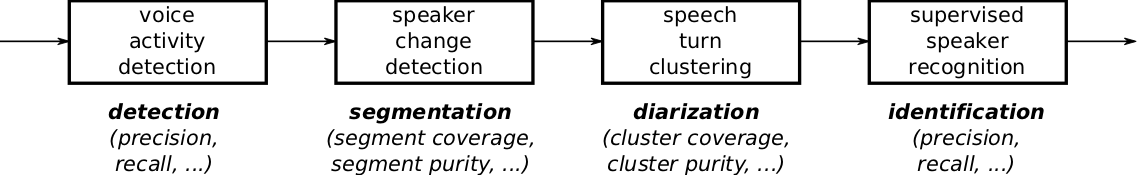
**Introduction**

Speech processing is an area of study dedicated to transforming speech into digital signals and interpreting them (sciencedirect). (why is processing speech done, why is it relevant, why should we spend time on it?) Speech is rich in complexity, and in a natural setting conversations are spontaneous and overlapping; models that perform well in a controlled setting completely fall apart in a noisy environment. This is addressed in early literature (earlySpeech cite) and is still unsolved, but the field has developed rapidly in recent years, driven by advancements in deep learning and big data (Xu et al. 2021) (Ng. et al. 2024). Current methods treat speech processing as a pipeline – torchaudio (torchaudio cite) and pyannote (pyannote cite) are leading examples of solving speech processing tasks using many sub-models as ‘building blocks’ to solve larger problems (fig: The pyannote pipeline for speech processing. Their metrics are shown in parentheses).

 Speaker diarisation, speaker separation and speech transcription have gained attention as focal tasks in speech processing pipelines, particularly in applications such as meeting transcription (Song et al. 2021), surveillance (Crocco et al. 2016) and human-computer interaction (Helander 2014). Speaker diarisation refers to the process of determining “who spoke when” in an audio stream. Unlike transcription, which focuses on the content of speech, diarisation segments audio by speaker identity (NEEDS CITE). Speaker separation aims to extract speech from audio, often a multi-speaker or noisy environment. (show a figure of each and visually show what they are). While most speech processing research is spent on these tasks, many pipelines fail due to unreliable upstream processes, or in some cases, a complete lack of them. For example, (n=2, n=3 tasks require n, but fail to provide a check for *n*.) There are some attempts to create larger *n* models, but these are too rigid. A major missing piece of the pipeline is counting the number of speakers effectively, at any given moment.

There are arguments that speaker counting is unnecessary; the task can be skipped via successful transcription/diarisation; if we know “who spoke when”, its clear that metadata would already contain speaker counts at any given moment.

Overlapping speech is particularly a significant challenge; Yella and Herv{\'e} (2014) emphasise that “overlapping speech [is] one of the main sources of errors in diarization”; the ability to robustly estimate how many speakers are active at a given moment remains an upstream bottleneck.

Without accurate speaker activity estimation, downstream tasks often fail to segment turns correctly, attribute speech accurately, or maintain coherent transcripts.

Notes:

“Overlapping speech, background noise, short backchannels ("hmm", "yeah", "ok", ...), non-speech vocalization (laughter, coughing, ...). All of these make existing systems fail completely.”

There are n\_srcs = *i* models available, but no easy way to find *i* at any moment. Our speaker counter finds *i* at any given moment which should be passable to external models; this model is a piece of the pipeline.

There are ‘speech or no speech’ models; even a model which depicts 0, 1 or many (>1). But no actual speaker counter (using mono). There are successful attempts made with multi-channel audio; it’s a much simpler task when a model is provided spatial information.

Another problem area is creating a low-latency model which can be pushed to production.

Chatgpt crap:

Much of the existing work in speaker counting relies on multi-channel inputs or controlled environments. These methods often require spatial cues that are not present in single-channel audio, limiting their applicability in real-world or resource-constrained scenarios. Mono-channel speaker count estimation remains underexplored, despite its practical value in applications like mobile devices, voice assistants, and call centers.

Several recent models have approached the speaker counting task using CNNs, RNNs, and attention-based architectures. These models are typically trained on synthetic mixtures or curated datasets like LibriMix or LibriSpeech. However, there is a gap in evaluating these systems on real-world, overlapping, mono-channel audio.

Overlap detection and the ability to model varying densities of speaker overlap are also key to performance. Some methods use frame-level classification to estimate count, while others regress over sliding windows or use auxiliary representations such as embeddings or spectrograms. These techniques vary in complexity, latency, and performance.

To date, real-time speaker count inference from single-channel overlapping audio remains a challenge, with most approaches either too slow for real-time use or too inaccurate for practical deployment. This work aims to bridge that gap by exploring efficient, low-latency models capable of handling varying overlap densities and generalizing across naturalistic audio.

Overlapping speech presents a major challenge in speech processing tasks such as diarisation and transcription. Accurate speaker counting is critical in environments with variable overlap densities, from no overlap to full concurrency.

Speaker counting approaches vary between offline and real-time methods. Models have included convolutional neural networks, recurrent networks, convolutional-recurrent hybrids, and transformer-based architectures. Most focus on offline processing.

Many existing systems rely on multi-channel or far-field microphone setups. Mono input, while limited, is more realistic in consumer and embedded settings. The constraint forces models to learn from time-frequency patterns without spatial cues.

Available datasets include LibriCSS, LibriMix, AMI, and others. These offer varying degrees of realism, overlap control, and microphone configuration. Most do not target mono, real-time inference with realistic acoustic conditions.

Prior work has focused on speaker counting using complex pipelines or multi-mic setups. Some explore CNNs on spectrograms for speaker activity detection. Others propose recurrent or CRN-based methods. Few operate on mono, single-segment audio in a truly real-time context.

There is a clear gap in models designed for mono input with low-latency inference on real-world audio containing dynamic overlap. No current approach targets all of these constraints simultaneously.

Applications include live meeting transcription, input to diarisation systems, or deployment on constrained hardware. These scenarios benefit from compact, accurate, and low-latency models.

This work proposes a lightweight, mono-input model capable of real-time inference across varied overlap densities, addressing a critical gap in current literature.