Vidushi ‘s Grammar Scoring Engine

Overview:

So, I built a grammar scoring engine that works on spoken data using a two-model approach. The main idea is to take transcribed audio, use Whisper to handle transcription, then use some fancy transformer embeddings (RoBERTa and SBERT) to understand meaning, and finally, run everything through LightGBM to predict grammar quality on a scale from 1 to 5. I also mixed in some custom grammatical features to help boost the prediction accuracy.

After diving into this for a couple of days and running through a bunch of tests, this setup ended up performing the best. Here’s a rundown of what I tried and why I landed on this approach:

1. **Transformer embeddings** (like RoBERTa and SBERT) turned out to be the best at picking up on the finer details of grammar. They understood sentence structure and meaning without needing me to program grammar rules manually.

2. I gave **LLM-based scoring** a shot with Mistral 7B, but it wasn’t great. The model’s scores were all over the place it’d lean too positive or negative depending on how I worded things. It wasn’t consistent enough, so I dropped it.

3. I also decided **not to fine-tune** the transformers or LLMs because the dataset was pretty small (only 444 samples). Fine-tuning with so little data would’ve risked overfitting, so I skipped that.

4. When it came to **feature engineering**, I focused on things like sentence complexity, structure, and word choices. These features worked well alongside the embeddings and didn’t add any biases that would skew the model.

In the end, the approach I ended up with was a good balance between being complex enough to capture the important details, but still simple enough to avoid overfitting. It works well with the limited data I had and provides solid predictions on grammar quality.

This code does the following step by step:

1. Install dependencies

- Installs key packages:
    - `openai-whisper` for audio transcription
    - `transformers` and `sentence-transformers` for language model embeddings (RoBERTa, SBERT)
    - `sentencepiece`, `datasets`, `evaluate` for text processing and evaluation
    - `lightgbm` for regression modeling
    - `torchaudio` for audio processing

---

2. Import libraries

- Loads essential Python libraries: `os`, `torch`, `pandas`, `numpy`, `tqdm`, `transformers`, `scikit-learn`, `lightgbm`, `evaluate`, `matplotlib`, `seaborn`.

---

3. Load and prepare dataset

- Defines dataset path and reads `train.csv` and `test.csv` metadata files.
- Updates dataframes with full audio file paths for training and test sets.
- Displays dataset shapes (444 training samples, 204 test samples).

---

4. Transcribe audio using Whisper

- Uses Whisper model to transcribe audio files into text.
  (Note: Whisper outputs were cached to avoid repeated runs.)

---

5. Extract features

- Uses transformer-based models (RoBERTa, SBERT) to generate semantic embeddings from transcripts.
- Computes handcrafted grammatical features:
    - Sentence structure
    - Word complexity
    - Quantitative text metrics

---

6. Train ensemble regression model

- Combines embeddings and handcrafted features.
- Splits data into training and validation sets.
- Trains a LightGBM regression model to predict grammar quality scores (1–5 scale).

---

7. Evaluate model performance

- Reports metrics:
    - Training RMSE ≈ 0.20
    - Validation RMSE ≈ 0.94
    - Pearson correlation ≈ 0.62
    - R^2 score ≈ 0.35
- Analyzes prediction errors across score bins.

---

8. Generate predictions on test set

- Applies the trained model to test data.
- Prepares predictions for submission or downstream use.

Summary

This submission introduces a grammar scoring engine optimized for spoken English audio, built using a hybrid, ensemble-based architecture that blends deep semantic modeling with handcrafted linguistic insights. The core pipeline begins with transcription via OpenAI's Whisper model, which demonstrated robust performance across various audio qualities in the dataset. These transcriptions are then processed through two pretrained transformer models: **RoBERTa** for capturing general contextual semantics, and **Sentence-BERT (SBERT)** for richer sentence-level embeddings tailored to similarity and regression tasks.

To complement these deep embeddings, I engineered a set of grammatical features that provide interpretable insights into the structural and syntactic quality of speech. These features include token counts, parts-of-speech distributions (e.g., number of nouns, verbs, adjectives), and average word lengths designed to reflect grammatical richness and coherence. Together, these representations were used to train a **LightGBM regressor**, chosen for its balance of speed, interpretability, and ability to handle small tabular datasets without overfitting.

I experimented with alternative modeling strategies over two days of focused development. This included using large language models like **Mistral 7B** for direct grammar scoring via prompting. However, despite their potential, LLM-based outputs were inconsistent and lacked repeatability, making them unsuitable for a reliable evaluation task. I also avoided fine-tuning any large pretrained models due to the small dataset size (444 training samples), which would likely have led to overfitting and poor generalization.

The final model was selected for its strong empirical performance and stability. It combines the generalization power of transformers with structured linguistic

features, resulting in a robust scoring system that maintains interpretability and reproducibility. The ensemble architecture also allows future extensibility for example, by integrating fluency or pronunciation scores without needing to overhaul the pipeline.

This solution reflects a thoughtful tradeoff between cutting-edge NLP tools and classical feature engineering, optimized specifically for grammar scoring in spoken English contexts.