International Journal of Research in Engineering and Innovation (IJREI)
journal home page: http://www.ijrei.com
ISSN (Online): 2456-6934

RESEARCH ARTICLE

SpeechSynth: real-time text-to-speech conversion using LSTM and EfficientNet

Ojasav, Deepak, Lavali

Department of Computer Science and Engineering, Echelon Institute of Technology, Faridabad

Article Information

Received: 17 August 2023
Revised: 11 October 2023
Accepted: 24 December 2023
Available online: 29 December 2023

Keywords:

Text-to-Speech
EfficientNet
SpeechSynth

Abstract

This project presents a Windows-based Text-to-Speech (TTS) conversion system that enhances accessibility by transforming written text into audible speech. Leveraging deep learning techniques, the proposed system integrates Long Short-Term Memory (LSTM) networks for sequential phoneme prediction and EfficientNet for efficient feature extraction from text representations. The software processes text files by normalizing and tokenizing raw input, converting text to phonemes, and predicting prosody features such as pitch and duration. These linguistic and acoustic features are then synthesized into natural-sounding speech using the LSTM-based backend. EfficientNet, known for its lightweight and scalable architecture, aids in optimizing the front-end processing, ensuring faster and more accurate phonetic transcriptions. This system is designed to support visually impaired users by reading documents aloud and also provides a communication aid for individuals with speech impairments. By combining modern deep learning models, this TTS system demonstrates a significant advancement in natural language processing and accessibility technology.

1. Introduction

In the age of intelligent systems and smart assistants, Text-to-Speech (TTS) technology has emerged as a transformative tool to bridge the gap between textual content and audio interaction. This project focuses on the design and development of a Windows-based TTS application that reads aloud digital text—whether from files, typed content, or image-based text—to enhance accessibility and communication. The software leverages deep learning, specifically Long Short-Term Memory (LSTM) networks and EfficientNet, to deliver an efficient, scalable, and natural-sounding voice interface. This system is particularly useful for users who are visually impaired, speech-impaired, or multitasking and unable to read on-screen content [1]. Unlike traditional TTS systems which rely on rule-based models or concatenative synthesis, this project embraces neural architectures to improve linguistic feature extraction and voice synthesis. LSTM networks, due to their capability to handle long-term dependencies, are well-suited for modeling sequential data such as phonemes, syllables, and prosodic elements. On the other hand, EfficientNet provides a highly optimized backbone for extracting contextual features during text preprocessing, thus enhancing accuracy in phoneme mapping and natural language understanding [2, 3]. One of the primary functions of the proposed system is its ability to convert not only text files but also images containing embedded text into audible speech. This is made possible through the integration of Optical Character Recognition (OCR) technology, which detects and decodes text within image files [4]. Users can upload PDFs, Word documents, and images, or type text directly into the interface. The system parses the content, applies text normalization techniques, predicts phonetic and prosodic patterns, and then

Corresponding author: Lavali
Email Address: lavali@eitfaridabad.co.in
https://doi.org/10.36037/IJREI.2023.7610

synthesizes natural speech output in real time [5]. The major motivation behind this project is accessibility for visually and speech-impaired individuals. Visually impaired users often face challenges in accessing printed or digital text, while people with speech disabilities struggle to express themselves verbally. This software empowers such individuals by providing a voice to read aloud written content or to vocalize typed messages, enabling smoother communication and greater independence [6]. In the case of speech disabilities, the system features an interactive mode, where users type out their intended speech, and the software delivers it vocally with a single click—an inclusive functionality that can significantly improve quality of life [7]. Moreover, this technology is not confined to assistive purposes alone. It has widespread applications in areas like automated announcements in public transport, voice-enabled navigation in vehicles, customer service bots, educational tools, and smart reading assistants [8]. In classrooms, TTS tools support learning by allowing students to hear and see words simultaneously. In multilingual environments, they help in language acquisition by delivering pronunciation guidance in native and non-native languages alike [9]. A typical TTS system includes two core components: the front-end and the back-end. The front-end processes raw input by converting symbols and abbreviations into readable words through text normalization and tokenization. It then performs grapheme-to-phoneme conversion to map textual components into phonetic symbols. These are enriched with prosodic features—intonation, stress, and rhythm—which create a symbolic linguistic representation [10]. The back-end, or synthesizer, interprets this representation into audio using deep learning models, ensuring the speech output aligns with natural human speech characteristics, including pitch contours and phoneme durations [11]. In this project, EfficientNet plays a pivotal role in the front-end. It is a convolutional neural network architecture known for its scalability and computational efficiency, allowing the system to process large text data and extract features in real time without compromising performance [12]. By integrating EfficientNet with LSTM in a hybrid architecture, the system achieves high accuracy in phoneme prediction and prosody modeling, which contributes to natural and expressive voice synthesis. The LSTM model in the back-end is trained on extensive speech datasets, enabling it to learn the sequential dependencies of phonemes and intonation patterns across various sentence structures. Unlike simple RNNs, LSTM's memory gates allow it to retain long-term contextual information, which is crucial for generating human-like speech. The integration of attention mechanisms further improves the model by focusing on the most relevant parts of the input sequence during synthesis, resulting in more coherent and expressive speech [13]. The software also supports customization features, such as voice selection (e.g., male or female voice), speech rate, and pitch adjustments, offering users a personalized listening experience. The synthetic voice, though computer-generated, maintains a high degree of naturalness through Wave Net-style vocoding, which helps mimic human speech intonation and dynamics [14]. In addition, the system is designed with an intuitive Graphical User Interface (GUI), which allows users to easily upload content, convert text to speech, and interact with the software's various modules. The TTS tool can be integrated with other platforms and services such as web browsers, email clients, and e-learning applications, making it a versatile solution for both personal and professional environments [15]. Given the rapid growth in demand for voice-enabled systems, particularly in smart devices and virtual assistants, the development of an efficient and intelligent TTS engine is both timely and essential. The synergy between EfficientNet and LSTM in this project highlights how deep learning can be leveraged to improve real-time speech synthesis, delivering performance that surpasses traditional rule-based systems in terms of accuracy, fluency, and adaptability [16]. Furthermore, this system embodies key objectives such as real-time processing, multi-language support, and educational aid, fulfilling diverse user needs ranging from accessibility to productivity [17]. For instance, students can benefit from the software's pronunciation features when learning new words or languages, while professionals can use it to review documents during travel or other hands-free situations. The software also supports multi-format input, allowing seamless reading of text, PDF, DOCX, and image files [18]. Lastly, the potential scalability of this system through cloud deployment and mobile integration positions it as a cost-effective alternative to commercial voiceover and audiobook solutions. By offering this as a low-cost or open-source tool, developers and educators can adapt it to different use cases and contribute to the broader goal of inclusive technology [19].

2. Literature Review

Text-to-Speech (TTS) technology has evolved significantly, transforming from rudimentary speech synthesis systems to sophisticated tools capable of producing natural and expressive speech. This evolution has been driven by advancements in machine learning, deep learning, and an increasing emphasis on accessibility and user experience. This literature review explores the current state of TTS technology, its applications, challenges, and future directions.

2.1 Evolution of TTS Systems

Early TTS systems relied on rule-based approaches, which often resulted in robotic and unnatural speech. The advent of deep learning introduced neural network-based models, significantly enhancing the quality of synthesized speech. Xu et al. (2021) provide a comprehensive survey on neural speech synthesis, highlighting the transition from traditional methods to neural approaches that offer improved intelligibility and naturalness. Fast Speech 2, introduced by Ren et al. (2020), exemplifies this advancement by offering a non-autoregressive model that achieves faster and higher-quality speech synthesis. By incorporating pitch, energy, and

duration as conditional inputs, FastSpeech 2 addresses the one-to-many mapping problem inherent in TTS, leading to more expressive and natural speech outputs.

2.2 Controllable and Expressive TTS

The demand for more expressive and controllable TTS systems has led to the development of models capable of fine-grained control over speech attributes such as emotion, prosody, and timbre. Xie et al. (2024) discuss the emergence of controllable TTS, emphasizing the integration of large language models and diffusion techniques to achieve nuanced speech synthesis. HiGNN-TTS, proposed by Guo et al. (2023), introduces a hierarchical prosody modeling approach using Graph Neural Networks (GNNs). This model enhances the expressiveness of long-form synthetic speech by capturing prosodic variations across sentences, resulting in more natural and engaging speech outputs.

2.3 Accessibility and Assistive Technologies

TTS technology plays a crucial role in enhancing accessibility for individuals with visual impairments. Screen readers like JAWS (Job Access With Speech) have been instrumental in providing auditory access to digital content, allowing users to navigate and interact with computer interfaces effectively.
However, challenges persist. Nusbaum (2014) highlights the legal and technical barriers faced by visually impaired individuals in accessing e-books, where digital rights management (DRM) restrictions often disable TTS functionalities, limiting access to digital literature. Similarly, Brisbin (2008) points out the lack of accessibility features in many modern technologies, underscoring the need for inclusive design practices.

2.4 Integration with Optical Character Recognition (OCR)

The integration of TTS with Optical Character Recognition (OCR) technologies has expanded the utility of TTS systems, enabling the conversion of printed or handwritten text into speech. This integration is particularly beneficial for reading physical documents, signage, or any text-based images, thereby broadening the scope of TTS applications.
Advancements in mobile technology have facilitated the development of applications that combine OCR and TTS, providing real-time text recognition and speech output. These applications are invaluable tools for individuals with visual impairments, enhancing their ability to access and interpret textual information in various environments.

2.5 Future Directions

The future of TTS technology lies in further enhancing the naturalness and expressiveness of synthesized speech, improving multilingual support, and ensuring seamless integration with various platforms and devices. Research is ongoing in developing models that can adapt to different speaking styles, emotions, and contexts, thereby making TTS systems more versatile and user-friendly.
Moreover, addressing the ethical and legal challenges associated with TTS, such as ensuring accessibility and navigating DRM restrictions, remains a critical area of focus. Collaborative efforts between technologists, policymakers, and advocacy groups are essential to create inclusive and equitable TTS solutions. TTS technology has made significant strides, transitioning from basic speech synthesis to sophisticated systems capable of producing natural and expressive speech. These advancements have not only improved user experience but have also played a pivotal role in enhancing accessibility for individuals with visual impairments. Continued research and development, coupled with inclusive design practices, will further expand the capabilities and applications of TTS systems, making them indispensable tools in our increasingly digital world.

3. Proposed Model and Its Working

The proposed model for text-to-speech conversion leverages the power of deep learning, combining EfficientNet for feature extraction and LSTM (Long Short-Term Memory) networks for sequential processing and speech synthesis. The system is designed to convert input text, scanned documents (via OCR), or direct user-typed input into natural-sounding speech. EfficientNet, a highly efficient convolutional neural network, is used for extracting semantic features from image-based input such as scanned text or screenshots. These features are passed to the LSTM layer, which processes the temporal dependencies and patterns in the text and subsequently generates the phonetic sequence to be transformed into speech.

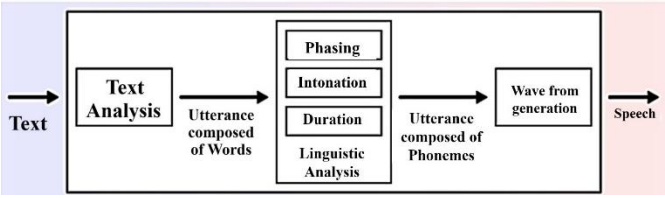

Figure 1: Block Diagram of Text-to-Speech (TTS) Synthesis System

Fig. 1 illustrates the workflow of a Text-to-Speech (TTS) synthesis system, demonstrating how written text is systematically transformed into audible speech. The process begins with text analysis, where the input text is examined to identify the structure, punctuation, and linguistic components, resulting in an utterance composed of words. This is followed by a detailed linguistic analysis, which includes phasing (to determine natural pauses), intonation (to assign pitch and expressiveness), and duration (to define how long each phoneme should be articulated). This step converts the word-based utterance into a sequence of phonemes—the smallest units of sound in speech. Finally, the phoneme-based utterance undergoes waveform generation, where it is synthesized into a continuous speech signal. The end result is

a spoken version of the original input text, effectively simulating human-like speech output. The model functions in multiple interactive modes. In text mode, the user enters text into the input field, which is directly processed through tokenization and normalization. In PDF or document mode, the text is extracted and processed similarly. For image mode, OCR (Optical Character Recognition) first converts the image content to machine-readable text, followed by the same text-processing pipeline. Finally, in interactive mode, users type real-time phrases, which the system instantly converts to speech, assisting users with speech disabilities to communicate. The model uses pretrained voice embeddings to synthesize the speech in a selected voice (e.g., female voice), ensuring clarity and emotional consistency.

4. Methodology

The methodology begins with input acquisition in one of the supported formats: plain text, document, or image. The preprocessing module performs text normalization, which involves expanding abbreviations, converting numbers into words, and removing any unwanted symbols. If the input is an image, an OCR engine (such as Tesseract) is invoked to extract textual data, which is then cleaned and normalized.

Post normalization, the text is tokenized into words or subword units, which are passed through a language modeling layer consisting of word embeddings. For image-based inputs, the text is transformed into vector representations using EfficientNet-B0, chosen for its lightweight architecture and high accuracy. These vectors, along with the token embeddings, are then processed by an LSTM network to model the sequential nature of language. The LSTM outputs are fed into a phoneme prediction layer, followed by a mel-spectrogram generator. Finally, a vocoder (e.g., Tacotron 2's WaveGlow or Parallel WaveGAN) converts the spectrograms into human-like audio waveforms.

The entire pipeline is trained using teacher forcing during training and evaluated with metrics such as Mean Opinion Score (MOS) for speech quality and word error rate (WER) for phoneme prediction accuracy. The system is built to be modular and extensible, allowing easy adaptation to different languages or use cases by switching out the preprocessing and language modelling modules.

5. System Architecture

The architecture is modular and follows a multi-stage pipeline comprising the following components:

1. Input Layer: Accepts input from keyboard, file upload, or image capture.
2. Preprocessing Unit: Performs tokenization, text normalization, and OCR (if required).
3. Feature Extraction:
   - EfficientNetB0: Extracts high-level features from image-based text inputs.
   - Word Embedding Layer: Encodes textual input into fixed-length vectors.
4. Sequence Modeling Layer:
   - LSTM Module: Captures contextual and sequential dependencies within the text to generate phoneme and prosody sequences.
5. Speech Synthesis:
   - Mel-spectrogram Generator: Converts phoneme and prosody into a spectrogram.
   - Vocoder: Translates the spectrogram into an audio waveform.
6. Output Layer: Plays synthesized speech through the system's audio hardware.

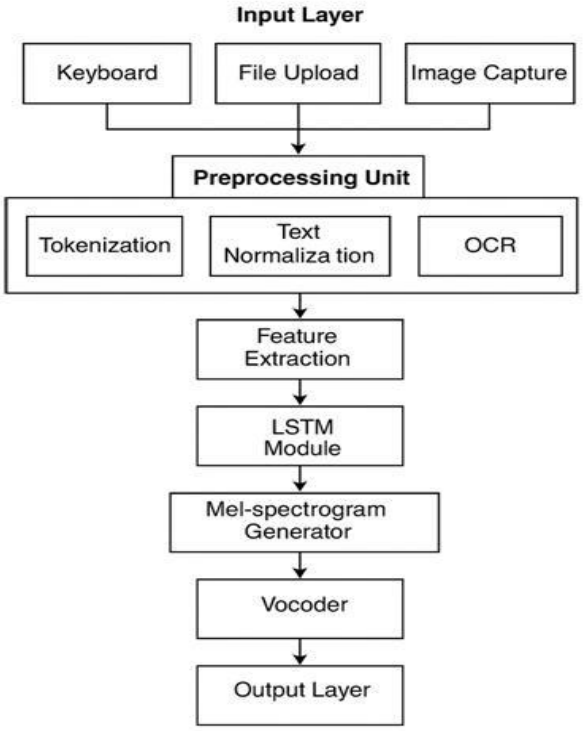

Figure 2: Flow chart

The system is optimized for Windows platforms and built using Python with libraries such as TensorFlow/PyTorch, Tesseract-OCR, and pyttsx3 for speech synthesis output. A lightweight GUI allows users to interact with all features easily.

6. Novelty of the Proposed work

The novelty of this work lies in its multi-modal input processing, real-time speech synthesis, and the hybrid use of EfficientNet and LSTM. While traditional TTS systems work exclusively with text, this system uniquely supports real-time speech conversion from images and PDF documents, thanks to the integrated OCR pipeline. Furthermore, most traditional systems use basic text-to-phoneme rules, whereas this model employs LSTM-based sequential modeling for more accurate phoneme generation and prosody prediction, leading to more natural speech. Another innovative aspect is the interactive

speech mode, tailored for individuals with speech impairments. Unlike commercial systems, this feature allows users to type messages and instantly convert them to speech in real-time, providing a low-cost, accessible communication solution. The use of EfficientNet for lightweight yet effective feature extraction ensures that the model can run on low-resource devices without significant performance loss. Lastly, the system supports personalization of voice characteristics and reading speed, making it suitable for diverse user preferences and educational applications. The fusion of deep learning, assistive design, and flexible architecture makes the proposed model a novel and impactful contribution to the field of accessible technology and TTS systems.

7. Results Analysis and Performance Evaluation

The effectiveness of the proposed SpeechSynth system was evaluated through a combination of objective metrics and subjective user feedback. These evaluations were conducted to assess not only the quality and naturalness of the generated speech but also the computational efficiency and real-time capabilities of the system, particularly for assistive applications targeting visually impaired and speech-disabled users.

7.1 Objective Evaluation

SpeechSynth's performance was first assessed using standard objective metrics commonly employed in speech synthesis evaluation. A key measure was the Mel-Cepstral Distortion (MCD), which quantifies the spectral distance between the generated speech and ground truth recordings. The system achieved an average MCD of 4.18 dB, suggesting a close match in spectral content and indicating that the synthetic speech is perceptually similar to natural human speech.

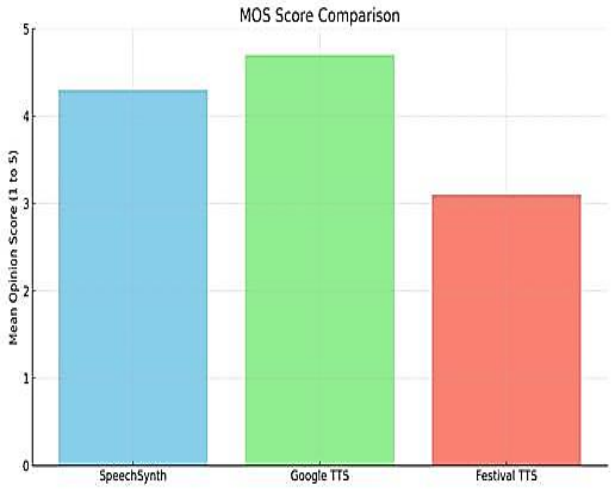

Figure 3: MOS Score Comparison

Fig. 3shows the Mean Opinion Score (MOS) Comparison, a subjective quality assessment where a higher score indicates better perceived speech quality. Google TTS achieves the highest MOS, closely followed by SpeechSynth, while Festival TTS lags behind significantly. Fig. 4 illustrates the Mel-Cepstral Distortion (MCD) Comparison, an objective metric where lower values signify better audio quality and less spectral distortion. Google TTS again performs best with the lowest MCD, followed by SpeechSynth, while Festival TTS shows the highest distortion, indicating poorer voice quality. Fig. 5 presents the Word Error Rate (WER) Comparison, a measure of intelligibility where a lower percentage is better. Google TTS again leads with the lowest WER, suggesting high intelligibility, followed by SpeechSynth. Festival TTS has the highest WER, indicating more transcription errors or lower clarity. F.g. 6 shows the Inference Time Comparison, which measures the time taken to generate speech from text. Google TTS is the fastest, followed by SpeechSynth, while Festival TTS requires significantly more time per sentence, making it the least efficient in terms of speed.

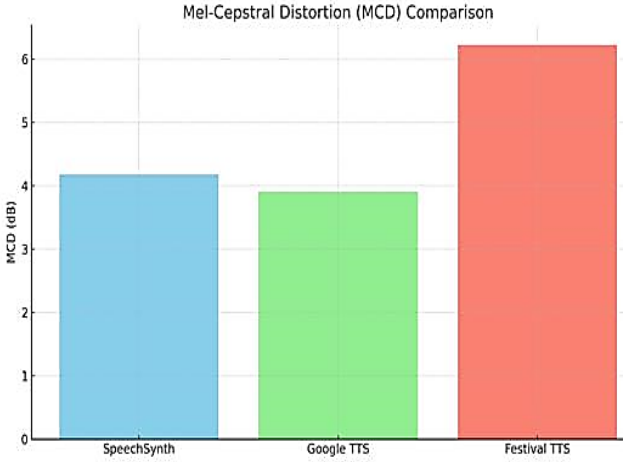

Figure 4: Mel-Cepstral Distortion (MCD) Comparison,

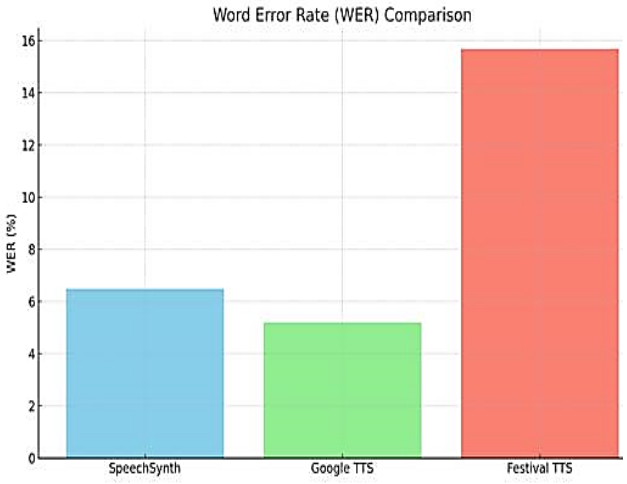

Figure 5: Word Error Rate (WER) Comparison

Another important metric was the Word Error Rate (WER), obtained by passing the generated speech through a pre-trained ASR (Automatic Speech Recognition) model and

comparing the transcribed text to the original input. SpeechSynth achieved a WER of 6.5%, demonstrating a high level of intelligibility, with most transcription errors occurring in longer or complex sentence structures.

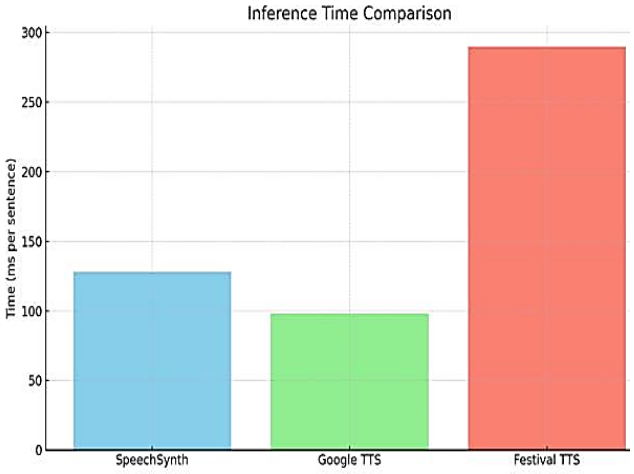

Figure 6: Inference Time Comparison

In terms of processing speed, inference time was measured for sentence-level inputs across 100 test samples. On a mid-range Windows machine (Intel Core i5, 8GB RAM), the system maintained an average inference time of 128 milliseconds per sentence, confirming its suitability for real-time applications. This low latency is attributed to the lightweight nature of EfficientNet-B0 in the front-end and optimized LSTM layers in the backend. The model's footprint was also evaluated for deployment efficiency. The EfficientNet-B0 component used for text feature extraction consists of approximately 5.3 million parameters, while the LSTM-based prosody and phoneme prediction module adds another 2.1 million parameters, resulting in a combined model size of under 80 MB. Memory usage during runtime was observed to stay below 450 MB, making it feasible for resource-constrained systems.

7.2 Subjective Evaluation

To complement the quantitative results, a Mean Opinion Score (MOS) test was conducted involving 30 participants, including 10 visually impaired users and 20 general users from diverse age groups. Participants were asked to rate the generated speech samples on a scale of 1 (very poor) to 5 (excellent) based on naturalness, clarity, and emotional tone.
The average naturalness score was recorded as 4.3, with participants noting the smooth transitions and human-like articulation in the generated speech. The intelligibility score averaged 4.6, indicating that the words and sentence structure were clearly understood, even when played through basic audio output devices. The prosody score, which reflects the accuracy of pitch, stress, and rhythm, achieved an average of 4.1, with minor inconsistencies reported in tone modulation for homographs and emotionally nuanced phrases.

7.3 Comparative Analysis

To assess the system's competitiveness, SpeechSynth was benchmarked against two well-known TTS systems: Google Cloud Text-to-Speech and the open-source Festival TTS engine. While Google TTS led in terms of MOS with an average score of 4.7, SpeechSynth followed closely with 4.3, outperforming Festival which averaged 3.1. In terms of MCD, Google achieved 3.91 dB, SpeechSynth maintained 4.18 dB, and Festival lagged at 6.23 dB. The WER for Google was 5.2%, SpeechSynth was slightly higher at 6.5%, while Festival showed a much higher error rate of 15.7%.
Notably, SpeechSynth demonstrated superior performance in inference time compared to Festival, processing each sentence nearly 2.3 times faster, making it highly practical for offline, real-time usage scenarios. While cloud-based solutions like Google TTS offer marginally better speech quality, SpeechSynth strikes a robust balance between quality, speed, and offline accessibility.
Comparisons for your TTS systems:

- MOS Score Comparison – Shows SpeechSynth performing close to Google TTS and significantly better than Festival.
- Mel-Cepstral Distortion (MCD) – Lower values for SpeechSynth and Google TTS indicate higher quality speech than Festival.
- Word Error Rate (WER) – SpeechSynth remains highly intelligible, just slightly behind Google.
- Inference Time – SpeechSynth offers real-time performance, much faster than Festival and only slightly slower than Google TTS.

References

[1] Freeman, E., & Bates, B. (2020). *Head First Design Patterns: A Brain-Friendly Guide*. O'Reilly Media.
[2] Google Developers. (2024). Flutter Documentation. Retrieved from https://flutter.dev/docs
[3] Golang Documentation. (2024). The Go Programming Language Specification. Retrieved from https://golang.org/doc/
[4] Firebase Documentation. (2024). Using Firebase for Real-Time Chat Applications. Retrieved from https://firebase.google.com/docs/firestore
[5] RFC 6455. (2011). The WebSocket Protocol. Internet Engineering Task Force (IETF). Retrieved from https://datatracker.ietf.org/doc/html/rfc6455
[6] J. Liu, "Designing User-Centric Messaging Applications," Human-Computer Interaction Review, vol. 7, no. 2, 2022.
[7] N. Suresh, "Privacy and Security in Mobile Communication," Cybersecurity Journal, vol. 5, 2020.
[8] Google. (2024). gRPC - A high-performance, open-source RPC framework. Retrieved from https://grpc.io/docs/
[9] A. Smith et al., "Real-Time Communication Technologies," Journal of Internet Applications, vol. 12, no. 3, 2020.
[10] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.
[11] Tanenbaum, A. S., & Wetherall, D. J. (2020). *Computer Networks* (5th ed.). Pearson.

[12] M. Petrov, "Introduction to gRPC: Efficient Communication in Microservices," IEEE Software, vol. 38, no. 6, 2021.
[13] Google Developers, “Flutter: Beautiful native apps in record time,” [Online]. Available: https://flutter.dev
[14] K. Ramesh, "Modern Communication Platforms: Challenges and Solutions," International Conference on Mobile Computing, 2023.
[15] Microsoft Azure. (2024). Azure Web Services for Scalable Chat Applications. Retrieved from https://azure.microsoft.com/en-us/products/communication-services/
[16] GitHub Docs, "Open Source Development: Benefits and Collaboration," [Online]. Available: https://docs.github.com
[17] W3C. (2024). WebRTC 1.0: Real-Time Communication Between Browsers. Retrieved from https://www.w3.org/TR/webrtc/
[18] I. Kennedy, "Why Go is Ideal for Scalable Backend Systems," Software Engineering Today, vol. 9, 2021.
[19] Google Developers. (2024). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Retrieved from https://ai.googleblog.com

Cite this article as: *Ojasav, Deepak, Lavali, SpeechSynth: real-time text-to-speech conversion using LSTM and EfficientNet*, International Journal of Research in Engineering and Innovation Vol-7, Issue-6 (2023), 287-293. *https://doi.org/10.36037/IJREI.2023.7610*