Final Project Proposal

Group 1: Andrew Ash and Marcus Mellor

March 26, 2025

Proposed Concept: Filler Word Counter

We propose a deep neural network designed to replicate the role of an “ah” counter, as in public speaking organizations such as Toastmasters. The network should take real-time audio as input and indicate whether the last few seconds of audio contained filler words such as “ah” or “um”. The system should be able to provide feedback within a few seconds of input.

Detailed Design

Training Dataset

The TED-LIUM dataset is a corpus of English-language TED talk audio with transcriptions. Crucially for our application, these transcripts include speech disfluencies such as filler words. The dataset was originally developed to help train speech recognition tasks.

The dataset consists of 452 hours of audio sampled at 16kHz with verbatim transcriptions in the text-based STM format. Filler words are mapped to a specific set of strings so that they are easily identifiable. This transcription scheme should allow us to identify whether the last few seconds of audio contained a filler word as a binary classificatoin task.

We also plan to use the PodcastFillers dataset, a corpus of 145 hours of English podcast audio with manually annotated filler words. This dataset was developed specifically to identify the location of filler words in audio. It uses a higher sampling rate, so we will most likely downsample to match the lower sampling rate of the TED-LIUM dataset.

Both datasets are available using the `datasets` Python module developed by Hugging Face.

Network Architecture

Since the primary function of this network is identifying whether a time series contains a particular keyword, we suspect that we will need an architecture which has some amount of memory.

For this reason we plan to use LSTM-based models. We may extend this with convolutional or time-delay neural networks as well, depending on initial results.

We plan to use Python with the Pytorch framework. We will likely also use Kaldi, a Python library for speech recognition tasks.

Performance Metrics

Because we intend to use real-time input and feedback, we want to heavily penalize false positives; it is more acceptable for the network to miss a few filler words than for the network to misclassify non-fillers. For this reason we intend to use F_β as our scoring metric. β will be tuned towards higher precision and lower recall.

We will evaluate performance through a combination of metrics. Accuracy on a test set will give us a baseline, and performance on our second dataset will give us a comparison representative of real-world application. ROC and DET curves will help us tune our false positive and false negative rates.

Schedule

Week 1: Build initial network and data processing
Week 2: Network and dataset tuning, evaluate whether we need to make any major changes
Week 3: Network rearchitecting (if needed) and fine-tuning
Week 4: Write any code needed to use inference for a live demo, prepare presentation

References and Resources

The links below are a few resources we have found so far in preparation for this project. We will continue to search publication databases such as IEEE Xplore for relevant articles as the project proceeds.

- https://www.semanticscholar.org/reader/3e908e4f31d95b321729d6a2893cb2c7c0071908
- https://art-jang.github.io/assets/pdf/kws.pdf
- https://huggingface.co/datasets/LIUM/tedlium
- https://huggingface.co/datasets/ylacombe/podcast_fillers_by_license
- https://en.wikipedia.org/wiki/Receiver_operating_characteristic
- https://en.wikipedia.org/wiki/Detection_error_tradeoff
- https://en.wikipedia.org/wiki/F-score#F%CE%B2_score