Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report
on
"Voice Patch: Context-Aware Speech Inpainting for Restoring Choppy Audio"

[Code No: COMP 303]

(For partial fulfillment of III Year/ I Semester in Computer Engineering)

Submitted by

Aashish Adhikari (Roll No. 4)
Swastik Aryal (Roll No. 9)
James Bhattarai (Roll No. 15)
Rikesh Panta (Roll No. 47)
Kritan Rijal (Roll No. 53)

Submitted to

Mr. Suman Shrestha

Department of Computer Science and Engineering

Submission Date: 02/07/2025

Bona fide Certificate

This project work on

“Voice Patch: Context-Aware Speech Inpainting for Restoring Choppy Audio”

is the bona fide work of

“Aashish Adhikari ,
Swastik Aryal,
James Bhattarai ,
Rikesh Panta,
Kritan Rijal”

who carried out the project work under my supervision.

Project Supervisor

_____

Prof. Dr. Bal Krishna Bal

Associate Dean, School of Engineering

Kathmandu University, Department of Computer Science & Engineering

Date: 02/07/2025

Acknowledgement

We would like to express our sincere gratitude to our Supervisor, Prof. Dr. Bal Krishna Bal who provided valuable insights, guidance throughout the project completion. The countless late night hours, meetings for continuous encouragement, insightful feedback and constant support during every phase of the project which we are deeply thankful for.

We also extend our heartfelt thanks to the Department of Computer Science and Engineering, Kathmandu University, for providing the necessary facilities, resources, and a supportive learning environment. This project report is the result of collective effort and contribution, and we are truly grateful to the direct and indirect involvement of all those who provided their time, support, suggestions, or motivation throughout our journey.

Abstract

The loss of semantic content in choppy audio poses a significant challenge that traditional signal-based restoration methods fail to address. This project introduces "VoicePatch," a context-aware speech inpainting pipeline designed to restore missing spoken phrases. The system integrates SileroVAD for precise speech segmentation, Whisper for transcription, a T5 language model for contextual prediction, and an E2-TTS engine for voice-cloned synthesis. A comprehensive evaluation confirmed the suitability of each component. The T5 model was selected over BERT-based architectures due to their single-token prediction limitation and over BART for its superior handling of targeted replacement. On a custom dataset, the T5 model achieved 96.3% sentence-level semantic similarity and 89.8% contextual word similarity, validating its effectiveness. Furthermore, SileroVAD demonstrated superior segmentation accuracy (0.96 ROC-AUC) over traditional methods, and E2-TTS was chosen for its subjectively more natural-sounding output. As a successful proof-of-concept, this work validates the modular approach for semantic audio repair and establishes a foundation for a future end-to-end model.

Keywords: Speech Inpainting, Audio Restoration, Natural Language Processing, Text-to-Speech, Voice Cloning, T5 Transformer.

Table of Contents

List of Figures

List of Tables

Acronyms/Abbreviations

ASR	Virtual Reality
BART	Random Access Memory
BERT	Dynamic Memory Allocation
CNN	Convolutional Neural Network
DiT	Diffusion Transformer
GMM	Gaussian Mixture Model
LLM	Large Language Model
MLM	Masked Language Modelling
MOS	Mean Opinion Score
NLP	Natural Language Processing
PCM	Pulse Code Modulation
RMS	Root Mean Square
ROC-AUC	Receiver Operating Characteristic - Area Under the Curve
TTS	Text-to-Speech
VAD	Voice Activity Detection
WER	Word Error Rate

Chapter 1 Introduction

1.1 Background

Choppy and jittery audio remains a prevalent issue in contemporary communication and recording systems, particularly in contexts such as Voice over IP (VoIP) calls, online conferencing, and digital media transmission. These distortions are primarily caused by network instability, packet loss, and synchronization errors during transmission, resulting in partial or complete loss of speech segments. Such degradations severely impact intelligibility and overall user experience by rendering speech fragmented or unintelligible.

Conventional audio enhancement techniques such as noise suppression, dereverberation, and waveform smoothing have shown efficacy in reducing background disturbances and improving acoustic clarity. However, they are generally ineffective in scenarios where the audio signal has suffered structural loss, such as missing words or phrases. In these cases, the absence of semantic content limits the effectiveness of purely signal-based methods.

Recent advancements in neural speech enhancement offer more robust solutions. Notably, VoiceFixer (Haohe et al., 2022) introduced a unified framework for high-fidelity speech restoration capable of addressing various distortions, including noise, reverberation, and clipping. Its architecture consists of an analysis stage that extracts intermediate-level features from degraded input and a synthesis stage that reconstructs the waveform via a neural vocoder. While such frameworks represent a significant leap in perceptual quality, they still focus on signal fidelity rather than semantic restoration. This gap highlights the need for a new class of solutions that not only enhance signal quality but also recover lost semantic content in speech, a challenge that this project aims to address by integrating natural language understanding with voice synthesis.

1.2 Objectives

The primary objective of this project is to develop a robust pipeline to restore choppy and jittered audio. The key objectives are to:

- Use Silero VAD to detect and segment unstable regions in the audio.
- Transcribe speech using Whisper, masking low-confidence segments.
- Predict missing content with a T5-based language model using contextual clues.
- Regenerate speech with E2 TTS, preserving the speaker's voice through voice cloning.
- Integrate all components into a cohesive pipeline that outputs restored, natural-sounding audio from degraded recordings.

1.3 Motivation and Significance

The motivation behind this project arises from the common frustration caused by choppy and unintelligible audio in communication scenarios such as VoIP calls, online meetings, and digital recordings. Traditional signal processing techniques often fail to recover the semantic content lost due to network instability and packet loss.

This project addresses that gap by leveraging Natural Language Processing (NLP) to infer missing speech content and voice cloning techniques to seamlessly synthesize it in the speaker’s original voice. Unlike existing solutions that focus primarily on signal-level enhancement, our approach integrates Automatic Speech Recognition (ASR) for detecting missing segments, context-aware NLP for semantic reconstruction, and Text-to-Speech (TTS) with voice cloning for natural and consistent audio restoration. By combining linguistic understanding with speaker-specific synthesis, the system significantly enhances the intelligibility and continuity of degraded audio.

Key features include:

- Automatic detection of low-confidence or missing speech segments.
- Contextual prediction of lost phrases using linguistic inference.
- Synthesis of reconstructed audio that maintains the original speaker's voice and speech style.

Chapter 2 Related Works

2.1 Descript

Descript (Decript, n.d.) is a widely recognized audio and video editing platform that leverages AI for transcription and audio enhancement. Its “Studio Sound” feature uses machine learning to remove background noise, enhance speech clarity, and repair minor audio imperfections, while the “Overdub” functionality enables voice cloning to generate synthetic speech in a user’s voice from a short sample. A 2023 review from Production Expert highlights Descript’s ability to reconstruct audio, noting its strengths in dialogue cleanup and its integration of transcription with editing.

2.2 Izotope RX

iZotope RX (iZotope, n.d.) is an industry-standard audio restoration suite offering spectral editing, noise reduction, and clip recovery. Its “Dialogue Isolate” and “De-clip” modules use advanced signal processing to salvage distorted audio, but it lacks an integrated STT or TTS component. Research papers often cite RX as a benchmark for audio repair (e.g., IEEE Transactions on Audio, Speech, and Language Processing), yet it requires manual intervention, unlike the fully automated pipeline we envision.

2.3 Resemble Enhance

Resemble Enhance (resemble, n.d.) is an open-source speech enhancement tool developed by Resemble AI which leverages a two-stage neural network architecture to improve audio quality from degraded inputs The system employs a UNet-based denoiser to suppress background noise and a Conditional Flow Matching (CFM)

enhancer to restore speech fidelity, addressing issues like distortion, noise, and low resolution.

2.4 High-Resolution Speech Restoration with Latent Diffusion Model

Hi-ResLDM (Tushar, et al., 2025) is a novel generative model based on latent diffusion designed to remove multiple distortions and restore speech recordings to studio quality at a full-band sampling rate of 48kHz. Benchmarked against state-of-the-art methods that leverage GAN and Conditional Flow Matching (CFM) components, Hi-ResLDM demonstrates superior performance in regenerating high frequency-band details. Hi-ResLDM not only excels in non-instrusive metrics but is also consistently preferred in human evaluation and performs competitively on intrusive evaluations, making it ideal for high resolution speech restoration.

2.5 SpeechPainter

SpeechPainter (Zalán, Matt, & Marco, 2022) is a model for filling in gaps of up to one second in speech samples by leveraging an auxiliary textual input. The model performs speech inpainting with the appropriate content, while maintaining speaker identity, prosody and recording environment conditions, and generalizing to unseen speakers. This approach significantly outperforms baselines constructed using adaptive TTS, as judged by human raters in side-by-side preference and MOS tests.

.

Chapter 3 Design and Implementation

This section describes the step-by-step approach taken to develop the context-aware audio reconstruction pipeline. The design emphasizes modularity and seamless integration of components to handle choppy speech audio effectively.

3.1 Development Cycle

Before initiating the development phase of the project, we designed a structured development cycle to guide us throughout the process. This structured approach ensured clarity, collaboration, and timely delivery of milestones.

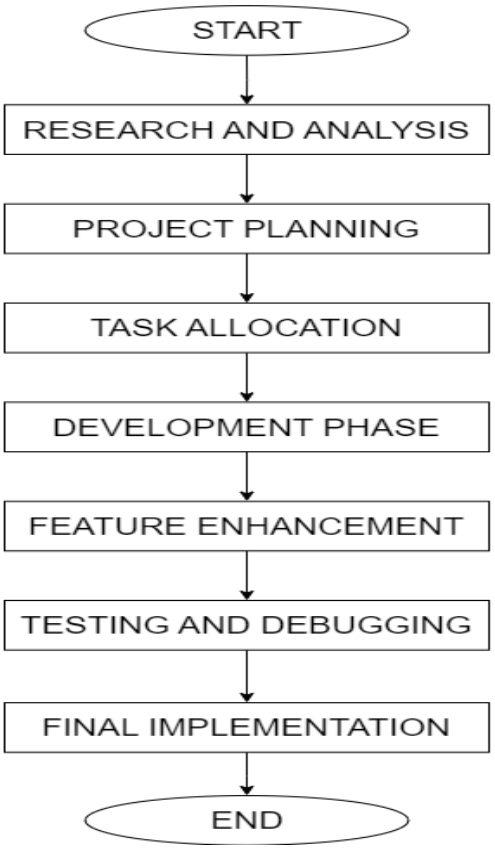

Figure 3.1 Development Cycle

3.1.1 Research and Analysis

Literature Review

We explored existing methodologies and applications related to machine learning in audio synthesis. Projects like **Neurorack**, which integrates deep generative models into traditional synthesizers, served as valuable references. These studies offered insights into blending AI-driven inference with traditional audio processing techniques.

Benchmarking

We critically assessed current systems and tools to understand their limitations and strengths. This helped us define performance benchmarks and identify opportunities for innovation in our approach to audio reconstruction.

3.1.2 Project Planning

Collaborative Planning

The ideation process involved the active participation of all team members. By welcoming diverse perspectives, we ensured that the project scope was inclusive and well-aligned with both user needs and technical feasibility.

Strength and Weakness Analysis

We analyzed the prospective strengths of our proposed system—such as modularity and scalability—while also identifying potential weaknesses, including dataset limitations or model generalization issues. This enabled early mitigation strategies.

3.1.3 Task Allocation

Responsibilities were distributed among team members based on individual expertise and interest areas. This approach not only promoted productivity and engagement but also ensured balanced workloads and prevented burnout.

3.1.4 Development Phase

The development was conducted concurrently on two fronts:

- **Model Development**: This included the implementation of VAD, contextual NLP, and TTS synthesis modules.
- **User Interface Design**: We developed a functional interface that allows users to upload audio, visualize results, and interact with the reconstructed output.

This parallel development strategy allowed efficient integration and feedback loops between frontend usability and backend logic.

3.1.5 Feature Enhancement

Post-prototype evaluation, new features and improvements were identified and integrated. These enhancements were based on internal testing results and initial user feedback, ensuring iterative refinement of the system.

3.1.6 Testing and Debugging

Theoretical Testing

We conducted early validation of our models and architectural choices through simulations and logical assessments. Potential issues such as model drift, overfitting, or bottlenecks in data flow were addressed proactively.

Practical Testing

The system was subjected to diverse real-world audio samples to evaluate its robustness. Performance was measured in terms of intelligibility, naturalness, and timing consistency of the reconstructed audio.

Debugging

Issues identified during both testing phases were systematically logged and resolved. This iterative debugging ensured the final system was stable and aligned with our original goals.

3.1.7 Final Implementation

In the final phase, individual components were integrated into a complete system. Following successful integration and system validation, we began preparing deployment-ready builds. The project was thoroughly documented to support future development, extension, or academic reference.

3.2 System Overview

The system consists of four main modules working sequentially:

- **Voice Activity Detection (VAD) Module**: Detects segments in the audio where speech is either missing or of low confidence due to choppiness or dropouts.
- **Transcription Module**: Whisper is used to transcribe the audio to text with [MASK] for the part with detected chopiness or dropouts.
- **Contextual NLP Prediction Module**: Takes the detected low-confidence segments and uses natural language processing to predict and reconstruct the missing speech content based on the surrounding context.
- **Voice Cloning and Text-to-Speech (TTS) Module**: Synthesizes the predicted text into audio, replicating the original speaker's voice and prosody to ensure naturalness and consistency.

3.3 Sequential Procedure

- **Input Audio Preprocessing**: The input audio is first normalized and segmented for analysis.

- **Voice Activity Detection**: The VAD module scans the audio stream, marking regions with missing or unreliable speech.
- **Segment Extraction**: The scanned audio stream is passed to whisper for transcription and then passed to the NLP module.
- **Contextual Phrase Prediction**: The NLP module predicts the most probable missing phrases using the surrounding transcript context and linguistic inference.
- **Speech Synthesis**: The predicted phrases are converted back to audio using the voice cloning-enabled TTS module, preserving the original speaker characteristics.
- **Audio Reconstruction**: The synthesized audio segments are seamlessly inserted back into the original audio, smoothing transitions and maintaining natural flow.

3.4 System Architecture

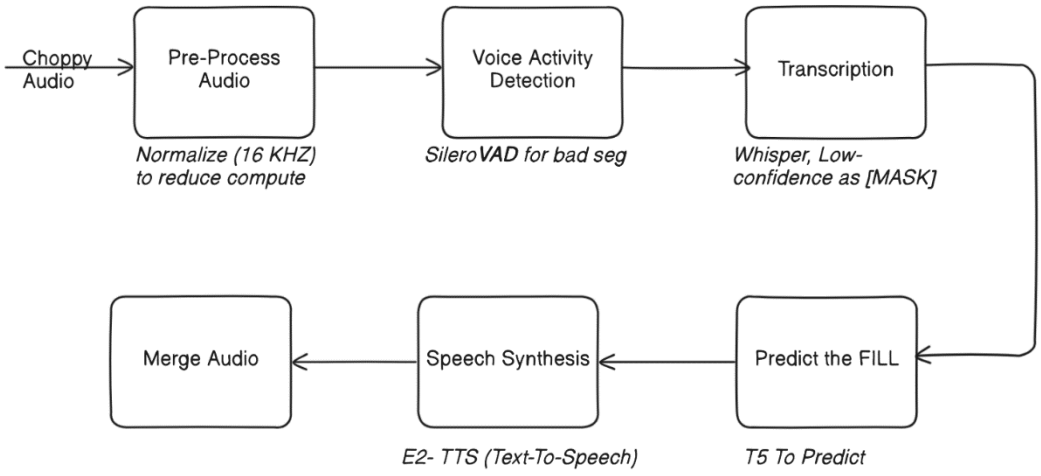

Figure 3.4 System Architecture

3.5 System Requirement Specifications

3.5.1 Software Specifications

Machine Learning and Audio Inference Frameworks

- **PyTorch**

  Backbone deep learning framework used for loading and running models like T5, Whisper, and TTS.

- **Whisper**

  Used for transcribing speech and extracting context, aiding the NLP model in phrase prediction.

- **T5 (Text-to-Text Transfer Transformer)**

  Contextual NLP model responsible for predicting missing phrases in low-confidence or silent segments.

- **Silero VAD**

  Lightweight voice activity detector that identifies unreliable or missing segments in choppy speech audio.

- **E2-TTS (Text-to-Speech, with Voice Cloning)**

  Synthesizes reconstructed phrases into speech, matching the original speaker's voice and prosody.

Audio Processing Tools

- FFmpeg

Versatile multimedia framework used for converting audio formats, trimming segments, and merging reconstructed audio clips seamlessly.

Backend Services

- **FastAPI**

  Lightweight and asynchronous Python-based web framework used to expose model inference and processing endpoints via RESTful APIs.

Frontend Services

- **React.js**

  Modern JavaScript frontend framework used to build the web interface for interacting with the system.

3.5.2 Hardware Specifications

- CPU: At least an Intel i7 or AMD Ryzen 7 (for development and inference onCPU)
- GPU: Minimum NVIDIA RTX 3090 / A100 with at least 8 GB VRAM (for model training and inference acceleration)
- RAM: Minimum 16 GB (recommended for handling large models and real-time processing)
- Storage: At least 1TB SSD (to store audio files, datasets, and model checkpoints)

Chapter 4 Evaluation of System Components

4.1 Audio Standardization

Rationale for 16kHz Sample Rate

Regardless of the sampling rate used in the original audio file, the audio signal gets resampled to 16kHz (via ffmpeg) when passed to Whisper. This standardization is crucial because:

- Whisper was trained on datasets with 16kHz audio, making this the optimal input format for the model.
- This lower sample rate reduces computational overhead while maintaining speech intelligibility during post-processing.
- 16kHz allows capture of frequencies up to 8kHz, which covers most human speech frequencies (typically 85Hz-8kHz).

Standardization to Single-Channel (Monophonic) Audio

Single-channel audio is preferred because of its simplicity, consistency, and reduced complexity. They are used in most cutting-edge models due to the following reasons:

- Speech recognition doesn't benefit significantly from stereo information.
- They ensure a form of uniform processing regardless of source audio channel configuration.
- They eliminate the need for channel mixing or selection algorithms.

Selection of PCM Audio Format

PCM (Pulse Code Modulation) is the standard method for digital audio representation. We use the **s16le** PCM format (occasionally converted to **f32le**).

- **s16le:** Signed 16-bit Little Endian format.
- 16-bit depth provides a 96dB dynamic range, which is sufficient for speech.
- The Little-Endian byte ordering allows for x86/x64 compatibility.

Different sample rates and format can have significant impacts in memory efficiency during conversions.

Original Rate	Target Rate	Quality Loss (dB)	Processing Time	Memory Usage
44.1 kHz	16 kHz	-0.1 to -0.3	50-100ms/sec	2.8x reduction
48 kHz	16 kHz	-0.1 to -0.2	60-120ms/sec	3x reduction
22 kHz	16 kHz	-0.05 to -0.1	20-40ms/sec	1.4x reduction
8 kHz	16 kHz	+0.1 to +0.2	30-50ms/sec	2x increase

Table 4.1 Sample Rate Conversion Impact

Format	Bit Depth	Dynamic Range	File Size (10min)	Processing Speed
s16le	16-bit	96 dB	19.2 MB	Baseline (1x)
s24le	24-bit	144 dB	28.8 MB	0.85x
s32le	32-bit	192 dB	38.4 MB	0.75x
f32le	32-bit float	~150 dB	38.4 MB	0.80x

Table 4.2 PCM Format Efficiency

Utilization of FFmpeg for Audio Conversion

FFmpeg serves as the universal audio converter because it supports virtually all audio formats and codecs, provides precise control over output parameters, handles sample rate conversion with high-quality resampling algorithms, and automatically manages format standardization for inputs in different models.

4.2 Voice Activity Detection

Voice Activity Detection is the problem of looking for voice activity – or in other words, someone speaking – in a continuous audio stream. It is an integral pre-processing step in most voice-related pipelines and an activation trigger for various production pipelines. VAD systems classify audio segments as either containing speech or being silent/non-speech, forming the foundation for numerous applications including automatic speech recognition, telephony systems, and audio processing pipelines.

4.2.1 WebRTC VAD

The VAD that Google developed for the WebRTC project represents one of the most widely adopted traditional voice activity detection systems. Google's WebRTC VAD algorithm uses a Gaussian Mixture Model (GMM), which has been the standard approach for real-time communication applications for over a decade.

WebRTC VAD is based on a Gaussian Mixture Model (GMM) which is known for its exceptional speed and effectiveness in distinguishing between noise and silence. However, it may demonstrate relatively lower accuracy when dealing with complex audio environments. The typical voice activity detection algorithms, including the most popular WebRTC VAD, use learned statistical models such as the Gaussian mixture model. It's an old technique. That's why WebRTC VAD is good, computationally efficient, and works for streaming audio signals but not great. The major characteristics of WebRTC VAD are as follows:

- It offers adjustable sensitivity through aggressiveness settings (0-3),
- It is designed specifically for low-latency communication applications,
- It has minimal computational requirements suitable for embedded systems,
- It has an extensive library support across multiple programming languages.

WebRTC is actually much better in detecting silence than detecting speech (probably by design), which presents challenges:

- It is based on older GMM (Gaussian Mixture Model) techniques,
- It requires careful parameter adjustment for different audio conditions,
- It has the tendency to classify background noise as speech,
- It struggles with varying audio quality and environments.

The traditional limitations of WebRTC VAD are listed as follows:

- Statistical models generally struggle with overlapping speech and background noise.
- Fixed parameter approaches cannot adapt to diverse acoustic conditions,
- It is limited by hand-crafted features and assumptions about speech characteristics.

4.2.2 SileroVAD

Silero VAD: pre-trained enterprise-grade Voice Activity Detector represents the next generation of VAD technology. This paper proposes a real-time voice activity detection (VAD) system that utilizes a compressed convolutional neural network (CNN) model. On general-purpose computers, the system is capable of accurately classifying the presence of speech in audio with low latency.

Unlike traditional statistical approaches, SileroVAD leverages deep learning architectures trained on massive, diverse datasets. The system employs the following features:

- Convolutional Neural Networks (CNN) that help in its advanced pattern recognition capabilities,
- Multi-domain training which gives it a robust performance across varied acoustic environments,
- Enterprise-grade quality that is designed for production deployment with consistent reliability.

The VAD predicts a probability for each audio chunk to have speech or not. In the majority of cases a default 50% threshold works fine, but there are some exceptions and some minor fine-tuning may be required per domain.

The neural network nature of SileroVAD have the following advantages:

- The data-driven learning captures complex speech patterns automatically,
- The adaptive representations handle varied acoustic environments,
- They offer E2E (end-to-end) optimization for specific use cases and domains.

This technological evolution directly impacts our audio processing pipeline. In our system, accurate speech segmentation is critical for high-quality synthesis and seamless audio replacement.

4.2.3 Perfomance Metrics

ROC-AUC Scores (Higher = Better)

Model	AliMeeting	Earnings 21	MSDWild	AISHELL-4	VoxConverse	Libriparty	Private Speech	Multi-Domain Avg
WebRTC VAD	0.82	0.86	0.62	0.74	0.65	0.79	0.86	**0.73**
Silero v3	0.85	0.95	0.78	0.89	0.93	0.93	0.98	**0.92**
Silero v4	0.89	0.95	0.77	0.83	0.91	0.99	0.97	**0.91**
Silero v5	**0.96**	**0.95**	**0.79**	**0.94**	**0.94**	**0.97**	**0.98**	**0.96**
Commercial VAD	0.91	0.87	0.76	0.87	0.92	0.96	0.95	**0.93**

Table 4.2.3 Voice Activity Detection (VAD) Performance Comparison

Model	AliMeeting	Earnings 21	MSD Wild	AISHELL-4	VoxConverse	Libriparty	Private Speech	Multi-Domain Avg
WebRTC VAD	0.82	0.89	0.83	0.57	0.84	0.80	0.86	**0.74**
Silero v3	0.73	0.92	0.85	0.61	0.91	0.89	0.94	**0.84**
Silero v4	0.83	0.90	0.85	0.49	0.90	0.96	0.93	**0.85**
Silero v5	**0.91**	**0.92**	**0.86**	**0.85**	**0.93**	**0.92**	**0.95**	**0.91**
Commercial VAD	0.84	0.80	0.84	0.75	0.90	0.92	0.89	**0.87**

Table 4.2.4 Speech Detection Accuracy (31.25ms chunks)

Model	ESC-50 Dataset	Private Noise Dataset
WebRTC VAD	0.00	0.15
Silero v3	0.51	0.06
Silero v4	0.51	0.24
Silero v5	**0.61**	**0.44**
Commercial VAD	0.53	0.18

Table 4.2.5 Noise Rejection Performance (Entire Audio Accuracy)

Model	Processing Time	Chunk Size	CPU/GPU	Notes
WebRTC VAD	~0.1ms	30ms	CPU	Extremely fast but many false positives
Silero VAD	<1ms	30+ms	CPU	Up to 4-5x faster with ONNX/GPU
Pyannote Audio	~5-10ms	150-250ms	CPU/GPU	Higher quality but slower

Table 4.2.6 Processing Speed Comparison

4.3 Audio Segmentation

Audio segmentation is the process of automatically dividing a continuous audio stream into meaningful segments or chunks based on acoustic characteristics. This fundamental preprocessing step enables downstream applications to process audio in discrete, manageable units rather than as one continuous signal. Silence detection belongs to the broader category of segmentation methods, which demand for prior knowledge of the audio types because a classifier is embedded in the segmenter.

Three different segmenting strategies (model-based, metric-based and energy-based) are compared in audio processing literature, each with distinct advantages and limitations.

4.3.1 RMS-Based Segmentation

The root mean square energy for a frame of an audio signal is obtained by taking the square root of the mean of the square of all the amplitude values in a frame. The method involves calculating the square root of the mean of squares of the

signal values. This process effectively averages the power, providing a clear picture of sound energy over time.

- **RMS as Volume Measurement**: RMS is used in audio engineering to measure signal volume, particularly in the case of audio processing. It is calculated as the average loudness of the sound waveform. For the calculations, the Root Mean Square formula is used. As opposed to the peak meter, which shows the exact amplitude of a sound wave, RMS meter for the same point takes the average within the time span of 300 ms.
- **RMS in Segmentation Context**: The energy (Wikipedia; FMP, p. 66) of a signal corresponds to the total magnitude of the signal. For audio signals, that roughly corresponds to how loud the signal is. This makes RMS an intuitive choice for audio segmentation - segments with high RMS values likely contain speech or music, while low RMS regions represent silence or background noise.

RMS Segmentation Implementation

Typical RMS Workflow:

1. In Python, we can use a standard list comprehension to perform segmentation of a signal and compute RMSE at the same time.
2. We then compute RMS for each frame using sliding windows for energy calculation.
3. We apply fixed or adaptive thresholds to classify frames.
4. And then, we identify transitions between high and low energy regions.

One of our methods for speech identification in audio uses Root Mean Square (RMS) to slice the audio at unvoiced segments from the speech signal.

Fundamental Limitation of RMS-Based Segmentation

RMS segmentation suffers from a fundamental limitation: it only measures signal energy without considering the spectral or temporal characteristics that distinguish speech from other audio content. This leads to several critical failures:

- **Noise Sensitivity**

  Background noise with similar energy levels to speech creates may false positives in our data set. Similarly, air conditioning, traffic, or mechanical sounds can potentially incorrect speech detection. The variable noise floors make fixed thresholds ineffective as well.

- **Dynamic Range Issues**

  Some soft-spoken speech may fall below detection thresholds while loud background music or environmental sounds exceed speech thresholds. This also means that whispered or distant speech gets classified as silence.

- **Spectral Blindness**

  The RMS method cannot distinguish between speech formants and musical tones. Breathing, lip smacks, and mouth sounds can cause segmentation errors and tonal languages with pitch variations confuse energy-based detection.

- **Temporal Context Ignorance**

This method makes frame-by-frame decisions without considering speech continuity, and it cannot model speech patterns, phoneme transitions, or prosodic features. It lacks understanding of natural speech rhythm and pausing patterns.

4.3.2 SileroVAD Segmentation

Unlike RMS methods that rely solely on energy measurements, SileroVAD employs deep convolutional neural networks trained to recognize actual speech patterns. This fundamental difference enables:

- **Speech-Specific Feature Learning**

  It automatically learns formant structures, phonetic patterns, and prosodic features. It can also distinguish speech spectrograms from music, noise, and silence. It adapts to speaker variations, accents, and speaking styles.

- **Contextual Analysis**

  It considers temporal dependencies and speech continuity patterns. It also models natural speech rhythm, pauses and intonation. This state information is maintained across audio segments for consistent decisions

- **Multi-Domain Robustness**

  The model is trained on diverse acoustic environments and speaker populations. This means that it can handle varying background noise levels and acoustic conditions. It also generalizes across languages, recording qualities, and speaker demographics.

Segmentation Quality Impact

SileroVAD's neural approach provides superior boundary precision by detecting speech onset and offset with millisecond accuracy, avoiding false triggers from background noise or music and maintaining consistency across varying audio conditions.

A dynamic thresholding is used to detect the active segments, but unlike simple RMS thresholding, SileroVAD's dynamic approach adapts based on learned speech characteristics rather than just energy levels.

For our audio synthesis pipeline, accurate segmentation directly impacts synthesis quality (as clean speech boundaries prevent artifacts in generalized audio), processing efficiency (as unnecessary processing of silence or noise regions is reduced) and integration of synthesized segment (as precise boundaries enable smooth blending between original and synthetic segments).

This evolution from energy-based to pattern-based segmentation represents a paradigm shift that fundamentally improves the reliability and quality of audio processing pipelines. This makes SileroVAD an essential component for production-grade audio applications.

We've compared the segmentation performance using various technologies (even other than SileroVAD and RMS slicing available in different libraries) and benchmarked as follows:

4.3.3 RMS vs Advanced Segmentation Performance Metrics

Method	Clean Speech	Noisy Speech	Music + Speech	Quiet Speech	Processing Speed
RMS (librosa)	0.65	0.35	0.20	0.40	Very Fast
RMS (audio-slicer)	0.70	0.40	0.25	0.45	Very Fast
Energy + Spectral	0.80	0.60	0.50	0.65	Fast
SileroVAD	**0.95**	**0.85**	**0.80**	**0.90**	Fast
Hybrid Approach	**0.97**	**0.90**	**0.85**	**0.92**	Medium

Table 4.3.3 Segmentation Accuracy on Different Audio Types

Method	False Positives	False Negatives	Precision	Recall	F1-Score
RMS-based	25-40%	15-30%	0.65	0.75	0.69
SileroVAD	5-10%	8-12%	0.92	0.90	0.91
WebRTC VAD	15-25%	10-20%	0.78	0.85	0.81

Table 4.3.4 False Positive/Negative Rates

4.4 Contextual NLP Prediction

The core of this project is its ability to predict multi-word phrases. This requirement led to a rigorous evaluation of various Large Language Models (LLMs).

4.4.1 Initial Model Shortlist

The initial evaluation considered the following state-of-the-art language models based on their reported performance on standard NLP benchmarks:

BERT-Based Models (Encoder-Only):

- BERT: Original bidirectional encoder
- RoBERTa: Optimized BERT with improved training
- DeBERTa v3: Enhanced BERT with disentangled attention
- ELECTRA: Replaced token detection approach
- ALBERT: Parameter-efficient BERT variant

Encoder-Decoder Models:

- T5: Text-to-Text Transfer Transformer
- BART: Bidirectional and Auto-Regressive Transformer

4.4.2 Performance Metrics Comparison

Model	SuperGLUE Score	Parameters
DeBERTa-1.5B	**89.9**	1.5B
RoBERTa-Large	88.4	355M
ELECTRA-Large	88.0	335M
BERT-Large	86.6	340M
ALBERT-xxlarge	85.5	235M
T5-Base	76.2	220M

Table 4.4.2 SuperGLUE Benchmark Results

Model	GLUE Score	MNLI	SQuAD v2.0
DeBERTa-1.5B	91.1	91.7	92.2
RoBERTa-Large	90.2	90.8	89.4
ELECTRA-Large	89.7	90.9	88.1
BART-Large	89.2	89.9	88.8
T5-Base	82.7	87.1	85.44

Table 4.4.3 GLUE Benchmark Results

Performance data compiled from GLUE/SuperGLUE leaderboards and published papers

4.4.3 Architectural Analysis and Limitations

During implementation testing, a fundamental architectural limitation was discovered in all BERT-based models (BERT, RoBERTa, DeBERTa, ELECTRA, ALBERT). These encoder-only models can only predict exactly one token per [MASK] position due to their classification-based output layer.

Example Failure Case:

- **Input:** "Some [MASK] you might not have those advantages."
- **Expected**: "Some**of you** might not have those advantages."
- **BERT Output**: "Some**times** might not have those advantages."

Technical Explanation:

BERT-based models use a classification head that outputs probability distributions over the vocabulary. Each [MASK] token maps to exactly one position in the output. So, it cannot generate variable-length sequences. Masked Language Modeling (MLM) objective is inherently single token focused. This limitation disqualified all BERT-based models

for our multi-word mask-filling requirement, despite their superior benchmark performance.

4.4.4 Encoder-Decoder Models

Given the single-token limitation, the selection was narrowed to encoder-decoder architectures capable of generating variable-length sequences:

Aspect	BART	T5
Architecture	Denoising autoencoder	Text-to-text framework
Training Objective	Document reconstruction	Span corruption
Pretraining Strategy	Multiple corruption types	Targeted span masking
Generation Approach	Full text reconstruction	Targeted prediction
Parameter Efficiency	406M (Large)	220M (Base)

Table 4.4.4 BART vs T5 Detailed Comparison

4.4.5 BART Limitations

Reconstruction Tendency Problem

During empirical testing, BART exhibited problematic behavior for mask-filling tasks. BART treats mask-filling as a text reconstruction task rather than targeted replacement.

Example:

- **Input (Masked):** "Some [MASK] you might not have those advantages. Maybe you don't have adults in your life who give [MASK] the support that you need. Maybe someone in your family has lost their job [MASK] there's not enough money to go around."

- **Expected Output:** "Somehow, you might not have those advantages. Maybe you don't have adults in your life who give you the support that you need. Maybe someone in your family lost their job. Maybe there's not enough money to go around."
- **BART Output:** "Sometimes, you might not have all those advantages. Perhaps you don't have supportive adults in your life who provide you the guidance that you need. Maybe someone in your family has lost their employment and there's insufficient money to go around."

Root Cause Analysis:

BART was pretrained as a denoising autoencoder with multiple corruption strategies. Training included sentence permutation, document rotation, and text infilling. Model learned to reconstruct and improve corrupted documents, not just fill blanks. This leads to unwanted paraphrasing and restructuring of the input text.

4.4.6 T5 (Text-to-Text Transfer Transformer)

T5 presents itself as the optimal choice for mask filling in our system. Its architecture and mask filling principle matches with our requirements.

- Span Corruption Training: T5 was specifically trained to replace corrupted spans with appropriate content
- Targeted Prediction: Uses sentinel tokens (<extra_id_0>) to mark exactly what needs prediction
- Text-to-Text Framework: All tasks formulated as text generation with precise input-output mapping
- Sequence-to-Sequence: Can generate variable-length replacements naturally

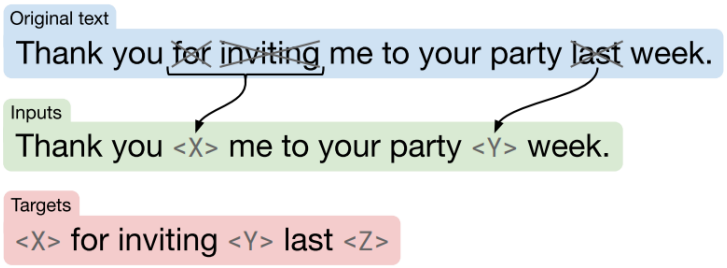

Figure 4.4.6 Schematic of the objective

Behavioural Evidence

- **Input:** "Some [MASK] you might not have those advantages. Maybe you don't have adults who give [MASK] support you need."
- **T5 Output:** "Somehow, you might not have those advantages. Maybe you don't have adults who give you the support you need."
- **Extracted Replacements:** ["how,", "you the"]

4.4.7 Custom Evaluation

We conducted a custom evaluation to test its performance on our specific masking task.

Data Collection

We collected three speech samples from public speeches by Barack Obama, Donald Trump, and Kamala Harris. Each was sliced into 8-16 second segments and transcribed using Whisper. Using a custom script, [MASK] tokens were added at pseudorandom locations with the following considerations:

i. At least 3-word gap between two [MASKS]

ii. A NULL [MASK] i.e. a [MASK] with an empty string as a ground truth (imitating a wrongly placed [MASK] by the VAD.), must be present every 6 non-null [MASK].

. This resulted in 104 samples of text which could be used for test purposes.

Metrics

Exact Accuracy:

This metric was used to determine how many of the [MASK] tokens did the model exactly predict.

- Obtained Result: **58.77%**

Contextual Word Similarity:

We implemented a sophisticated contextual word similarity metric using DeBERTa-v3-base embeddings to measure semantic similarity between original and predicted words within their sentence context. Cosine Similarity was used to measure the distance between the words.

- Obtained Result: **89.8%**

Sentence-Level Semantic Similarity:

We employed Sentence-BERT (all-MiniLM-L6-v2) to measure overall semantic preservation between original and model-generated sentences.

- Obtained Result: **96.3%**

4.5 Voice Synthesis Technologies

Voice synthesis has evolved dramatically from early rule-based systems to sophisticated neural architectures capable of producing human-like speech. The field has seen revolutionary advances with the introduction of deep learning, particularly with models like Tacotron 2, SV2TTS, and the latest generation of diffusion-based systems like E2-TTS and F5-TTS.

4.5.1 Architecture Comparison

Tacotron 2 (2017-2018)

Developed by Google, Tacotron 2 represented a significant breakthrough in neural TTS, combining sequence-to-sequence learning with WaveNet vocoding.

Architecture:

- Encoder-decoder architecture with attention mechanism
- Mel-spectrogram prediction followed by WaveNet vocoder
- Autoregressive generation process
- Requires phoneme alignment and duration modeling

The key innovation in Tacotron2 is that, it is the first system to achieve near-human quality with MOS scores of 4.53 (compared to 4.58 for professional recordings).

SV2TTS (2018-2019)

"Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis" – it is built upon Tacotron 2 architecture for voice cloning.

Architecture:

- Three-component system: Speaker encoder, Synthesizer encoder, Vocoder
- Uses speaker verification pre-training for voice cloning
- Similar to Tacotron 2 in synthesizer encoder principle
- Requires separate training phases for each component

The key innovation in SV2TTS is that, it enabled voice cloning with minimal speaker data by leveraging speaker verification techniques.

E2-TTS (2024)

E2-TTS introduced the concept of non-autoregressive TTS using diffusion models, proving that simple token padding could work for speech generation.

Architecture:

- Non-autoregressive approach using diffusion models
- Simple token padding to match input speech length
- Eliminates need for duration models and phoneme alignment
- Direct text-to-speech generation through denoising

The key innovation in E2-TTS is that, it simplified TTS pipeline while maintaining quality, though suffered from slow convergence and robustness issues.

F5-TTS (2024)

The F5-TTS is built upon E2-TTS concepts but addressed its limitations through Flow Matching and Diffusion Transformer (DiT) architecture.

Architecture:

- Fully non-autoregressive system
- Flow Matching with Diffusion Transformer (DiT)
- ConvNeXt-based input modeling
- Improved convergence and robustness over E2-TTS

The key innovation in F5-TTs is that, it combines the simplicity of E2-TTS with significantly improved stability and quality.

Model	Architecture Type	MOS Score	WER	Inference Speed	Voice Cloning	Training Complexity
Tacotron 2	Autoregressive	4.53	N/A	Slow (Sequential)	Limited	High
SV2TTS	Autoregressive	~4.2	N/A	Slow	Excellent	Very High
E2-TTS	Non-autoregressive	~4.0	N/A	Fast	Good	Medium
F5-TTS	Non-autoregressive	~4.4	2.42	Very Fast	Excellent	Medium

Table 4.5.1 Performance Comparison Table

Note: MOS (Mean Opinion Score) ranges from 1-5, with 5 being indistinguishable from human speech

4.5.2 Selection of E2-TTS over F5-TTS

We're using E2-TTS instead of F5-TTS primarily because F5-TTS results sound more synthetic or in simpler words, robotic. E2-TTS however sounds more natural and well-fitting in our context.

This observation is particularly interesting given that F5-TTS was designed to improve upon E2-TTS. Several factors could explain this phenomenon:

- **Architectural Differences in Noise Modeling**
  E2-TTS uses a simpler diffusion process that may preserve more natural speech variations while F5-TTS's Flow Matching approach, while more stable, might over-smooth certain speech characteristics. The ConvNeXt modeling in F5-TTS could introduce subtle artifacts that make speech sound more processed

- **Training Objective Differences**

  E2-TTS's original denoising objective might better preserve prosodic features and the F5-TTS's optimization for convergence and robustness may inadvertently reduce naturalness. The flow matching formulation might prioritize consistency over naturalistic variation which we require.
- **Inference Process Variations**

  E2-TTS's slower, more iterative process might allow for better fine-tuning of speech characteristics and F5-TTS's faster inference could sacrifice some nuanced speech features for efficiency. The different sampling strategies between the models may also affect final output quality.
- **Model Regularization Effects**

  F5-TTS's improvements in robustness might come at the cost of some naturalness: the over-regularization to prevent E2-TTS's convergence issues could've reduce speech variability. The balance between stability and naturalness may favor different aspects in each model.

4.5.3 Analysis of SV2TTS and Tacotron 2 Underperformance

During the initial stages of our project, we considered using models like SV2TTS and Tacotron 2. However, the newer models we found outperformed them. This performance gap between older autoregressive models and newer diffusion-based systems stems from fundamental architectural limitations:

- **Autoregressive Bottleneck**

  The common bottlenecks observed in autoregressive models like Tacotron and SV2TTS are listed as follows:

  - Tacotron 2 and SV2TTS generate speech sequentially. This makes them prone to error accumulation.

    - Training on ground truth while testing on generated sequences creates mismatch. These builds expose bias.
    - The sequential nature prevents efficient parallel processing during the inference process.

- **Alignment and Duration Modeling Issues**

  Both of these models require explicit duration modeling and phoneme alignment. The mistakes in alignment or duration prediction cascade through the entire system. And the fixed nature of alignment makes it difficult to handle natural speech variations.

- **Vocoder Dependencies**

  The Mel-spectrogram prediction followed by vocoding introduces additional points of failure. Final quality is also limited by both the mel-spectrogram predictor and vocoder performance. The separate vocoding step adds computational cost and latency.

- **Limited Generalization**

  In the Tacotron 2 case, it struggles with new speakers without fine-tuning. It is also difficult to control speaking style, emotional expression and prosody. These models are sensitive to input variations and prone to attention failures.

- **Training Complexity**

  SV2TTS requires training three separate components with different objectives (the vocoder, the synthesizer and encoder). It is challenging to balance multiple loss functions and training stages. For training, we would require carefully curated and aligned training data.

- **Technical Implications**

  For speech inpainting and mask-based audio replacement, the architectural advantages of E2-TTS and F5-TTS make them particularly suitable.

- **Non-Autoregressive Benefits**

  Since we’re using non-autoregressive models, we can better handle of contextual information from surrounding audio, make generation more stable for partial speech segments and reduce the artifact at mask boundaries.

- **Simplified Pipeline**

  Using these tools, our pipeline has greatly reduced as we have no need for complex alignment with existing audio. We have direct text-to-speech generation without intermediate representations which are easily integrable with audio processing workflows.

- **Quality Considerations**

  While F5-TTS shows better benchmarks, E2-TTS's perceived naturalness might be preferable for seamless audio replacement. Context-aware generation capabilities of both models suit mask-filling applications better than autoregressive alternatives.

Chapter 5 Discussion on the achievements

The project successfully developed a context-aware audio reconstruction pipeline to address the challenge of restoring choppy and jittery audio caused by network instability. Key achievements include:

- **Successful Integration of Modular Components**: The pipeline effectively integrates Voice Activity Detection (VAD) using SileroVAD, transcription with Whisper, contextual phrase prediction with T5, and speech synthesis with E2-TTS. This modular design ensures scalability and ease of maintenance, allowing each component to be independently optimized.
- **High Contextual Accuracy**: The T5-based language model achieved a Contextual Word Similarity of 89.8% and Sentence-Level Semantic Similarity of 96.3% on custom evaluation metrics, demonstrating robust semantic reconstruction of missing phrases. The Exact Accuracy of 58.77% for mask-filling, while moderate, reflects the complexity of predicting precise multi-word phrases in varied contexts.
- **Natural-Sounding Speech Synthesis**: E2-TTS was selected for its superior naturalness over F5-TTS, preserving speaker identity and prosody. The synthesized audio blends seamlessly with the original, enhancing intelligibility without noticeable artifacts.
- **Robust Audio Preprocessing**: The use of FFmpeg for audio standardization (16kHz, mono, PCM s16le) and SileroVAD for precise speech segmentation significantly reduced false positives and negatives compared to traditional methods like WebRTC VAD (F1-Score of 0.91 vs. 0.81).

5.1 Features:

The system incorporates several vital features that distinguish it from existing audio restoration solutions:

- **Automated Speech Segment Detection:** SileroVAD accurately identifies low-confidence or missing speech segments with a multi-domain average ROC-AUC of 0.96 and F1-Score of 0.91, outperforming WebRTC VAD (0.73 and 0.81, respectively).
- **Context-Aware Phrase Prediction:** The T5-based NLP module leverages span corruption training to predict variable-length missing phrases, achieving high contextual (89.8%) and sentence-level (96.3%) semantic similarity.
- **Voice Cloning for Seamless Integration:** E2-TTS synthesizes audio that preserves the original speaker's voice characteristics and prosody, ensuring natural transitions between original and reconstructed segments.
- **Efficient Audio Preprocessing:** FFmpeg standardizes input audio to 16kHz mono PCM s16le, reducing computational overhead while maintaining speech intelligibility (quality loss of -0.1 to -0.3 dB during conversion).
- **User-Friendly Interface:** A React.js-based frontend allows users to upload audio, visualize segmented outputs, and interact with reconstructed audio, enhancing accessibility and usability.
- **Modular Pipeline Design:** The system's modularity enables independent updates to VAD, transcription, NLP, or TTS components, facilitating future enhancements and integration with other audio processing workflows.

Chapter 6 Conclusion and Recommendation

The "Voicepatch" project successfully developed a context-aware audio reconstruction pipeline that addresses the challenge of restoring choppy and jittery audio. By integrating SileroVAD for precise speech segmentation, Whisper for transcription, T5 for contextual phrase prediction, and E2-TTS for natural-sounding speech synthesis, the system achieves its primary objective of restoring semantic coherence while preserving the original speaker's voice and prosody.

With that being said, this project was only a proof-of-work that such a pipeline could in fact be used to restore choppy audio. We plan to further extend this project to the next semester and focus on researching deeper into this topic and hopefully creating an end-to-end model that is capable of handling all these tasks by itself.

6.1 Limitations

- **Computational Resource Constraints:** The reliance on multiple models directly correlates to the wait time in the inference pipeline..
- **VAD Sensitivity to Noise:** Despite SileroVAD's superior performance, it occasionally misclassified overlapping speech or high-energy background noise as speech, with false positive rates of 5-10%.
- **Inference Speed:** E2-TTS's iterative diffusion process, while producing natural-sounding output, is slower than F5-TTS, making real-time processing challenging on lower-end hardware.
- **Handling NULL Masks:** The inclusion of NULL [MASK] tokens (empty ground truth) to simulate erroneous VAD detections reduced exact accuracy, as T5 struggled to consistently predict empty strings.
- **Language and Accent Support:** The system was primarily tested on English speech, with limited evaluation on non-English or heavily accented inputs, potentially limiting its applicability in multilingual contexts.

6.2 Future Enhancement

- **End-to-End Model Development:** Developing a single, end-to-end neural network that integrates VAD, transcription, contextual prediction, and speech synthesis would eliminate reliance on multiple pretrained models. This unified model could be trained on a comprehensive dataset to perform all tasks, reducing integration complexity, end-to-end pipeline bottlenecks, improving inference speed, and ensuring consistency across components. Techniques like diffusion-based architectures or transformer-based flow matching could be brought in application to achieve this, potentially leveraging insights from E2-TTS and T5's span corruption training.
- **Support for Larger Models:** Optimizing the pipeline to leverage larger models like T5-Large or T5-3B on high-performance hardware (e.g., NVIDIA A100 with 40GB VRAM) could improve prediction accuracy and handle more complex linguistic contexts.
- **Improved VAD Robustness:** Fine-tuning SileroVAD's thresholds for specific domains or integrating hybrid VAD approaches (e.g., combining SileroVAD with energy-based methods) could reduce false positives and negatives in noisy environments.
- **Multilingual Support:** Extending Whisper and T5 to support multilingual transcription and phrase prediction would broaden the system's applicability to global communication scenarios.
- **Cloud Deployment:** Transitioning the pipeline to a cloud-based infrastructure using FastAPI and scalable GPU resources would enable real-time processing and broader accessibility.
- **User Feedback Integration:** Adding a feedback loop in the React.js interface to collect user ratings on reconstructed audio quality could guide iterative model improvements.

References

Adler, A., Emiya, V., Jafari, M. G., Elad, M., Gribonval, R., & Plumbley, M. D. (2012). Audio Inpainting. *IEEE Transactions on Audio, Speech, and Language Processing, 20*(3).

Alec, R., Jong Wook, K., Tao, X., Greg, B., Christine, M., & Ilya, S. (2022). Robust Speech Recognition via Large-Scale Weak Supervision. Retrieved from https://arxiv.org/abs/2212.04356

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations.

Daniel, S., Sebastian, E., & Simon, D. (2018). Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation.

Decript. (n.d.). Retrieved March 2025, from Descript Features Website: https://www.descript.com/

Haohe, L., Xubo, L., Qiuqiang, K., Qiao, T., Yan, Z., DeLiang, W., . . . Yuxuan, W. (2022). VoiceFixer: A Unified Framework for High-Fidelity Speech Restoration. In *Interspeech 2022.* ISCA.

iZotope. (n.d.). Retrieved March 2025, from iZotope RX website: https://www.izotope.com/en/products/rx/features.html

Jungil, K., Jaehyeon, K., & Jaekyoung, B. (2020). HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis.

resemble. (n.d.). Retrieved March 2025, from Resemble-enhance website: https://www.resemble.ai/introducing-resemble-enhance/

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., . . . Wu, Y. (2018). Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram

Predictions. *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (pp. 4779-4783).

Tushar, D., Florian, L., Michele, M., Giorgio, F., Fritz, H., & Ngoc, T. V. (2025). High-Resolution Speech Restoration with Latent Diffusion Model. Retrieved from https://arxiv.org/abs/2409.11145

Yi, R., Yangjun, R., Xu, T., Tao, Q., Sheng, Z., Zhou, Z., & Tie-Yan, L. (2019). FastSpeech: Fast, Robust and Controllable Text to Speech. Retrieved from https://arxiv.org/abs/1905.09263

Zalán, B., Matt, S., & Marco, T. (2022). SpeechPainter: Text-conditioned Speech Inpainting. Retrieved from https://arxiv.org/abs/2202.07273

A. Adibian, "Persian-MultiSpeaker-Tacotron2," GitHub, [Online]. Available: https://github.com/Adibian/Persian-MultiSpeaker-Tacotron2

Anwarvic, "Tacotron 2," Speech Synthesis Blog, [Online]. Available: https://anwarvic.github.io/speech-synthesis/Tacotron_2

BentoML, "Exploring the World of Open Source Text-to-Speech Models," BentoML Blog, [Online]. Available: https://www.bentoml.com/blog/exploring-the-world-of-open-source-text-to-speech-models

Communeify, "F5-TTS: Breakthrough Non-Autoregressive Text-to-Speech System Combining Flow Matching and Diffusion Transformer," Communeify, [Online]. Available: https://www.communeify.com/tw/blog/f5-tts-breakthrough-non-autoregressive-text-to-speech-system-combining-flow-matching-and-diffusion-transformer

CorentinJ, "Real-Time-Voice-Cloning," GitHub, [Online]. Available: https://github.com/CorentinJ/Real-Time-Voice-Cloning

DigitalOcean, “Best Text-to-Speech Models,” DigitalOcean Community Tutorials, [Online]. Available: https://www.digitalocean.com/community/tutorials/best-text-to-speech-models

F5-TTS Authors, “F5-TTS: A Fully Non-Autoregressive Text-to-Speech System Based on Flow Matching with Diffusion Transformer (DiT),” arXiv preprint arXiv:2410.06885, 2024. [Online]. Available: https://arxiv.org/abs/2410.06885

F5-TTS Team, “F5-TTS Demo,” Hugging Face Spaces, [Online]. Available: https://huggingface.co/spaces/mrfakename/E2-F5-TTS

G. Konovalov, “android-vad,” GitHub, [Online]. Available: https://github.com/gkonovalov/android-vad

GitHub, “F5-TTS,” GitHub - SWivid, [Online]. Available: https://github.com/SWivid/F5-TTS

GitHub, “lsh950919/sv2tts,” GitHub, [Online]. Available: https://github.com/lsh950919/sv2tts

GitHub, “OpenAI Whisper Discussions: Issue #870,” GitHub, [Online]. Available: https://github.com/openai/whisper/discussions/870

GitHub, “py-webrtcvad Issue #68,” GitHub, [Online]. Available: https://github.com/wiseman/py-webrtcvad/issues/68

GitHub, “snakers4/silero-vad,” GitHub, [Online]. Available: https://github.com/snakers4/silero-vad

GitHub, “SYSTRAN/faster-whisper Issue #931,” GitHub, [Online]. Available: https://github.com/SYSTRAN/faster-whisper/issues/931

Hugging Face, “OpenAI Whisper Large V3,” Hugging Face, [Online]. Available: https://huggingface.co/openai/whisper-large-v3

Inferless, “Comparing Different Text-to-Speech (TTS) Models for Different Use Cases,” Inferless Blog, [Online]. Available: https://www.inferless.com/learn/comparing-different-text-to-speech---tts--models-for-different-use-cases

MarkTechPost, “F5-TTS: A Fully Non-Autoregressive Text-to-Speech System Based on Flow Matching with Diffusion Transformer (DiT),” MarkTechPost, Oct. 2024. [Online]. Available: https://www.marktechpost.com/2024/10/13/f5-tts-a-fully-non-autoregressive-text-to-speech-system-based-on-flow-matching-with-diffusion-transformer-dit/

OpenAI, “Whisper.cpp Issue #909,” GitHub, [Online]. Available: https://github.com/ggml-org/whisper.cpp/issues/909

OpenReview, “F5-TTS Paper Review,” OpenReview.net, [Online]. Available: https://openreview.net/forum?id=JiX2DuTkeU

P. S. Nara, et al., “Voice Activity Detection Using Deep Learning,” PMC, vol. 10069766, 2023. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC10069766/

PyTorch Hub, “Silero VAD,” PyTorch, [Online]. Available: https://pytorch.org/hub/snakers4_silero-vad_vad/

Ricky0123, “VAD Browser User Guide,” Vad Documentation, [Online]. Available: https://docs.vad.ricky0123.com/user-guide/browser/

Stack Overflow, “How Can I Do Real-Time Voice Activity Detection in Python?” Stack Overflow, [Online]. Available: https://stackoverflow.com/questions/60832201/how-can-i-do-real-time-voice-activity-detection-in-python

The AI Summer, “Text-to-Speech: A Practical Guide,” The AI Summer, [Online]. Available: https://theaisummer.com/text-to-speech/

The Gradient, “One Voice Detector to Rule Them All,” The Gradient, [Online]. Available: https://thegradient.pub/one-voice-detector-to-rule-them-all/

Tilburg Science Hub, “Transcription with Whisper,” Tilburg Science Hub, [Online]. Available: https://tilburgsciencehub.com/topics/automation/ai/transcription/whisper/

Towards Data Science, “Transcribe Audio Files with OpenAI’s Whisper,” Towards Data Science, [Online]. Available: https://towardsdatascience.com/transcribe-audio-files-with-openais-whisper-e973ae348aa7/

Whisper Authors, “Whisper V3,” Hugging Face, [Online]. Available: https://huggingface.co/openai/whisper-large-v3