Project Documentation

Building a grammar scoring system for spoken audios.

Problem Statement

To build a Grammar Scoring Engine that evaluates the grammatical fluency of a person's spoken English using audio input. The model receives `.wav` audio files of 45–60 seconds and outputs a continuous grammar score between 0 to 5 based on human-rated MOS (Mean Opinion Score) Likert grammar levels.

This task has real-world implications for assessing language proficiency, supporting HR, EdTech, and language training tools.

Dataset Description

The dataset consists of two primary folders:

- `train.csv` — 444 audio samples with MOS grammar scores
- `test.csv` — 195 audio samples with no labels (for submission)

Audio files are stored in:

- `/audios/train/*.wav`
- `/audios/test/*.wav`

The grammar score rubric ranges from 1 (poor grammar control) to 5 (high grammatical accuracy and complexity).

Evaluation Metrics

- **Training Evaluation:**
  - Root Mean Squared Error (RMSE)
  - Pearson Correlation Coefficient (for interpretability)

Preprocessing Pipeline

Given the variability in audio length, volume, and silence, I applied a consistent cleaning process:

Steps:

1. **Stereo to Mono** — ensure single channel for uniformity
2. **Silence Removal** — remove unnecessary pauses using `torchaudio.transforms.VAD`
3. **Amplitude Normalization** — normalize values to [-1, 1] range
4. **Resampling to 16kHz** — ensure compatibility with HuBERT backbone
5. **Padding** — pad all sequences to the same length per batch using ‘`pad_sequence`’.

This ensures all inputs are standardized and model-ready.

Baseline Approach (Wav2Vec2 + LightGBM)

Method:

- Use `facebook/wav2vec2-base` to extract embeddings
- Mean-pool over sequence dimension to get one vector per audio
- Train a **LightGBM regressor** on top

Limitations:

- Embeddings are static, pre-trained only for ASR.
- Fails to capture nuances like sentence transitions, pauses, fluency
- Low Pearson correlation (~0.51)

Final Model (HuBERT + Custom Head)

Why HuBERT?

- Self-supervised speech model trained to capture phonetic + prosodic structure
- Does not rely on labeled text

Architecture:

- **Backbone:** `facebook/hubert-base-ls960`
    - All layers frozen (to save GPU)
- **Custom Regression Head:**

```
self.regressor = nn.Sequential(

    nn.Linear(768, 256),

    nn.ReLU(),
```

```
    nn.Dropout(0.3),

    nn.Linear(256, 64),

    nn.ReLU(),

    nn.Linear(64, 1)

)
```

Training:

- Batch size: `1` (large audio + large model)
- Loss: `MSELoss`
- Optimizer: `Adam (lr=1e-3)`
- Epochs: `5`
- Training only the regression head

Inference Pipeline

- Preprocess audio as above
- Pass through HuBERT (frozen)
- Mean-pool the hidden states
- Feed to MLP head
- Output score

NOTE:

I experimented with multiple modeling strategies, including classical ML, pretrained audio models, and hybrid architectures. These two approaches (Wav2Vec2 + LGBM and HuBERT + Custom Head) stood out as the most effective within compute constraints.

The final model could likely achieve significantly better results if trained on **faster or more memory-efficient GPUs**, allowing for larger batch sizes or full backbone fine-tuning.

Credits:

1. Articles on **Towards Data Science**
2. **ChatGPT (Free version)** for all the errors.
3. **Perplexity AI** for efficient research and clarifications.