future internet

MDPI

Article

RAISE: A Conversational AI Framework for Real-Time Transcription, Analysis of Sentiment, and Insights for Support and Engagement Coaching

Ajay Bandi [1,*], Benitha Sri Panchagiri [1], Rushika Reddy Pingili [1], Akash Thanneeru [1], Jonathan Grant Geren [2] and Popin Bose Roy [3]

[1] School of Computer Science and Information Systems, Northwest Missouri State University, 800 University Dr, Maryville, MO 64468, USA; panchagiribenithasri@gmail.com (B.S.P.); pingilirushikareddy@gmail.com (R.R.P.); athanneeru@outlook.com (A.T.)
[2] School of Business, Northwest Missouri State University, 800 University Dr, Maryville, MO 64468, USA; grantgeren9@gmail.com
[3] Zillow Group, 1301 2nd Avenue, Floor 31, Seattle, WA 98101, USA; popinb@zillowgroup.com
* Correspondence: ajay@nwmissouri.edu

Abstract

RAISE (Real-time AI Insights for Sentiment and Engagement) is a preliminary framework that combines speech transcription with Whisper, speaker diarization with pyannote.audio, text-based emotion classification with DistilRoBERTa, and feedback generation with GPT-3.5. We examined the workflow using 53 audio clips: 12 author-recorded conversations, 23 YouTube clips, and 18 clips from public emotion-labeled datasets, including RAVDESS and IEMOCAP. The experiments provide an initial demonstration of how these components can be integrated and identify speaker diarization as a major source of processing latency. Because the dataset is small and the evaluation is exploratory, the results do not establish generalizable accuracy or reliable real-time performance. Further evaluation using larger, independently labeled datasets and longer conversations is needed.

Keywords: conversational AI; speech transcription; speaker diarization; emotion classification; generative AI; AI agents; AI sales coaching; customer engagement

Check for updates

Academic Editors: Eirini Eleni Tsiropoulou and Symeon Papavassiliou

Received: 21 August 2025
Revised: 15 October 2026
Accepted: 23 October 2026
Published: 5 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The growing integration of artificial intelligence (AI) into everyday business operations has driven a transformation in how organizations interact with data and engage with customers. Two major forces behind this shift are AI analytics [1] and conversational AI [2,3]. While AI analytics enables organizations to extract real-time insights from huge volumes of data, conversational AI facilitates natural, human-like interactions through chatbots, virtual assistants, and live dialogue systems.

However, these technologies often function independently: AI analytics focus on data insights, while conversational AI emphasizes interactive dialogue [4]. There is an emerging need for a unified approach that combines the real-time analytical power of AI with the dynamic, contextual flow of live conversations. This convergence can enable more intelligent systems that not only understand what is being said but also interpret the tone, intent, and level of user engagement.

For instance, as shown in Figure 1, a customer says, "I can't log into my account after updating my password." The AI analyzes this input and immediately offers a tip to the agent: check if the customer is using a VPN or browser auto-fill, which are common causes of login

issues. Guided by this suggestion, the agent replies, "Sometimes a saved password or VPN causes issues. Can we try logging in using a private window?" This quick, context-aware support streamlines issue resolution and improves customer satisfaction.

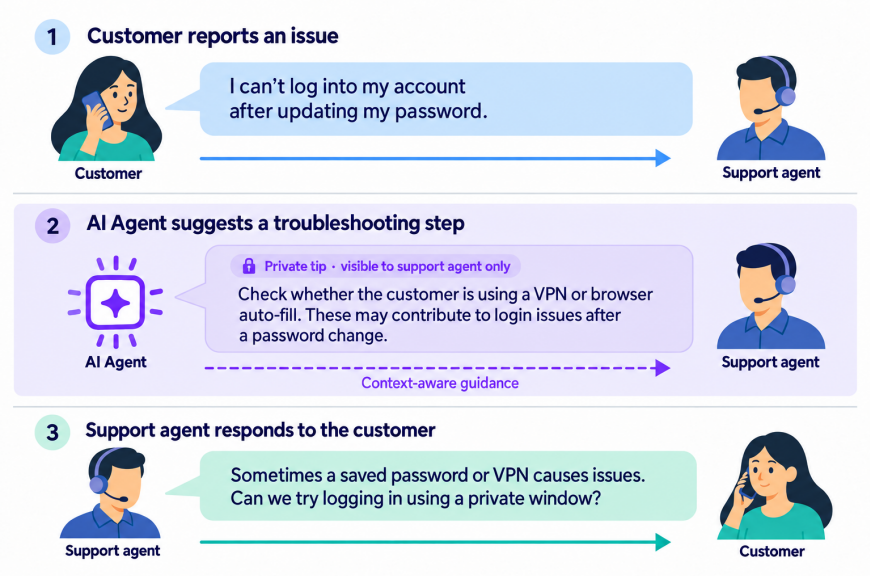


Figure 1. AI Agent provides real-time troubleshooting guidance to a support agent during a customer live calls.

In another case, as demonstrated in Figure 2, during a live customer interaction, if a customer says, "These shoes are a bit expensive compared to others," the AI system can immediately detect a potential objection, identify the sentiment, and suggest that the agent highlight features such as durability or long-term value. At the same time, it can pull up more affordable options from the inventory, allowing the agent to respond both informatively and empathetically. These examples illustrate how a future implementation of RAISE could provide contextual prompts to a human agent. The suggestions are illustrative rather than measured outcomes; whether they improve issue resolution, customer satisfaction, or agent performance would require evaluation with users in realistic settings.

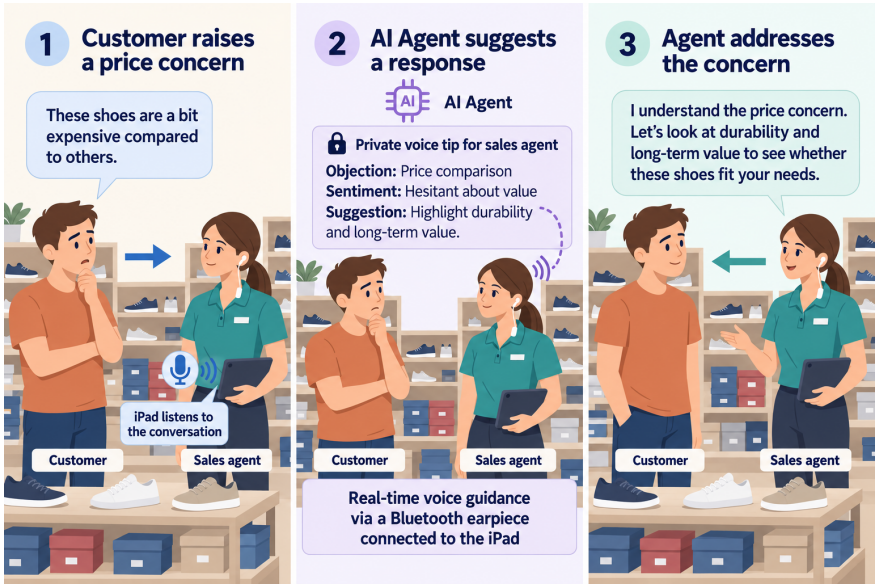


Figure 2. AI Agent guides a sales agent in addressing a customer's price concern.

Existing tools vary in whether they analyze conversations during or after an interaction and in how they combine transcription, speaker attribution, emotion classification, and contextual feedback [5–8]. RAISE is proposed as a modular workflow that brings these stages together. Its intended use cases include customer-support assistance, sales conversations, organizational coaching, and, with appropriate safeguards and domain-specific validation, educational or healthcare communication. This study focuses on describing the architecture and demonstrating its operation on a small collection of audio clips; it does not establish effectiveness in these application domains.

This research introduces the RAISE framework, designed to enhance live customer–agent conversations through real-time transcription, emotion detection, contextual understanding, and AI-generated coaching. In this initial study, RQ1 is addressed by implementing and examining the integration of transcription, diarization, and text-based emotion classification. RQ2 is explored through illustrative GPT-generated feedback and human review of its clarity, kindness, and helpfulness. These activities constitute a preliminary feasibility assessment, not a controlled test of real-world effectiveness. Larger-scale validation, component-level analysis, and evaluation with intended users remain future work.

RQ1: How can a conversational AI system effectively integrate speech transcription, speaker diarization, and emotion detection to provide personalized real-time feedback during sensitive or emotionally expressive conversations?

RQ2: Can a modular AI framework accurately interpret and respond to emotionally rich interactions in a way that supports coaching and self-reflection?

The rest of this paper is organized as follows: Section 2 presents related work; Section 3 introduces the RAISE framework, with several components discussed in detail in its subsections; Section 4 describes the experimental setup, including an overview of the dataset and details on system and software requirements; Section 5 provides the results and analysis of the research findings; Section 6 discusses the limitations of the study; and Section 7 concludes this paper.

2. Related Work

Table 1 summarizes selected features described in the cited work and the design goals of RAISE. Because the systems were not evaluated under a common protocol and RAISE was tested on a small dataset, the table should be read as a qualitative comparison of scope and architecture, not as evidence that RAISE outperforms the other systems.

Table 1. Comparison of RAISE with Related Frameworks.

Framework	Real-Time Transcription	Sentiment Analysis	Adaptive Coaching	Open Framework	Merits	Limitation
Woebot [9]	No	Yes (predefined)	Yes	No	Mental health-focused	Limited real-time interaction
Replika [10]	No	Yes	Yes	No	Human-like conversations	Black-box design
EngageMeter [11]	Yes (via video)	No	No	No	Visual engagement tracking	Lacks NLP context
RAISE (This work)	Yes	Yes (real-time NLP)	Yes	Yes	Proposed modular workflow; preliminary prototype	Early-stage evaluation

Recent progress in generative AI has opened up many new uses in different fields. A review paper [12] described a general overview of generative AI models, input and output types, and how they are measured. However, their work only considers the theoretical level and does not include real-time or interactive systems. RAISE puts these ideas into action by supporting live conversations between multiple speakers with personalized and emotion-aware coaching. Similarly, Shahin, Chen, and Hosseinzadeh [13] focused on feedback tools like surveys and interviews, especially in the service industry. They show how GPT-3.5 can be used to find insights in customer-service talks. While useful, their work looks only at static Voice of Customer (VoC) analysis. RAISE builds on this by offering live, flexible VoC insights, which can be very helpful in sales.

Bandi et al. [14] discussed how well generative AI answers perform using clear evaluation methods, but only in single-turn settings. It does not consider emotional depth or multiple speakers. RAISE improves this by working with live conversations, identifying speakers, recognizing emotions, and understanding context for more natural, human-like interaction. Huijbregts [15] explored making AI more usable through voice interfaces. Their approach uses voice tools, which help with usability, but the interaction is basic. RAISE improves this by adding speaker context and emotion detection for more meaningful conversation. Krugmann and Hartmann [16] reviewed how well LLMs like GPT-3.5, GPT-4, and LLaMA handle sentiment analysis in zero-shot settings. But these are tested on short, single-turn inputs. RAISE works beyond that by working with ongoing conversations, adding emotion and speaker awareness for more detailed, real-time coaching.

In attempting to improve chatbot accuracy using knowledge graphs [17], generative AI answers have been tested using structured prompts and feedback, but only in text form. RAISE changes this by working with live, speaker-tagged conversations that include emotional context, giving smarter and more flexible feedback. Moattar and Homayounpour [18] offer a deep review of speaker diarization, explaining how to divide and group speaker segments. Their work, though detailed, focuses only on offline methods.

In a case study on the life cycle of generative AI projects using large language models, ref. [19] showed how LLMs such as Google Flan can support domain-specific chatbots using structured inputs. But their system runs into limits with CPU power and does not support real-time, multi-modal use. Across these studies, prior work addresses individual tasks or particular conversational settings, while RAISE explores how transcription, diarization, text-based emotion classification, and generated feedback can be connected in one workflow. This is an architectural distinction, not a demonstrated performance advantage. The current evaluation does not support claims that RAISE is more accurate, natural, or effective than the cited approaches.

3. RAISEArchitecture

Figure 3 presents the proposed RAISE architecture and the modules intended to support audio input, transcription, speaker diarization, text-based emotion classification, contextual analysis, and feedback presentation. The prototype uses Whisper, pyannote.audio, a DistilRoBERTa-based emotion classifier, and GPT-3.5. Other elements shown in the architecture, including streaming, storage, security, and dashboard functions, should be described as implemented only where they were actually built and tested. The current study primarily examines the processing workflow and its behavior on a limited set of clips.

The components were selected to assemble a prototype from available tools and services. Their inclusion reflects implementation choices rather than a claim that they are optimal. The framework combines open-source components with GPT-3.5, a proprietary model service. Reproducibility depends on recording model versions, settings, prompts, and execution conditions.

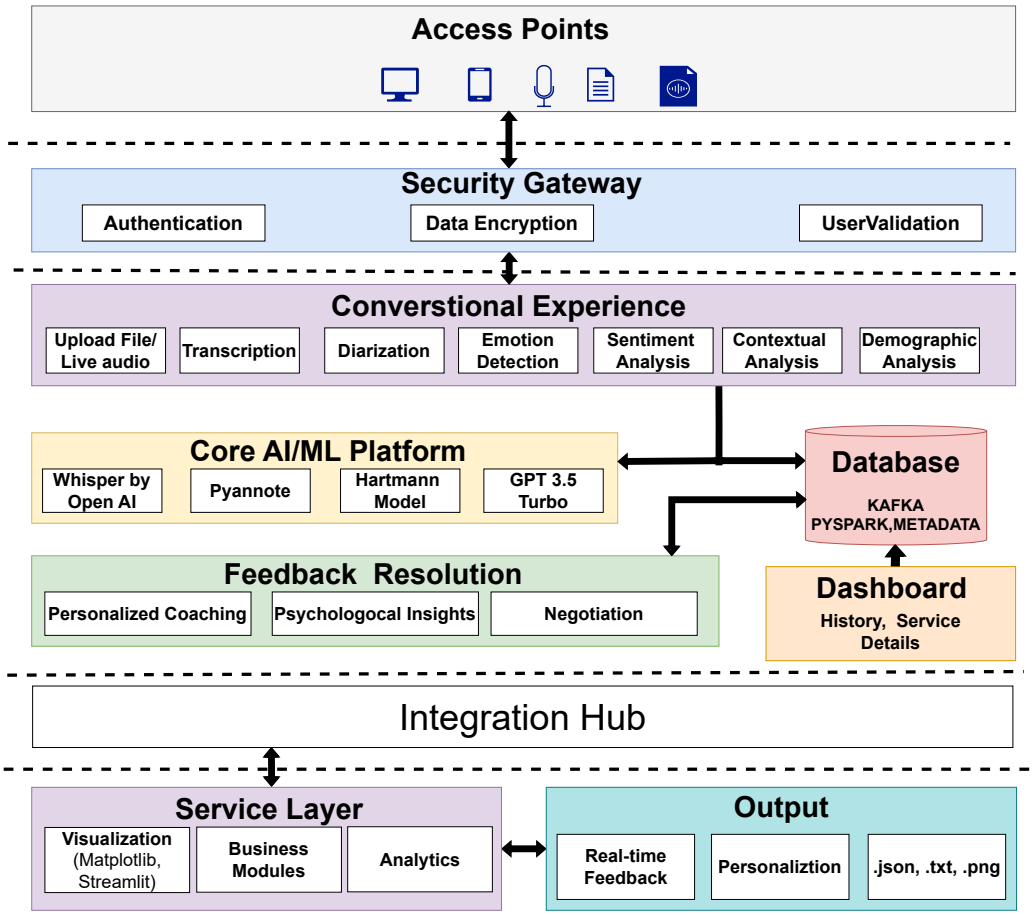


Figure 3. Proposed RAISE architecture. The arrows represent the interactions between the layers or components.

3.1. Transcription

Transcription is the first operational layer in the RAISE framework, transforming raw spoken audio into structured, time-aligned text. This output serves as the foundation for processes such as speaker diarization, emotion analysis, and contextual coaching. Precision and real-time performance at this stage are essential to ensure accurate and meaningful interpretations throughout the system.

Whisper is a Transformer-based automatic speech recognition model developed by OpenAI [3,20]. Its published training and design characteristics motivated its selection for the prototype. Transcription performance in RAISE depends on audio quality, accents, noise, and processing settings; the present evaluation does not establish its accuracy across these conditions. The transcription pipeline comprises four sequential stages, explained below and shown in Figure 4.

1. Input Audio Acquisition: Audio can be captured live through a user's microphone or uploaded as a pre-recorded WAV file. Both live and file-based inputs are supported via RAISE's web and mobile interfaces.
2. Preprocessing:
   (a) Resampling: The raw audio is resampled to 16 kHz, ensuring consistent frame lengths and frequency resolution across sources.

(b) Framing & STFT: It is segmented into overlapping frames of 400 samples (25 ms) with a hop of 160 samples (10 ms). Each frame undergoes a short-time Fourier transform (STFT) [21]:

$$X[n,k] = \sum_{m=0}^{W-1} x[nH+m]\, \mathrm{e}^{-j2\pi\, k\, m/W} \tag{1}$$

where $X[n,k]$ is the complex coefficient for frame n and bin k, W the window size, H the hop size, and $x[n]$ the resampled waveform.

(c) Mel Filterbank Projection: The magnitude spectrum $|X[n,k]|$ is mapped onto an 80-channel Mel filterbank to match human auditory sensitivity [22]:

$$M[n,m] = \sum_{k} |X[n,k]|\, \phi_m(f_k) \tag{2}$$

where $\phi_m(f_k)$ is the response of the m-th Mel filter at frequency f_k.

(d) Logarithmic Compression: Finally, the Mel spectrogram is log-compressed to stabilize its dynamic range [23]:

$$S_{\text{log-mel}}[n,m] = \log\big(M[n,m] + \epsilon\big) \tag{3}$$

with $\epsilon = 10^{-6}$ for numerical stability. The resulting $S_{\text{log-mel}}$ is the input to Whisper's Transformer encoder.

3. Model Inference: The log-Mel spectrogram $S_{\text{log-mel}}$ is fed into Whisper's encoder–decoder Transformer, which uses multi-head self-attention, positional encodings, and layer normalization. It auto-regressively predicts a sequence of tokens that form the transcript.
4. Output Generation: Whisper produces
   (a) a human-readable plain-text transcript (.txt);
   (b) a structured metadata file (`.json`) with word-level timestamps and confidence scores.

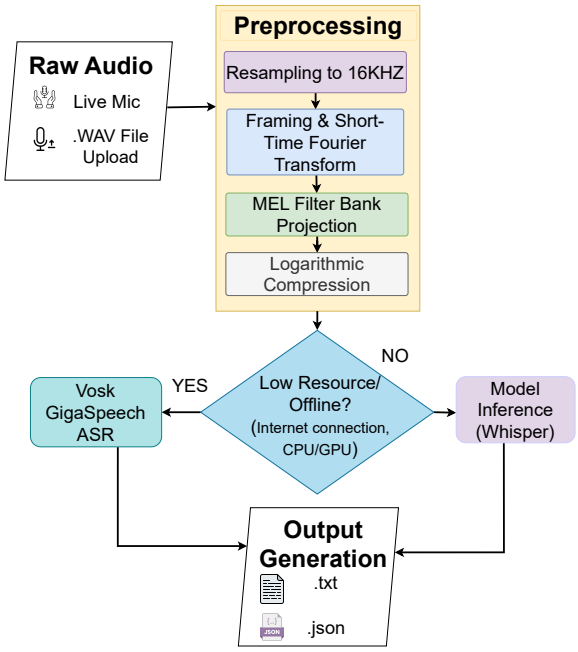


Figure 4. Transcription stages in the RAISE framework.

To minimize end-to-end latency, RAISE processes audio in overlapping 30 s windows, enabling near-real-time inference. In low-resource or offline environments, it seamlessly

falls back to the lightweight Vosk GigaSpeech ASR engine [24] transcription with reduced computational demands.

3.2. Speaker Diarization

Speaker diarization is the task of segmenting a continuous stream of multi-speaker conversational audio into temporally aligned segments, each attributed to a distinct, anonymous speaker. It answers the core question—"Who spoke when?"—and serves as a foundational layer for systems like RAISE that analyze emotional tone, conversational intent, and behavioral patterns. By segmenting audio into speaker-specific intervals, diarization allows downstream modules—such as emotion classification and generative coaching—to operate on correctly attributed utterances, without requiring any prior speaker identity.

In the RAISE architecture, we implement speaker diarization using the PyAnnote.audio framework [25], which provides an end-to-end neural pipeline for unsupervised speaker segmentation. The system begins by detecting speech-active regions using a pretrained Voice Activity Detection (VAD) model [26], isolates speaker-specific acoustic features via ECAPA-TDNN embeddings [27], and clusters them using Ward's hierarchical clustering [28], enhanced with overlap detection for multi-speaker regions. The diarization output provides speaker-labeled time segments for downstream emotion classification and feedback generation. Attribution errors, overlapping speech, and processing latency can affect subsequent stages; these effects were not comprehensively quantified in the current evaluation. The diarization pipeline follows five key stages, as shown in Figure 5.

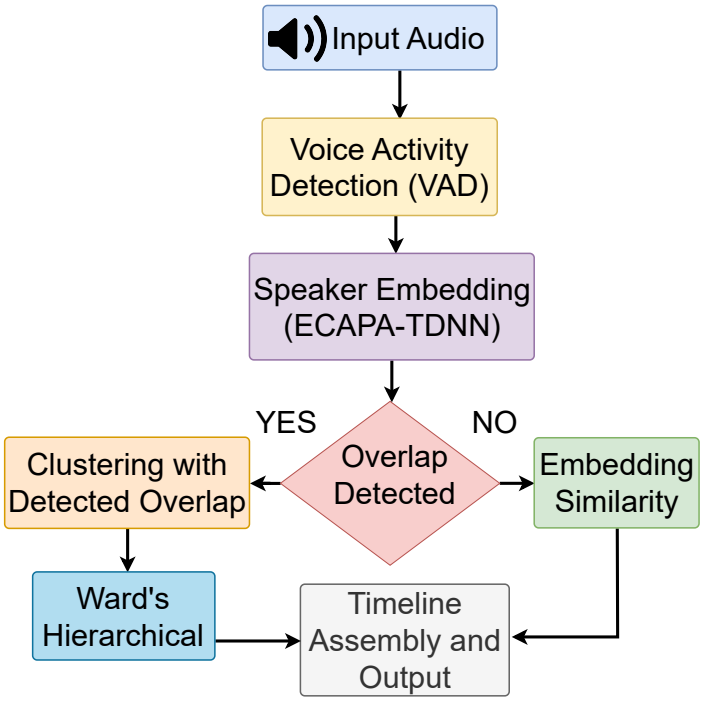


Figure 5. Speaker diarization in RAISE.

1. Voice Activity Detection (VAD): The diarization process begins by isolating segments of human speech from raw audio input. We apply pretrained Voice Activity Detection (VAD) model V to waveform $x(t)$, identifying intervals where speech is present and filtering out silence or background noise [26]. Here, $x(t)$ denotes the audio signal amplitude at time t, sampled at 16 kHz. The VAD model outputs a set of N time-aligned speech intervals.

$$\mathcal{S} = \{(t^i_{\text{start}}, t^i_{\text{end}})\}_{i=1}^{N} \quad (4)$$

Each pair $(t^i_{\text{start}}, t^i_{\text{end}})$ represents the start and end times (in seconds) of the ith detected speech segment. The complete set $\mathcal{S}$ constitutes the active speech timeline. Each interval is then used to extract a corresponding speech-only waveform segment.

$$s_i = x(t), \quad t \in \left[t^i_{\text{start}},\ t^i_{\text{end}}\right] \tag{5}$$

Here, s_i denotes the ith isolated speech segment, extracted by slicing the original waveform $x(t)$ within the bounds of $\mathcal{S}$. These segments s_i serve as input to the subsequent speaker embed stage.

2. Speaker Embedding: Each isolated speech segment s_i is encoded into a fixed-dimensional vector that characterizes the unique vocal features of the corresponding speaker. ECAPA-TDNN is used to generate speaker embeddings for segment comparison and clustering [27]. These embeddings are intended to capture speaker-related acoustic characteristics, but their quality can be affected by recording conditions, segment duration, noise, and overlap. The embedding process is defined as

$$\vec{e}_i = f_{\text{ECAPA}}(s_i) \in \mathbb{R}^{192} \tag{6}$$

Here, f_{ECAPA} denotes the nonlinear transformation learned by the ECAPA-TDNN model, which maps the input waveform s_i to a 192-dimensional speaker embedding $\vec{e}_i$. These embeddings encode speaker-specific characteristics such as vocal timbre, prosody, formant structure, and speaking rate, while remaining invariant to lexical content and background noise. The resulting vectors $\vec{e}_i$ serve as the fundamental representation for downstream speaker comparison and clustering operations.

3. Embedding Similarity: To determine which speech segments likely belong to the same speaker, we compute pairwise cosine similarity between their corresponding embeddings [28]. Given two embeddings $\vec{e}_i$ and $\vec{e}_j \in \mathbb{R}^{192}$, similarity is defined as

$$\text{sim}(\vec{e}_i, \vec{e}_j) = \frac{\vec{e}_i \cdot \vec{e}_j}{\|\vec{e}_i\| \cdot \|\vec{e}_j\|} \tag{7}$$

$$\text{dist}(\vec{e}_i, \vec{e}_j) = 1 - \text{sim}(\vec{e}_i, \vec{e}_j) \tag{8}$$

Here, $\vec{e}_i \cdot \vec{e}_j$ denotes the dot product, while $\|\vec{e}_i\|$ represents the Euclidean norm of the embedding. A lower cosine distance indicates a higher likelihood that segments s_i and s_j were spoken by the same individual. These pairwise distances serve as input to the clustering stage.

4. Clustering with Overlap Detection: To group speech segments belonging to the same speaker, we apply agglomerative hierarchical clustering with Ward's linkage criterion, which minimizes the total within-cluster variance. The clustering process iteratively merges pairs of clusters based on cosine distance until no inter-cluster distances fall below a predefined threshold:

$$\text{dist}(\vec{e}_i, \vec{e}_j) > \theta, \quad \theta \approx 0.6 \tag{9}$$

Here, $\text{dist}(\vec{e}_i, \vec{e}_j)$ is the cosine distance between two speaker embeddings $\vec{e}_i$ and $\vec{e}_j$, as defined earlier. The parameter θ controls the sensitivity of the clustering: smaller values result in more granular speaker segmentation.

To handle instances of overlapping speech—where multiple speakers talk simultaneously—we incorporate an auxiliary binary classifier. This model estimates the probability that a given time frame contains concurrent speech activity.

$$P_{\text{overlap}}(t) > \delta, \quad \delta \approx 0.5 \tag{10}$$

In this formulation, $P_{\text{overlap}}(t)$ denotes the model-predicted probability of overlapping speech at time t and δ is an empirical threshold. Segments with $P_{\text{overlap}}(t) > \delta$ are

flagged as multi-speaker regions and handled accordingly during timeline generation [25].

5. Timeline Assembly and Output: After clustering, each speech segment is assigned a speaker label s_k, resulting in a chronologically ordered, speaker-attributed timeline.

$$\mathcal{D} = \left\{\left(t^i_{\text{start}}, t^i_{\text{end}}, s_k\right)\right\}_{i=1}^{N} \tag{11}$$

Here, t^i_{start} and t^i_{end} denote the start and end times (in seconds) of the ith speech segment and s_k is the anonymous speaker label assigned to that interval. The resulting set $\mathcal{D}$ constitutes the complete diarized representation of the audio stream, which becomes a foundational input to downstream tasks such as per-speaker emotion classification and GPT-based coaching interventions. The diarization output is initially returned as a `pyannote.core.Annotation` object [26]. For integration with the RAISE architecture's feedback generation and visualization modules, it is converted into a JSON-serializable list of dictionaries.

```
# Convert Annotation to a structured list for JSON export
tracks = []
for turn, _, speaker in diarization.itertracks(yield_label=True):
    tracks.append({
        "start_time": round(turn.start, 3),
        "end_time":   round(turn.end,   3),
        "speaker_id": speaker
    })
# Example output:
# [
#   {"start_time": 0.200, "end_time": 1.500, "speaker_id": "SPEAKER_00"},
#   {"start_time": 1.800, "end_time": 3.900, "speaker_id": "SPEAKER_01"},
#   ...
# ]
```

3.3. Emotion Classification

Emotion classification is a core component of the RAISE framework, enabling the system to contextualize utterances with affective meaning in real time. By detecting emotional cues embedded in user conversations, the model provides a deeper understanding of speaker intent, tone, and interpersonal dynamics.

Emotion detection is the capture of specific moment-to-moment emotional expressions, such as joy, confusion, or concern, in the voice of a speaker, helping to reveal their immediate affective state. This typically takes place immediately following speaker diarization as part of the real-time conversational analysis. Sentiment analysis, on the other hand, evaluates the broader tone of the conversation, whether it feels generally supportive, neutral, or critical, based on the context and content of the dialogue. It is applied later in the pipeline, during the interpretive stage that informs the feedback and coaching layers.

RAISE applies the j-hartmann/emotion-english-distilroberta-base model to transcribed English text [29]. Its output is a predicted text–emotion label among seven classes: anger, disgust, fear, joy, neutral, sadness, and surprise [30]. These labels represent the model's classification of the words in an utterance; they should not be interpreted as a direct measurement of a speaker's internal emotional state or vocal prosody. Sarcasm, mixed emotions, cultural variation, transcription errors, and context may affect predictions. The six stages are shown in Figure 6.

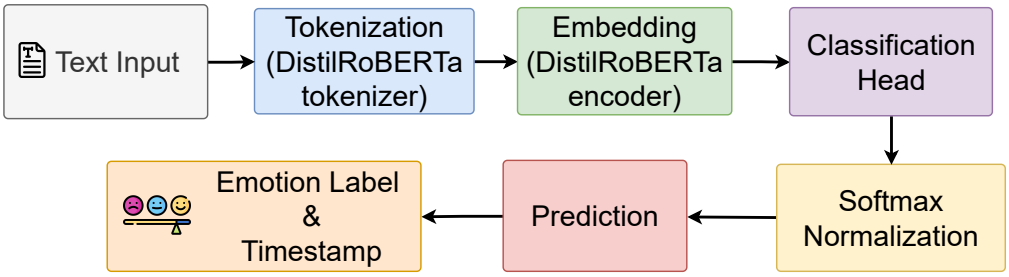


Figure 6. Sequential stages of emotion classification.

1. Text Input: The input is a natural language utterance, such as: "I'm tired of pretending everything's fine."
2. Tokenization: The sentence is tokenized using the DistilRoBERTa tokenizer, which splits it into WordPieces, prepends `[CLS]` (Classification), appends `[SEP]` (Separator) [4], and pads or truncates the sequence to a maximum length of 512 tokens.
3. Embedding: The tokenized input is passed through the DistilRoBERTa encoder [31], which maps each token to a high-dimensional vector called an *embedding*. These embeddings are contextualized, meaning that each vector reflects not just the token itself but also its relation to surrounding tokens. Among these, the embedding corresponding to the `[CLS]` token is extracted as a holistic representation of the input sentence:

$$\vec{h}_{\text{CLS}} = f_{\text{DistilRoBERTa}}(\text{input}) \tag{12}$$

Here, $f_{\text{DistilRoBERTa}}$ is the transformer encoder and $\vec{h}_{\text{CLS}} \in \mathbb{R}^d$ is the embedding vector (with $d = 768$).
4. Classification Head: The sentence-level representation $\vec{h}_{\text{CLS}}$, extracted from the final layer of the DistilRoBERTa encoder, is used as a fixed-length summary of the input utterance. This vector is then passed through a fully connected linear transformation—also known as the classification head—which projects it into a 7-dimensional vector space corresponding to the emotion classes:

$$\vec{z} = W \cdot \vec{h}_{\text{CLS}} + \vec{b} \tag{13}$$

Here, $W \in \mathbb{R}^{7\times d}$ is the weight matrix that maps the hidden state dimension d (typically 768) to the seven emotion classes. $\vec{b} \in \mathbb{R}^7$ is the bias vector added to each logit output. $\vec{z} \in \mathbb{R}^7$ is the output logits representing the unnormalized confidence scores for each of the seven emotions. This transformation enables the model to learn decision boundaries between emotion categories based on semantic and affective patterns in the hidden space.
5. Softmax Normalization: To interpret the raw logits $\vec{z}$ as probabilities, they are passed through the softmax function [32]:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{7} e^{z_j}}, \quad i = 1, \ldots, 7 \tag{14}$$

Here, z_i is the logit value for the ith emotion class. $P(y_i) \in [0, 1]$ is the normalized probability that the utterance belongs to the emotion class . This transformation ensures that the outputs are interpretable as a probability distribution over the full emotion label set, with $\sum_{i=1}^{7} P(y_i) = 1$.
6. Prediction and Alignment: The final predicted emotion label $\hat{y}$ is obtained by selecting the class with the maximum softmax probability:

$$\hat{y} = \arg\max_i P(y_i) \tag{15}$$

The predicted label $\hat{y} \in \{\text{anger, disgust, fear, joy, neutral, sadness, surprise}\}$ represents the most probable emotion associated with the utterance. This classification is then temporally aligned with the corresponding diarized segment s_k, which contains the speaker identity and timestamp metadata. This alignment allows the system to construct an emotion-aware speaker timeline, enabling per-speaker emotion tracking, contextual response generation, and affective coaching interventions within the RAISE framework.

To enable emotionally intelligent feedback, RAISE classifies each utterance into one of seven emotion categories, based on Ekman's basic emotion theory. These categories are shown in Table 2.

Table 2. Emotion categories and their psychological descriptions.

Emotion	Description
Anger	Hostility, frustration, aggression
Disgust	Aversion, moral repulsion
Fear	Anxiety, threat perception
Joy	Positive affect, satisfaction
Neutral	Factual or emotionally flat tone
Sadness	Loss, disappointment, despair
Surprise	Unexpectedness, astonishment

Each input utterance generates a 7-dimensional probability vector representing the model's confidence for each emotion class:

$$\vec{E}(x) = \left[P_{\text{anger}}, P_{\text{disgust}}, P_{\text{fear}}, P_{\text{joy}}, P_{\text{neutral}}, P_{\text{sadness}}, P_{\text{surprise}}\right] \tag{16}$$

The emotion with the highest probability in $\vec{E}$ is treated as the predicted emotion. The sum of all probabilities equals 1. Table 3 shows the heuristic settings used in this prototype.

To interpret these scores meaningfully during runtime, RAISE uses the following confidence strategy:

$$\text{EmotionLevel}(P(y_i)) = \begin{cases} \text{Strongly expressed} & \text{if } P(y_i) \geq 0.85 \\ \text{Context-dependent} & \text{if } 0.65 \leq P(y_i) < 0.85 \\ \text{Ambiguous/Weak} & \text{if } 0.50 \leq P(y_i) < 0.65 \\ \text{Neutral/Unclear} & \text{if } P(y_i) < 0.50 \end{cases} \tag{17}$$

These thresholds help RAISE distinguish between clear, context-sensitive, and uncertain emotional tones, enhancing the reliability of the generated feedback and coaching insights.

Table 3. Heuristic settings used in this prototype.

Emotion	Heuristic Settings	Trigger Examples
Joy	0.70–0.98	"I love this", "That's amazing"
Sadness	0.65–0.95	"I feel hopeless", "This hurts"
Anger	0.65–0.96	"This is infuriating", "I'm so mad"
Fear	0.50–0.88	"I'm scared", "Something feels wrong"
Disgust	0.55–0.92	"That's disgusting", "I hate this"
Surprise	0.60–0.92	"Wait, what?", "No way!"
Neutral	0.60–0.85	"It's 3 PM", "The file is saved"

3.4. Contextual Analysis Using GPT-3.5

After converting speech to text, identifying who said what, and detecting emotions, the RAISE system uses GPT-3.5 [33,34] to understand the deeper meaning behind each

statement. This includes recognizing the emotional tone, hidden intent, and flow of conversation. These insights help generate thoughtful, personalized coaching responses that are both timely and relevant.

The contextual analysis using GPT-3.5 happens in four main steps as shown in Figure 7.

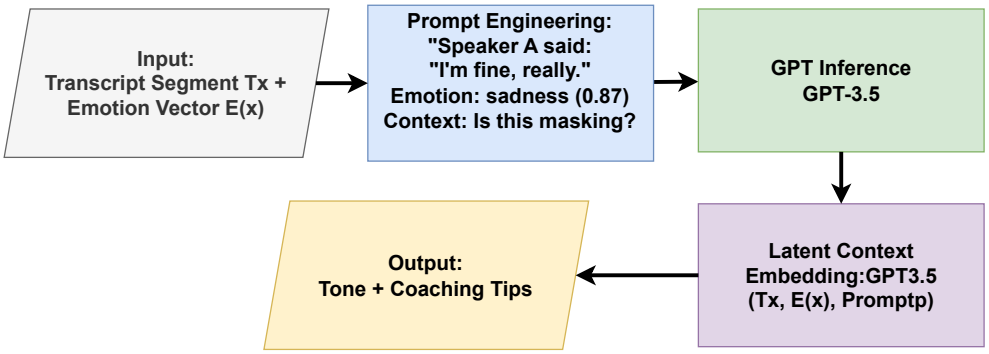


Figure 7. Contextual analysis in RAISE.

1. Input: The system takes transcript segments T_x, each labeled with the speaker and aligned with time. These are paired with a 7-dimensional emotion vector $\vec{E}(x) \in \mathbb{R}^7$, which represents the emotional state of the speaker [35].
2. Prompt Engineering: The prompt generation process in RAISE is dynamically driven. For each transcript segment, the system uses a combination of emotion scores and conversational context to generate a suitable prompt. For example,

```
Speaker A said: "I'm fine, really."
Emotion: sadness (0.87)
Context: Is this masking? What's a suitable empathetic response?
```

3. Model Inference: GPT-3.5 receives the transcript segment, predicted text–emotion label or scores, and a prompt requesting contextual feedback. It may generate interpretations or suggestions, but these outputs are model-generated hypotheses rather than verified detections of hidden intent, sarcasm, or emotional masking. Outputs can vary with prompts and context and should be reviewed by a person before being used in sensitive settings [34].
4. Output: The model returns a description of the speaker's tone (e.g., passive aggressive) and suggestions that guide the coaching system.

GPT-3.5 combines the transcript, emotion vector, and prompt into a single representation:

$$C(x) = \text{GPT-3.5}\left(T_x,\ \vec{E}(x),\ \text{Prompt}_p\right) \tag{18}$$

In this equation, T_x is the speaker's message, $\vec{E}(x) \in \mathbb{R}^7$ is the emotion vector for that message, and Prompt_p is the question used to guide the model [36].

This method allows GPT-3.5 to look beyond the literal words and understand the emotional and social context. As a result, RAISE can deliver feedback that feels more human, empathetic, and meaningful [37].

3.5. Feedback Resolution Layer

The Feedback Resolution Layer is the interpretive and decision-making engine of the RAISE framework. It synthesizes outputs from speaker diarization [38], emotional classification [29], and contextual reasoning [33] to generate personalized coaching cues, micro-interventions, or behavioral nudges tailored to each speaker.

This layer adapts its feedback based on the psychological tone and functional goal of the conversation. In negotiation-oriented settings, it emphasizes strategic reframing and compromise tactics or highlights leverage points (e.g., "Reframe your ask to align with mutual goals"). The feedback layer combines speaker labels, text-emotion predictions, and GPT-generated interpretations to produce suggestions such as reflection questions or active-listening prompts. These examples illustrate the intended output format; they do not establish that the suggestions improve communication or are suitable as therapeutic guidance. In sensitive settings, feedback should be treated as optional decision support and reviewed by a qualified person.

Each coaching instance is computed using the speaker identifier s_k, utterance segment T_x, emotion vector $\vec{E}(x)$, and contextual tag $C(x)$ from Equation (18), using the function

$$F(s_k) = f_{\text{coach}}(T_x,\ \vec{E}(x),\ C(x)) \tag{19}$$

where $F(s_k)$ represents the personalized feedback for speaker s_k. The output may be a reflection prompt, empathy cue, or active listening suggestion, dynamically selected based on the speaker's emotional state and dialogue context.

Feedback is rendered either in real-time—such as sidebar notifications during conversation—or post-session via a reflection dashboard. Each message is timestamped, attributed to its respective speaker, and aligned with the emotional and contextual data, ensuring transparent and actionable coaching [20].

3.6. Integration Hub

The integration layer aligns module outputs using speaker labels and timestamps and passes them to downstream components. Its modular design is intended to allow individual components to be replaced, although compatibility and the effects of such replacements would need to be tested.

3.7. Demographics Analysis

RAISE uses Named Entity Recognition (NER), an HuggingFace library, to identify key demographic details, such as the names of people, places or organizations, from the comments of the speakers. By detecting location-based phrases (for example, 'I'm calling from Houston'), the system builds a basic profile of the speaker's background. This helps tailor feedback and responses based on region-specific needs, such as local regulations, offers, or support guidelines, making the interaction more relevant and personalized.

3.8. Database

The database layer in the RAISE framework plays a critical role in ensuring real-time, context-aware responses during conversations. It stores session metadata, user profiles, and processed analysis outputs such as emotion labels, speaker attributions, and GPT feedback. When a user mentions specific items (e.g., "running shoes"), the system dynamically queries the database to retrieve related descriptions, product details, or recommendations, enabling intelligent responses aligned with the conversation context. Kafka and PySpark were used to enable real-time data streaming and parallel processing within the RAISE framework [39]. Apache Kafka facilitates real-time data streaming between services, while PySpark supports efficient processing of large-scale audio and metadata for both real-time and historical analysis.

3.9. Service Layer and Output

The Service Layer in the RAISE framework encapsulates business logic and presentation modules to enhance user interaction and system interpretability. It powers features like

emotional trend visualizations using Matplotlib and personalized coaching suggestions based on speaker-wise emotional timelines. Output components consolidate all generated insights into accessible formats (.json, .txt, .png), delivering real-time feedback and downloadable reports. These results are seamlessly pushed to users via dashboards or interfaces, providing actionable, emotion-aware guidance that enhances both user understanding and agent performance.

4. Experimental Setup

4.1. Dataset Overview

To evaluate the RAISE framework, we curated a dataset of 53 audio clips that reflect a wide range of emotional expressions, speaker interactions, and recording conditions. The 53 clips were selected to include variation in emotion, speaker count, and recording conditions. This selection provides a small exploratory sample, but it does not ensure robustness or generalizability. These files ranging from 30 s to 3 min and featuring 1 to 4 speakers are stored in the Test_Samples directory of the GitHub repository (https://github.com/bandiajay/RAISE (accessed on 22 October 2026)). Each clip was converted to .wav format and passed through the RAISE pipeline for transcription, speaker diarization, emotion classification, and, in selected cases, GPT-generated feedback.

The dataset consists of 53 audio clips, carefully curated to ensure diversity in speech patterns, emotional tones, and recording conditions. It includes 12 author-recorded conversations that capture distinct emotional expressions such as anger, sadness, and joy using high-quality audio. Additionally, 23 real-world YouTube clips were selected from publicly available interviews, debates, and vlogs to represent varied accents, speaking styles, and background noise conditions. Each of these clips was manually reviewed and annotated for emotional content and speaker identity. The dataset also includes 18 clips from established public datasets such as RAVDESS and IEMOCAP, which provide speech labeled as spontaneous and emotional. Although the dataset is relatively small, it was intentionally designed to support an early evaluation of the RAISE framework's performance across a range of realistic and controlled emotional speech scenarios. Future work will focus on expanding the dataset to include more speakers, additional languages, and longer interactions.

4.2. System and Software Requirements

The system was tested on a MacBook Air with an Apple M2 chip (8-core CPU, 8-core integrated GPU), 16 GB unified memory, and running macOS Sequoia 15.3.2. Despite support for Metal Performance Shaders (MPS), hardware acceleration provided no significant speedup and showed partial incompatibility with PyTorch 2.1.2-based modules like pyannote.audio and speechbrain, leading to unstable behavior. All inference tasks—including transcription, speaker diarization, emotion classification, and feedback generation—were executed on CPU, delivering consistent real-time performance for audio clips up to 3 min without latency, thermal throttling, or memory issues, confirming its suitability for low-resource environments. The software requirements are given in Table 4.

Table 4. Software used in RAISE framework.

Package	Version	Purpose
Speech Recognition		
`openai-whisper`	517a94e (Git)	Speech transcription (Whisper base)
`ffmpeg-python`	0.2.0	Audio decoding and conversion
`librosa`	0.9.2	Audio preprocessing and resampling
`soundfile`	0.13.1	WAV I/O operations
Speaker Diarization		
`pyannote.audio`	3.1.1	End-to-end speaker diarization
`speechbrain`	0.5.16	ECAPA-TDNN embeddings for speaker clustering
`torchaudio`	2.1.2	Audio feature extraction for PyTorch
`scikit-learn`	1.6.1	Agglomerative clustering for the graph
Emotion Classification		
`transformers`	4.27.4	Hartmann's DistilRoBERTa for emotion prediction
`torch`	2.1.2	ML inference backend
`speechbrain`	0.5.16	Supplementary emotion modeling
NLP Feedback Engine		
`openai`	0.28.0	GPT-3.5 Turbo API for conversational feedback
`python-dotenv`	1.1.0	Secure environment variable management
Audio I/O and Signal Processing		
`pydub`	0.25.1	MP3 to WAV conversion
`resampy`	0.4.3	Resampling for compatibility
`audread`	3.0.1	Backend audio decoding abstraction
`scipy`	1.15.2	Signal filtering and numerical operations
Visualization and UI		
`streamlit`	1.44.1	Interactive web interface
`streamlit-webrtc`	0.46.2	Real-time audio streaming and input capture
`matplotlib`	3.0.1	Emotion timeline plotting
`plotly`	5.5.0	Dynamic bar charts for emotion summaries
Real-Time and System		
`aiortc`	1.1.0	WebRTC streaming infrastructure
`asyncio`	Built-in	Asynchronous server communication
`websockets`	10.5.0	Real-time signaling support
`watchdog`	3.0.0	File system event monitoring
Utilities and Support		
`numpy`	1.26.4	Core numerical processing
`pandas`	2.2.2	Tabular data manipulation
`joblib`	1.4.2	Caching and performance optimization
`tqdm`	4.62.3	Progress tracking in batch processing
`typing-extensions`	4.1.2	Type hinting support
`protobuf`	3.20.1	Model serialization and config validation

5. Results and Analysis

To test RAISE, we used an emotional conversation and ran it through the system. The implementation of the RAISE framework is shown in Figure 8. The steps were as follows: first, turn the audio into text with Whisper, then separate who said what using PyAnnote, find emotions with the Hartmann model, and, finally, create personalized feedback with GPT-3.5 Turbo. All the results were shown on a visual dashboard with transcripts, emotions, and coaching tips.

Across the ten sample files summarized in Tables 5 and 6, diarization accounted for the largest share of processing time and, in several cases, took longer than the audio duration.

These measurements indicate that the tested pipeline did not consistently process audio fast enough to support a real-time interaction. The results are limited to these samples and the reported hardware/software configuration; they do not establish performance for longer conversations or concurrent sessions. Future evaluation should report end-to-end latency, real-time factors, hardware configuration, and variability across repeated runs. The current experiments are exploratory and do not establish the overall reliability or effectiveness of RAISE.

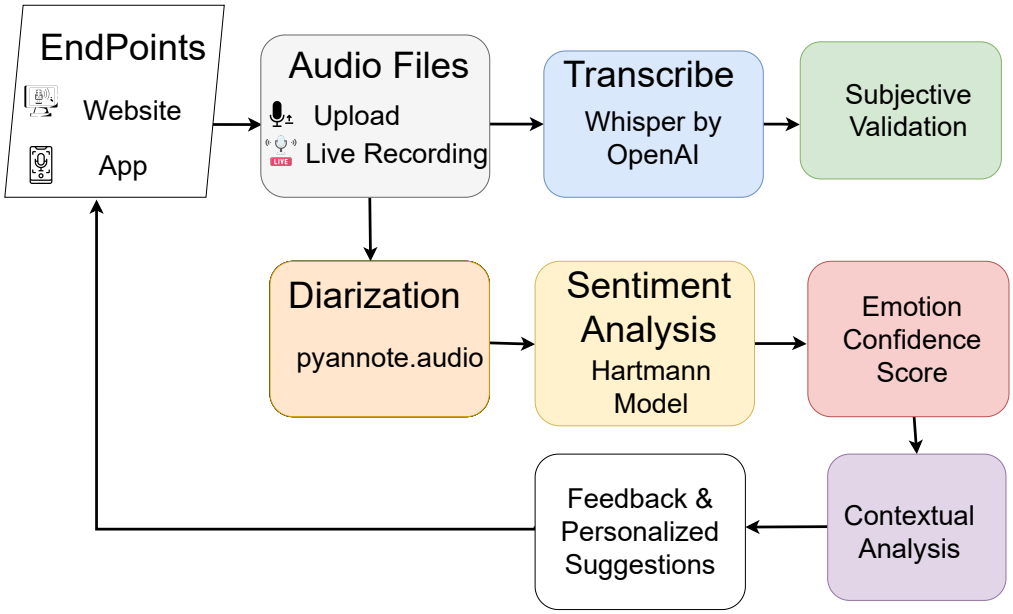


Figure 8. Implementation of RAISE framework.

Table 5. Latency metrics (in seconds) for selected audio samples.

Audio File Name	Audio Length	Transcription Time	Diarization Time	Emotion & Sentiment	GPT Feedback	Total Duration	Time Difference
Friends/input.wav	49	6.59	71.32	0.18	2.15	80.24	31.24
Indian Accent/input.wav	53	4.80	77.82	0.12	1.05	83.79	30.79
Kid/input.wav	60	7.10	93.04	0.14	1.37	101.65	41.65
Oppenheimer/input.wav	55	5.12	79.33	0.14	1.44	86.03	31.03
Weekend/input.wav	38	5.13	67.91	0.10	1.43	74.57	36.57
KFC/input.wav	50	4.00	51.30	0.06	1.45	56.80	19.71
Police Station/input.wav	43	4.69	58.59	0.11	1.98	66.42	23.72
UPSC Interview/input.wav	76	10.57	116.71	0.08	1.62	131.14	54.59
Sales Pitch/input.wav	77	9.13	120.25	0.16	1.91	134.20	56.52
Car Sales/input.wav	56	6.34	84.07	0.12	4.74	96.97	40.24

Another noticeable trend is the variation in total processing time, even for audio clips of similar lengths. For example, both of the original audio files UPSC_Interview and Sales_Pitch are around 76–77 s long, but their total durations go beyond 130 s due to long diarization times. In contrast, audio files like KFC and Police_Station were processed much faster, suggesting that simpler or clearer recordings lead to better performance. In addition, most of the emotions and sentiment analysis steps take very little time, which shows that this part is well optimized. In general, audio clarity, the number of speakers, and content complexity can all impact processing time.

Table 6. RAISE evaluation results for sample audio files.

Audio	Audio Time	Diarization	Emotion	Sentiment Analysis	Personal Feedback
`Friends.mp3`	49 s	[0:00:00–0:00:02] S1, [0:00:02–0:00:03] S1	Neutral, Surprise, Anger, Disgust	S1: 0.84 < surprise < 0.87, 0.88 < neutral < 0.91, 0.83 < disgust < 0.88	Overall, you did a great job keeping the conversation flowing. You showed interest in the other person's thoughts...
`Indian Accent.wav`	53 s	[0:00:00–0:00:02] S1, [0:00:02–0:00:03] S1	Joy, Neutral, Fear	S1: 0.84 < neutral < 0.86, 0.89 < joy < 0.92, 0.87 < fear < 0.89	You are engaging in the conversation very well. Your responses are clear and thoughtful.
`kid.mp3`	60 s	[0:00:00–0:00:01] S2, [0:00:01–0:00:02] S2	Surprise, Neutral, Sadness, Anger, Joy	S1: 0.86 < joy < 0.91, 0.87 < sadness < 0.88	Good job engaging with the child. Keep the conversation light. Be aware of context...
`Oppenheimer.wav`	55 s	[0:00:01–0:00:02] S1, [0:00:03–0:00:04] S1	Surprise, Neutral, Anger, Disgust, Fear	S1: 0.84 < anger < 0.86, 0.87 < fear < 0.87, 0.86 < surprise < 0.89	The conversation felt emotionally intense. Try to manage tone and avoid sounding too aggressive.
`trip_plan.mp3`	48 s	[0:00:01–0:00:04] S1, [0:00:05–0:00:06] S2	Neutral, Joy, Surprise, Anger	S1: 0.90 < joy < 0.91, 0.87 < surprise < 0.88, 0.86 < anger < 0.89	Flows well and the suggestion was timely. Try clarifying points when making plans.
`Reservation.wav`	38 s	[0:00:00–0:00:02] S1	Neutral, Joy, Surprise, Disgust	S1: 0.91 < disgust < 0.91, 0.90 < joy < 0.91, 0.90 < surprise < 0.91	Confirmed availability and plans. Consider being more open when discussing changes.
`UPSC_Interview.wav`	77 s	[0:00:00–0:00:02] S1, ..., [0:00:14–0:00:15] S1	Neutral, Disgust, Fear, Joy	S1: 0.87 < disgust < 0.92, fear < 0.92, joy < 0.91	Prioritizing basic needs is noble. Consider balancing short and long-term goals.
`Share_Sales.wav`	76 s	[0:00:00–0:00:02] S1, ..., [0:00:22–0:00:26] S1	All emotions	S1: surprise, joy, sadness, disgust; S2: anger, fear	Started strong. Later diverged due to unclear focus. Try maintaining topic clarity.
`Car_Sales.wav`	56 s	[0:00:00–0:00:04] S1, [0:00:34–0:00:35] S4	Neutral	S1: 0.86 < neutral < 2.16, S2: 1.2 < neutral < 1.88, S4: 2.5 < neutral < 3.38	Solid negotiation. Try being more clear on payment and pricing.
`Police_Station.wav`	42 s	[0:00:00–0:00:02] S1, [0:00:25–0:00:26] S1	Neutral, Fear, Disgust, Surprise, Sadness, Joy	S1: 0.84 < neutral < 1.94, 0.86 < disgust < 0.92, 0.87 < fear < 1.2, 0.90 < surprise < 1.05; S2: 1.0 < surprise < 1.67	Good effort. Focus on structuring your points. Avoid jumping between unrelated ideas.

Table 6 provides illustrative outputs generated by the prototype for selected clips. These examples show the kinds of text the system can produce; they are not a systematic evaluation of feedback quality or user benefit.

6. Limitations

This study has several limitations. First, the evaluation uses 53 clips, with detailed results reported for a subset of ten files. The sample is too small and heterogeneous to support broad claims about accuracy or generalizability, and the current description does not establish a standardized independent validation protocol. Second, speaker diarization is the main processing bottleneck in the reported timing results; the tested pipeline therefore cannot yet be assumed to provide timely feedback during live conversations. Performance on 10–15 min conversations and concurrent sessions has not been established. Third, the emotion classifier operates on transcribed text and predicts one of seven labels, so it does not directly analyze vocal prosody and may miss mixed, nuanced, culturally dependent, or context-specific expressions. Fourth, GPT-generated feedback can be generic, inconsistent, or incorrect and should not be treated as a verified assessment of intent or emotional state. Finally, conversations may contain sensitive personal information. Any real-world use would require clear consent, data minimization, secure storage and transmission, defined retention/deletion practices, access controls, and an assessment of the risks of using inferred emotion or demographic information. These limitations mean that RAISE should be regarded as a preliminary methodological framework and proof-of-concept prototype, not a deployment-ready system.

RAISE offers an integrated approach to speech transcription, speaker diarization, emotion analysis, and AI-generated coaching feedback. However, its deployment raises ethical concerns regarding conversation privacy, data security, potential bias across accents and languages, and the reliability of emotion analysis and coaching suggestions. Human oversight, informed consent, transparency, and secure data handling are essential to address these risks. Furthermore, the evaluation of 53 audio clips and the observed processing latency do not establish broad generalizability or real-time performance. Therefore, RAISE should be considered a preliminary research framework until larger-scale validation, fairness testing, and appropriate governance measures are completed.

7. Conclusions

RAISE is a preliminary methodological framework that brings together speech transcription, speaker diarization, text-based emotion classification, and GPT-generated contextual feedback in a single workflow. The prototype was examined using 53 audio clips, with timing and output examples reported for a subset of ten files/Test_Samples directory (https://github.com/bandiajay/RAISE (accessed on 22 October 2026)). These experiments demonstrate the feasibility of connecting the selected components, but they do not establish generalizable accuracy or reliable, real-time performance. In the tested samples, diarization was the main source of processing latency. The emotion model is limited to seven text-based categories, and generated feedback may be incomplete or incorrect. The framework may warrant further study for agent-assistance and organizational-coaching scenarios, provided that feedback is reviewed by a human and privacy safeguards are in place. Educational or healthcare uses would require additional domain-specific validation and oversight. Future work should evaluate the modules against independently labeled data, test longer conversations and concurrent users, measure latency under clearly specified conditions, assess feedback quality using a documented protocol, and address consent, security, data retention, and privacy-preserving processing. RAISE should therefore be understood as a proof of concept that motivates further evaluation rather than a validated real-time system.

Author Contributions: A.B. contributed to the conceptualization, formal analysis, methodology, project administration, supervision, visualization, and funding acquisition (70%), and was responsible for writing the original draft and reviewing and editing the manuscript. B.S.P., R.R.P., and A.T.

contributed to data curation, investigation, software development, visualization, and writing the original draft. J.G.G. contributed to conceptualization, formal analysis, and validation of the research. P.B.R. contributed to funding acquisition (30%) and writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The audio dataset used to evaluate the RAISE framework, consisting of 53 audio clips, is publicly available in the Test_Samples directory of the project's GitHub repository (https://github.com/bandiajay/RAISE (accessed on 22 October 2026)). The dataset includes variations in emotional expressions, number of speakers, and recording conditions.

Acknowledgments: During the preparation of this manuscript, the authors used OpenAI ChatGPT (GPT-6.1 Sol) to generate Figures 1 and 2 based on prompts and instructions provided by the authors. The authors reviewed and edited the generated figures and take full responsibility for the accuracy, originality, and content of this publication.

Conflicts of Interest: Author Popin Bose Roy was employed by Zillow Group. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. IBM. What Is AI Analytics? Available online: https://www.ibm.com/think/topics/ai-analytics (accessed on 21 April 2025).
2. Adiwardana, D.; Luong, M.T.; So, D.R.; Hall, J.; Fiedel, N.; Thoppilan, R.; Yang, Z.; Kulshreshtha, A.; Nemade, G.; Lu, Y.; et al. Towards a human-like open-domain chatbot. *arXiv* **2020**, arXiv:2001.09977.
3. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
4. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*; ACL: Stroudsburg, PA, USA, 2019; pp. 4171–4186.
5. Camilleri, M.A.; Troise, C. Live support by chatbots with artificial intelligence: A future research agenda. *Serv. Bus.* **2023**, *17*, 61–80. [CrossRef]
6. Cui, L.; Huang, S.; Wei, F.; Tan, C.; Duan, C.; Zhou, M. Superagent: A customer service chatbot for e-commerce websites. In *Proceedings of ACL 2017, System Demonstrations*; ACL: Stroudsburg, PA, USA, 2017; pp. 97–102.
7. Stoilova, E. AI chatbots as a customer service and support tool. *ROBONOMICS J. Autom. Econ.* **2021**, *2*, 21. [CrossRef]
8. Leocádio, D.; Guedes, L.; Oliveira, J.; Reis, J.; Melão, N. Customer service with AI-powered human-robot collaboration (HRC): A literature review. *Procedia Comput. Sci.* **2024**, *232*, 1222–1232. [CrossRef]
9. Fitzpatrick, K.K.; Darcy, A.; Vierhile, M. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (Woebot): A randomized controlled trial. *JMIR Ment. Health* **2017**, *4*, e7785. [CrossRef] [PubMed]
10. Hakim, F.Z.M.; Indrayani, L.M.; Amalia, R.M. A dialogic analysis of compliment strategies employed by replika chatbot. In *Proceedings of the Third International Conference of Arts, Language and Culture (ICALC 2018)*; Atlantis Press: Dordrecht, The Netherlands, 2019; pp. 266–271.
11. Hassib, M.; Schneegass, S.; Eiglsperger, P.; Henze, N.; Schmidt, A.; Alt, F. EngageMeter: A System for Implicit Audience Engagement Sensing Using Electroencephalography. In *CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*; ACM: New York, NY, USA, 2017; pp. 5114–5119. [CrossRef]
12. Bandi, A.; Adapa, P.V.S.R.; Kuchi, Y.E.V.P.K. The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges. *Future Internet* **2023**, *15*, 260. [CrossRef]
13. Shahin, M.; Chen, F.F.; Hosseinzadeh, A. Harnessing customized AI to create voice of customer via GPT3. 5. *Adv. Eng. Inform.* **2024**, *61*, 102462. [CrossRef]
14. Bandi, A.; Zeng, R. Evaluation of the Effectiveness of Prompts and Generative AI Responses. In *Proceedings of the Computer Applications in Industry and Engineering*; Hu, G., Kambhampaty, K.K., Roy, I., Eds.; Springer: Cham, Switzerland, 2025; pp. 56–69. [CrossRef]
15. Henricus, M. Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled. Ph.D. Thesis, University of Twente, Enschede, The Netherlands, 2008.
16. Krugmann, J.O.; Hartmann, J. Sentiment analysis in the age of generative AI. *Cust. Needs Solut.* **2024**, *11*, 3. [CrossRef]

17. Bandi, A.; Babu, J.; Zeng, R.; Muthyala, S.R. Enhancing Generative AI Chatbot Accuracy Using Knowledge Graph. In *Proceedings of the Software and Data Engineering*; Feng, W., Rahimi, N., Margapuri, V., Eds.; Springer: Cham, Switzerland, 2025; pp. 157–167. [CrossRef] [PubMed]
18. Moattar, M.H.; Homayounpour, M.M. A review on speaker diarization systems and approaches. *Speech Commun.* **2012**, *54*, 1065–1103. [CrossRef]
19. Bandi, A.; Kagitha, H. A Case Study on the Generative AI Project Life Cycle Using Large Language Models. In *Proceedings of the 39th International Conference on Computers and Their Applications*; Bandi, A., Hossain, M., Jin, Y., Eds.; EPiC Series in Computing; EasyChair: Stockport, UK, 2024; Volume 98, pp. 189–199. [CrossRef]
20. Radford, A.; Kim, J.W.; Xu, T.; Brockman, G.; Mcleavey, C.; Sutskever, I. Robust Speech Recognition via Large-Scale Weak Supervision. In *Proceedings of the 40th International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023*; Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J., Eds.; Proceedings of Machine Learning Research (PMLR); JMLR: New York, NY, USA, 2023; Volume 202, pp. 28492–28518.
21. Allen, J.; Rabiner, L. A unified approach to short-time Fourier analysis and synthesis. *Proc. IEEE* **1977**, *65*, 1558–1564. [CrossRef]
22. Davis, S.; Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. Acoust. Speech Signal Process.* **1980**, *28*, 357–366. [CrossRef]
23. Kubichek, R. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of the IEEE Pacific Rim Conference on Communications Computers and Signal Processing*; IEEE: Piscataway, NJ, USA, 1993; Volume 1, pp. 125–128. [CrossRef]
24. Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P.; et al. The Kaldi speech recognition toolkit. In *Proceedings of the IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*; IEEE Signal Processing Society: Piscataway, NJ, USA, 2011.
25. Bredin, H.; Yin, R.; Coria, J.M.; Gelly, G.; Korshunov, P.; Lavechin, M.; Fustes, D.; Titeux, H.; Bouaziz, W.; Gill, M.P. Pyannote.Audio: Neural Building Blocks for Speaker Diarization. In *Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*; IEEE: Piscataway, NJ, USA, 2020; pp. 7124–7128. [CrossRef]
26. Bredin, H.; Plaquet, A. `pyannote/speaker-diarization-3.1` Model Card. Benchmark Table. Available online: https://huggingface.co/pyannote/speaker-diarization-3.1 (accessed on 22 October 2026).
27. Desplanques, B.; Thienpondt, J.; Demuynck, K. Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. *arXiv* **2020**, arXiv:2005.07143.
28. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
29. Hartmann, J. Emotion English DistilRoBERTa-Base. 2022. Available online: https://huggingface.co/j-hartmann/emotion-english-distilroberta-base/ (accessed on 20 April 2025).
30. Ekman, P. An argument for basic emotions. *Cogn. Emot.* **1992**, *6*, 169–200. [CrossRef]
31. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
32. Bishop, C.M.; Nasrabadi, N.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006; Volume 4.
33. OpenAI. GPT-3.5 Technical Report, 2023. Available online: https://openai.com/research/gpt-4 (accessed on 22 October 2026).
34. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
35. Amin, M.M.; Mao, R.; Cambria, E.; Schuller, B.W. A wide evaluation of ChatGPT on affective computing tasks. *IEEE Trans. Affect. Comput.* **2024**, *15*, 2204–2212. [CrossRef]
36. Chen, Y.; Xiao, Y. Recent advancement of emotion cognition in large language models. *arXiv* **2024**, arXiv:2409.13354.
37. Bălan, D.A.; Ordelman, R.J.; van den Heuvel, H. A Systematic Benchmark for Dutch Automatic Speech Recognition. In Proceedings of the CLARIAH Annual Conference 2024, Leiden, The Netherlands, 13 June 2024.
38. Bredin, H. pyannote.audio 2.1 speaker diarization pipeline: Principle, benchmark, and recipe. In *Proceedings of the 24th INTERSPEECH Conference (INTERSPEECH 2023)*; ISCA: Baixas, France, 2023; pp. 1983–1987.
39. Bandi, A.; Hurtado, J.A. Big data streaming architecture for edge computing using Kafka and Rockset. In *Proceedings of the 2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*; IEEE: Piscataway, NJ, USA, 2021; pp. 323–329. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.