**Survey on Existing TTS, STT, and S2S Models**

**Introduction**

Speech technologies have become fundamental to human-computer interaction. This survey aims to compare existing models in the fields of:

* **STT (Speech-to-Text)**
* **TTS (Text-to-Speech**)
* **S2S (Speech-to-Speech)**

Existing text-to-speech (TTS) and speech-to-text (STT) models are advancing rapidly, with a focus on accuracy, efficiency, and multilingual support. Open-source models like Whisper (STT) and Kokoro (TTS) are gaining traction for their performance in low-resource environments. Deep learning architectures, particularly those based on transformers, are becoming increasingly dominant in both areas.

**Speech-to-Text (STT) Models:**

Whisper: Developed by OpenAI, Whisper is known for its multilingual capabilities and performance in noisy environments.

Kyutai STT: A streaming STT model, meaning it transcribes audio in real-time, suitable for applications like voice assistants.

Deep learning models: Transformers are particularly effective in real-time transcription.

Other models: AssemblyAI, Deepgram, and others are also prominent in the STT landscape.

**Text-to-Speech (TTS) Models:**

Kokoro:A highly efficient TTS model with only 82 million parameters, suitable for deployment on resource-constrained devices.

XTTS-v2:A popular voice cloning model that can clone voices across multiple languages with minimal input.

Deep learning models:Deep learning is driving advancements in TTS, with models capable of generating more natural and expressive speech.

Open-source solutions:Coqui TTS is another notable open-source TTS toolkit.

Speech-to-text (STT) and text-to-speech (TTS) are two revolutionary technologies that have fundamentally changed how we interact with computers and digital devices. Major tech companies like Google, IBM, and Amazon are in a constant race to develop the most accurate and advanced speech recognition systems. While both STT and TTS involve converting between spoken and written language, they serve different purposes and are used in a variety of applications.

**Current STT Models**

The speech-to-text (STT) landscape is abuzz with models that push the boundaries of what's possible in voice recognition. From the pioneering Hidden Markov Models (HMMs) to sophisticated Recurrent Neural Networks (RNNs) and groundbreaking Transformers, these frameworks form the backbone of how machines interpret our spoken words. The journey from audio waves to written text is a complex one, and each model brings its strengths to the table.

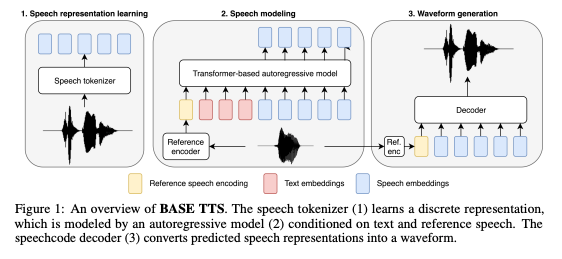
**Hidden Markov Models (HMMs):** HMMs have long been the standard in STT, relying on statistical models to predict the probability of sequences of sounds. In environments where speech is clear and noise levels are minimal, HMMs perform with commendable accuracy. Yet, they may stumble in more dynamic settings.

**Recurrent Neural Networks (RNNs):** RNNs, particularly those employing Long Short-Term Memory (LSTM) units, excel at capturing context from audio data. Their design enables them to remember long-term dependencies, making them well-suited for tasks like transcribing conversations where context is key.

**Transformers:** A newer entrant, Transformers have revolutionized STT with their attention mechanisms, which allow them to weigh the importance of different parts of the input data. This model thrives in real-time transcription scenarios due to its ability to process entire sentences and even paragraphs simultaneously. Deepgram is a Transformer-based STT model.

**Text-to-Speech (TTS) Technologies in 2025**

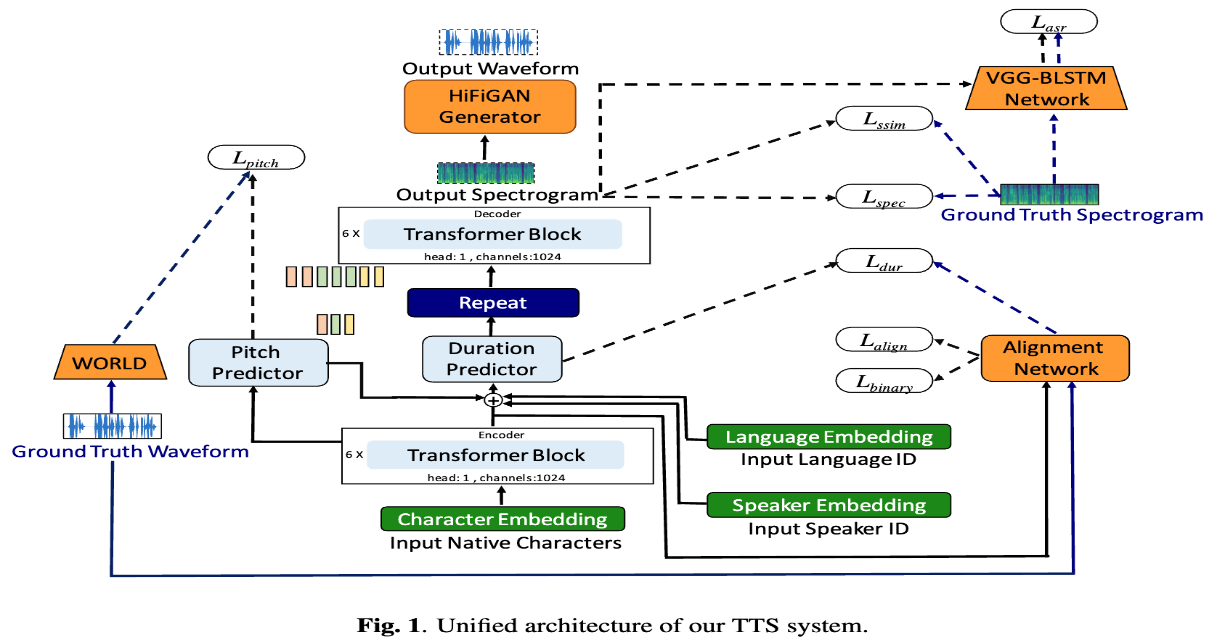
Text-to-Speech (TTS) technology has undergone remarkable advancements, driven by open-source models that are making high-quality, human-like speech synthesis widely accessible. This section explores leading frameworks, benchmark comparisons, and how innovations like Kokoro-82M and other models are shaping the field.ref[1]



The open-source TTS landscape is rapidly evolving with powerful tools:

* **ESPnet-TTS** offers a unified platform supporting Tacotron 2, Transformer TTS, and multilingual, multi-speaker pipelines with pre-trained models.
* **BASE TTS** is a billion-parameter model trained on 100k hours of speech, delivering highly natural and expressive output.
* **Kokoro-82M**, built on StyleTTS 2 and ISTFTNet, achieves state-of-the-art performance with just 82M parameters, outperforming much larger models like XTTS v2 and MetaVoice.

**Unified Architecture** Ref: [[2]](https://github.com/AI4Bharat/Indic-TTS)



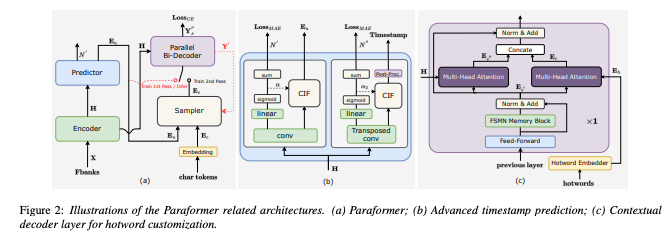
**Speech-to-Text (STT) Technologies**

Speech-to-Text (STT) technology continues to evolve rapidly, with open-source models driving innovation in accuracy, scalability, and deployment. This section examines the leading frameworks, recent advancements in end-to-end systems, and performance benchmarks shaping the STT landscape in 2025

In 2025, open-source TTS and STT technologies have transformed speech AI by offering powerful, accessible, and efficient solutions. TTS models like BASE TTS and Kokoro-82M deliver highly natural and expressive speech, with Kokoro-82M outperforming much larger models despite its compact size. Frameworks such as ESPnet-TTS, CoquiTTS, and MozillaTTS enable fast, multilingual, and customizable speech synthesis. On the STT side, DeepSpeech remains a go-to for low-resource devices, while Fairseq S2T and Paraformer lead in multilingual, end-to-end transcription. Tools like TTS Arena and smarter data strategies are further accelerating innovation, making speech technology more inclusive and industry-ready,application like

* **WhisperSTT**: Developed by OpenAI, this model leverages large-scale pretraining on multilingual datasets to deliver exceptional accuracy across various languages and dialects. Its robust performance in noisy environments makes it ideal for transcription services and accessibility tools​.
* **CoquiSTT**: Built as a modular and extensible platform, CoquiSTT emphasizes community collaboration. It provides fine-tuning capabilities for domain-specific applications, such as transcription of legal and medical content​.ref[1]

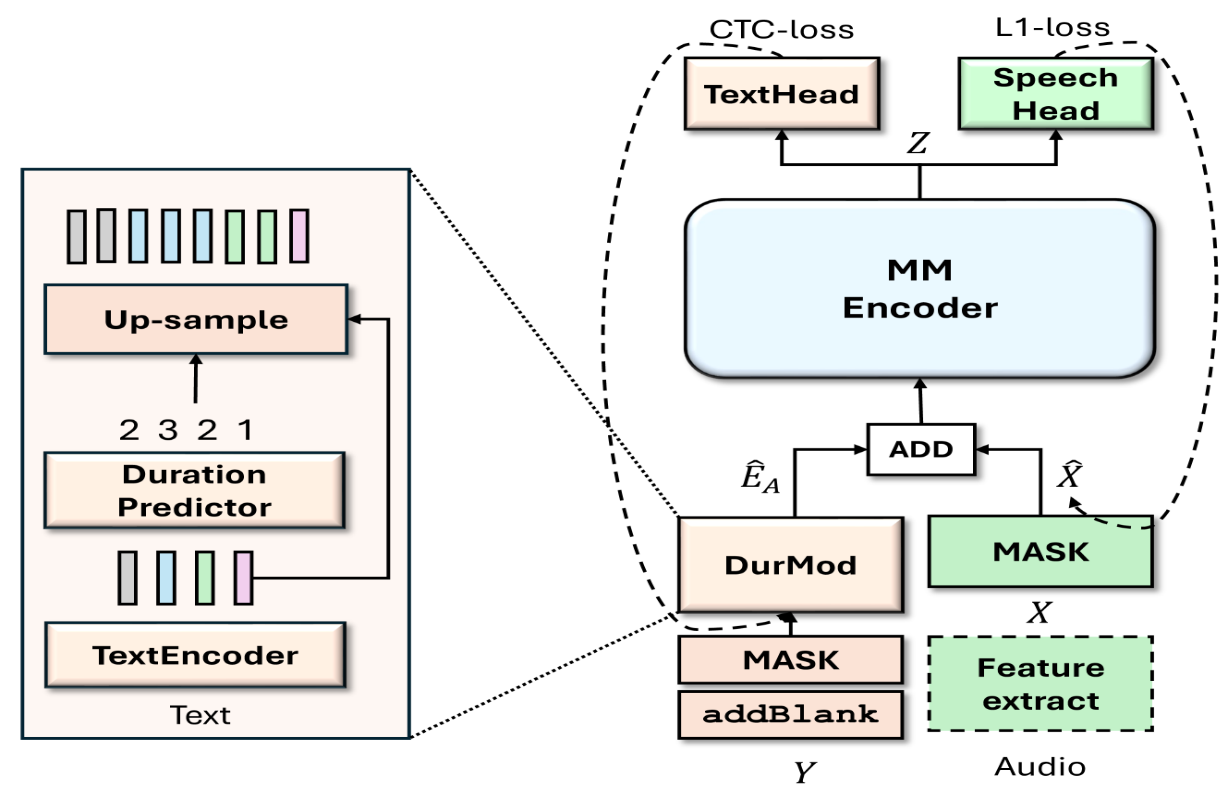
**A Fundamental End-to-End Speech Recognition Toolkit**



<https://blog.premai.io/the-rise-of-open-source-audio-models-text-to-speech-and-speech-to-text/>

**Model for both text and speech**

model design is such that it can take speech, text, or a combination of both features as input. Let X be a sequence of log-mel filterbank (LFB) features and Y be the corresponding transcript. The multimodal model consists of four major components and outlined below . ref[3]



The model starts by converting input text (e.g., "CAT") into an aligned form for speech synthesis.

* Text is interleaved with blanks: "CAT" → "C\_A\_T".
* A **Text Encoder** (2-layer conformer) processes this to get embeddings.
* During training, each character is repeated (R) based on ground-truth CTC alignment.
* During inference, a **Duration Predictor** estimates how many times to repeat each character (notated as R^).
* This repetition produces an upsampled version (EA) matching the length of the target speech.
* The **loss** L\_dur trains the model to predict R^ accurately using cross-entropy.

This is inspired by **FastSpeech**, but uses **character-based** CTC instead of phonemes.

### ****2. Masking Strategy****

To support **self-supervised learning** and robustness, the model applies **masking**:

#### **Text Masking (MaskY):**

* Random characters are replaced with <mask>.
* If a masked character is followed by a blank, that blank is also masked.
* This creates a noisy but structured input for learning.

#### **Speech Masking**:

* **MaskX1 (Wav2Vec-style)**: Random log-mel frames are zeroed out in short chunks.
* **MaskX2 (Refinement-style)**: Large chunks of time and frequency bins are masked to simulate missing data during inference refinement.

This masking helps the model learn even from **unpaired** or incomplete data.

### ****3. Multimodal Encoder (MMEncoder)****

The core of the system is a **shared encoder** that handles both text and speech.

* Takes two inputs: masked speech features (X^) and masked duration-aligned text embeddings (EA^).
* Both are linearly transformed and normalized to match dimensions.
* If either modality is missing, it’s replaced with a zero-vector or <mask> embeddings.
* The inputs are added together and passed to a **2-layer conformer encoder** to produce a unified representation Z.

This allows the model to work with:

* **Text only** (for TTS),
* **Speech only** (for STT),
* Or **both** together.

### ****4. Task-Specific Heads****

The final output Z is used for either speech or text prediction:

* **Speech Head**: Outputs predicted log-mel features (OX) used in TTS, trained using L1 loss.
* **Text Head**: Outputs text logits (OY) used in STT, trained with CTC loss.

Both heads are also 2-layer conformers followed by linear projection layers

ref: <https://arxiv.org/html/2501.09104v2#bib.bib11>

**Speech to Speech Model**

speech-to-speech translation, involve translating speech from one language into speech in another language. These models can be built using a cascade of systems: automatic speech recognition (ASR) to transcribe the input speech, text-to-text machine translation (MT) to translate the text, and text-to-speech (TTS) to synthesize the translated speech. Many text-to-speech models are available, including those that can generate speech in multiple languages and for multiple speakers.

Machine learning models analyze speech patterns, vocal characteristics, and linguistic elements to generate natural-sounding voice transformations. Neural networks process features like pitch, timbre, and pronunciation to create voice outputs that are nearly indistinguishable from human speech. The models continuously improve through training on diverse voice datasets, enabling increasingly realistic and expressive voice conversion.

Tavus API is a leading solution in this space, enabling developers to offer AI-generated voice and video personalization at scale. With Tavus, end users can create dynamic, hyper-personalized voice content that enhances customer engagement and automation while maintaining a natural, human-like experience.ref[4]

**How AI Speech-to-Speech Technology Works**

Modern speech AI operates through three key neural networks: a **speech recognition engine**, a **language processor**, and a **voice synthesizer**. When you speak, your voice is converted to text, the meaning is analyzed, and a new voice—matching your tone or transformed—is generated.

These systems are trained on millions of recordings, learning nuances like tone, rhythm, and pauses. As a result, today’s models produce natural, emotionally expressive speech that closely mirrors real conversation—far beyond the robotic voices of the past.ref[4]

**S2S APIs**

**1.Tavus API:**

The Tavus API is a video generation tool that uses speech-to-speech technology to create realistic AI voices. It enables users to produce personalized videos—such as marketing, training, or educational content—at scale, without needing AI knowledge or coding skills. With just a two-minute training video, Tavus handles the entire creation process.

**2. Replica Studios:**

specializes in replicating human voice using text-to-speech and speech-to-speech AI voice technology The platform's API enables developers to transform voices for games, animation, and interactive media.

**3.** **Microsoft Azure Speech Services**

It offers speech recognition and speech-to-speech capabilities and Azure ecosystem integration for workflow automation. It offers real-time and batch processing services**.**

**How does speech-to-speech conversion differ from text-to-speech?**

Speech-to-speech conversion analyzes spoken audio input and generates new audio in a different voice, maintaining the original speaker's tone, pace, and emotion. Text-to-speech reads written text aloud using predefined voice models. Converting between speech requires precise neural processing to capture subtle vocal elements like pitch variation, speaking rhythm, and emotional undertones.

For example, when a marketing team needs to localize video content, speech-to-speech APIs can transform the narrator's voice into multiple languages while keeping their unique speaking style intact.

**How can I integrate a speech-to-speech API into my application?**

1. Adding speech-to-speech capabilities requires:
2. Creating an API account and generating access credentials
3. Installing language-specific SDK (Python, Node.js, etc.)
4. Configuring audio input/output parameters
5. Making API calls to send source audio and receive converted speech
6. Implementing error handling and retry logic

With Tavus API, you can access speech-to-speech technology without the labor-intensive process of configuring the AI model—you can provide high-quality AI video generation without any experience with artificial intelligence or coding.Implement Tavus API today.

**Reference**

**[1]** <https://blog.premai.io/the-rise-of-open-source-audio-models-text-to-speech-and-speech-to-text/>

**[2]** <https://github.com/AI4Bharat/Indic-TTS>

**[3]** <https://arxiv.org/html/2501.09104v2#bib.bib11>

**[4]**<https://www.tavus.io/post/speech-to-speech#what-is-speech-to-speech-technology>