Self-supervised learning for dysarthric speech recognition

Introduction

This essay proposes a self-supervised learning pipeline for training a streaming automatic speech recognition (ASR) system for dysarthric speech. It is based on the techniques described in the paper "Deploying SSL in the wild for hybrid ASR".

ASR Model Architecture

The proposed ASR model architecture follows the design presented in Figure 1 of the referenced paper. It consists of a self-supervised pre-training component (Lfb2vec) and a fine-tuned streaming ASR model.

Training Data Collection

Collecting a large and diverse dataset is crucial for effective self-supervised learning. The proposed approach aims to gather as much dysarthric speech data as possible, including curated corpora and unannotated recordings. Existing dysarthric speech corpora, as listed in the "Survey of dysarthric ASR" paper, will be utilized. To mitigate the potential scarcity of dysarthric speech data, the dataset will be augmented with synthesized dysarthric speech. This can be achieved by transforming normal speech or generating dysarthric speech using a trained text-to-speech system.

Data Preprocessing Pipeline

The data preprocessing pipeline involves several steps:

1. **Audio conversion**: All audio files will be converted to 16kHz, 16-bit PCM format using ffmpeg.
2. **Voice Activity Detection (VAD)**: A VAD algorithm will be employed to filter out long silences (exceeding 1 second) from the audio recordings.
3. **Audio segmentation**: The audio data will be segmented into chunks with a maximum length of 20 seconds.
4. **Audio Event Detection (AED)**: An AED system will be used to distinguish speech portions from background noises such as music. Three AED filters will be applied:
    - **Speech-filter**: Ignores utterances without any speech events.
    - **Speech-crop**: Crops utterances based on speech events to include only the speech portion.
    - **Rand-crop**: Randomly crops long utterances (≥ 5s) for data augmentation purposes.

    The AED filters will be applied on-the-fly during training to enhance the model's robustness.

Self-Supervised Pre-Training

The Lfb2vec model (left side of Figure 1) will be pre-trained using all available training data. Similar to wav2vec 2.0, random time steps will be masked with a probability of 0.065, and the subsequent 10 time steps will also be masked. The masked features will then be fed into the Encoder, a 6-layer Latency-Controlled BLSTM (LC-BLSTM) with 600 hidden dimensions.

If the training data size is limited, the number of layers and dimensions may be slightly reduced to prevent overfitting. The pre-training process will optimize the contrastive loss between masked positions of context vectors and target vectors. 100 negative samples will be randomly drawn from the same utterance, and the flatNCE loss function will be used as it has shown slightly better performance than InfoNCE. The AdamW optimizer will be employed for pre-training due to its stability in learning rate scheduling with large maximum iterations.

Fine-tuning for Streaming ASR

The streaming ASR model (right side of Figure 1) will be fine-tuned using the labeled subset of the training data. The Encoder will be initialized from the pre-trained model and frozen during fine-tuning. Only the linear projection layer will be trained from scratch. The training loss will be the cross-entropy loss, and the Adam optimizer will be used since fine-tuning typically requires fewer iterations than pre-training.

Continuous Learning

Once the streaming ASR system is deployed, it will continuously monitor its accuracy on new speech data. Speech inputs resulting in low ASR accuracy will be saved for periodic fine-tuning of the ASR model. In contrast, the computationally intensive pre-training process need not be performed frequently, as the pre-trained model can be effectively fine-tuned to adapt to new data.