SPEECH EMOTION RECOGNITION

Source

- Paper: http://arxiv.org/abs/2110.06309

- Code: https://github.com/b04901014/FT-w2v2-ser
  - *should be able to implemented following the given instructions, but some of the libraries(e.g. fairseq)/code require NVIDIA GPU and NCCL*
  - *replace python with python3 if needed*

- Dataset: IEMOCAP
  - *can be downloaded and prepared by running following command*
  - *replace python with python3 if needed*
  -
    ```
    cd Dataset/IEMOCAP && python make_16k.py IEMOCAP_DIR && python gen_meta_label.py IEMOCAP_DIR && python generate_labels_sessionwise.py && cd ../..
    ```

Main idea of the paper

- Fine-tuning Wav2vec 2.0 for speech emotion recognition (SER)
  - *About wav2vec 2.0:*
  - *proposed for speech recognition (ASR)*
  - *a transformer-based model trained to extract contextualized representations from raw audio signal*

- Different fine-tuning strategies:
  - *Two baseline methods:*
    - vanilla fine-tuning (V-FT)
    - task adaptive pretraining (TAPT), which is an existing NLP fine-tuning strategy
  - *A novel fine-tuning method proposed by the authors:*
    - pseudo-label task adaptive pretraining (P-TAPT), which modifies the TAPT objective to learn contextualized emotion representations

Overview of methods

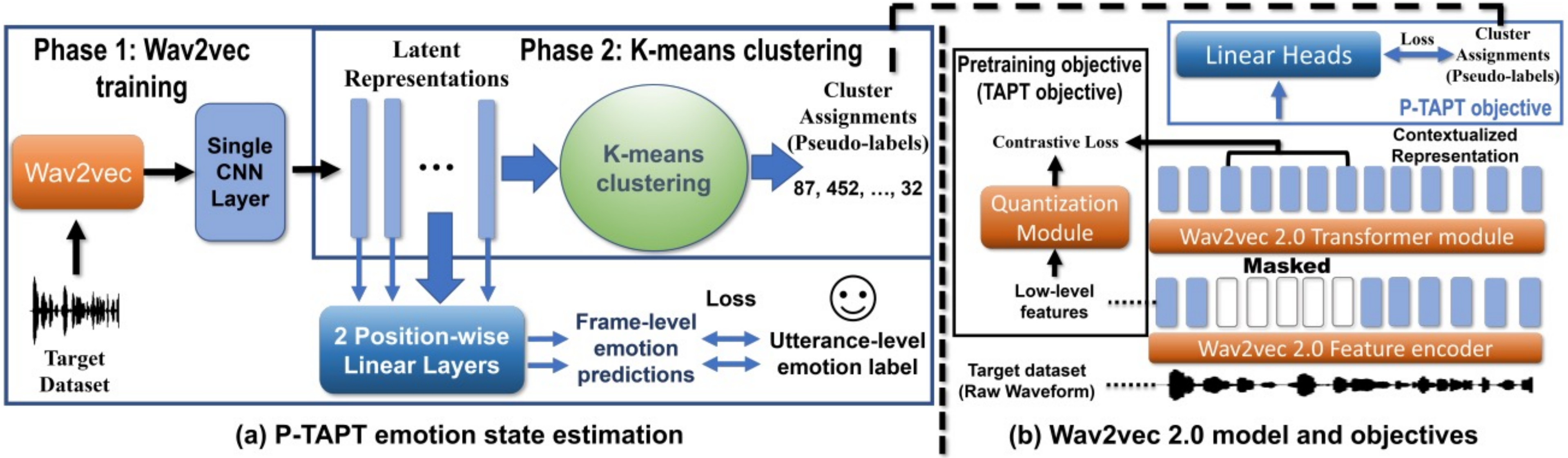

Phase 1: Wav2vec training
Wav2vec
Single CNN Layer
Target Dataset
Latent Representations
Phase 2: K-means clustering
...
K-means clustering
Cluster Assignments (Pseudo-labels)
87, 452, ..., 32
2 Position-wise Linear Layers
Frame-level emotion predictions
Loss
Utterance-level emotion label
(a) P-TAPT emotion state estimation
Pretraining objective (TAPT objective)
Contrastive Loss
Quantization Module
Low-level features
Target dataset (Raw Waveform)
Linear Heads
Loss
Cluster Assignments (Pseudo-labels)
P-TAPT objective
Contextualized Representation
Wav2vec 2.0 Transformer module
Masked
Wav2vec 2.0 Feature encoder
(b) Wav2vec 2.0 model and objectives

Fig. 1: System overview of our methods. (a) Emotion state estimation phase of P-TAPT. An additional CNN with stride 2 is used to align the time steps between wav2vec and wav2vec 2.0. The output of cluster assignments will be used as pseudo-labels for the P-TAPT objective. (b) Model architecture and pretraining objective of wav2vec 2.0 along with our P-TAPT objective.

More on Wav2vec 2.0

- Consists of three sub-modules:
  - *Feature encoder: a multi-layer CNN that processes the input signal into low-level features*
  - *Transformer module: applied to the above representation produced by the encoder to produce contextualized representation*
  - *Quantization module: discretizes the low-level features into a trainable codebook*
- To train the model, part of the low-level features are masked from the transformer module, and the objective is to identify the quantized version of the masked features based on its context

More on vanilla fine-tuning (V-FT)

- Intuition: for Wav2vec 2.0, there is no utterance level pretraining task to naturally form a sentence representation. So aggregation across time step is required to fine-tune on utterance level classification tasks.

- Method: average pooling on the final layer
  - *i.e. the final contextualized representation extracted by wav2vec 2.0 → a global average pooling across the time dimension → the ReLU activation → a single linear layer to predict the emotion categories*

More on Task adaptive pretraining (TAPT)

- a method to fine-tune pretrained language models on domain-specific tasks.
- bridges the difference between the pretraining and target domain by continuing to pretrain on the target dataset.

More on pseudo-label task adaptive pretraining (P-TAPT)

- Intuition: TAPT adapts to the emotive speech by continual training with the pretraining objective, but it does not make use of the emotion labels. i.e. the contextualized representations obtained will be general features suitable for various downstream tasks. So P-TAPT propose to adapt this objective to generate emotion specific features.

- Methos: focus on predicting the emotion state of the masked sequence
  - *Step 1, recognize frame-level emotion states: finetune wav2vec to extract frame-level emotion representation → run k-means clustering algorithm on all of the extracted representations from the target dataset (intermediate result: cluster assignment <=> a pseudo-label that represents local emotion state)*
    - PS: only utterance-level emotion labels are given for most of the SER dataset, but frame-level emotion representation is also useful for predicting an utterance-level emotion label
  - *Step 2: replace the TAPT objective with the new P-TAPT objective*

- Advantages: better data efficiency, less vulnerable to over-fitting, simplifies the fine-tuning stage

Results

- All three outperformed state-of-the-art models
- P-TAPT > TAPT > V-FT

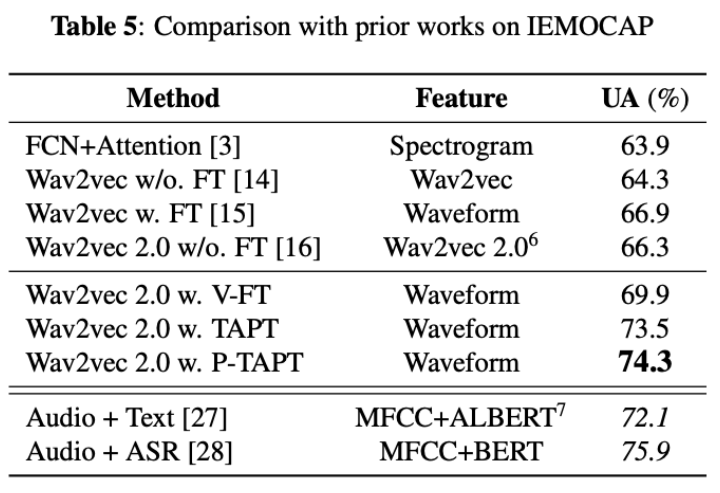


Table 5: Comparison with prior works on IEMOCAP

Method	**Feature**	**UA (%)**
FCN+Attention [3]	Spectrogram	63.9
Wav2vec w/o. FT [14]	Wav2vec	64.3
Wav2vec w. FT [15]	Waveform	66.9
Wav2vec 2.0 w/o. FT [16]	Wav2vec 2.0[6]	66.3
Wav2vec 2.0 w. V-FT	Waveform	69.9
Wav2vec 2.0 w. TAPT	Waveform	73.5
Wav2vec 2.0 w. P-TAPT	Waveform	**74.3**
Audio + Text [27]	MFCC+ALBERT[7]	*72.1*
Audio + ASR [28]	MFCC+BERT	*75.9*

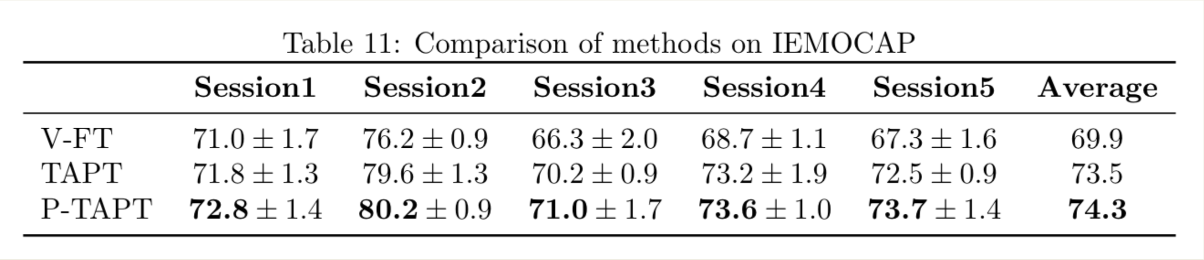
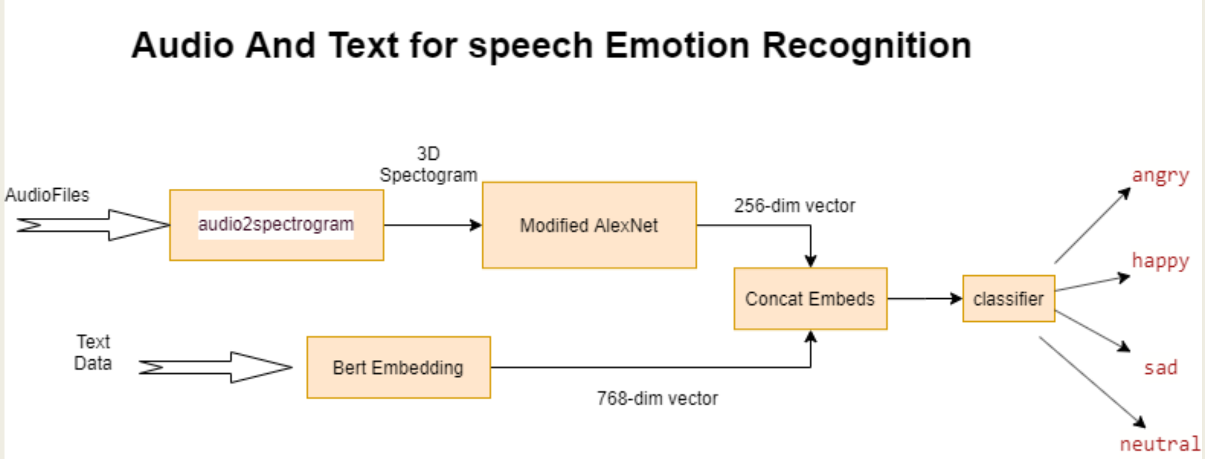


Table 11: Comparison of methods on IEMOCAP

	Session1	**Session2**	**Session3**	**Session4**	**Session5**	**Average**
V-FT	71.0 ± 1.7	76.2 ± 0.9	66.3 ± 2.0	68.7 ± 1.1	67.3 ± 1.6	69.9
TAPT	71.8 ± 1.3	79.6 ± 1.3	70.2 ± 0.9	73.2 ± 1.9	72.5 ± 0.9	73.5
P-TAPT	**72.8** ± 1.4	**80.2** ± 0.9	**71.0** ± 1.7	**73.6** ± 1.0	**73.7** ± 1.4	**74.3**

Possible limitation

- The models in the paper only utilized sound waves, it did not explore the usefulness of the fine-tuned models in multi-model setting, i.e. analyze both audio and text to potentially further improve the performance. So we may try it out, although result is not guaranteed
  - *We may try something similar to graph below but replace AlexNet with one of the three fine-tuned models*


Audio And Text for speech Emotion Recognition
AudioFiles
audio2spectrogram
3D Spectogram
Modified AlexNet
256-dim vector
Concat Embeds
classifier
angry
happy
sad
neutral
Text Data
Bert Embedding
768-dim vector