🎙️ AI 기반 음성 요약 및 화자 태깅 파이프라인 구축 제안

1. 프로젝트 개요: 핵심 목표 및 기술 선택 근거

1.1. 프로젝트 목표 (Task)

녹취 파일 내 **다수 발화자**를 **자동 분류 및 Tagging**하고, 동시에 **고정밀 STT(Speech-to-Text)**를 수행하여 최종적으로 화자별 요약본을 생성하는 인공지능 시스템의 **Baseline Pipeline**을 구축한다.

1.2. 핵심 난제 및 해결 전략

난제 (Pain Point)	해결 전략 (Adopted Solution)
다자 발화 환경	**Pyannote.audio**를 통한 전문적인 화자 분리(Diarization) 선행 적용
녹취 품질 및 노이즈	**Whisper large-v3**의 방대한 사전 학습 기반 **강건성(Robustness)** 활용
처리 속도 및 배포 효율	**Faster-Whisper** 및 **WhisperX**를 통한 추론 속도 및 정렬 정밀도 극대화

2. 제안 파이프라인 구성 (Baseline Architecture)

본 파이프라인은 정확도(Accuracy)와 처리 속도(Inference Speed)를 동시에 최적화하는 4단계 모듈 조합으로 구성됩니다.

순서	모듈 (모델)	핵심 역할	기술 스택
STEP 1	**화자 분리 (Diarization)**	오디오 파일에서 **'누가 말했는지'** 화자 ID 및 발화 구간(Segment) 추출	`Pyannote.audio` (Hugging Face)
STEP 2	**ASR (음성 인식)**	분리된 발화 구간의 음성을 텍스트로 변환 (ASR)	`Faster-Whisper` (Whisper large-v3 기반)
STEP 3	**정렬 (Alignment)**	ASR 결과 텍스트에 **워드(단어)별 정밀 시간 스탬프** 부여	`WhisperX`
STEP 4	**결과 병합 & 출력**	Pyannote의 화자 ID와 WhisperX의 워드 스탬프를 **병합**하여 최종 태깅된 녹취록 생성	Custom Python Script

3. 핵심 모듈별 유의 사항 및 최적화 전략

모듈	유의 사항 (Action Items)	최적화 팁
Pyannote.audio	① **16kHz 샘플링 레이트** 표준화 전처리 필수. ② Hugging Face **인증 토큰 발급 및 설정** 필요.	① 초기 학습 데이터셋(AI 허브 등)을 활용하여 **모델 성능을 미세 조정(Fine-tuning)**할 수 있음.

Whisper large-v3	① **대용량 GPU 메모리(10GB 이상 VRAM)**가 요구됨. ② `Faster-Whisper` 사용 시 CTranslate2 환경 설정을 확인해야 함.	① GPU 환경에서 ** `float16` 또는 `int8` 양자화(Quantization)**를 적용하여 메모리 효율성 및 속도를 높일 것.
Pipeline Flow	STT 모델에 오디오를 **한 번에 대규모로 입력하지 않고**, Pyannote로 분리된 세그먼트를 **배치(Batch)** 형태로 처리하여 GPU 활용률을 극대화해야 함.	**배치 사이즈 최적화**를 통해 추론 속도를 모니터링하고 조정해야 함.

4. 데이터 확보 계획 (Baseline Data)

프로젝트 단계	데이터셋 (추천 소스)	활용 목적
STT 정확도 개선	AI 허브: **'한국인 대화 음성'** 데이터셋	한국어 녹취 환경 및 다양한 화자 유형에 대한 ASR 모델의 성능 검증 및 미세 조정.