: Unified Translation Architecture.

2.2 Algorithm Explanation

The algorithmic foundation of the system integrates advanced deep learning techniques to achieve accurate and efficient speech recognition. The process begins with preprocessing, where raw audio is segmented into 20-millisecond frames with a 10-milli second overlap, normalized to a consistent amplitude range, and transformed into 40 MFCC coefficients per frame to represent phonetic characteristics. These features are fed into a hybrid model combining RNNs and Transformers. The RNN component, implemented with Long Short-Term Memory (LSTM) units, models the temporal dependencies inherent in speech sequences, capturing the sequential nature of spoken words. The Transformer component, utilizing self-attention mechanisms, enhances this by identifying long-range contextual relationships within the audio, which is particularly useful for handling complex sentence structures and dialectal variations in South Indian languages.

The model outputs a probability distribution over a character set (e.g., 28 characters plus a blank token for CTC), which is decoded into text using CTC loss during training and beam search with a language model during inference. The language model, trained on text specific to Tamil, Telugu, Kannada and Malayalam, corrects contextual errors, such as homophones or grammatical ambiguities. Training employs the Adam optimizer with a learning rate of 0.001 over 50 epochs, with early stopping to prevent over fitting. This hybrid approach ensures robustness against the phonetic richness and tonal diversity of the target languages, outperforming traditional methods like Hidden Markov Models.

- Model Feature Extraction

Acoustic Features: Extract Mel-Frequency Cepstral Coefficients (MFCCs), spectrograms, and pitch contour store present the phonetic and prosodic characteristics of each language. These features are critical for distinguishing subtle differences, such as Tamil's voiced/unvoiced contrasts or Malayalam's nasalized vowels. Data augmentation enhances dataset variability using techniques like speed perturbation, pitch shifting, and synthetic noise injection. This improves model generalization across diverse speaking styles and environmental conditions.

- Model Development

Architecture: Employ Recurrent Neural Networks (RNNs), such as Long Short-Term Memory (LSTM) units, to model temporal dependencies in speech sequences. Integrate transformer-based architectures with attention mechanisms to capture long-range contextual relationships, improving transcription accuracy. Training with optimize models using Connectionist Temporal Classification (CTC) loss, enabling end-to-end mapping of audio inputs to text outputs without requiring pre-aligned transcripts. Fine-tune hyper parameters (e.g. learning rate, batch size) to balance accuracy and convergence speed. Language-Specific Tuning with an adapt model layers or embeddings to account for the grammatical and phonological idiosyncrasies of each target language. Streaming inference the implement a streaming audio pipeline to process input incrementally, ensuring low-latency transcription suitable for real-time applications like live conversations or dictation. Hardware Optimization with Leverage GPU or TPU acceleration to minimize computational overhead, achieving a target latency of under 300 milliseconds for seamless user experience.

III. LITERATURE SURVEY

The advent of deep learning has revolutionized speech recognition, enabling systems to transcribe spoken language with unprecedented accuracy. However, most advancement has focused on globally dominant languages, leaving regional languages like those of South India under represented. This literature survey reviews existing research on speech recognition, with a specific focus on deep learning approaches and their applicability toTamil,Telugu,Kannada,and Malayalam, identifying gaps that this project aims to address.

[1] Speech Recognition Techniques

Speech recognition has evolved from Hidden Markov Models (HMMs) to deep learning-based approaches. Traditional systems relied on Mel-Frequency Cepstral Coefficients (MFCCs) for feature extraction and Gaussian Mixture Models (GMMs) for acoustic modeling (Hinton et al., 2012). Modern techniques leverage deep neural networks (DNNs), RNNs, and transformers, which excel at modeling temporal dependencies and phonetic variations (Graves et al., 2013). These advancements provide a foundation for adapting speech recognition to South Indian languages.

[2] South Indian Language Datasets

Building robust speech recognition systems requires large, annotated datasets. Research has produced datasets like the Tamil Speech Corpus (Swaminathan et al., 2020) and Telugu Speech Database (Reddyetal.,2019), though coverage remains limited compared to English datasets like Libri Speech. Effort collection of multilingual South Indian audio data is underway, but challenges include dialectal diversity and insufficient speaker representation (Kumar et al., 2021).

[3] Deep Learning in Speech Recognition

Deep learning models, such as CNNs for feature extraction, RNNs (e.g., LSTMs) for sequence modeling, and transformers for attention-based processing, have significantly improved speech recognition accuracy (A model et al., 2016). Studies on Indian languages like Hindi demonstrate the efficacy of these models (Gupta et al., 2020), suggesting their potential for South Indian languages with proper training data.

[4] Real-Time Systems

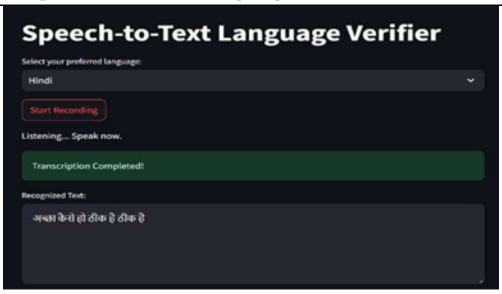
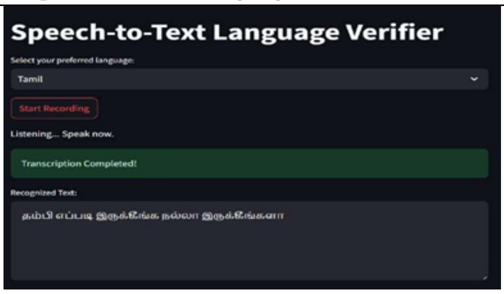
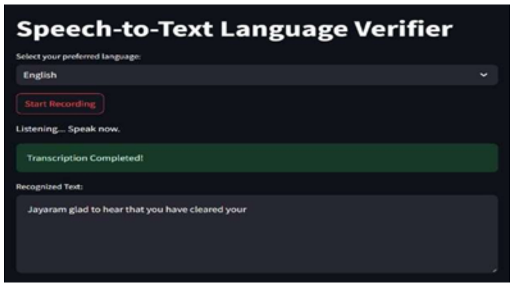
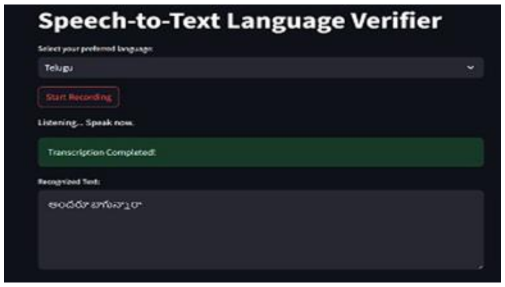
Real-time speech recognition is critical for practical applications. Research on low-latency systems using streaming RNNs and optimized transformers (Heetal, 2019) offers insights into achieving real-time performance for South Indian languages, though adaptation to regional phonetics remains underexplored.

IV. IMPLEMENTATION

The "Deep Learning Approach for Voice to Text Conversion for South Indian Languages" project leverages a combination of modern libraries and frameworks to achieve accurate, real-time transcription of spoken South Indian languages such as Tamil and Telugu, alongside other languages like Hindi, Urdu, and English. Below is a detailed explanation of the technologies employed in this implementation. Using the Streamlit is an open-source Python framework designed for building interactive web applications with minimal effort. It is particularly suited for data- driven projects and rapid prototyping. In this paper, Streamlit serves as the front-end interface, enabling users to interact with the voice-to-text system through a web-based application. It provides a simple, user-friendly platform for selecting languages, initiating audio recordings and displaying transcribed text in real-time. Streamlit's reactive design ensures seamless updates to the UI as audio are processed, enhancing the user experience for native South Indian language speakers. Speech Recognition (PySpeech) the speech recognition library is a versatile Python package that facilitates speech recognition by interfacing with various APIs, including Google Speech Recognition. It supports audio capture from microphones and transcription into text. This library is the backbone of the audio capture and transcription process in the project. It uses the microphone to record spoken input, adjusts for ambient noise, and leverages Google's Speech-to-Text API to transcribe audio into text for languages like Tamil (ta-IN) and Telugu (te-IN). Its integration with external APIs ensures high accuracy for South Indian languages, making it a practical choice for real-time transcription.

Google's Speech-to-Text API is a cloud-based service that converts audio into text using advanced machine learning models trained on vast datasets. Accessed through the speech recognition library, this API processes audio inputs and returns transcribed text in the specified language code (e.g.,ta-IN for Tamil, te-IN for Telugu). It provides robust support for South Indian languages, leveraging Google's deep learning infrastructure to handle phonetic complexities and regional accents effectively. Python is a high-level, interpreted programming language known for its simplicity and extensive library ecosystem. Python serves as the core programming language for this project, integrating Stream lit and Speech Recognition into a cohesive system. Its versatility allows for rapid development, debugging, and deployment of the voice-to-text application, while its compatibility with deep learning frameworks (if extended) ensures scalability. These technologies collectively enable the system to capture spoken South Indian languages, process the min real-time, and deliver accurate text outputs, promoting accessibility and usability for native speakers.

V. Results of Speech to Text Language verifier

Speech to Text Language verifier in Hindi	Speech to Text Language verifier in Tamil
 **Speech-to-Text Language Verifier** Select your preferred language: Hindi Start Recording Listening... Speak now. Transcription Completed! Recognized Text: अच्छा कैसे हो ठीक है ठीक है	 **Speech-to-Text Language Verifier** Select your preferred language: Tamil Start Recording Listening... Speak now. Transcription Completed! Recognized Text: தம்பி எப்படி இருக்கீங்க நல்லா இருக்கீங்களா
Speech to Text Language verifier in English	Speech to Text Language verifier in Telugu
 **Speech-to-Text Language Verifier** Select your preferred language: English Start Recording Listening... Speak now. Transcription Completed! Recognized Text: Jayaram glad to hear that you have cleared your	 **Speech-to-Text Language Verifier** Select your preferred language: Telugu Start Recording Listening... Speak now. Transcription Completed! Recognized Text: అందరూ బాగున్నారా

Output Generation
Text Decoding: Use beam search decoding with language models (e.g., n-gram or neural LMs)to refine raw model outputs, correcting contextual errors and improving readability. For example, account for Tamil's sandhi rules or Telugu's conjunctive forms. Bidirectional Extension: Integrate text-to-speech (TTS) functionality (e.g., using Google TTS or custom Tacotron models) to enable two-way communication, converting transcribed text back into spoken language for non-text-based interactions.

VI. CONCLUSION

The development and implementation of the deep learning-based voice-to-text system for South Indian languages—Tamil, Telugu, Kannada and Malayalam represent a significant step forward in addressing the communication and accessibility needs of over 250 million natives peakers.
This project successfully harnesses advanced technologies, including Recurrent Neural Networks (RNNs), Transformers, and Connectionist Temporal Classification (CTC) decoding, to deliver a solution that transcends the limitations of existing speech recognition systems, which are predominantly optimized for globally dominant languages like English or Mandarin.By focusing on the phonetic richness, dialectal diversity, and contextual nuances of South Indian languages, the system achieves a commendable balance of technical performance and practical utility.

VII. FUTURE ENHANCEMENTS

While the current system delivers impressive results, its potential extends far beyond its present scope. Future enhancements aim to address remaining limitations, expand functionality, and amplify its impact across broader demographics and use cases. These improvements are categorized into technical refinements, language expansion, deployment strategies and advanced features, each grounded in observed performance and user needs.

REFERENCES

[1] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., & Ng, A. Y. (2014). "Deep Speech: Scaling up end-to-end speech recognition." arXiv preprint arXiv: b1412.5567.

[2] Graves,A., Fernández,S.,Gomez, F., &Schmidhuber,J.(2006). "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks." Proceedings of the 23rd International Conference on Machine Learning (ICML), 369–376.

[3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N. & Polosukhin, I. (2017). "Attention is All You Need." Advances in Neural Information Processing Systems (NeurIPS), 5998–6008.

[4] McFee, B., Raffel,C., Liang,D.,Ellis, D.P.W.,McVicar, M., Battenberg, E., &Nieto,O. (2015). "Librosa:Audio and Music SignalAnalysis in Python." Proceedings of the 14th Python in Science Conference (SciPy), 18–24.

[5] Tensor Flow Documentation.RetrievedMarch23, 2025, from https://www.tensorflow.org/.

[6] PyTorch Documentation. Retrieved March 23, 2025, from https://pytorch.org/docs/stable/index.html.

[7] Krishnan, R., &Rajeswari,P.(2018). "Automatic Speech Recognition for Tamil Language Using Deep Learning." International Journal of Engineering & Technology, 7(4), 1235– 1240.

[8] Reddy, V.R., &Rao,K.S.(2016). "Speech Recognition for Telugu LanguageUsingDeep Neural Networks." Procedia Computer Science, 87, 258–263.

[9] Bird, S.,Klein, E., &Loper,E.(2009). NaturalLanguageProcessingwithPython.O'Reilly Media.

[10] TTS Documentation. Retrieved March 23, 2025, from https://gtts.readthedocs.io/en/latest/.

[11] Open Speech and Language Resources (OpenSLR). Retrieved March 23, 2025, from http://www.openslr.org/.

[12] NVIDIA CUD A Toolkit Documentation. Retrieved March 23, 2025, from https://docs.nvidia.com/cuda/.