**Model**

Pytorch, Hugging Face Transformers, Hugging Face Datasets.

**Pandas**

For reading and processing labels and metadata in CSV format.

**Librosa**

For loading and basic processing of audio signals (such as sampling rate conversion).

**Scikit-learn**

For label encoding (LabelEncoder) and dataset partitioning (train\_test\_split).

**Regular Expressions (re)**

For cleaning Korean Jamo text.

**Jamo library**

For converting Korean text to Jamo (Korean syllable decomposition) to improve speech recognition accuracy.

**Data Augmentation**

SpecAugment

**Customized multi-task neural network structure**

The main body is the ASR model (Wav2Vec2ForCTC), which outputs the speech recognition results.

Add three additional parallel classification heads (fully connected layers) for classification of age, gender, and accent respectively.

Supports improved designs such as attention pooling, shared feature layer, task-specific feature layer, and task weight adaptation.

**Loss function**

Speech recognition: CTC loss (F.ctc\_loss)

Auxiliary classification task: cross entropy loss (F.cross\_entropy)

The total loss is the weighted sum of the main task and auxiliary task losses, supporting task weight adaptation (learnable parameters).

**Batch processing and data loading**

Use custom collate\_fn to achieve batch audio, label, auxiliary task label alignment and padding.

Support multi-process data loading and pin\_memory acceleration.

**Optimization and scheduling**

Optimizer: Adam (torch.optim.Adam)

Learning rate scheduler: Linear scheduling (LinearLR)

Support mixed precision training (torch.cuda.amp)

**Evaluation indicators**

Speech recognition: WER (word error rate, jiwer library and custom implementation)

Classification task: Accuracy

**Training process management**

Support model saving, best model tracking, training logging, training timing, etc.

**Batch Inference and Label Inverse Encoding**

Support batch prediction and auxiliary task label inverse encoding (LabelEncoder.inverse\_transform).

Use jamotools for Jamo to Korean syllable synthesis display.

**Result Visualization and Analysis**

Support sampling prediction, detailed WER analysis and training process visualization (such as matplotlib).

**NumPy**

Used for random seed setting and basic numerical calculations.

**Matplotlib**

Used for visualization of the training process (such as loss curves, accuracy changes, etc.).

**Jiwer**

Calculates WER and detailed edit distance indicators for speech recognition.

**Jamotools**

Converts Jamo to Korean syllables for easy result display and analysis.

**OS, sys, time, gc**

Auxiliary functions such as file path management, system operations, timing, memory management, etc.

**User Interface**

**Streamlit**

Used to quickly build an interactive Web front end to implement functions such as audio file uploading and result display.

**Pydub**

Used for audio format conversion (such as m4a to wav), improving the compatibility of audio input.

**re**

For cleaning and formatting Korean text.