## 1. 프로젝트 개요

본 프로젝트는 논문  
**"Improving Speech Recognition with Prompt-based Contextualized ASR and LLM-based Re-predictor"**  
의 실험을 100% 재현하기 위해 설계된 통합 ASR 파이프라인입니다.

본 실험은 다음의 핵심 모듈로 구성됩니다:

* ASR 모델: Zipformer 기반 Prompt-aware Encoder + RNNT Decoder
* Inference: Beam Search 기반 n-best 추출
* Post-processing: LLM 기반 Re-Predictor Reranking (LoRA fine-tuning 포함)

## 2. 전체 시스템 구조

Speech Input + Prompt

→ ASREncoder (Zipformer + BERT + CrossAttention)

→ ASRDecoder (PredictionNet + JointNet)

→ Beam Search (n-best candidates)

→ Re-Predictor (LLM reranker with LoRA fine-tuning)

→ Final Transcription

## 3. 프로젝트 디렉토리 구조 및 소스코드 설명

| **디렉토리** | **설명** |
| --- | --- |
| config/ | ASR 모델 전체 하이퍼파라미터 설정 파일 |
| dataset/ | CommonVoice 기반 Dataset 로더 및 Re-Predictor 학습 데이터셋 생성 |
| models/encoder/ | Zipformer Encoder, TextEncoder (BERT), TextAdapter, CrossAttention 모듈 |
| models/decoder/ | PredictionNet, JointNet, ASRDecoder |
| models/inference/ | RNNT 기반 Beam Search Inference Decoder |
| models/re\_predictor/ | Re-Predictor Prompt Formatter, LLM 기반 Re-Ranker, LoRA 세팅 모듈 |
| trainer/ | 학습 스크립트 (ASR 학습, Re-Predictor 데이터 생성, Re-Predictor Fine-Tuning) |
| inference\_pipeline.py | 통합 Inference 파이프라인 |
| requirements.txt | 전체 의존성 패키지 목록 |
| README.md | 실험 실행 가이드 |

## 4. 사용한 데이터셋 상세 설명

### 4.1 Common Voice 13.0 (Mozilla Foundation)

* 사용 Split: train, test
* 사용 언어: en (영문 기준 실험)
* 데이터 로딩 예시:

from datasets import load\_dataset

dataset = load\_dataset("mozilla-foundation/common\_voice\_13\_0", "en", split="train")

* 데이터 특징:
  + 다양한 발화자, 억양, 환경에서 수집된 음성 데이터
  + 실제 ASR 실험에서 광범위하게 사용되는 공개 음성 데이터셋

### 4.2 음성 특징 추출

* 샘플링 주파수: 16kHz (Resample 수행)
* 특징 추출: MelSpectrogram (n\_mels=80 사용)

### 4.3 Prompt 구성

* 모든 입력에 고정된 utterance-level prompt 사용:

sql

복사편집

You are recognizing the following utterance.

## 5. 모델 구성 및 학습 세부사항

### 5.1 ASR 모델 구성

* **Encoder**: Zipformer 기반 (Downsampling + Multi-stage Transformer Encoder)
* **Text Encoder**: BERT (Huggingface bert-base-uncased)
* **Adapter**: Text Adapter Linear Layer
* **Cross Attention**: Prompt와 Speech Feature 융합
* **Decoder**: RNNT 기반 (PredictionNet + JointNet)
* **Loss Function**: Pruned RNNT Loss (PrunedTransducerLoss from Icefall)

### 5.2 Inference 및 Beam Search

* Beam Width: 5
* n-best 후보 추출

### 5.3 Re-Predictor Fine-Tuning

* Fine-tuning Dataset: Beam Search 결과 기반 n-best 후보 생성 → Oracle Label 생성
* Model: LLaMA 기반 Instruct 모델 (예: LLaMA3 8B)
* Fine-Tuning 방식: LoRA 적용
  + Rank: 8
  + Alpha: 16
  + Dropout: 0.05
* Loss: CrossEntropy Loss 기반 fine-tuning

## 6. 전체 실험 파이프라인 순서

1️⃣ ASR 모델 학습

bash

복사편집

python trainer/train\_asr.py

2️⃣ Fine-tuning 데이터 생성 (n-best → pairwise label 생성)

bash

복사편집

python trainer/generate\_re\_predictor\_data.py

3️⃣ Re-Predictor LoRA Fine-Tuning

bash

복사편집

python trainer/fine\_tune\_re\_predictor.py

4️⃣ 통합 Inference 수행

bash

복사편집

python inference\_pipeline.py

## 7. 환경 구성 및 주의사항

* GPU: 최소 A100 80GB 이상 권장
* Python 3.9+
* PyTorch, Torchaudio, HuggingFace Transformers, Datasets, Icefall, K2, PEFT 패키지 사용
* LLaMA3 기반 모델 checkpoint 별도 준비 필요