Introduction: ***

Paper Overview

The basis for our project is the paper *Personalized Adaptation With Pre-Trained Speech Encoders.*

In this paper, the researchers propose novel architectural and technical modifications to speech encoder models to improve their performance at speech recognition tasks, beating prior benchmarks on the MSP-Podcast dataset.

(1) Personalized Adaptive Pre-training:

- A method of making models 'personalize' and learn robust speech-speaker relationships during pre-training.
- Vector/feature-ized speech utterances are fed through a pre-trained speech encoder [authors use Meta's HuBERT]. Speaker features for these audio are combined with a learned speaker embedding.
- The results of this robust, personalized speech representation step are then used for fine-tuning for emotion recognition.

(2) Personalized Label Distribution Calibration:

- The authors identify label-shift—where the probability distribution of emotional indicators predicted for a test speaker differs substantially from those seen in the test-data—as a substantial problem models face in evaluating unseen speakers.
- So what PLDC does is the following: during evaluation, the model will also search back into saved training data-files to find speakers with the most similar personalized embeddings as the currently evaluated speaker, and then use these to shift the mean of its predictions in order to make them more robust.

Why This Paper?

We chose this paper because we find the sort of HCI work it deals with very interesting and, of course, more and more relevant as AI seeps into our everyday lives. The paper was published by USC's Institute for Creative Technologies as part of their work in the field of Affective Computing.

The authors' description of the work's relevance resonated with us in particular—they noted that speech emotion recognition is not an obvious and easily generalizable task due to the variance in how people of different backgrounds and cultures express themselves. AI taking this into account is important to make sure AI tools and developments can be accessible to all.

What Type of Project Is It?

I think it's not super clear that it's one type of project or another! But, of course there are elements of basic regression supervised learning (using CCC loss which is basically MSE Loss on steroids for valence regression during the fine-tuning process, for example) but also self-supervised learning (which is what happens when you continue the pre-training process of the HuBERT model through PAPT) at play.

The core of the project, PAPT, will be self-supervised learning for sure.

Project Objectives *** **Updated: Please give due consideration**

Changes:

Upon expert consultation and given the limitations—time, resource, and ability— of this project, we will not be re-implementing the PAPT technique.

There are a few reasons for this.

(1) HuBERT is trained using a **proprietary self-supervised clustering technique**.
    - It generates pseudo-labels for batches of data it is given, and minimizes its loss based on said generated pseudo-labels.
    - The details of this clustering approach are **not known to the public**.
    - The authors **approximate** how HuBERT is pre-trained.

(2) The authors did not include their code for approximating HuBERT's clustering algorithm in the code they provided to us.
    - **Attempting to replicate** HuBERT's pre-training clustering algorithm would require work and research **well beyond the scope of this project**.

(3) Even if we had the adaptive pre-training code, HuBERT is a **1B parameter speech model**. We do not have the computing power necessary to pre-train HuBERT.

(4) Even if we had the computing power to pre-train HuBERT, there is a certain sense in which we **ought not pre-train the model** on the grounds of using good research methodology.
    - HuBERT is a foundation model in the realm of speech encoding—other models that fall into this class include: T-5, GPT-4 in generative text models, EfficientNet, ViT in image recognition.
    - The idea of foundation models is that they provide **extremely robust** representations of data in their modalities; their pre-training weights should **not** be touched.
    - A real criticism of the paper that can be made is that they move straight from their experimental evidence showing simple fine-tuning is not enough to arguing for the necessity of a complete pre-training cycle of audio foundation models on data.
    - This is both a prohibitively computationally expensive task as well as a drastic decision that generally shouldn't be made w.r.t foundation models.

#--------------------

New Goals:

We will implement similar feature engineering methods as proposed by the authors—namely, featurizing and embedding speaker vectors—in order to test their effectiveness and impact on fine-tuning HuBERT for emotion recognition in speech.

We will implement variations on this approach (i.e **how** to personalize; how to append this info to the base speech encodings generated by HuBERT) for ablation studies.

We will probably stick with the implementation of the original simple MLP for fine-tuning classification the authors used, though we were definitely interested in using a transformer encoder classification—we will write about this in the ultimate submission just as it is an intriguing idea and direction, though ultimately might have been too complicated to perform.

We believe re-implementing model architecture is a strong **target/baseline** goal to accomplish.

If we are able to do this to good standards, we are very interested in implementing their PLDC technique for label shift too (as a **reach** goal) but will have to see the rate at which work progresses.

Challenges:

A lot of the difficulty in this project so far has been knowing how to pivot and decide what we can and cannot do in our work on this paper.

Most notably, we initially went in a bit naively thinking that the full results and methodology of the paper would be reproducible and within the scope of a DL final project.

So it took a bit of 'wriggling' with ideas and asking advisors, experts about what is really feasible before realizing the enormity of (the computational power required for) what the authors are actually doing, as well as the technical difficulties of this.

It was a difficult experience to try to understand what researchers are doing and decide what works, what doesn't, what we can reimplement and should reimplement, and make these sorts of choices. But it was also very valuable to teach how to read research papers and make such judgements.

On the code side, we had never worked either with audio modalities nor preprocessing large batches of data, so we had to learn a lot of the tools (writing/saving to files and storing/loading data in TensorFlow).

It was actually a lot of fun to write scripts for preprocessing nonetheless as we learned a lot about data pipelining!

Insights:

Unfortunately, apart from the insight that not every research paper is very reproducible, we don't have any good insights into our data and the way models can be used to process it yet 😞 as we have not trained our model yet.

Plan:

We are further behind than we would like on this project because, while we've finished preprocessing data (though have some adjustments to format to make, the overall conception is good and we have a plan for that), we have not trained our model on it yet and the hyperparameter tuning as we all know can be rather difficult…

We will definitely be dedicating a lot of time to this in the coming week.

Our plan is the following:

(1) Clean up the pre-processing script. Action items are:
   - (a) Calculate speaker-embeddings differently. Initially, I did a raw MFCC on the speaker embeddings, then padded them all to size (216, 13) and then intended to average them out after storing them in a dict (JSON) file.
   - (b) But this is maybe not the smartest way to do things. It is better to do a raw MFCC w/ (13 coefficients/feature size) on the speaker embeddings, do average pool (just a fancy way of saying tf.reduce mean 🙂) on each embedding to make it a 1x13 vector, then average pool again to find the embedding. This way no padding is required and no noise is introduced into the speaker embedding.

(2) Load the main CSV data (we have both a CSV as a general dataset table and a JSON file for speaker ⇒ speaker embeddings) into datasets to start the data loading process.

(3) Process the raw audio with HuBERT, make a new dataset with the speech encodings, and move this into our fine-tuning pipeline asap.

(4) Experiment with various ways of calculating and learning speaker embedding; namely:
   - (a) Learnable embedding (dense layer to turn it into higher dimensional embedding space = in size to the HuBERT output; sum with the output)
   - (b) Unlearned embedding (simply use the averaged out MFCC features to track speaker embedding; concatenate speaker embeddings with audio sample for a given speaker)

(5) If time, implement PLDC method for test robustness.