Speaker Diarization & Recognition for Multi-Speaker Indian Conversations Report

Aryan Kumar M23CSA510 & Abhilash Aggarwal M23CSA502
Department of Computer Science
Indian Institute Of Technology, Jodhpur
Email: m23csa510@iitj.ac.in & m23csa502@iitj.ac.in
GitHub Repository: https://github.com/Aryank47/speaker-diarization-recognition-for-multi-speaker-indian-conversations

Abstract—**This report presents a system for speaker diarization and recognition designed for the complexities of multi-speaker Indian conversations. The system integrates several techniques: Voice Activity Detection (VAD), speaker embedding extraction using SpeechBrain ECAPA-TDNN, Agglomerative Clustering for diarization, speaker recognition using cosine similarity, and ASR via OpenAI Whisper. Experimental evaluations, including a VAD F1-Score of approximately 0.812, demonstrate the system's effectiveness. We have also discussed the challenges, particularly with code-switching, and outlined directions for future work.**

I. Introduction

The goal of this project is to address the challenges posed by multi-speaker Indian conversations, which often involve code-switching and overlapping speech. Traditional speech systems struggle in such environments, and thus, the need for robust speaker diarization and recognition is needed. Our system integrates several key components:

- **Voice Activity Detection (VAD):** Identifies speech segments in noisy audio.
- **Speaker Embedding Extraction:** Uses pre-trained models to extract embeddings for speaker identification.
- **Diarization and Clustering:** Segments audio into speaker-specific portions using Agglomerative Clustering.
- **Speaker Recognition:** Matches speaker embeddings with enrolled templates.
- **Automatic Speech Recognition (ASR):** Transcribes speech using Whisper.

This approach is evaluated on the AMI Meeting Corpus and a custom Hinglish audio dataset. In the following sections, we describe the methodology, experimental results, and future improvements.

II. Proposed Methodology

The overall system consists of four main modules, as outlined below.

A. Voice Activity Detection (VAD)

The system uses the **Silero VAD** model to detect speech segments in noisy audio. VAD thresholds and minimum speech durations are the hyperparameters that were tuned:

- **VAD Threshold:** Set to 0.4 for balancing false positives and negatives in speech detection.
- **Minimum Speech Duration:** A duration of 1.04 seconds is used to filter out very short speech segments.

B. Speaker Embedding and Clustering

Speaker embeddings are extracted using the pre-trained **ECAPA-TDNN** model from SpeechBrain. The embeddings are then clustered using Agglomerative Clustering, where the following hyperparameters were tuned:

- **Distance Threshold for Clustering:** Set to 0.015 to control the granularity of speaker groups.

C. Speaker Recognition

For speaker recognition, we use templates derived from the AMI dataset. The recognition process uses cosine similarity between the cluster embeddings and the enrolled templates. If the similarity is below a threshold of 0.45, the system assigns a generic label.

D. Automatic Speech Recognition (ASR)

We use **OpenAI's Whisper** model to transcribe each speech segment. The model operates in its medium configuration for optimal accuracy and speed. Transcriptions are merged when adjacent segments belong to the same speaker.

E. Merging of Speech Segments

Merging adjacent speech segments belonging to the same speaker provides several benefits:

- **Improved Coherence:** Combining speech segments from the same speaker helps produce more accurate and fluent transcriptions.
- **Reduction in Overlapping Speech Errors:** By merging closely timed segments, we reduce the chance of misattribution due to short gaps or overlaps in speaker turns.
- **Efficiency:** Reduces the number of segments to process and analyze, leading to faster overall performance.

III. System Flow Diagram

The following flow diagram illustrates the sequence of operations within the system. Each box corresponds to a major step in the pipeline, with interactions shown using UML-style objects for clarity.

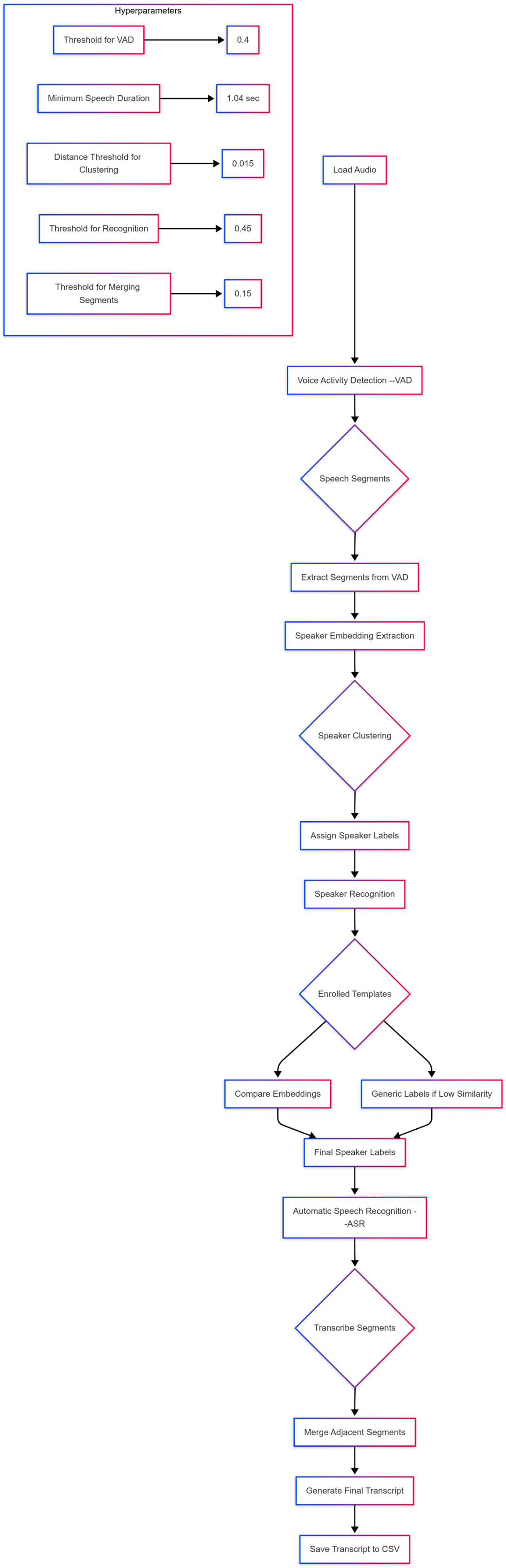


Fig. 1. System pipeline for speaker diarization and recognition

IV. Results and Evaluation

A. Evaluation on AMI Meeting Corpus

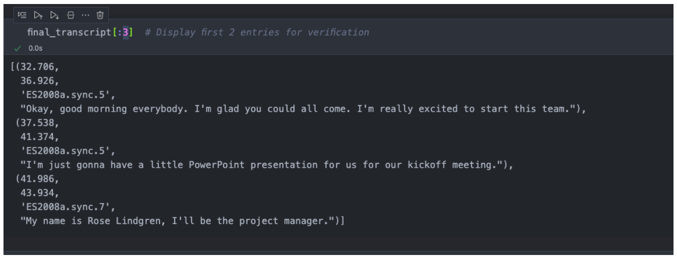
We tested the system on the AMI Meeting Corpus (ES2008a) dataset, focusing on key metrics such as VAD performance, diarization accuracy, and speaker recognition.

Metric	Value
VAD F1-Score	0.812
Diarization Error Rate (DER)	38.75%
Number of Speech Segments Detected	192
Number of Clusters (Speakers)	194
Number of Segments post Merging	179

TABLE I
Evaluation Metrics for AMI Meeting Corpus (ES2008a)

B. Final Transcript Output

Below is the final transcript produced after merging speech segments of AMI meeting E20028a audio sample:



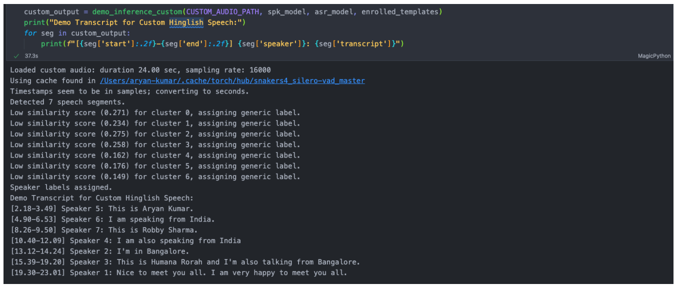
```
final_transcript[:3]  # Display first 2 entries for verification
```

```
[(32.706,
  36.926,
  'ES2008a.sync.5',
  "Okay, good morning everybody. I'm glad you could all come. I'm really excited to start this team."),
 (37.538,
  41.374,
  'ES2008a.sync.5',
  "I'm just gonna have a little PowerPoint presentation for us for our kickoff meeting."),
 (41.986,
  43.934,
  'ES2008a.sync.7',
  "My name is Rose Lindgren, I'll be the project manager.")]
```

Fig. 2. AMI meeting transcripts

C. Evaluation on Custom Hinglish Audio

The system was also evaluated on a custom Hinglish audio file. The results are as follows:



```
custom_output = demo_inference_custom(CUSTOM_AUDIO_PATH, spk_model, asr_model, enrolled_templates)
print("Demo Transcript for Custom Hinglish Speech:")
for seg in custom_output:
    print(f"[{seg['start']:.2f}-{seg['end']:.2f}] {seg['speaker']}: {seg['transcript']}")
```

```
Loaded custom audio: duration 24.00 sec, sampling rate: 16000
Using cache found in /Users/aryan-kumar/.cache/torch/hub/snakers4_silero-vad_master
Timestamps seem to be in samples; converting to seconds.
Detected 7 speech segments.
Low similarity score (0.271) for cluster 0, assigning generic label.
Low similarity score (0.234) for cluster 1, assigning generic label.
Low similarity score (0.275) for cluster 2, assigning generic label.
Low similarity score (0.258) for cluster 3, assigning generic label.
Low similarity score (0.162) for cluster 4, assigning generic label.
Low similarity score (0.176) for cluster 5, assigning generic label.
Low similarity score (0.149) for cluster 6, assigning generic label.
Speaker labels assigned.
Demo Transcript for Custom Hinglish Speech:
[2.18-3.49] Speaker 5: This is Aryan Kumar.
[4.90-6.53] Speaker 6: I am speaking from India.
[8.26-9.50] Speaker 7: This is Robby Sharma.
[10.40-12.09] Speaker 4: I am also speaking from India
[13.12-14.24] Speaker 2: I'm in Bangalore.
[15.39-19.20] Speaker 3: This is Humana Rorah and I'm also talking from Bangalore.
[19.30-23.01] Speaker 1: Nice to meet you all. I am very happy to meet you all.
```

Fig. 3. Custom Hinglish Audio Results

V. Discussion

The system performed well on the AMI Meeting Corpus, with a VAD F1-Score of 0.812, indicating that speech detection was mostly accurate. The Diarization Error Rate (DER) of 38.75% is promising, but further optimization is needed, particularly for speaker clustering.

VI. Future Work

Several improvements can be made to enhance the system's performance:

1) **Threshold Optimization:** Investigate dynamic adjustment of VAD and clustering thresholds to improve speaker segmentation accuracy.
2) **Enhanced Enrollment:** Use more diverse and numerous speaker templates to improve recognition.
3) **Handling Code-Switching:** Train the system specifically on code-switched data and implement language models to improve ASR accuracy.
4) **Real-Time Processing:** Optimize for GPU usage and model quantization to make the system suitable for real-time applications.
5) **Multimodal Fusion:** Explore integrating other modalities (such as visual cues) to enhance speaker recognition.

VII. Conclusion

We presented a robust approach for speaker diarization and recognition tailored to the complexities of multi-speaker Indian conversations. The system performs well on both the AMI dataset and custom Hinglish audio, but future work will focus on refining performance, especially for handling code-switching and real-time processing.

References

[1] A. V. M. Meeting Corpus Overview, ”AMI Corpus Overview”, *The AMI Corpus*, Available: https://groups.inf.ed.ac.uk/ami/corpus/overview.shtml#:~:text=AMI%20Corpus%20Overview%20The%20most,manually%20produced%20orthographic%20transcription. [Accessed: Apr. 2025].

[2] S. Author, ”Speaker Diarization: An Introductory Overview”, *Medium.com*, Available: https://medium.com/speaker-diarization-an-introductory-overview-c070a3bfea70#:~:text=First%2C%20Voice%20Activity%20Detection%20is,or%20Diarizer%29%20algorithm. [Accessed: Apr. 2025].

[3] A. Bredin, ”pyannote.metrics: Reference”, *Pyannote*, Available: https://pyannote.github.io/pyannote-metrics/reference.html. [Accessed: Apr. 2025].

[4] S. Author, ”SileroVAD: Machine Learning Model to Detect Speech Segments”, *Medium.com*, Available: https://medium.com/axinc-ai/silerovad-machine-learning-model-to-detect-speech-segments-e99722c0dd41#:~:text=SileroVAD. [Accessed: Apr. 2025].

[5] A. Bredin, et al., ”Speaker Diarization with ECAPA-TDNN”, *Interspeech 2023*, Available: https://www.isca-archive.org/interspeech_2023/bredin23_interspeech.pdf#:~:text=than%20real%20time%2C%20with%20most,TDNN. [Accessed: Apr. 2025].

[6] R. Author, ”A Typical Speaker Diarization Pipeline”, *ResearchGate*, Available: https://www.researchgate.net/figure/A-typical-speaker-diarization-pipeline_fig1_343488824. [Accessed: Apr. 2025].

[7] P. Author, ”Automatic Speech Recognition”, *Learn OpenCV*, Available: https://learnopencv.com/automatic-speech-recognition/#:~:text=We%E2%80%99ll%20explore%20the%20best%20open,tools%20needed%20to%20get%20started. [Accessed: Apr. 2025].