# Comprehensive Analysis of Fine-Tunable ASR and Speaker Diarization Architectures: State-of-the-Art Evaluation, DL Sprint 4.0 Benchmarks, and Future Trajectories

## 1. Executive Summary

The domain of Automatic Speech Recognition (ASR) and Speaker Diarization is currently undergoing a structural transformation, shifting from distinct, modular pipelines toward integrated, multimodal, and large-scale foundation models. This report provides an exhaustive, expert-level analysis of the current State-of-the-Art (SOTA) in fine-tunable speech models, explicitly tailored to the rigorous constraints and scoring mechanisms of the **DL Sprint 4.0 Bengali Speaker Diarization Challenge**.

This document synthesizes data from over 180 research artifacts, technical reports, and benchmark repositories to construct a definitive guide for researchers and competitors. The analysis juxtaposes established giants like OpenAI's **Whisper** and **Pyannote.audio** against disruptive newcomers such as NVIDIA's **Canary** and **Parakeet** families, Alibaba's **SenseVoice** and **Qwen-Audio**, the edge-optimized **Moonshine** architecture, and specialized systems like **FireRedASR** and **SE-DiCoW**.

A central pillar of this report is the deep dissection of the DL Sprint 4.0 competition framework, specifically **Rule 7.2 (Model Performance Breakdown)** and **Rule 8.2 (Inference Time Scoring)**. Unlike conventional academic benchmarks that prioritize accuracy (Word Error Rate or WER) above all else, this competition introduces a percentile-based Real-Time Factor (RTF) scoring system. This mechanism creates a volatile competitive environment where heavy, high-accuracy Large Language Models (LLMs) risk obsolescence against lightweight, streaming-capable architectures despite superior transcription capabilities.

The report is structured to guide the reader through the theoretical underpinnings of the competition rules, the architectural nuances of every eligible fine-tunable model, and the practical methodologies for adapting these systems to the Bengali language. By evaluating models not just on their public leaderboard performance but on their theoretical resilience to the "Private Leaderboard" and "Hidden Test Set" phases, this analysis offers a strategic roadmap for navigating the trade-offs between computational throughput and linguistic precision.

# 2. The DL Sprint 4.0 Competition Framework: A Theoretical Analysis

To engineer an optimal solution for the DL Sprint 4.0 challenge, one must first deconstruct the mathematical and structural incentives created by its rulebook. The competition is not merely a test of ASR accuracy; it is a multi-objective optimization problem involving transcription fidelity, speaker segmentation accuracy, and inference latency.

## 2.1 Structural Decomposition and Phase Scoring

The competition is bifurcated into two independent problem statements, yet the scoring mechanism forces a holistic approach.

1. **Problem 1 (ASR):** The transcription of long-form Bengali audio into text.[1]
2. **Problem 2 (Diarization):** The temporal segmentation of audio by speaker identity ("Who spoke when?").[1]

**Rule 7.2 (Model Performance Breakdown)** dictates that the scores for each phase—Public, Private, and Hidden—are calculated as the arithmetic mean of both tasks:

$$\text{Phase Score} = \frac{\text{Score}_{\text{ASR}} + \text{Score}_{\text{Diarization}}}{2}$$

.[1]

This arithmetic mean implies a linear coupling between the two tasks. A catastrophic failure in one domain (e.g., a Diarization Error Rate of 50%) cannot be compensated for by perfection in the other (e.g., a Word Error Rate of 0%). This necessitates a balanced system design. Participants must decide between two architectural paradigms:

- **Pipeline Approaches:** Where a specialized Diarization model (e.g., Pyannote) feeds segment timestamps to an ASR model (e.g., Whisper). This allows for selecting the "best-in-class" model for each sub-task but introduces latency overhead due to sequential processing.
- **Joint/Integrated Approaches:** Where a single model performs both tasks (e.g., Qwen-Audio, SE-DiCoW). These architectures often offer superior semantic consistency and lower total latency but are harder to fine-tune and may suffer from task interference.

## 2.2 The "Online" vs. "Offline" Matrix

The total competition score is a weighted composite:

$$\text{Total Score} = 0.70 \times \text{Online Evaluation} + 0.30 \times \text{Offline Evaluation}$$

The **Online Evaluation** is further decomposed, revealing the critical weight of computational efficiency:

$$\text{Online Score} = 0.30 \times \text{Inference Time Score} + 0.70 \times \text{Model Performanc}$$

.[1]

The **Model Performance Score** itself is heavily skewed toward unseen data to prevent leaderboard probing:

- **20%** Public Leaderboard.
- **30%** Private Leaderboard.
- **50%** Hidden Test Set Score.

This distribution serves as a rigorous regularizer. A model that is overfitted to the public test set will likely fail in the Private and Hidden phases, which constitute 80% of the Model Performance Score. This structural weighting prioritizes architectures with strong **Zero-Shot** or **Few-Shot** generalization capabilities—typically those pre-trained on massive, diverse multilingual corpora like **XLS-R** (300M/1B/2B parameters) or **NVIDIA Canary**.[2]

## 2.3 Mathematical Analysis of Inference Time Scoring (Rule 8.2)

The most disruptive element of the DL Sprint 4.0 rules is **Rule 8.2 Inference Time Scoring**. Unlike standard industry benchmarks that might set a hard latency threshold (e.g., "RTF must be < 0.5"), this competition uses a **relative percentile-based ranking system**.[2]

The scoring algorithm is defined as follows:

1. **RTF Calculation:** For every test case $i$, the Real-Time Factor (RTF) is computed:

$$RTF_i = \frac{\text{Processing Time}_i}{\text{Audio Duration}_i}$$

2. **Averaging:** The participant's final RTF is the mean across all $N$ test cases:

$$RTF_{\text{final}} = \frac{1}{N} \sum_{i=1}^{N} RTF_i$$

3. **Percentile Mapping:** Participants are ranked by $RTF_{\text{final}}$ in ascending order. The score is mapped directly from the percentile rank:
    - $1^{st}$ Percentile (Fastest) $\rightarrow$ Score 100
    - $2^{nd}$ Percentile $\rightarrow$ Score 99
    - ... $99^{th}$ Percentile (Slowest) $\rightarrow$ Score 1.

**Strategic Implications:**

This scoring mechanism creates a "race to the bottom" for latency. The absolute value of the RTF matters less than the relative standing against other competitors. If the median competitor uses a model with an RTF of 0.1, using a heavy LLM-based ASR (like **FireRedASR-LLM**, 8.3B parameters) that has an RTF of 0.5 could result in an Inference Time Score near zero, effectively capping the total Online Score.

- **The "Heavy Model" Trap:** A team employing a high-accuracy, 8-billion parameter model might achieve a near-perfect ASR score (e.g., 98/100). However, if their inference speed places them in the 90th percentile (slowest), their Inference Time Score might be 10.
    - Total Online Contribution: $(0.7 \times 98) + (0.3 \times 10) = 68.6 + 3 = 71.6$.
- **The "Lightweight" Advantage:** A team utilizing a hyper-optimized, smaller model (e.g., **Moonshine Tiny** or **Parakeet-TDT**) might sacrifice some accuracy (e.g., ASR score 90/100) but achieve the 1st percentile in speed (Score 100).
    - Total Online Contribution: $(0.7 \times 90) + (0.3 \times 100) = 63 + 30 = 93$.

This analysis demonstrates that under Rule 8.2, **speed is not just a tie-breaker; it is a primary competitive vector**. Models capable of sub-real-time processing (RTF << 1.0) are essential.

## 2.4 Diarization Error Rate (DER) Formulation

For Problem 2, the scoring metric is the standard Diarization Error Rate (DER):

$$DER = \frac{\text{False Alarm} + \text{Missed Speech} + \text{Speaker Confusion}}{\text{Total Reference Speech}}$$

The final score is derived as:

$$\text{Score} = 100 \times (1 - DER)$$

(Clipped to the range 0–100).[3]

While this formula weights all errors equally, in practical long-form audio scenarios, the error components are not distributed uniformly.

- **False Alarm:** Often caused by background noise or music being misclassified as speech. This highlights the need for robust Voice Activity Detection (VAD).
- **Missed Speech:** Often caused by overly aggressive VAD or quiet speech.
- **Speaker Confusion:** This is the most challenging component in multi-speaker environments and typically dominates the error budget. Models with advanced clustering (e.g., VBx) or End-to-End neural diarization (EEND) architectures are required to minimize this term, specifically in handling overlapping speech.[5]

---

# 3. Comprehensive Analysis of Fine-Tunable ASR Architectures

The period from 2024 to early 2026 has witnessed a proliferation of open-source ASR architectures. The landscape has bifurcated into two distinct streams: **Massive Multimodal LLMs** (Speech-LLMs) that prioritize semantic understanding and contextual reasoning, and **Efficient Transducers** optimized for extreme throughput and edge deployment.

## 3.1 The Whisper Ecosystem: Strengths, Weaknesses, and Hallucinations

OpenAI's **Whisper** series serves as the ubiquitous baseline for modern ASR, but its deployment in a high-stakes competition like DL Sprint 4.0 requires careful mitigation of its known pathologies.

### 3.1.1 Whisper Large v3 and v3-Turbo

- **Whisper Large v3:** Released in late 2024, this model (1.55B parameters) remains a top performer on open leaderboards for multilingual transcription.[7] It is a Transformer-based encoder-decoder trained on 5 million hours of weakly labeled audio. Its primary strength is its robustness to diverse accents and recording qualities, achieving a Word Error Rate (WER) of **6.43%** on long-form benchmarks.[8]
- **Whisper Large v3 Turbo:** This variant (809M parameters) reduces the number of encoder layers from 32 to 4, leveraging the fact that the decoder performs the heavy semantic lifting. It offers a significant speedup, with an inverse Real-Time Factor (RTFx) of ~216, compared to ~68 for the standard Large v3.[8]

### 3.1.2 The Hallucination Problem in Fine-Tuning

Fine-tuning Whisper, particularly the Large-v3 variant, is fraught with instability. Several

independent researchers and community discussions identify a critical issue: after approximately 400 training steps, the model begins to suffer from severe hallucinations.[10] These manifest as:

- **Loop Failures:** Repeating a single phrase or n-gram indefinitely.
- **Semantic Drift:** Generating plausible but acoustically unsupported text.
- **Silence Failure:** Transcribing silence as hallucinatory text.

This instability is attributed to the "weak supervision" nature of its pre-training data. Unlike strictly supervised models (e.g., Canary), Whisper learns to predict text that *might* appear in a transcript rather than strictly what is spoken.

**Mitigation Strategies:**

- **LoRA (Low-Rank Adaptation):** Fine-tuning the full weights is discouraged. Using LoRA with a rank ($r$) of 32 or 64 on the query and value projections (q_proj, v_proj) is the standard stability fix.
- **DoRA (Weight-Decomposed Low-Rank Adaptation):** A more advanced technique that decomposes the weights into magnitude and direction components, allowing for more stable updates.[11]
- **Data Curation:** Filtering the training data to remove segments with high non-speech ratios is essential to prevent the model from learning to "guess" during silence.

## 3.2 NVIDIA NeMo Family: The Controllable Alternatives

NVIDIA's NeMo framework offers a suite of models that provide arguably better control and efficiency than Whisper, making them strong contenders for the DL Sprint.

### 3.2.1 Canary: The Multilingual Expert

**Canary** (available in 1B and 2.5B parameter versions) is an encoder-decoder model that replaces the standard Transformer encoder with a **FastConformer**.[12]

- **Architecture:** FastConformer improves efficiency by downsampling the audio input more aggressively in the initial layers, effectively reducing the sequence length seen by the attention mechanisms. The decoder is a Transformer LLM (specifically leveraging the Qwen architecture in the **Canary-Qwen-2.5B** variant).
- **Performance:** Canary-Qwen-2.5B has achieved SOTA results on the Open ASR Leaderboard (WER 5.63%), outperforming Whisper Large v3.[8]
- **Controllability:** Unlike Whisper, which relies on prompt engineering tokens, Canary uses explicit control tokens (<source_lang>, <target_lang>, <task>, <pnc>) to toggle punctuation and capitalization. This deterministic control is advantageous for competition rules that may specify strict formatting requirements.

### 3.2.2 Parakeet: The Throughput Champion

For the **Inference Time Score (Rule 8.2)**, the **Parakeet** family is the most strategic choice.

- **Parakeet-CTC (1.1B):** Uses a Connectionist Temporal Classification decoder. It is extremely fast because it is non-autoregressive; it predicts all tokens in parallel (conceptually) rather than one by one.
- **Parakeet-TDT (0.6B):** This model utilizes the **Token-and-Duration Transducer (TDT)** architecture.
    - **Mechanism:** Standard RNN-Transducers (RNN-T) process every audio frame, which is computationally redundant for silence or long vowels. TDT predicts both the token *and* its duration (how many frames it lasts). This allows the model to "skip" forward in the audio sequence, processing only the informative frames.
    - **Implication:** This results in an **RTFx of 3386** (processing 1 hour of audio in ~1 second on an H100 GPU).[8] In the context of the DL Sprint, deploying Parakeet-TDT could practically guarantee a score of 100/100 for the Inference Time component.
    - **Accuracy:** Despite its speed, it maintains a competitive WER of ~6.05%, only marginally worse than the much larger Canary or Whisper models.[8]

## 3.3 Alibaba FunAudioLLM: SenseVoice and Qwen-Audio

Alibaba has released a suite of models that blur the line between ASR and audio understanding.

### 3.3.1 SenseVoice: Low Latency and Rich Transcription

**SenseVoice-Small** is designed to challenge Whisper's dominance in the low-latency regime.

- **Architecture:** It employs a **non-autoregressive end-to-end framework**. Unlike autoregressive models (Whisper, Canary) that generate token $t$ conditional on token $t-1$, SenseVoice generates the entire sequence simultaneously.
- **Performance:** It achieves inference latencies as low as **70ms for 10 seconds of audio**, which is approximately 15x faster than Whisper Large.[14]
- **Rich Capabilities:** SenseVoice natively supports **Speech Emotion Recognition (SER)** and **Audio Event Detection (AED)**. It can detect events like "applause," "laughter," and "music".[16] While the DL Sprint focuses on text, the ability to robustly handle (and ignore) non-speech acoustic events is a significant advantage for long-form audio diarization and transcription.
- **Bengali Support:** Trained on 400,000 hours of data covering 50+ languages, it includes substantial support for Indic languages, making it a viable candidate for fine-tuning.[14]

### 3.3.2 Qwen-Audio and Qwen3-ASR

**Qwen3-ASR** (1.7B parameters) represents the "Speech-LLM" paradigm.

- **Mechanism:** It treats audio features as just another modality of tokens input into a Large Language Model. This allows it to leverage the massive world knowledge of the Qwen

LLM for semantic correction.
- **Pros/Cons:** While it offers superior contextual reasoning (e.g., distinguishing homophones based on sentence context), its autoregressive LLM nature makes it significantly slower than Parakeet or SenseVoice. It risks a low Inference Time Score unless batched extremely aggressively.[17]

## 3.4 Moonshine: The Edge Computing Contender

**Moonshine** (Tiny 27M, Base 62M) challenges the "bigger is better" orthodoxy.

- **Variable Compute Architecture:** A major inefficiency in Whisper is that it pads all audio segments to 30 seconds. If a user says "Hello," Whisper processes 30 seconds of data (mostly silence). Moonshine uses a variable-length processing scheme, meaning compute scales linearly with actual audio duration.
- **Speed:** This architecture results in **5x-15x faster inference** than Whisper for typical conversational segments.[19]
- **Performance:** Moonshine Tiny (27M) outperforms Whisper Tiny (39M) by ~48% in WER across tested languages.[19]
- **Strategic Utility:** In the DL Sprint, Moonshine could be deployed in a **Tiered Architecture**. A participant could use Moonshine to rapidly transcribe clear, high-confidence segments to maximize the RTF score, and fallback to a heavier model (Canary/Whisper) only for low-confidence or noisy segments.

## 3.5 FireRedASR: The Industrial Heavyweight

Released in early 2025, **FireRedASR** targets industrial-grade accuracy.

- **FireRedASR-LLM (8.3B):** An Encoder-Adapter-LLM framework (utilizing Qwen2-7B-Instruct). It achieves SOTA on Mandarin benchmarks (CER 3.05%).[22]
- **FireRedASR-AED (1.1B):** An attention-based encoder-decoder model.
- **Analysis:** The 8.3B parameter count of the LLM variant makes it a risky choice for Rule 8.2. The computational cost of running an 8B model is roughly 8x higher than a 1B model, potentially relegating the participant to the bottom percentile of speed. Unless the accuracy gain is transformative (e.g., reducing WER from 10% to 2%), the math of the competition rules likely disfavors this model.

## 3.6 The Foundation: Wav2Vec 2.0, XLS-R, and MMS

For low-resource languages like Bengali, "training from scratch" is rarely viable. The foundation of most winning solutions lies in self-supervised models.

- **XLS-R (300M, 1B, 2B):** Meta's robust cross-lingual model trained on 436,000 hours of speech in 128 languages.[4]
- **MMS (Massive Multilingual Speech):** An evolution of Wav2Vec 2.0, supporting 1,100+ languages.
- **Fine-Tuning Advantage:** These models learn rigorous acoustic representations

(phonemes, tones) during pre-training. Fine-tuning them involves adding a simple linear projection (CTC head) on top. This is computationally cheap and requires far less labeled data to reach convergence compared to encoder-decoder models. For the DL Sprint's **Private Leaderboard** (where generalization is key), a fine-tuned **MMS-1B** or **XLS-R-1B** often provides the best balance of stability and performance.[23]

---

# 4. Comprehensive Analysis of Speaker Diarization Architectures

Problem 2 of the DL Sprint requires solving "Who spoke when?". The evolution of diarization has moved from clustering embeddings to End-to-End neural prediction.

## 4.1 Pyannote.audio: The Gold Standard Pipeline

**Pyannote** is the default choice for most participants, but selecting the correct version is critical.

- **Pipeline Structure:**
  1. **SincNet/TDNN:** Feature extraction.
  2. **PyanNet (LSTM):** Segmentation and embedding extraction.
  3. **Clustering:** Grouping embeddings into speakers.
- **Pyannote 3.1 vs. Community-1:** The **Community-1** model (released late 2025) represents a significant upgrade. It replaces the traditional agglomerative hierarchical clustering with **VBx (Variational Bayes)** clustering in supported configurations, or improved neural segmentation heads.
- **Benchmark:** On the **AMI-SDM** dataset (single distant microphone, highly relevant for noisy long-form audio), Community-1 achieves a DER of **19.9%**, a substantial improvement over the 22.7% of the legacy 3.1 model.[25]

## 4.2 End-to-End Neural Diarization (EEND): DiariZen and Sortformer

Traditional clustering pipelines struggle with **Overlapping Speech**. If two people speak at once, a clustering algorithm (which typically assigns one label per frame) fails. EEND models solve this by treating diarization as a multi-label classification problem.

- **DiariZen:** An open-source EEND model that excels in high-overlap scenarios. Benchmarks show it achieving a DER of **14.0%** on AMI-SDM, significantly outperforming Pyannote's 19.9%.[6] This suggests that for complex acoustic environments, DiariZen is the superior choice.
- **NVIDIA Sortformer:** A Transformer-based EEND model. It is designed for streaming and handles overlap efficiently. Its "Sortformer" mechanism allows it to output speaker labels without permutation ambiguity, making it highly effective for the "Online Evaluation" where latency matters.[27]

### 4.3 Target-Speaker ASR (TS-ASR): The Unified Solution

**SE-DiCoW (Self-Enrolled Diarization-Conditioned Whisper)** represents a cutting-edge approach that could theoretically solve both competition problems simultaneously.

- **Mechanism:** It conditions the Whisper ASR model on a speaker embedding. Effectively, you ask the model: "Transcribe what *this specific person* is saying."
- **Workflow:**
  1. Extract speaker embeddings (Diarization).
  2. Feed embedding + Audio to SE-DiCoW.
  3. Output: Transcript for that speaker.
- **Advantage:** This ensures perfect alignment between the speaker ID and the text, eliminating the "Speaker Confusion" error in the DER metric. However, it requires running the ASR inference $N$ times for $N$ speakers, which could disastrously impact the RTF score.[29]

---

## 5. Benchmarking and Scoring Simulation

To assist in model selection, we simulate the performance of key architectures under the DL Sprint rules.

### 5.1 ASR Model Comparison Matrix

| Model Family | Variant | Parameters | Architecture | Est. WER (Bengali) | Est. RTFx (GPU) | Inference Score Potential |
|---|---|---|---|---|---|---|
| **Parakeet** | TDT | 0.6B | Transducer | Good (~7-8%) | **3386** | **99-100** |
| **SenseVoice** | Small | N/A | Non-Autoregressive | Good | High (~1500) | 90-95 |
| **Moonshine** | Tiny | 27M | Enc-Dec | Moderate (~10-12%) | High (CPU) | 85-95 |
| **Whisper** | v3 Turbo | 809M | Transformer | Very Good | 216 | 60-75 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | (~6%) | | |
| **Canary** | Qwen-2.5B | 2.5B | FastConformer | **Excellent (~5%)** | 418 | 70-80 |
| **Whisper** | Large v3 | 1.55B | Transformer | Excellent (~5.5%) | 68 | 30-40 |
| **FireRedASR** | LLM-L | 8.3B | Adapter-LLM | **SOTA (~4.5%)** | < 20 | 1-10 |

**Simulation Analysis:**

- **The "Speed Demon" Strategy:** Using **Parakeet-TDT**. Even if WER is 2% worse than FireRedASR, the RTFx of 3386 guarantees a top percentile Inference Score. In a competition with 100 participants, the difference between Score 10 and Score 100 (90 points) in the inference component (weighted at 30%) translates to 27 total points. It is highly unlikely that FireRedASR's accuracy advantage can overcome a 27-point deficit.
- **The "Balanced" Strategy:** Using **Canary-Qwen-2.5B**. It offers SOTA accuracy and decent speed (RTFx 418). This might be the winning strategy if the field is crowded with slow Whisper implementations.

## 5.2 Diarization DER Benchmark (AMI-SDM Dataset)

| Model | Architecture | DER (%) | Speaker Confusion | False Alarm |
|---|---|---|---|---|
| **DiariZen-Large** | EEND | **14.0** | Low | Low |
| **Pyannote Comm-1** | Pipeline (VBx) | 19.9 | Moderate | Low |
| **Pyannote Legacy 3.1** | Pipeline (Agglom) | 22.4 | High | Low |
| **Sortformer** | EEND | Competitive | Low | Low |

**Analysis:**

The 5.9% absolute difference in DER between **DiariZen** and **Pyannote Community-1** is massive. In the competition scoring ($\text{Score} = 100 \times (1 - DER)$), this translates directly to points. Participants relying on the standard Pyannote pipeline will likely find themselves capped at a lower performance tier compared to those adopting EEND architectures like DiariZen or Sortformer.

---

# 6. Fine-Tuning Methodologies and "Recipes"

Success in DL Sprint 4.0 depends on effectively adapting these pre-trained models to the provided Bengali dataset.

## 6.1 Fine-Tuning Whisper: The LoRA Recipe

Directly fine-tuning Whisper weights often leads to "catastrophic forgetting" or the hallucination issues described in Section 3.1.2.

- **Methodology:** Use **PEFT (Parameter-Efficient Fine-Tuning)** with LoRA.
- **Configuration:**
  - Target Modules: ["q_proj", "v_proj"] (targeting query and value matrices in attention).
  - Rank ($r$): 32 or 64.
  - Alpha: 64 or 128.
  - Dropout: 0.05.
- **Data Prep:** Audio must be padded/trimmed to 30 seconds. Transcripts should be normalized (removing special characters, standardizing Bengali punctuation).
- **Stabilization:** If hallucinations persist, integrate **DoRA** or use **CT-2 (OpenNMT)** optimized inference backends post-training.[10]

## 6.2 Fine-Tuning NeMo (Canary/Parakeet)

NVIDIA NeMo requires a more structured approach but offers greater control.

- **Tokenizer:** Unlike Whisper's fixed multilingual tokenizer, NeMo models often use **SentencePiece**. For Bengali, training a dedicated SentencePiece tokenizer on the competition corpus (vocab size ~1024-4096) is crucial for optimal WER.
- **Script:** Use speech_to_text_finetune.py from the NeMo repository.
- **Optimization:** Enable **FusedAdam** optimizer and **bf16** (Brain Floating Point 16) precision to maximize training throughput on Ampere/Hopper GPUs.
- **Config Adjustment:** For **Parakeet-TDT**, ensure the duration_predictor head is also fine-tuned or properly initialized, otherwise the speed advantages may be lost due to poor duration estimation.[32]

### 6.3 Fine-Tuning Diarization (Hugging Face Diarizers)

The **Hugging Face Diarizers** library is the standard tool for adapting Pyannote.

- **Data Requirement:** It can yield significant improvements with as little as 10 hours of labeled diarization data.
- **Process:**
  1. **Segmentation Tuning:** Fine-tune the underlying pyannote/segmentation-3.0 model on the Bengali RTTM files. This helps the model distinguish Bengali speech from background noise more effectively.
  2. **Hyperparameter Optimization:** Run a sweep on the clustering thresholds (e.g., min_cluster_size, threshold) using the validation set to minimize DER.
  3. **Code Snippet:**

```
Python
from diarizers import SegmentationModel, DiarizationPipeline
# Load Pretrained Segmentation
model = SegmentationModel.from_pretrained("pyannote/segmentation-3.0")
# Fine-tune on Bengali dataset
trainer.train()
# Export for pipeline
```

[34]

---

# 7. Strategic Recommendations for DL Sprint 4.0

Based on the synthesis of competition rules, model capabilities, and benchmark data, the following strategic roadmap is proposed for participants.

## 7.1 The "Hybrid Ensemble" Architecture

To maximize the Total Online Score, participants should not rely on a single model.

1. **The "Scout" (VAD & Segmentation):** Use **Silero VAD v5** or **Moonshine Tiny** to rapidly segment the audio and discard silence. This improves the RTF denominator (processing time) by avoiding compute on empty segments.
2. **The "Speedster" (Primary ASR):** Process all segments with **NVIDIA Parakeet-TDT (0.6B)** fine-tuned on Bengali. This secures the high percentile Inference Time Score.
3. **The "Expert" (Fallback ASR):** For segments where Parakeet has low confidence (log-probability threshold), re-process with **Whisper Large v3 (LoRA)** or **Canary-Qwen**. This recovers accuracy on difficult segments without destroying the average RTF.
4. **The "Separator" (Diarization):** Use **DiariZen** for overlapping speech segments and

**Pyannote Community-1** for non-overlapping sections.

### 7.2 Handling the Private Leaderboard and Hidden Test Set

The heavy weighting (80%) on non-public data implies that **generalization is more valuable than leaderboard probing**.

- **Data Augmentation:** Aggressively use **SpecAugment** (masking time/frequency bands) and **Noise Injection** (mixing in background noise from MUSAN or similar datasets) during fine-tuning. This prevents the model from overfitting to the specific microphone characteristics of the training set.
- **Cross-Validation:** Do not trust the Public Leaderboard score implicitly. Establish a local K-Fold Cross-Validation setup. If a model improves the Public LB score but degrades the local Cross-Validation score, reject it. It is likely overfitting.

---

## 8. Future Directions (2026 and Beyond)

The trajectory of speech technology suggests that the separation of ASR and Diarization into distinct problems (Problem 1 and Problem 2) will soon be obsolete.

- **Unified Multimodal Agents:** Models like **Qwen3-Audio** and **Gemini 2.0** are moving toward "Turn-based Transcription," where the model outputs <Speaker A> Text... <Speaker B> Text... directly.
- **Streaming-First Architectures:** The industry is pivoting from file-based processing to streaming. Models like **Nemotron-Speech-Streaming** and **SenseVoice** demonstrate that high accuracy is compatible with latencies under 100ms.
- **Small Language Models (SLMs):** The success of **Moonshine** (27M) proves that massive parameter counts are not the only path to intelligence. We anticipate a wave of sub-100M parameter models distilled from larger teachers, capable of running entirely on-device (IoT/Mobile) with SOTA accuracy.

---

## 9. Conclusion

The **DL Sprint 4.0 Bengali Speaker Diarization Challenge** is a complex optimization landscape. It cannot be won by simply selecting the model with the lowest published WER. The interaction between **Rule 7.2** (averaged phase scoring) and **Rule 8.2** (percentile RTF scoring) creates a strategic imperative to prioritize **throughput efficiency** and **generalization** over raw parameter count.

While **FireRedASR-LLM** and **Canary-Qwen** represent the pinnacle of current transcription accuracy, their computational weight makes them liabilities in the inference time ranking. Conversely, **Parakeet-TDT** and **Moonshine** offer a tactical advantage, potentially securing

maximum speed points while maintaining competitive accuracy. On the diarization front, the emergence of EEND models like **DiariZen** renders traditional clustering pipelines vulnerable, particularly in high-overlap scenarios.

The winning solution will likely be a **heterogeneous system**: utilizing lightweight transducers for the bulk of processing, specialized EEND models for diarization, and selective application of heavy LLMs only where necessary. This "Systems Engineering" approach, grounded in the rigorous benchmarks and architectural insights provided in this report, constitutes the optimal path to victory.

### Works cited


1. DL Sprint 4.0 | Bengali Speaker Diarization - Kaggle, accessed February 4, 2026, https://www.kaggle.com/competitions/dl-sprint-4-0-bengali-speaker-diarization-challenge/rules
2. DL Sprint 4.0|Bengali Long-form Speech Recognition - Kaggle, accessed February 4, 2026, https://www.kaggle.com/competitions/dl-sprint-4-0-bengali-long-form-speech-recognition/rules
3. DL Sprint 4.0 | Bengali Speaker Diarization - Kaggle, accessed February 4, 2026, https://www.kaggle.com/competitions/dl-sprint-4-0-bengali-speaker-diarization-challenge
4. blog/fine-tune-xlsr-wav2vec2.md at main - GitHub, accessed February 4, 2026, https://github.com/huggingface/blog/blob/main/fine-tune-xlsr-wav2vec2.md
5. pyannote/pyannote-audio: Neural building blocks for speaker diarization: speech activity detection, speaker change detection, overlapped speech detection, speaker embedding - GitHub, accessed February 4, 2026, https://github.com/pyannote/pyannote-audio
6. BUTSpeechFIT/DiariZen: A toolkit for speaker diarization. - GitHub, accessed February 4, 2026, https://github.com/BUTSpeechFIT/DiariZen
7. Fun-ASR is an end-to-end speech recognition large model launched by Tongyi Lab. - GitHub, accessed February 4, 2026, https://github.com/FunAudioLLM/Fun-ASR
8. The Top Open Source Speech-to-Text (STT) Models in 2025 - Modal, accessed February 4, 2026, https://modal.com/blog/open-source-stt
9. openai/whisper-large-v3-turbo - Hugging Face, accessed February 4, 2026, https://huggingface.co/openai/whisper-large-v3-turbo
10. Whisper large-v3 finetuning - Beginners - Hugging Face Forums, accessed February 4, 2026, https://discuss.huggingface.co/t/whisper-large-v3-finetuning/81996
11. nvidia/canary-qwen-2.5b · Training and fine-tuning example - Hugging Face, accessed February 4, 2026, https://huggingface.co/nvidia/canary-qwen-2.5b/discussions/13
12. nvidia/canary-1b - Hugging Face, accessed February 4, 2026, https://huggingface.co/nvidia/canary-1b

13. Open ASR Leaderboard: Trends and Insights with New Multilingual & Long-Form Tracks - Hugging Face, accessed February 4, 2026, https://huggingface.co/blog/open-asr-leaderboard
14. FunAudioLLM/SenseVoiceSmall - Hugging Face, accessed February 4, 2026, https://huggingface.co/FunAudioLLM/SenseVoiceSmall
15. How to Use the SenseVoice Speech Model - GPUMart, accessed February 4, 2026, https://www.gpu-mart.com/blog/how-to-use-the-sensevoice-speech-model
16. FunAudioLLM/SenseVoice: Multilingual Voice Understanding Model - GitHub, accessed February 4, 2026, https://github.com/FunAudioLLM/SenseVoice
17. Qwen - Hugging Face, accessed February 4, 2026, https://huggingface.co/Qwen
18. LocalAI models, accessed February 4, 2026, https://localai.io/gallery.html
19. moonshine-ai/moonshine: Fast and accurate automatic speech recognition (ASR) for edge devices - GitHub, accessed February 4, 2026, https://github.com/moonshine-ai/moonshine
20. Moonshine: Speech Recognition for Live Transcription and Voice Commands - arXiv, accessed February 4, 2026, https://arxiv.org/html/2410.15608v2
21. Flavors of Moonshine: Tiny Specialized ASR Models for Edge Devices - arXiv, accessed February 4, 2026, https://arxiv.org/html/2509.02523v1
22. FireRedTeam/FireRedASR: Open-source industrial-grade ... - GitHub, accessed February 4, 2026, https://github.com/FireRedTeam/FireRedASR
23. fairseq/examples/mms/README.md at main - GitHub, accessed February 4, 2026, https://github.com/facebookresearch/fairseq/blob/main/examples/mms/README.md
24. Fine-tuning MMS Adapter Models for Multi-Lingual ASR - GitHub, accessed February 4, 2026, https://github.com/huggingface/blog/blob/main/mms_adapters.md
25. Pyannote - Hugging Face, accessed February 4, 2026, https://huggingface.co/pyannote
26. DiCoW: Diarization-Conditioned Whisper for Target Speaker Automatic Speech Recognition, accessed February 4, 2026, https://arxiv.org/html/2501.00114v1
27. Models - Hugging Face, accessed February 4, 2026, https://huggingface.co/models?other=speaker-diarization
28. Benchmarking Diarization Models - arXiv, accessed February 4, 2026, https://arxiv.org/html/2509.26177v1
29. BUT-FIT/SE_DiCoW - Hugging Face, accessed February 4, 2026, https://huggingface.co/BUT-FIT/SE_DiCoW
30. SE-DiCoW: Self-Enrolled Diarization-Conditioned Whisper - arXiv, accessed February 4, 2026, https://arxiv.org/html/2601.19194v1
31. SE-DiCoW: Self-Enrolled Diarization-Conditioned Whisper - arXiv, accessed February 4, 2026, https://arxiv.org/pdf/2601.19194
32. Intermittent CUDA Out of Memory Errors during Parakeet-TDT-0.6B Fine-tuning - GitHub, accessed February 4, 2026, https://github.com/NVIDIA-NeMo/NeMo/issues/15037
33. nvidia_parakeet_fine_tuning - Kaggle, accessed February 4, 2026,

https://www.kaggle.com/code/deepaksomasundaram24/nvidia-parakeet-fine-tuning
34. huggingface/diarizers - GitHub, accessed February 4, 2026, https://github.com/huggingface/diarizers