孟加拉语长篇语音转录与说话人分离的多种方法研究

Epshita Jahan孟加拉国工程与技术大学计算机科学与工程系孟加拉国达卡

Khandoker Md TanjinulIslam孟加拉国工程与技术大学计算机科学与工程系孟加拉国达卡

Pritom Biswas孟加拉国工程与技术大学计算机科学与工程系孟加拉国达卡

Tafsir AlNafin孟加拉国工程与技术大学计算机科学与工程系孟加拉国达卡

arXiv:2603.03158v1 [cs.SD] 3 Mar 2026

摘要——**孟加拉语在语音技术领域仍属于低资源语言，尤其是在长篇转录和说话人分离等复杂任务中。本文介绍了一种多阶段方法，该方法为 Kaggle 上的“DL Sprint 4.0 — 孟加拉语长篇语音识别”和“DL Sprint 4.0 — 孟加拉语说话人分离”开发，旨在应对长达一小时的录音中“谁在何时说了什么”的挑战。我们采用了在孟加拉语数据上微调的 Whisper Medium（bengaliAI/tugstugi bengaliai-asr whisper-medium）进行转录，并将 pyannote/speaker-diarization-community-1 与我们自训练的分段模型相结合，以处理多样且嘈杂的声学环境。通过两遍处理方法和超参数调优，我们在私有排行榜上取得了 0.27 的说话人分离错误率，在公开排行榜上则为 0.19。对于转录，分块、背景噪声清理和算法后处理使我们在私有排行榜上的词错误率达到 0.38。这些结果表明，针对性调优和策略性的数据利用能够显著提升南亚语言的人工智能包容性。所有相关代码均可在以下地址获取：https://github.com/Short-Potatoes/Bengali-long-form-transcription-and-diarization.git**

索引词—**孟加拉语语音识别、说话人分离、Whisper、自动语音识别、低资源语言、Pyannote、语音活动检测**

I. 引言

语音识别领域的最新进展得益于大规模数据集和强大模型，但孟加拉语等低资源语言尚未充分受益。本研究通过提出针对孟加拉语的稳健长篇转录与说话人分离方法来弥补这一差距。

长篇转录颇具挑战，因为 Whisper 等系统将输入时长限制在 30 秒，因此必须对较长的音频进行分段。为了解决这一问题，我们使用语音活动检测（VAD）识别语音片段，使用 Whisper 进行转录，并通过后处理对齐输出、去除冗余内容。我们的数据集由带有转录文本的长达一小时的录音组成，这凸显了采用有效策略的必要性。

对于说话人分离，我们在其流水线中对 Pyannote 的分段模型进行了微调。两遍处理方法——先进行粗分段，再进行精细分析——提高了准确率。评估基于带有 JSON 格式真实说话人标注的长音频。

总之，将基于分段的预处理、基于 Whisper 的自动语音识别与微调后的说话人分离模型相结合，显著提升了孟加拉语及其他低资源语言的语音识别效果。

II. 方法

A. 说话人分离的模型选择

孟加拉语会话说话人分离面临诸多挑战，包括说话人重叠、声学变化以及不受限制的说话人数。我们评估了三种当代说话人分离框架：Pyannote Community [2], NVIDIA NeMo Sortformer [3],和 DiariZen [4]。

初步实验包括 Pyannote Speaker Diarization v3.1；然而，较新的 *community-1*流水线在多个基准测试中表现出持续的改进。该框架将神经说话人嵌入提取与基于 PLDA 的聚类相结合 [5],，在领域变化下实现了稳健的说话人区分。根据官方模型基准测试，报告的说话人分离错误率（DER）在 ALSHELL-4 上为 11.7，在 VoxCon-verse 上为 11.2。

NeMo Sortformer 采用端到端 Transformer 架构，但每段录音最多支持四位说话人，这限制了其在不受约束的对话场景中的可扩展性 [3]。DiariZen 报告称其性能优于 Pyannote v3.1；然而，它在开源基准测试环境中的验证仍不够充分 [4]。

综合考虑基准性能、聚类鲁棒性、可扩展性和实现稳定性，我们采用 **Pyannote Community 流水线** 作为说话人分离的骨干方案。

B. 自动语音识别（*ASR*）的模型选择

针对孟加拉语自动语音识别，我们评估了多种基于Transformer的架构，这些架构构建于Whisper和wav2vec2框架之上。

我们的基线系统采用了 bengaliAI/tugstugi bengaliai-asr_whisper-medium，这是一种针对孟加拉语进行微调的Whisper-medium模型，并使用手动分块进行长音频推理。选择该模型是因为它在孟加拉语基准测试中表现出色，并且近期文献中的实证结果也对其进行了验证。

我们进一步尝试了WhisperX流水线，将以下模型结合起来：anuragshas/whisper-large-v2-bn、tugstugi bengaliai-asr whisper-_medium、tanm_oyio/wav2vec2-large-xlsr-bengali和ai4bharat/indicwav2vec v1 bengali。

WhisperX框架通过基于语音活动检测的分块以及对齐优化，实现了更好的时间戳对齐和分段。然而，**tugstugi bengaliai-asrwhisper-medium** 始终取得最低的词错误率（WER）。因此，我们采用该模型作为主要的自动语音识别骨干模型 [6]。

C. 说话人分离的预处理

所有录音均使用 **DEMUCS**进行语音增强预处理。该模型用于抑制背景噪声，并在进行说话人分离前隔离语音成分。

为保留说话人区分性声学特征，未采用其他频谱变换或激进的滤波处理。

D. 自动语音识别的预处理

我们评估了语音增强和分段策略在转录前的效果。尽管最初尝试了基于 DEMUCS 的去噪，但其并未改善性能，因此未纳入最终系统。

在分段方面，我们比较了 Silero VAD 与经过微调的任务专用分段模型。经过微调的模型能够更准确地进行边界检测，因此被采用到最终的自动语音识别流程中。由于基于 Transformer 的模型可直接处理由波形导出的对数梅尔特征，因此未采用其他特征工程。

E. 说话人分离训练

本任务中，我们仅训练了 Pyan- note diarization-3.1 模型的分割组件。我们尝试了几种训练策略。最初，我们仅使用竞赛数据集训练分段模型，并取得了最佳表现。我们还尝试了一个规模更大的多语言数据集，该数据集结合了竞赛数据集中的孟加拉语语料库和额外的印地语数据集。我们的理由是，通常认为说话人分离不受语言影响，因此纳入更多样化的数据或许能提升泛化能力。然而，这种方法的表现略微逊于仅使用孟加拉语数据集进行训练的表现。

训练实验主要围绕三个维度进行调整：学习率、批量大小和正则化强度。较低的学习率、适中的批量大小以及最小的权重衰减，带来了最稳定的收敛，并最终取得了最佳的说话人分离效果。提高学习率和正则化强度，同时减小批量大小，虽然加快了训练速度，却导致训练不稳定、性能下降。增加训练轮次带来了些许提升，但仍未能超越采用保守超参数时所达到的稳定性。总体而言，最有效的配置兼顾了渐进式学习与有限的正则化，凸显了说话人分离分段对训练动态的敏感性。

除了训练阶段的优化外，我们还使用训练集作为开发集，针对聚类和后处理超参数进行了系统的网格搜索。搜索分为三个阶段：(i) 在单次推理下扫描聚类阈值；(ii) 缓存最优阈值下的原始预测；(iii) 扫描后处理参数（例如最短时长和合并间隔），而不必重新运行整个流水线。这种分阶段策略在尽量减少冗余推理开销的同时，实现了高效的超参数调优。

F. 长篇转录训练

对于长篇转录，我们尝试使用外部数据集对自动语音识别模型进行微调。然而，由于训练数据中很可能存在不一致性和噪声，这种方法产生的结果极差。为了解决这一问题，我们将此前为说话人分离训练的分段模型集成到自动语音识别流程中。该分段模型用于生成连贯的音频片段，再由自动语音识别系统对这些片段进行转录。由于结合说话人分离信息的分段改善了转录结果的对齐和一致性，这一策略为处理长时录音提供了更可靠的框架。

III. 结果

A. 说话人分离结果

经过约 3–4 个训练轮次后，我们的分段模型相较于基线说话人分离错误率（DER）0.40 取得了显著提升，在公开评估集上将其降至 0.199。在此基础上，我们采用了结合后处理优化的两遍解码策略，使公开评估集上的 DER 进一步降至 0.192。在私有评估集上，相同方法取得了 0.27 的 DER。这些结果表明，迭代训练和多阶段推理能够显著提升说话人分离性能，尤其是在公开基准上，同时在私有数据集上也保持了具有竞争力的结果。

B. 自动语音识别结果

使用*Tugstugi Whisper-medium* 模型并通过*SileroVAD* 进行分段后， [7], 我们的初始系统取得了词错误率

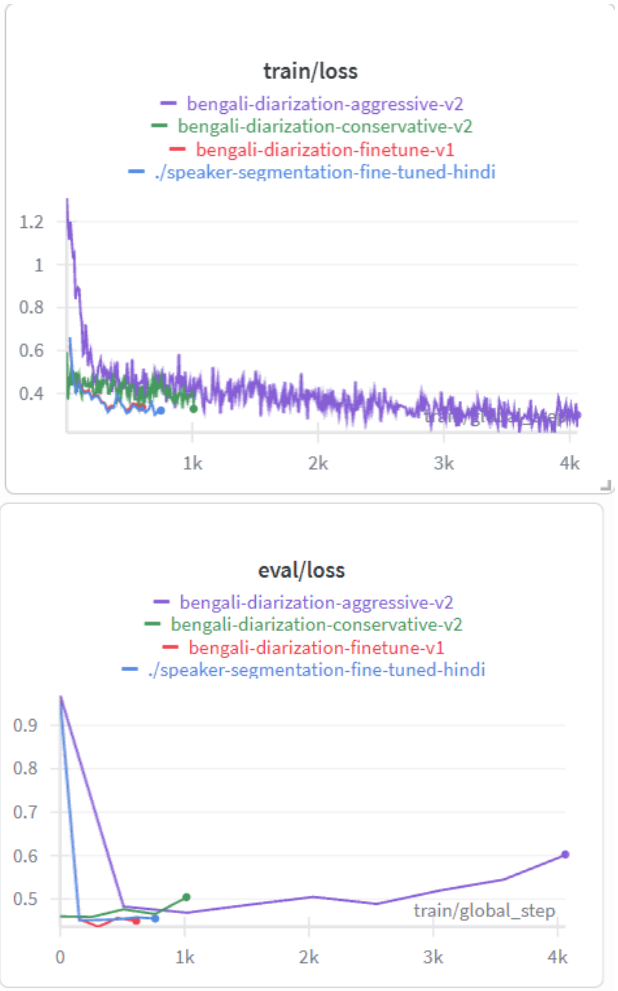

图 1. 说话人分离性能结果，展示了训练进程和对比分析。

为**0.36771**，在公开评估集上的表现如此。采用基于通义千问大语言模型的后处理后，词错误率升至**0.37921**；加入基于*DEMUCS*的语音增强后，词错误率为**0.37614**，表明其表现并未优于基线配置。最后，用我们的微调分段 模型替代 Silero VAD，取得了最佳公开评估表现，将词错误率降至**0.36674**。

在私有评估集上，相应配置的词错误率分别为**0.38970**（*Tugstugi +Silero*）、**0.39657**（添加通义千问 *LLM*）、**0.39426**（添加*DEMUCS*）和 **0.38005**（**微调分段**）微调分段模型再次取得了最佳结果。

IV. 讨论

A. 使用 *WhisperX* 搭配孟加拉语微调版 *Whisper* 和对齐模型的效果

我们尝试使用 WhisperX 流水线，并以*tugstugi bengaliai-asr whisper-medium* 作为基础模型，并以 *tanmoyio/wav2vec2-large-xlsr-bengali* 和 *ai4bharat/indicwav2vec v1 bengali* 作为对齐模型。它们的结果非常差，基线词错误率为 0.79。对齐模型未能按预期生成清晰的转录文本。

B. 使用 *Demucs* 过滤背景噪声的效果

我们使用 Demucs 库为这两项任务过滤背景噪声。它显著提升了说话人分离任务的表现，明显有助于更有效地检测说话人片段。然而，在长篇语音识别中使用该库并未带来明显的正面效果，这可能是因为增强后的信号与 Whisper 的预训练分布不匹配 [8]。

C. 任务专用分割的影响

任务专用分割比通用语音活动检测更有优势。经过微调的分割模型在各项任务中分别取得了最低的词错误率和说话人分离错误率。

D. 两遍处理方法对说话人分离的影响

在说话人分离任务中，我们使用自己的流水线检测给定音频文件中的说话人数。随后，我们将说话人数作为固定约束传入下一遍处理，从而有助于减少幻觉。这种方法取得了极佳的效果，将说话人分离错误率从最初的 0.199 降至 0.192。这一提升表明，在推理前明确将说话人数传入流水线能够显著提高准确性。

E. 后处理的影响

后处理对说话人分离和转录任务均产生了积极影响。在转录任务中，我们注意到，原始自动语音识别输出往往会频繁重复单词和短语，这很可能是因为模型将其与背景音乐混淆。为解决这一问题，我们采用了一种算法，可在单词、短语和字母层面去除重复内容，同时保留自然重复。对于说话人分离任务，如果 B 的持续时间非常短，我们就移除 A-B-A 片段并将其合并为 A-A，因为这可能是模型产生的幻觉。同样，我们还移除了极短片段，并合并了被极短静音隔开的同一说话人片段。这些步骤带来了显著改进。

V. 局限性与伦理考量

尽管性能有所提升，但仍存在若干局限。在说话人分离流程中，仅对分段组件进行了微调，而说话人嵌入模型并未使用特定任务数据进行训练。尝试微调 WeSpeaker 嵌入模型并未带来性能提升，而改用 ECAPA-TDNN 嵌入则导致性能下降，这可能是由于填充和对齐不一致所致。此外，由于时间限制，未能系统地探索其他聚类策略。

对于自动语音识别，尽管众所周知，针对特定任务进行微调通常可以提升性能，但我们的微调尝试反而导致准确率低于未微调的

表 I 说话人分离方法与错误率（DER）

方法	错误率（私有数据的 DE
Pyannote（微调版 lucius-40/speaker-segmentation-fine-tuned-bn） +Demucs（使用两遍处理）	0.28286
Pyannote （微调版 lucius-40/speaker-segmentation-fine-tuned-bn） + Demucs	0.27318
Pyannote （微调版 lucius-40/speaker-segmentation-fine-tuned-bn）	0.27773
Pyannote (微调：speaker-segmentation-bengali-optimized-cfg-1-balanced)	0.27346
Pyannote (微调：speaker-segmentation-fine-tuned-hindi-and-english)	0.28413
Pyannote (微调 lucius-40/speaker-segmentation-bengali-optimized-conservative)	0.25921
Pyannote (微调 lucius-40/speaker-segmentation-fine-tuned-bn)，使用超调参数（采用两遍处理）	0.26388
Pyannote(微调 lucius-40/speaker-segmentation-bengali-optimized-conservative) +Demucs（采用两遍处理）	0.27988
DiariZen	0.46624

表 II 长篇语音检测方法及错误率（词错误率）

方法	错误率（词错误率）
bengaliAI/tugstugi_bengaliai-自动语音识别_Whisper-medium + lucius-40/speaker-segmentation-fine-tuned-bn	0.38005
bengaliAI/tugstugi_孟加拉语ai-自动语音识别_Whisper-medium + Silero VAD	0.38970
Demucs + lucius-40/speaker-segmentation-fine-tuned-bn + bengaliAI/tugstugi_bengaliai-自动语音识别_Whisper-medium	0.39426
孟加拉语AI/Tugstugi_孟加拉语AI-自动语音识别_Whisper-medium + Silero VAD + Qwen2.5-7B 少量-镜头校正	0.39657
Whisper-X	0.79

模型，这可能是由于流水线和方法上的不一致所致。

从伦理角度来看，说话人分离系统可能会扩展到说话人画像应用，从而引发潜在的隐私问题。此外，系统可能会因数据集不平衡而表现出固有偏见，进而影响其在不同说话人群体特征或方言上的表现。

VI. 结论

我们利用基于Transformer的架构和任务特定分割策略，提出了一种有效的孟加拉语说话人分离与自动语音识别框架。

以往的说话人分离方法大多依赖传统的聚类流水线和固定的分割模块。我们的工作表明，将现代神经网络说话人分离框架与经过微调的分段方法相结合，可改善孟加拉语会话语音中的边界检测效果。尽管 Pyannote 框架提供了稳健的端到端流水线，但受计算资源和时间限制，我们无法单独微调说话人嵌入模型。我们认为，对嵌入提取器进行领域自适应训练可进一步降低说话人分离错误率。

对于自动语音识别，我们采用了针对孟加拉语微调的 Whisper 架构。虽然分段优化带来了可衡量的提升，但声学模型本身并未在竞赛数据集上进一步微调。引入结合基于CTC的对齐或混合对齐策略的任务特定微调，或许能够增强音素层面的一致性并降低词错误率。我们预计，这种对齐感知优化将进一步增强在自发对话场景中的转录鲁棒性。

VII. 贡献

我们的工作提出了一种实用的孟加拉语说话人分离与自动语音识别多阶段框架。在说话人分离方面，我们采用两遍推理策略：首先运行 Pyannote 以估计说话人数量上限，然后在流水线中限制为推断出的说话人数量进行第二遍处理，从而提高说话人分配稳定性。我们还通过使用特定任务数据微调分割组件，进一步提升了性能。

对于自动语音识别，我们采用分割感知解码，并将 n 元语法重复移除作为一种轻量级后处理步骤，以减少冗余词元生成。这些改动相结合，在不改变核心预训练骨干网络的情况下，带来了稳定的性能提升。

参考文献

[1] H.M. Shadman Tabib、Istiak Ahmmed Rifti 等，《Bengali-Loop：孟加拉语长篇自动语音识别与说话人分离的社区基准测试》，*arXiv* 预印本 *arXiv:2602.14291*，2026年。[2] H. Bredin、R. Yin、J. Coria 等，《pyannote.audio：说话人分离的神经网络构建模块》，载于*ICASSP* 会议论文集，2020年。[3] NVIDIA，《Sortformer：端到端神经说话人分离》，*arXiv:2409.06656*，2024年。[4] Jiangyu Han、Federico Landini 等，《利用自监督学习进行说话人分离》，*arXiv:2409.09408*，2024年。[5] S. Ioffe，《概率线性判别分析》，载于欧洲计算机视觉会议（*ECCV*）论文集，2006年，第531–542页。[6] Tawsif Tashwar Dipto、Azmol Hossain 等，《面向低资源语言的自动语音识别基础模型是否具备足够的泛化能力来捕捉地区方言特征？》，*IJCNLP-short* 论文集，2025年。获取地址：https://aclanthology.org/2025.ijcnlp-short.17.pdf[7] Herve Bredin、Antoine Laurent，《面向重叠感知重新分割的端到端说话人分割》，*arXiv:2104.04045v2*。

[8] Alexandre D´efossez，Nicolas Usunier，《Demucs：使用额外未标注数据混音的音乐源深度提取器》，*arXiv:1909.01174v1*。