Introduction to Deep Learning for Speech and Language Processing: Coding Project

Maximilian Kimmich, Pascal Tilli, Ngoc Thang Vu

Winter Term 24/25

Introduction

- The Coding Project starts on 28th January, 2025 at 3:30 pm.
- `https://dlcourse.ims.uni-stuttgart.de/` is used to publish data as well as for evaluating your model.
- There are two stages.
  1. Submission Test: Train & tune your model, and test submission by submitting your predictions on **ser_test_1.json**; unlimited uploads.
     Starts on 28th January, 2025 at 3:30 pm and ends on 28th February, 2025 at 11:59pm.
  2. Submission Final: submit your predictions once on **ser_test_2.json** for the final evaluation; only one upload allowed.
     Starts on 24th February, 2025 at 12am and ends on 28th February, 2025 at 11:59pm.
- Data becomes available as the stages become active.
- For evaluation, upload your predictions in the same format as the download.
  - Each data sample in a file has a unique id (be aware of its type).
  - Upload the data sample id with your prediction.
  - You don't have to upload the input data (there is a size limit for each upload).
- On 3rd, 4th and 6th February, 2025, there will be live coding sessions where we will provide help and answer questions.

Task: Speech Emotion Recognition (SER)

Description

Predict the emotion class given the features of a speech input.
There are 4 classes, where we use a two-dimensional representation:

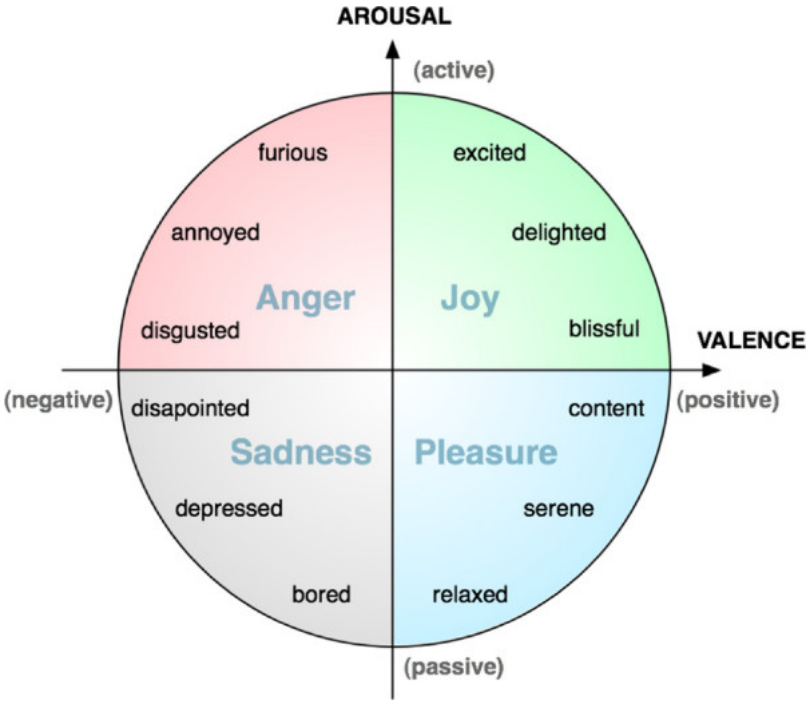

Figure 1: Multimedia content analysis for emotional characterization of music video clips (Yazdani et al., 2013)

Both valence and activation can be either 0 or 1. Your model will be evaluated by means of the accuracy of the emotion class (i.e., both activation and valence have to be correct to predict the correct class).

Data

- `features` entry is our input, it contains a list of a list with 26 items.
- length of inner list: 26 (float numbers - represent one preprocessed speech frame (logMel)).
- length of outer list: number of frames per data-point, e.g. 10 or 15, ...
- you can directly create a numpy array or PyTorch tensor from a list using `numpy.array([0,1])` and `torch.tensor([0,1])` respectively.
- Data on which you should do the predictions is missing the labels, i.e, activation and valence.
- Your predictions should have the exact same format.

The following files will be provided:

- train.json, dev.json: The SER training and development data.
- ser_test_1.json: The data with which you can test the submission, therefore you should submit your predictions for ‘stage 1: Submission Test‘ on this data.

- ser_test_2.json: The data on which you should submit your final predictions for ‘stage 2: Submission Final‘.

This is the format of the training data:

```
{
    "0": {
        "valence": 0,
        "activation": 1,
        "features": [[5.502810676891276, 5.389630715979907, ...], [...], ...]
        },
    "1": {
        "valence": 1,
        "activation": 1,
        "features": [[3.502810676891276, 5.389630715979907, ...], [...], ...]
        },
    ...
}
```

And this is how the prediction should look like (omitting features):

```
{
    "0": {
        "valence": 0,
        "activation": 1,
        },
    "1": {
        "valence": 1,
        "activation": 1,
        },
    ...
}
```

Upload your predictions without *features* fields!

You can read the JSON files as in the following:

```
import json

# read the data
data = json.load(open('train.json', 'r'))

# print the first item
print(next(iter(data.items())))
```