Grenze International Journal of Engineering and Technology, June Issue

Voicify - A Multilingual and Expressive Text to Speech System

Sandip Shinde[1], Anushka Dabhade[2], Nirwani Adhau[3], Madhura Chitupe[4] and Niharika Deshmukh[5]
[1-5]Vishwakarma Institute of Technology, Pune, Computer Engineering, Pune, Maharashtra, India
Email: anushka.dabhade23@vit.edu, sandeep.shinde@vit.edu, nirwani.adhau23@vit.edu, madhura.chitupe23@vit.edu
niharika.deshmukh231@vit.edu

***Abstract*— The human voice is essential for storytelling, but creating speech that feels real and emotional instead of robotic is a significant challenge today. Many current Text-to-Speech (TTS) systems face an "uncanny valley" effect. They produce monotone voices that struggle with the details of Indic languages and fail to handle structured numbers accurately. Most existing models choose between being expressive and remaining stable. When processing long texts this often leads to issues like audio glitches or system crashes. This paper introduces Voicify, a new TTS framework developed for RPM Studios to solve these problems and improve broadcast media synthesis. Unlike standard single-engine systems, Voicify employs an Intelligent Routing mechanism. This mechanism directs English narrative text to a Parler-TTS engine for rich emotional expression while routing Hindi, Marathi, and structured data to gTTS for accurate language processing. This two-path system is supported by a real-time RAM monitor that keeps everything stable during large document conversions. By combining smart text normalization with efficient resource management, Voicify offers an audio pipeline that sounds more human-like and outperforms current solutions in both expressiveness and reliability. It is specifically designed to meet the diverse needs of the Indian broadcast industry.**

***Index Terms*— Hybrid TTS, Parler-TTS, Multilingual Synthesis, Resource-Aware Computing, Broadcast Media, Indic Languages.**

I. INTRODUCTION

In this digital era, there is a rising demand for automated voice-over solutions in the broadcast media industry, including OTT platforms, television, and film. Companies like RPM Studios, which provide technical solutions for broadcast content, find that moving from human voice-overs to AI-generated speech requires more than just clear pronunciation. It needs emotional depth, language flexibility, and the ability to manage complex text without crashing. Modern neural TTS models often struggle to balance expressiveness and accuracy. The generative systems like Parler-TTS definitely excel at producing expressive speech with emotions but they sometimes have trouble normalizing specific elements like currency symbols or numbers in languages such as Hindi and Marathi. The traditional engines , on the other hand, focus on delivering accuracy but usually lack the warmth , emotions and empathy needed for storytelling.

This study presents a Hybrid TTS Pipeline to address this issue. The system includes a pre-processing layer that uses regex rules for phonetic normalization along with a sentence-based chunking method.

Grenze ID: 01.GIJET.12.2.228_24

A key feature of this research is the Intelligent Routing Controller, which examines language and content type to direct text to the best engine. Specifically, Hindi and Marathi content, along with structured data like numbers and dates, is sent to gTTS for linguistic accuracy. Meanwhile, English narrative text is processed through Parler-TTS with guidance from a dynamic Voice Style Controller to produce human-like expressions. Furthermore, to ensure that the system can effectively manage large-scale media projects, we use a resource-aware monitor powered by the psutil library for Adaptive Chunk Sizing based on real-time RAM usage. This paper outlines how we implemented this architecture and evaluates its effectiveness in delivering smooth audio output suitable for professional broadcasting standards.

II. Literature Review

The earliest text-to-speech (TTS) systems were mainly concatenative and statistical based syntheses. Concatenative speech synthesis was a type of speech synthesis that produced natural speech by picking segments of speech that had been previously recorded and thus achieved good speech synthesis with large speech databases but with spectral discontinuities and little flexibility as a result of memory limitations. Statistical parametric synthesis overcame these problems and created speech with learned acoustic models to provide smooth transitions and less storage, but frequently created over-smoothed speech. Late hybrid synthesis methods used the merits of the two paradigms to enhance continuity and at the same time maintain computational efficiency [1].

Deep learning made a dramatic shift in the research of speech synthesis, and current neural text-to-speech (NTTS) systems are able to learn how to generate speech directly through text. Neural TTS brought text processing, acoustic modeling, and waveform generation together in end-to-end networks and reduced traditional pipelines, enhancing speech intelligibility and naturalness [2]. These achievements were the basis of the contemporary neural speech synthesis that is geared towards human-level quality.

New generation end to end systems minimized further the perceptual gaps between the synthesized and human speech. In the modern context, the absence of statistically significant perceptual differences between generated and recorded speech is defined as human-level quality and uses unified text-to-waveform architectures which collectively learn linguistic representation, time alignment, and acoustic variation [3]. Although these systems are very natural, they demand a lot of training environment and are still difficult to implement effectively in practical scenarios.

With the spread of TTS applications around the world, the multilingual speech synthesis was regarded as a significant area of research. Zero-shot multilingual systems use multilingual text pretraining to supervise speech generation of languages that lack corresponding speech data, enhancing access to low-resource languages [4]. The communication in practice can involve the code-mixed text inside one utterance of two or more languages. Code-mixed TTS systems overcome this with language recognition, spelling normalization, and transliteration rules, and increases pronunciation accuracy over monolingual systems [5]. Flow-based generative models and speech-prompted generative models are also more efficient multilingual models (enhancing inference speed and data efficiency) that still leave the challenge of robustness and consistent pronunciation across a wide range of different inputs unsolved [6].

In addition to the capability of more than one language, human-like speech production demands proper modeling of prosody, such as variation of the intonation of the pitch, rhythm, and expression dynamics. More recent models use pretrained speech representations to encode fined grained prosodic cues in synthesis. BERT-based approaches encode and decode speech segment-wise to obtain expressive embeddings, which are combined in decoding to enhance temporal continuity and perceived nature [7]. Yet, these techniques tend to add complexity to architectures and are aimed at improving them at the model-level instead of the system-level.

Although significant progress has been achieved in the area of speech naturalness, multilingual synthesis and expressive generation, current studies have focused more on the development of models under controlled environments. Real-world deployment applications include long documents, mixed-language documents, abbreviations, number expressions, and hardware constraints, in which traditional TTS pipelines often experience speech discontinuities and lack of fluency in narration.

To counter this, this paper is proposing Voicify, which is a hybrid multilingual and expressive TTS system that combines several synthesis engines through an adaptive pipeline. The system incorporates context sensitive text preprocessing, dynamic selection of engine, memory conscious long document narration as well as seamless audio concatenation in facilitating continuous and humanized speech generation. Voicify fills the gap between state-of-the-art neural TTS and real-world usable speech synthesis systems by integrating neural expressiveness and runtime adaptability.

III. METHODOLOGY

A. System Overview

Voicify is a proposed hybrid neural Text-To-Speech(TTS) system that will transform texts of documents into expressive and natural-sounding speech without loss of the linguistic accuracy in the multilingual environment. The conventional TTS pipelines are based on a single synthesis engine and usually trade expressiveness and pronunciation quality. To overcome this drawback, we are combining two complementary speech generation engines, parler-TTS, which is based on expressive neural speech synthesis and google Text-to-speech(gTTS), which is more accurate and has pronunciation of structured information including numbers, currency and mixed language pieces.

There are six major stages of the system architecture:

(1) Input Acquisition, (2) Document Text Extraction, (3) Standardization and Pre-processing of the text, (4) Linguistic Routing and Linguistic Analysis, (5) Hybrid Speech Generation and (6) Audio Post-processing and Succession. The entire workflow aims at providing maximum expressiveness , linguistic accuracy, and effective processing of long textual data like scripts, reports, and multimedia narration content.

Input Acquisition

Where users can input the required document in text, pdf or word format into the system. Select the preferred voice personalities to be present in the synthesized audio file, and the language the audio should be in out of Hindi/Marathi/English. The user also has the liberty to select the tone of the synthesized audio, example, dramatic, emotional, natural, documentary etc. Once a user inputs his preferred choices, the system acquires the files, and further processing and synthesization takes place.

Document Processing and Input Acquisition

Some of the input formats supported by the system are plain text (.txt), PDF (.pdf), and Word documents (.docx). These are the formats of the media production and content creation pipelines. Specialized document parsing libraries are used to extract the text. The processing of PDF documents is done by PyPDF2, which extracts textual layers in the document structure. The python-docx is used to extract paragraphs in Word documents but maintains the order of the document.

TABLE I: ACCEPTED INPUT FORMAT

Format	Extraction Method	Library Used
Plain Text (.txt)	Direct file read	Python I/O
PDF (.pdf)	Page-wise text extraction	PyPDF2
Word (.docx)	Paragraph parsing	python-docx

TABLE II: MAPPING OF INPUTS WITH OUTPUTS

Input Type	Example Input	Normalized Output
Numbers	2500	two thousand five hundred
Currency	₹500	five hundred rupees
Percentage	5%	five percent
Dates	12/10/2024	October twelve twenty twenty four

B. Text Normalization and Preprocessing

Raw text that has been copied out of documents frequently incorporates format discrepancies, abbreviations, numeral figures, date phrases, and phrases that might negatively affect the naturalness of the speech that has been synthesized. That is why a text normalization module is used to transform the raw textual data into speech inputs. Normalization incorporates the processes as follows:

Numerical Expansion

This is done by using the num2words library which translates numerical expressions into textual ones. Such change forbids the TTS system to pronounce numbers one by one.

Example:

2500 → "two thousand five hundred"

23.5 → "twenty three point five"

Conversion of currencies and units.

Currency symbols and measuring units are recognized with the assisting of regular expression rules. These words are elaborated into complete spoken words.

Example:
Pr 2500- "two thousand five hundred rupees"
5kg → "five kilograms"

Date Normalization
The dateparser library is used to convert dates across formats to a standard format of speaking.
Example:
12/10/2024 → October 12, 2024
This stage of normalization plays a large role in enhancing speech intelligibility and accuracy in pronunciation.

C. Linguistic Analysis and Language Detection

In order to process multilingual scripts and code-mixed data, the system analyzes sentences at the linguistic level.
Regular expression-based punctuation detection is used to group the mixture of text into individual sentences.
The sentences are then analyzed with the help of a language detection library, where the language is recognized as the dominant language based on probabilistic language models.
Languages that are detected are classified under the following:
- English
- Hindi
- Marathi
- Other detected languages

This is the classification that allows smart routing of what is being typed to the most appropriate speech synthesis engine.

D. Hybrid Speech Generation Architecture

One important and prominent input of the suggested system is the hybrid speech synthesis mechanism, which is dynamically used to choose two TTS engines based on linguistic and structural features of the text.

Parler-TTS Expressive Neural Speech Generation
The use of Parler-TTS is in expressive narration and storytelling. It is a transformer-based neural speech generation model that is able to condition speech output through descriptive prompts.
The speech production system works by feeding the model with two inputs:
- Content Text
- Style Description Prompt

Example prompt: Warm and emotional Indian female storyteller with emotional tones.
This type of conditioning is based upon the use of prompts, enabling control of such prosodic features as:
Pitch variation, Speech rhythm, Emotional tone, Narrative emphasis

Speech Generation: Accuracy-Focused (gTTS)
When dealing with blocks of complicated numeric phrases, currency values, or particular language sections, the system sends text to Google Text-to-Speech (gTTS).
This engine ensures:
- Proper articulation of numbers.
- Proper construction of dates and units.
- Hindi and Marathi have better pronunciation.

E. Memory-Aware Chunking Method.

The problem is that long documents may overflow the memory or decrease the prosodic quality in case they are treated as one line of input.
In order to handle this issue, the system uses a sentence-level chunking system with a runtime memory monitoring.
Before speech production, the document is further subdivided into smaller parts. Segmentation is done separately to avoid high memory usage.
The usage of system memory is checked with the help of the psutil library. When the memory use surpasses a configured limit (e.g. 80 percent), the system dynamically changes over to the lightweight gTTS engine to keep the system stable

TABLE III: CHUNK PROCESSING STRATEGY.

Step	Description
Sentence Segmentation	Split input text into individual sentences
Memory Check	Monitor RAM utilization via psutil
Engine Selection	Assign Parler-TTS or gTTS based on conditions
Audio Generation	Produce speech segment for current chunk
Cache Clearing	Release memory resources after each segment

F. Audio Post-Processing and Succession

Once individual speech segments are created, the system undertakes the audio post-processing to make sure that there is smooth playback.

The Pydub library is used to merge audio pieces with a constant sampling rate. Also, punctuation pauses are added between the segments resembling natural speech rhythm.

Random modification of the speed of playback is introduced slightly to allow human speech to have variability. Lastly, the combined audio stream is saved in the form of a WAV file, which is also lossless audio that can be used in professional media operations.

Pause durations are determined based on punctuation:

A slight random variation in playback speed is introduced to simulate human speech variability.G.

G. Implementation

The system is coded in Python, with the combination of deep learning, natural language processing, and audio processing libraries.

H. Technologies Used

Following are the technologies used for the proposed system:

Python 3.x is used as the main programming language, Parler TTS for the neural speech engine, gtts as the Secondary TTS engine, PyTorch for Deep Learning Framework, langdetect, num2words, dateparser, regex for NLP libraries, pydub, soundfile for audio processing, PyPDF2, python-docx for Document Processing and psutil for System Monitoring

System Architecture

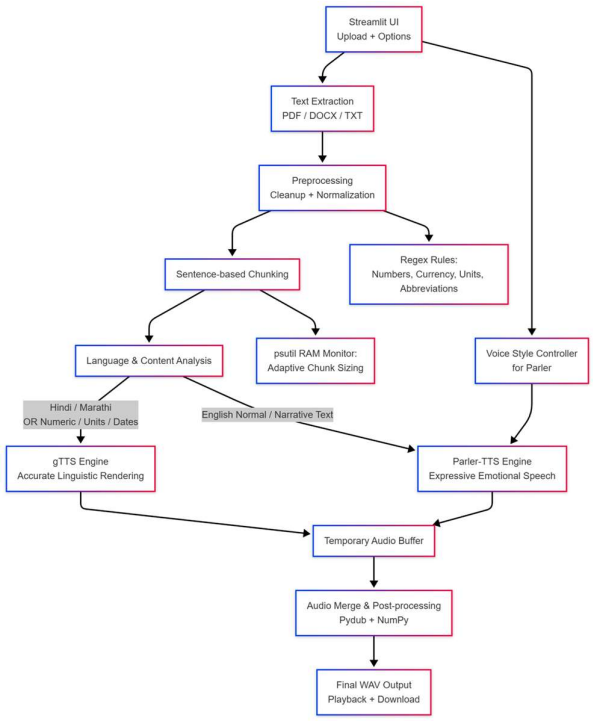

Fig.1. System architecture of the proposed Voicify TTS framework

IV. RESULTS

The Voicify system was evaluated and checked for accuracy on various collections of multilingual documents to assess the accuracy and robustness of the system in real world text-to-speech scenarios. The evaluation testing dataset consisted of documents containing numerical expressions, percentages and other mathematical symbols, currency values, and multilingual text segments. The system was tested on a machine with 16GB and 8 GB RAM, and Intel i7 processors.

A total of 15 participants (students and developers who are familiar with speech systems) participated in the survey, and fairly evaluated the audio generated by the system. Additionally, 80% participants had previously used voice assistants or TTS systems, ensuring that responses were informed by prior experience with speech synthesis systems and technologies.

A. Naturalness of synthesized speech

Participants evaluated the naturalness of the generated speech using a five-level perceptual scale.

TABLE IV: NATURALNESS RATINGS OF SYNTHESIZED SPEECH

Rating	Percentage
Very Robotic	6.7%
Mostly Robotic	6.7%
Somewhat Natural	6.7%
Natural	53.3%
Very Natural	26.7%

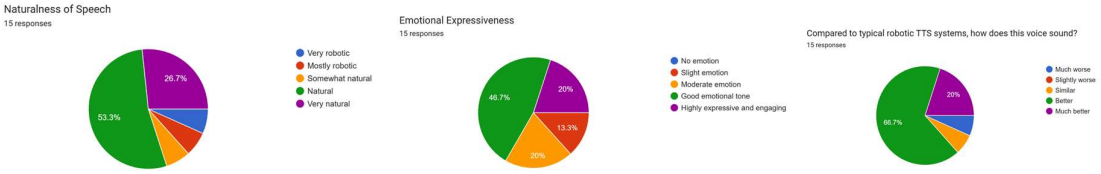

Fig.2 Statistics of speech naturalness ratings for the proposed system

B. Emotional expressiveness

Participants also evaluated the emotional expressiveness of the synthesized voice.

TABLE V: EMOTIONAL EXPRESSIVENESS RATINGS OF SYNTHESIZED SPEECH

Rating	Percentage
No emotion	0%
Slight emotion	13.3%
Moderate emotion	20%
Good emotional tone	46.7%
Highly expressive	20%

Interpretation:

- 66.7% rated the voice synthesized by the system as emotionally expressive.
- No participants reported the voice having no emotional tone.

C. Comparison to Traditional TTS systems

Participants also compared Vocify with typical robotic TTS systems that they had experienced before.

These results demonstrate that Voicify- A multilingual and expressive text to speech system produces natural, expressive, and comfortable speech suitable for long-form multilingual content generation.

REFERENCES

[1] S. Tiomkin, D. Malah, S. Shechtman and Z. Kons, "A Hybrid Text-to-Speech System That Combines Concatenative and Statistical Synthesis Units," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 19, no. 5, pp. 1278-1288, July 2011, doi: 10.1109/TASL.2010.2089679.

[2] X. Tan, T. Qin, F. Soong, and T.-Y. Liu,"A survey on neural speech synthesis,"arXiv preprint arXiv:2106.15561, 2021.

[3] X. Tan et al.,“NaturalSpeech: End-to-End Text-to-Speech Synthesis With Human-Level Quality,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 46, no. 6, pp. 4234–4245, Jun. 2024, doi: 10.1109/TPAMI.2024.3356232.
[4] T. Saeki, S. Maiti, X. Li, S. Watanabe, S. Takamichi, and H. Saruwatari, “Learning to speak from text: Zero-shot multilingual text-to-speech with unsupervised text pretraining,” in Proc. Int. Joint Conf. Artif. Intell. (IJCAI), 2023.
[5] S. Sitaram and A. W. Black, “Speech synthesis of code-mixed text,” in Proc. Annu. Conf. Int. Speech Commun. Assoc. (INTERSPEECH), 2016, pp. 1973–1977.
[6] S. Kim, K. J. Shih, R. Badlani, J. F. Santos, E. Bhakturin, M. Desta, R. Valle, S. Yoon, and B. Catanzaro,“P-Flow: A fast and data-efficient zero-shot TTS through speech prompting,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), 2023, pp. 1–5.
[7] L. Chen, Y. Deng, X. Wang, F. K. Soong and L. He, "Speech Bert Embedding for Improving Prosody in Neural TTS,”ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 2021, pp. 6563-6567, doi: 10.1109/ICASSP39728.2021.9413864.