Text-to-Audio Model Research

1. Introduction

In the digital era, enhancing accessibility and user engagement with multimedia content is increasingly important. The project in focus involves processing a list of video URLs provided by a user, extracting their audio, transcribing the speech using OpenAI's Whisper model, and enabling retrieval-based interaction with the transcribed content. My specific role in this project is to research and propose a suitable Text-to-Speech (TTS) model to convert retrieved textual content back into audio. This adds a dynamic, audio-based interaction layer to the system, enhancing usability for diverse user groups, including those with visual impairments or learning differences.

2. Use Case Context

The complete system pipeline involves:

1. Input: User uploads video URLs.

2. Processing:

   - Audio is extracted from each video.

   - OpenAI Whisper model is used to generate transcripts.

3. Retrieval: Users can query or search the transcribed content.

4. Output: The relevant text is returned.

5. Text-to-Audio Layer : The selected text is converted to speech using a TTS model.

This audio output enhances interactivity and accessibility, particularly in educational or assistive applications.

3. Text-to-Speech (TTS) Technology Overview

TTS systems transform written text into spoken audio. Modern systems are often composed of the following modules:

- Text Normalization: Converts numbers, dates, abbreviations, etc., into readable text.

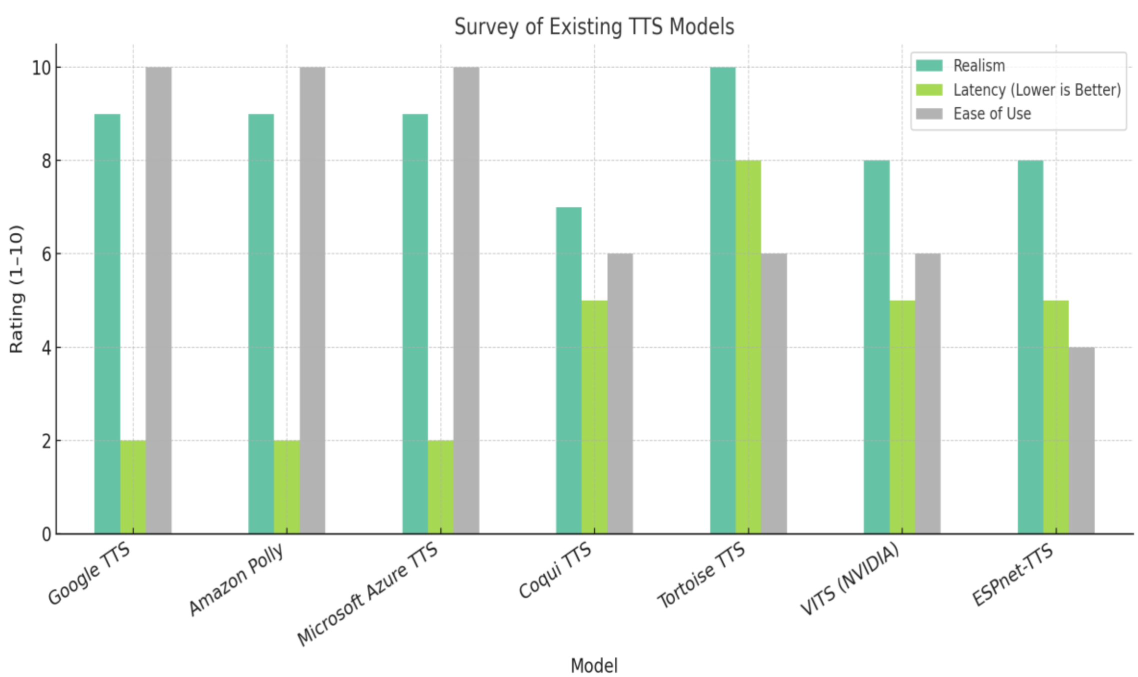
- Linguistic Analysis: Processes text to understand syntax and prosody.

- Acoustic Modeling: Predicts speech features from processed text.

- Vocoder: Synthesizes audio waveforms (e.g., using WaveNet, HiFi-GAN, or Griffin-Lim).

TTS technology has rapidly evolved, with neural TTS models offering high naturalness and expressive quality.

4. Survey of Existing TTS Models



5. Evaluation Criteria

To evaluate and select the most suitable TTS model, the following criteria were considered:

- Audio Quality (Naturalness and Intelligibility): How human-like and expressive is the generated audio?

- Multilingual and Accent Support: Does the model support various languages or dialects?

- Voice Cloning and Customization: Can it generate voices based on samples or allow custom tuning?

- Real-time Compatibility: Can it deliver low-latency performance suitable for live interaction?

- Resource Efficiency: What are the model's compute and memory requirements?

- Integration Ease: How straightforward is it to integrate with our existing tech stack?

- Licensing and Cost: Is it open-source or does it require a paid API?

6. Model Comparison and Recommendations

After evaluating the surveyed models against the criteria above, the following insights emerged:

- Tortoise TTS is ideal for offline, high-quality speech synthesis. Its realism is unmatched, but it has higher latency and compute demands. It is best suited for non-real-time playback.

- Coqui TTS offers a good balance between quality, flexibility, and open-source availability. It supports multilingual TTS, basic voice cloning, and is relatively easy to customize.

- VITS is suitable for real-time scenarios with relatively good naturalness and wide language support.

- Google, Amazon, and Microsoft TTS APIs offer high-quality, scalable services with very low latency and excellent language support, but incur usage costs and rely on cloud infrastructure.

Recommended Model(s):

1. Tortoise TTS – for use cases where quality is top priority and latency is tolerable.

2. Coqui TTS – for general-purpose, on-device, open-source use.

3. Google Cloud TTS – if cloud-based, fast, and scalable deployment is preferred.

7. Proposed Integration in System

- The TTS module will receive text output from the retrieval system.

- It will process the text using the selected TTS engine and generate corresponding audio.

- The audio can then be played back to the user via the web/mobile UI.

- Optional features:

  - Downloadable audio file.

  - Multiple voices or language selection.

  - Speed or pitch controls for playback.

8. Conclusion

This report reviewed the importance of TTS in augmenting a transcription-based video processing system. Based on a comprehensive survey and evaluation, Tortoise TTS and Coqui TTS emerge as strong open-source candidates for integration, while Google Cloud TTS stands out for its ease and scalability. Integrating TTS will significantly improve the system's accessibility and usability, catering to a broader audience and enabling multimodal content interaction.

9. References

- Tortoise TTS GitHub: https://github.com/neonbjb/tortoise-tts

- Coqui TTS GitHub: https://github.com/coqui-ai/TTS

- Google Cloud TTS: https://cloud.google.com/text-to-speech

- Amazon Polly: https://aws.amazon.com/polly/

- Microsoft TTS: https://azure.microsoft.com/en-us/products/cognitive-services/text-to-speech

- ESPnet: https://github.com/espnet/espnet

- VITS GitHub: https://github.com/jaywalnut310/vits