```
# Custom Dataset
class CustomVideoDataset(Dataset):
	def __init__(self, df, window_size=10, stride=5):
	self.df = df
	self.df['arousal'] = self.df['arousal']
	#print(df)
	self.df['valence'] = self.df['valence']
	self.window_size = window_size
	self.stride = stride
	self.video_windows, self.labels_windows = self.prepare_windows()

	def __len__(self):
	return len(self.video_windows)

	def __getitem__(self, idx):
	window_frames = self.video_windows[idx]
	embeddings = [self.df.loc[self.df['path'] == frame, self.df.columns[3:259]].values for
frame in window_frames]
	frames_tensor = torch.tensor(embeddings, dtype=torch.float32).squeeze(1)

	labels = self.labels_windows[idx]
	labels_tensor = torch.tensor(labels, dtype=torch.float32)

	return frames_tensor, labels_tensor

	def prepare_windows(self):
	video_frames = {}
	labels = {}
	for _, row in self.df.iterrows():
	video_id = self.extract_video_info(row['path'])
	if video_id not in video_frames:
		video_frames[video_id] = []
		labels[video_id] = []
	video_frames[video_id].append(row['path'])
	labels[video_id].append((row['arousal'], row['valence']))

	video_windows = []
	labels_windows = []
	for video_id in video_frames:
	frames = video_frames[video_id]
	label_vals = labels[video_id]
	for i in range(0, len(frames) - self.window_size + 1, self.stride):
		video_windows.append(frames[i:i + self.window_size])
		window_labels = label_vals[i:i + self.window_size]
		avg_arousal = sum([label[0] for label in window_labels]) / len(window_labels)
		avg_valence = sum([label[1] for label in window_labels]) / len(window_labels)
		labels_windows.append((avg_arousal, avg_valence))
```

```
        return video_windows, labels_windows

        def extract_video_info(self, file_path):
        parts = file_path.split('/')
        video_id = parts[-2]
        return video_id
```

I have audio and video datasets: SEWA_radiant_fog_160_dev, SEWA_radiant_fog_160_train and SEWA_radiant_fog_160_test with videos and SEWA_features_wav2vec_1_seconds_dev, SEWA_features_wav2vec_1_seconds_train and SEWA_features_wav2vec_1_seconds_test with audio of the same videos.
Here is the structure of these datasets as far as i know:
Load the datasets audio_train = pd.read_csv('/mnt/data/SEWA_features_wav2vec_1_seconds_train.csv') audio_dev = pd.read_csv('/mnt/data/SEWA_features_wav2vec_1_seconds_dev.csv') audio_test = pd.read_csv('/mnt/data/SEWA_features_wav2vec_1_seconds_test.csv') video_train = pd.read_csv('/mnt/data/SEWA_radiant_fog_160_train.csv') video_dev = pd.read_csv('/mnt/data/SEWA_radiant_fog_160_dev.csv') video_test = pd.read_csv('/mnt/data/SEWA_radiant_fog_160_test.csv') # Examine the first few rows of each dataset to understand their structure (audio_train.head(), video_train.head()) Result (filename start_timestep \ 0 /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav 0.0 1 /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav 1.0 2 /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav 2.0 3 /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav 3.0 4 /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav 4.0 end_timestep arousal valence features x_0 x_1 x_2 \ 0 0.99 0.069682 0.000000 NaN -0.072897 0.002067 0.171740 1 1.99 0.289680 0.315083 NaN -0.076561 0.011748 0.031308 2 2.99 0.418364 0.416254 NaN -0.009226 0.002203 -0.032764 3 3.99 0.393300 0.285306 NaN -0.037265 -0.007950 0.089631 4 4.99 0.256235 0.215973 NaN -0.019666 0.024242 -0.020480 x_3 ... x_758 x_759 x_760 x_761 x_762 x_763 \ 0 -0.039302 ... 0.139346 -0.054141 -0.112815 0.280597 -0.293145 0.000861 1 -0.077866 ... 0.265668 -0.055287 -0.020320 0.442153 -0.263152 -0.006163 2 -0.067221 ... 0.236319 -0.056110 -0.053254 0.394420 -0.360009 0.001394 3 -0.055752 ... 0.194469 -0.052152 -0.093884 0.356597 -0.207222 -0.006739 4 -0.037980 ... 0.192281 -0.054829 -0.057758 0.173721 -0.196372 0.002582 x_764 x_765 x_766 x_767 0 -0.012984 -0.122969 -0.000401 -0.110178 1 -0.011856 0.014565 -0.008225 -0.080227 2 -0.012047 -0.105393 -0.019488 -0.054505 3 -0.011744 -0.240672 -0.015567 -0.036216 4 -0.015311 -0.069752 0.006805 -0.009604 [5 rows x 774 columns], path timestamp arousal \ 0 /work/home/dsu/Datasets/SEWA/preprocessed/SEW1... 2.04 0.294194 1 /work/home/dsu/Datasets/SEWA/preprocessed/SEW1... 2.38 0.335762 2 /work/home/dsu/Datasets/SEWA/preprocessed/SEW1... 2.72 0.395458 3 /work/home/dsu/Datasets/SEWA/preprocessed/SEW1... 3.06 0.419050 4 /work/home/dsu/Datasets/SEWA/preprocessed/SEW1... 3.40 0.420003 valence emb_0 emb_1 emb_2 emb_3 emb_4 emb_5 ... \ 0 0.330165 0.000092 -0.000697 0.078171

-0.000050 -0.000146 -0.533671 ... 1 0.408600 0.000097 -0.000782 0.025183 0.001174 0.000013 -0.032633 ... 2 0.414567 0.000091 -0.000815 0.010308 0.000997 0.000059 -0.063469 ... 3 0.415242 0.000097 -0.000853 -0.011648 0.000785 0.000045 0.147162 ... 4 0.386110 0.000099 -0.000661 0.155788 -0.000450 -0.000002 0.000105 ... emb_246 emb_247 emb_248 emb_249 emb_250 emb_251 emb_252 \ 0 0.000312 0.000625 0.000035 0.654218 -0.996944 0.000028 0.001518 1 0.000435 0.000334 0.000035 0.516561 -0.995212 0.000020 0.001552 2 0.000373 0.000218 0.000035 0.536721 -0.985959 0.000021 0.001660 3 0.000411 0.000229 0.000031 -0.325153 -0.989091 0.000016 0.002085 4 0.000353 0.000346 0.000033 0.407132 -0.984939 0.000022 0.001722 emb_253 emb_254 emb_255 0 0.000032 -0.687274 -0.000585 1 0.000063 -0.833125 -0.000759 2 0.000070 -0.763110 -0.000735 3 0.000058 -0.876193 -0.000657 4 0.000053 -0.860464 -0.000594 [5 rows x 260 columns]) The datasets for both audio and video have been loaded successfully, and their structures are as follows: Audio Dataset filename: Path to the audio file. start_timestep and end_timestep: Indicate the start and end times of the audio segment in seconds. arousal and valence: Emotional annotations for the segment. features (seems to be empty) and x_0, x_1, ..., x_767: Audio features extracted (768 features in total). Video Dataset path: Path to the video file. timestamp: The timestamp of the video segment in seconds. arousal and valence: Emotional annotations for the segment. emb_0, emb_1, ..., emb_255: Video features extracted (256 features in total).

The reason why i am uploading these files to you is because i need to synchronize my datasets, because i am planning to do a fusion of audio and video later (does not concern you).

Idea:
- Write a DataLoader (https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#creating-a-custom-dataset-for-your-files) , which takes files (audio and video) as input and then chops them into windows
- The chopped windows are stored inside the DataLoader in the RAM
- and outputs them using def __getitem__(self, idx) (see the article I sent the link to)

Example of a DataLoader i did for video data only:

```
# Custom Dataset
class CustomVideoDataset(Dataset):
        def __init__(self, df, window_size=10, stride=5):
        self.df = df
        self.df['arousal'] = self.df['arousal']
        #print(df)
        self.df['valence'] = self.df['valence']
        self.window_size = window_size
        self.stride = stride
        self.video_windows, self.labels_windows = self.prepare_windows()

        def __len__(self):
        return len(self.video_windows)

        def __getitem__(self, idx):
        window_frames = self.video_windows[idx]
```

```
        embeddings = [self.df.loc[self.df['path'] == frame, self.df.columns[3:259]].values for
frame in window_frames]
        frames_tensor = torch.tensor(embeddings, dtype=torch.float32).squeeze(1)

        labels = self.labels_windows[idx]
        labels_tensor = torch.tensor(labels, dtype=torch.float32)

        return frames_tensor, labels_tensor

        def prepare_windows(self):
        video_frames = {}
        labels = {}
        for _, row in self.df.iterrows():
        video_id = self.extract_video_info(row['path'])
        if video_id not in video_frames:
                video_frames[video_id] = []
                labels[video_id] = []
        video_frames[video_id].append(row['path'])
        labels[video_id].append((row['arousal'], row['valence']))

        video_windows = []
        labels_windows = []
        for video_id in video_frames:
        frames = video_frames[video_id]
        label_vals = labels[video_id]
        for i in range(0, len(frames) - self.window_size + 1, self.stride):
                video_windows.append(frames[i:i + self.window_size])
                window_labels = label_vals[i:i + self.window_size]
                avg_arousal = sum([label[0] for label in window_labels]) / len(window_labels)
                avg_valence = sum([label[1] for label in window_labels]) / len(window_labels)
                labels_windows.append((avg_arousal, avg_valence))

        return video_windows, labels_windows

        def extract_video_info(self, file_path):
        parts = file_path.split('/')
        video_id = parts[-2]
        return video_id
```

Important notice:
- i don't want you to create a completely new dataset with synchronized data, i want a DataLoader which i can “reuse” in my neural network later on (not right now)
- i don't want you to do the synchronization for me where you create some new files, cause my data sets are too large and also i just want a code for the DataLoader

There are no windows at all in the video data right now, there are frame-by-frame features there.

I should try to take different number of seconds of window slicing, so it is easier to make this value (window length) dynamic and DataLoader slices accordingly with some window length I set. I have to set it in seconds, NOT in number of frames.
Since the window length is dynamic, I need to somehow determine how many frames of ONE video can fit in the window length (e.g 2 seconds). This can be done, for example, in a video dataset i try to see how many corresponding frames there from a to b (a and b are some numbers, e.g i take SEW2117 and see how many frames there at 0 to 2, so start timestep is 0 and end timestep is 2 and check the amount of frames at 0 till 2 seconds), I do it several times with differents second, e.g the next time i can take 1 till 3 seconds and check how many frames fit etc.And after several times i take the maximum amount of frames in the defined window length. In audio, one time checking (e.g for 0 to 2 seconds) is enough, as there are no skips in audio. For example, for a window length of 2 seconds, that's 6 frames in video and 2 lines in audio.
Why do I need to do this?
Look, I go my window length, say 2 seconds, that window may sometimes include not 6 frames but 5, 4 or 3. What do I do now?
- If the window includes 3 frames or less, then I throw out that particular window and its audio too. So it's like I'm throwing out that time period, both in the video and in the audio.
- If I have 4 frames in the window, I have to duplicate the last line twice, then I get 6 frames. If I have 5 frames in the window, I duplicate the last line once to get 6 frames.
- If there are 6 frames at once, we don't duplicate anything.

Example: I am looking at the data "SEWA_radiant_fog_160_dev.csv" and my window length is for example 2 seconds. I examine 1 to 3 seconds and see that there are 4 frames (in this example I have excluded other features except timestep, but they are of course important!) : /work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_1_7.png,1.7, /work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_04.png,2.04,/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_38.png,2.38,/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_72.png,2.72. The maximum number of frames in a window length of 2 seconds is 6 frames according to my calculations. Therefore, I take the last line of /work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_72_72.png,2.72 and duplicate it twice to make it 6 frames.
As you can see, it's very important to have the same number of elements in the window.

What's also important:
I have all the videos in my video data at once in one file, so it's important to make sure each video is separate. In Path you can select the name of the video, for example SEW2117, I allocate it in a separate column and create a dictionary, in it the key is the name of the video, and the value is the frames corresponding to this video, sorted by timestep. I do the same thing with audio: in the dictionary I take unique video names (keys) and extract features for a particular video. That is, I look at video 2117, extract audio and video features, slice each one as described earlier and then save the sliced windows. This is what I do for EVERY video!

What about arousal and valence:
I take the last state in the video. For example: I am looking at the data "SEWA_radiant_fog_160_dev.csv" and my window length is for example 2 seconds. I examine 2 to 4 seconds. This corresponds to the lines (in this example I have excluded the

other features, but they are of course important!) :
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_04.png,2.04,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_38.png,2.38,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_72. png,2.72,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_06.png,3.06,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_4.png,3.4,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_74. png,3.74. So we take the arousal and valence of the last line, i.e. timestamp 3.74. Arousal and valence are visible in the line immediately after the timestamp (path,timestamp,arousal,valence,emb_0,...) and in THIS case arousal = 0.2173389999999999, valence = 0.2938926.
Basically, I take arousal and valence from video, but synchronise with audio on features too.

What is also important about DataLoader: it is better that DataLoader gives not one array (audio and video), but two: let's say I cut into windows and make sure that these windows correspond to each other by index, then when I get a random index, for example 125 (example does not correspond to reality), then this index corresponds to the 125th window in my array, which I cut into windows and this window is (for example) from 4 to 6 seconds. Then we take from video 4 to 6 seconds and from audio also 4 to 6 seconds. As a result I output two arrays: in list, or tuple and labels separately, i.e. on output I have tuple/list of size 3 and inside it (list) I have: (video features, audio features, labels).

Example sequence for a video, for example how we can slice a SEW2117 video (from SEWA_features_wav2vec_1_seconds_dev.csv and SEWA_radiant_fog_160_dev.csv) :
1) I form a window of 2 seconds (this variable is dynamic, but in this example the window length is 2 seconds)
2) I consider from zero to the last timestep in the interval of 2 seconds (since the window is 2)
This turns out in audio : start_timestep = 0 and end_timestep = 2
3) I filter frames in video from zero to two: that is 0<timestep<2 (timestep greater than zero but less than two). We see that only frame 1.7 fits (/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_1_7.png,1.7,) , i.e. only one frame in the video. In audio I got two lines where start_timestep is greater than or equal to zero and end_timestep is less than or equal to two. These two lines are output:
/work/home/dsu/Datasets/SEWA/audio/SEW2117.wav,0.0,0.99,
/work/home/dsu/Datasets/SEWA/audio/SEW2117.wav,1.0,1.99.
4) See what we got: 2 lines in audio, 1 frame in video. Since one frame is too small, we need always 4 or more, so we get rid of this frame, i.e. just ignore it. How we decide what to keep and what to ignore is:”Look, I go my window length, say 2 seconds, that window may sometimes include not 6 (you have to find out the maximum amount of frames in the window length which you have set in the beginning, so basically if you “walk” through your time interval (window length) and see the amount of frames there, then take a step, e.g one second and see the next time interval and see how many frames are there, you need to find out the maximum of frames in this time interval from which you can decide if you need to duplicate frames or not; with the audio you dont need to go through all the data, you can check once and its enough) frames but 5, 4 or 3. What I do now:
- If the window includes 3 frames or less, then I throw out that particular window and its audio too. So it's like I'm throwing out that time period, both in the video and in the audio.

- If I have 4 frames in the window, I have to duplicate the last line twice, then I get 6 frames. If I have 5 frames in the window, I duplicate the last line once to get 6 frames.
- If there are 6 frames at once, we don't duplicate anything.”
5)Let's go one step further. My one step is half a window. So if the window length is 2 seconds, my step is 1 second. (If the window is 4 seconds long, then my step is 2 seconds, and so on).
6)If my step is one second, we now consider 1 to 3 seconds and do the same as before. That is, looking at the number of lines of audio and frames in the video. Here I have two lines of audio and 4 frames in the video. Since the frames are 4 and the maximum is 6, I duplicate the last line in the video in this gap twice to make it 6 frames and save.
7) I do this until the end of video 2117.
8) And this is what I do with every video in my data!
What about arousal and valence:
I take the last state in the video. For example: I am looking at the data "SEWA_radiant_fog_160_dev.csv" and my window length is for example 2 seconds. I examine 2 to 4 seconds. This corresponds to the lines (in this example I have excluded the other features, but they are of course important!) :
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_04.png,2.04,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_38.png,2.38,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_72. png,2.72,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_06.png,3.06,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_4.png,3.4,
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_3_74. png,3.74. So we take the arousal and valence of the last line, i.e. timestamp 3.74. Arousal and valence are visible in the line immediately after the timestamp (path,timestamp,arousal,valence,emb_0,...) and in THIS case arousal = 0.2173389999999999, valence = 0.2938926.
Basically, I take arousal and valence from video, but synchronise with audio on features too.

Important points:
- dynamic variable window length that I can change
- step is half of the window
- keep an eye on timesteps so that there is no out of sync!
-arousal and valence I take from the last state in the video (the last line in the video is in my seconds interval, I explained this earlier)

New text:

i have video data sets and the audios of the same videos as audio datasets :SEWA_radiant_fog_160_dev, SEWA_radiant_fog_160_train and SEWA_radiant_fog_160_test (videos) and SEWA_features_wav2vec_1_seconds_dev, SEWA_features_wav2vec_1_seconds_train, SEWA_features_wav2vec_1_seconds_test (audio).

Here is their structure (the first few rows of each dataset were examined):

```
(                                  filename  start_timestep  \
 0  /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav            0.0
 1  /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav            1.0
 2  /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav            2.0
 3  /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav            3.0
 4  /work/home/dsu/Datasets/SEWA/audio/SEW1101.wav            4.0

   end_timestep   arousal   valence  features       x_0       x_1       x_2  \
0          0.99  0.069682  0.000000       NaN -0.072897  0.002067  0.171740
1          1.99  0.289680  0.315083       NaN -0.076561  0.011748  0.031308
2          2.99  0.418364  0.416254       NaN -0.009226  0.002203 -0.032764
3          3.99  0.393300  0.285306       NaN -0.037265 -0.007950  0.089631
4          4.99  0.256235  0.215973       NaN -0.019666  0.024242 -0.020480

       x_3  ...     x_758     x_759     x_760     x_761     x_762     x_763  \
0 -0.039302  ...  0.139346 -0.054141 -0.112815  0.280597 -0.293145  0.000861
1 -0.077866  ...  0.265668 -0.055287 -0.020320  0.442153 -0.263152 -0.006163
2 -0.067221  ...  0.236319 -0.056110 -0.053254  0.394420 -0.360009  0.001394
3 -0.055752  ...  0.194469 -0.052152 -0.093884  0.356597 -0.207222 -0.006739
4 -0.037980  ...  0.192281 -0.054829 -0.057758  0.173721 -0.196372  0.002582

      x_764     x_765     x_766     x_767
0 -0.012984 -0.122969 -0.000401 -0.110178
1 -0.011856  0.014565 -0.008225 -0.080227
2 -0.012047 -0.105393 -0.019488 -0.054505
3 -0.011744 -0.240672 -0.015567 -0.036216
4 -0.015311 -0.069752  0.006805 -0.009604

[5 rows x 774 columns],
                                        path  timestamp   arousal  \
0  /work/home/dsu/Datasets/SEWA/preprocessed/SEW1...       2.04  0.294194
1  /work/home/dsu/Datasets/SEWA/preprocessed/SEW1...       2.38  0.335762
2  /work/home/dsu/Datasets/SEWA/preprocessed/SEW1...       2.72  0.395458
3  /work/home/dsu/Datasets/SEWA/preprocessed/SEW1...       3.06  0.419050
4  /work/home/dsu/Datasets/SEWA/preprocessed/SEW1...       3.40  0.420003

    valence     emb_0     emb_1     emb_2     emb_3     emb_4     emb_5  ...  \
0  0.330165  0.000092 -0.000697  0.078171 -0.000050 -0.000146 -0.533671  ...
1  0.408600  0.000097 -0.000782  0.025183  0.001174  0.000013 -0.032633  ...
```

```
2  0.414567  0.000091 -0.000815  0.010308  0.000997  0.000059 -0.063469  ...
3  0.415242  0.000097 -0.000853 -0.011648  0.000785  0.000045  0.147162  ...
4  0.386110  0.000099 -0.000661  0.155788 -0.000450 -0.000002  0.000105  ...

    emb_246   emb_247   emb_248   emb_249   emb_250   emb_251   emb_252  \
0  0.000312  0.000625  0.000035  0.654218 -0.996944  0.000028  0.001518
1  0.000435  0.000334  0.000035  0.516561 -0.995212  0.000020  0.001552
2  0.000373  0.000218  0.000035  0.536721 -0.985959  0.000021  0.001660
3  0.000411  0.000229  0.000031 -0.325153 -0.989091  0.000016  0.002085
4  0.000353  0.000346  0.000033  0.407132 -0.984939  0.000022  0.001722

    emb_253   emb_254   emb_255
0  0.000032 -0.687274 -0.000585
1  0.000063 -0.833125 -0.000759
2  0.000070 -0.763110 -0.000735
3  0.000058 -0.876193 -0.000657
4  0.000053 -0.860464 -0.000594

[5 rows x 260 columns])
```

Audio Dataset

- **filename**: Path to the audio file.
- **start_timestep** and **end_timestep**: Indicate the start and end times of the audio segment in seconds.
- **arousal** and **valence**: Emotional annotations for the segment.
- **features** (seems to be empty) and **x_0, x_1, ..., x_767**: Audio features extracted (768 features in total).

Video Dataset

- **path**: Path to the video file.
- **timestamp**: The timestamp of the video segment in seconds.
- **arousal** and **valence**: Emotional annotations for the segment.
- **emb_0, emb_1, ..., emb_255**: Video features extracted (256 features in total).

I need to create a custom DataLoader (https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#creating-a-custom-dataset-for-your-files import os
import pandas as pd
from torchvision.io import read_image

```
class CustomImageDataset(Dataset):
    def __init__(self, annotations_file, img_dir, transform=None, target_transform=None):
        self.img_labels = pd.read_csv(annotations_file)
        self.img_dir = img_dir
        self.transform = transform
```

```
        self.target_transform = target_transform

    def __len__(self):
        return len(self.img_labels)

    def __getitem__(self, idx):
        img_path = os.path.join(self.img_dir, self.img_labels.iloc[idx, 0])
        image = read_image(img_path)
        label = self.img_labels.iloc[idx, 1]
        if self.transform:
            image = self.transform(image)
        if self.target_transform:
            label = self.target_transform(label)
        return image, label
```

__init__

The __init__ function is run once when instantiating the Dataset object. We initialize the directory containing the images, the annotations file, and both transforms (covered in more detail in the next section).

The labels.csv file looks like:

```
tshirt1.jpg, 0
tshirt2.jpg, 0
......
ankleboot999.jpg, 9
```

```
def __init__(self, annotations_file, img_dir, transform=None, target_transform=None):
    self.img_labels = pd.read_csv(annotations_file)
    self.img_dir = img_dir
    self.transform = transform
    self.target_transform = target_transform
```

__len__

The __len__ function returns the number of samples in our dataset.

Example:

```
def __len__(self):
    return len(self.img_labels)
```

__getitem__

The __getitem__ function loads and returns a sample from the dataset at the given index `idx`. Based on the index, it identifies the image's location on disk, converts that to a tensor using `read_image`, retrieves the corresponding label from the csv data in `self.img_labels`, calls the transform functions on them (if applicable), and returns the tensor image and corresponding label in a tuple.

```
def __getitem__(self, idx):
    img_path = os.path.join(self.img_dir, self.img_labels.iloc[idx, 0])
    image = read_image(img_path)
    label = self.img_labels.iloc[idx, 1]
    if self.transform:
        image = self.transform(image)
    if self.target_transform:
        label = self.target_transform(label)
    return image, label )
```

that synchronizes audio and video data.

I have done a custom DataLoader in the past but for video data only, you can use it as a guide "# Custom Dataset

```
class CustomVideoDataset(Dataset):
        def __init__(self, df, window_size=10, stride=5):
        self.df = df
        self.df['arousal'] = self.df['arousal']
        #print(df)
        self.df['valence'] = self.df['valence']
        self.window_size = window_size
        self.stride = stride
        self.video_windows, self.labels_windows = self.prepare_windows()

        def __len__(self):
        return len(self.video_windows)

        def __getitem__(self, idx):
        window_frames = self.video_windows[idx]
        embeddings = [self.df.loc[self.df['path'] == frame, self.df.columns[3:259]].values for
frame in window_frames]
        frames_tensor = torch.tensor(embeddings, dtype=torch.float32).squeeze(1)

        labels = self.labels_windows[idx]
        labels_tensor = torch.tensor(labels, dtype=torch.float32)

        return frames_tensor, labels_tensor

        def prepare_windows(self):
        video_frames = {}
        labels = {}
        for _, row in self.df.iterrows():
```

```
        video_id = self.extract_video_info(row['path'])
        if video_id not in video_frames:
            video_frames[video_id] = []
            labels[video_id] = []
        video_frames[video_id].append(row['path'])
        labels[video_id].append((row['arousal'], row['valence']))

        video_windows = []
        labels_windows = []
        for video_id in video_frames:
        frames = video_frames[video_id]
        label_vals = labels[video_id]
        for i in range(0, len(frames) - self.window_size + 1, self.stride):
            video_windows.append(frames[i:i + self.window_size])
            window_labels = label_vals[i:i + self.window_size]
            avg_arousal = sum([label[0] for label in window_labels]) / len(window_labels)
            avg_valence = sum([label[1] for label in window_labels]) / len(window_labels)
            labels_windows.append((avg_arousal, avg_valence))

        return video_windows, labels_windows

        def extract_video_info(self, file_path):
        parts = file_path.split('/')
        video_id = parts[-2]
        return video_id

# Load data
train_df = pd.read_csv('SEWA_radiant_fog_160_train.csv')
dev_df = pd.read_csv('SEWA_radiant_fog_160_dev.csv')
test_df = pd.read_csv('SEWA_radiant_fog_160_test.csv')

# Hyperparameters
window_size = 5
input_size = 256  # Number of features (embeddings) per frame
hidden_size = 128  # Number of features in hidden state of GRU
output_size = 2  # Output size (arousal and valence)
num_layers = 2  # Number of layers
learning_rate = 0.01te
batch_size = 32
epochs = 100

# Create datasets and dataloaders
train_dataset = CustomVideoDataset(train_df, window_size)
dev_dataset = CustomVideoDataset(dev_df, window_size)
test_dataset = CustomVideoDataset(test_df, window_size)

train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
dev_loader = DataLoader(dev_dataset, batch_size=batch_size, shuffle=False)
```

test_loader = DataLoader(test_dataset, batch_size=batch_size, shuffle=False)".

The idea is:

- define a dynamic window_length parameter
- considering the window_length that was set in the beginning start synchronizing the datasets like this:
    1) start by finding out how many frames (amount of timestamps) fit in the time interval. The first time interval begins always with 0 and ends at window_length, e.g start_interval = 0 sec, end_interval = 2 sec in the beginning. Look how many video frames fit there? Remember it, then take a step (step = half of window_length, here step = 1 sec) to the next time interval which would be start_interval = 1 sec, end_interval = 3 sec. Count how many frames (amount of timestamps) fit in this time interval? Remember it again.
       Obviously, you have to look at the timestamps of the same video.
       Example: we look at the video SEW2117 for the time interval with start_interval = 0 sec, end_interval = 2 sec and look what timestamps would fit in there:
       "path,timestamp,arousal,valence,emb_0,emb_1,emb_2,emb_3,emb_4
       ,emb_5,emb_6,emb_7,emb_8,emb_9,emb_10,emb_11,emb_12,emb_
       13,emb_14,emb_15,emb_16,emb_17,emb_18,emb_19,emb_20,emb_
       21,emb_22,emb_23,emb_24,emb_25,emb_26,emb_27,emb_28,emb_
       29,emb_30,emb_31,emb_32,emb_33,emb_34,emb_35,emb_36,emb_
       37,emb_38,emb_39,emb_40,emb_41,emb_42,emb_43,emb_44,emb_
       45,emb_46,emb_47,emb_48,emb_49,emb_50,emb_51,emb_52,emb_
       53,emb_54,emb_55,emb_56,emb_57,emb_58,emb_59,emb_60,emb_
       61,emb_62,emb_63,emb_64,emb_65,emb_66,emb_67,emb_68,emb_
       69,emb_70,emb_71,emb_72,emb_73,emb_74,emb_75,emb_76,emb_
       77,emb_78,emb_79,emb_80,emb_81,emb_82,emb_83,emb_84,emb_
       85,emb_86,emb_87,emb_88,emb_89,emb_90,emb_91,emb_92,emb_
       93,emb_94,emb_95,emb_96,emb_97,emb_98,emb_99,emb_100,emb
       _101,emb_102,emb_103,emb_104,emb_105,emb_106,emb_107,emb
       _108,emb_109,emb_110,emb_111,emb_112,emb_113,emb_114,emb
       _115,emb_116,emb_117,emb_118,emb_119,emb_120,emb_121,emb
       _122,emb_123,emb_124,emb_125,emb_126,emb_127,emb_128,emb
       _129,emb_130,emb_131,emb_132,emb_133,emb_134,emb_135,emb
       _136,emb_137,emb_138,emb_139,emb_140,emb_141,emb_142,emb
       _143,emb_144,emb_145,emb_146,emb_147,emb_148,emb_149,emb
       _150,emb_151,emb_152,emb_153,emb_154,emb_155,emb_156,emb
       _157,emb_158,emb_159,emb_160,emb_161,emb_162,emb_163,emb
       _164,emb_165,emb_166,emb_167,emb_168,emb_169,emb_170,emb
       _171,emb_172,emb_173,emb_174,emb_175,emb_176,emb_177,emb
       _178,emb_179,emb_180,emb_181,emb_182,emb_183,emb_184,emb
       _185,emb_186,emb_187,emb_188,emb_189,emb_190,emb_191,emb
       _192,emb_193,emb_194,emb_195,emb_196,emb_197,emb_198,emb
       _199,emb_200,emb_201,emb_202,emb_203,emb_204,emb_205,emb
       _206,emb_207,emb_208,emb_209,emb_210,emb_211,emb_212,emb
       _213,emb_214,emb_215,emb_216,emb_217,emb_218,emb_219,emb

_220,emb_221,emb_222,emb_223,emb_224,emb_225,emb_226,emb
_227,emb_228,emb_229,emb_230,emb_231,emb_232,emb_233,emb
_234,emb_235,emb_236,emb_237,emb_238,emb_239,emb_240,emb
_241,emb_242,emb_243,emb_244,emb_245,emb_246,emb_247,emb
_248,emb_249,emb_250,emb_251,emb_252,emb_253,emb_254,emb
_255
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_1_7.png,1.
7,0.031651,0.02683,8.278729364974424e-05,-0.0010977666825056
076,0.11482857912778854,0.0037840548902750015,0.00059905787
92989254,0.8374770283699036,-0.00436581252142787,0.00056470
18551826477,5.1805702241836116e-05,0.9937756061553955,0.000
11138235747930 5938,-0.9758582711219788,0.6578124761581421,0.
003843883750960231,0.0002830072771757841,0.000610832590609
7889,-0.36042481660842896,0.7443380951881409,-0.99963802099
2279,0.8587721586227417,0.0005427891737781465,-0.0752624198
7943649,1.4582237781723961e-05,-7.81839553383179e-05,-0.9118
15345287323,0.007516421843320131,0.0002428362931823358,-7.1
6923241270706e-05,0.07104001939296722,-0.00098027486819773
91,-0.006544844713062048,-0.01787266507744789,-1.68131664395
33234e-05,-0.00011111972708022222,-5.785585744888522e-05,0.00
3305444261059165,-0.018848085775971413,0.24273565411567688,
-4.773588079842739e-05,-8.310743578476831e-05,-0.00069519155
65878153,-0.008282341994345188,-0.14291632175445557,-0.00053
3070124220103,0.37671759724617004,0.7305417060852051,0.0003
3724543754942715,-1.914680797199253e-05,0.0006465002661570
907,-0.0001513697934569791,-0.9086379408836365,2.6542815248
831175e-05,-0.00016315400716848671,0.004204676952213049,0.33
9608758687973,-0.00016994673933368176,0.000205526317586191,
0.0001393527927575633,-0.0017512073973193765,-0.00385362468
65987778,-4.846405136049725e-05,-0.0004707049811258912,2.554
760612838436e-05,0.8654772043228149,0.0001543979742564261,-
0.002122943988069892,5.91152893321123e-05,0.000217462453292
68277,-0.000575572659727186,-0.0001459687773603946,9.245940
600521863e-05,6.752640911145136e-05,0.0001586063444847241,3.
059744994970 97e-05,-0.007541518192738295,-0.000623934203758
8358,-0.0009676724439486861,0.0004352473479229957,-0.123052
30647325516,0.9730011224746704,-0.0032401876524090767,-0.005
157123319804668,-0.000409907748689875,0.000358279125066474
1,-0.0009944421472027898,0.0006803263095207512,-0.001732153
119519353,-0.0006785026635043323,-0.0008710701949894428,-0.0
003579272888600826,0.0003747049777302891,-0.64299345016479
49,-0.9878367185592651,-0.07663071900606155,-0.0327868163585
66284,-5.805886030429974e-05,0.0039597260765731335,-0.001848
259475082159,-0.2442782074213028,0.0001524997060187161,-0.0
035669992212206125,-0.9949198365211487,0.07461810111999512,
0.0002057302772300318,0.0005184652982279658,0.458370149135
5896,-0.0017859520157799125,-0.0004683601663820446,5.16318
5051060282e-06,-8.264321877504699e-06,-0.000943227612879127

3,0.00024892069632187486,0.005542142782360315,-0.0008537991
670891643,0.0001463885128032416,-0.9289689064025879,4.37577
5461085141e-05,-0.5296065807342529,0.0011079948162660003,-0.
0004401896730996668 3,-0.0007649569888599217,0.630681633949
2798,5.562193473451771e-05,-0.0003677310887724161,0.3886303
0076026917,0.006448973901569843,-0.0006473723915405571,0.00
020941039838362485,-0.00022693035134579986,-0.000209699835
97751707,-0.265739768743515,0.00995600875467062,0.992443859
577179,-0.000406514824135229,0.0060124178417027,-0.00208992
09193885326,0.0001272594527108594 8,-0.0007242285646498203,3
.815854142885655e-05,0.00024405935255344957,-0.001307032187
4693036,0.7638520002365112,5.0458958867238835e-05,-1.5592946
738252067e-06,-0.0008716300944797695,0.004084752406924963,-
0.5338962078094482,0.8930138349533081,-0.000251798948738724
,0.26396095752716064,-0.8086360692977905,0.9713665246963501,
-0.0003721569664776325,-0.00011921324039576575,0.0051272679
1203022,-0.055645305663347244,-0.9751937389373779,-0.0001089
6651656366885,-0.04712747782468796,0.8771294355392456,8.078
714017756283e-05,-0.004462333396077156,0.9990233182907104,9
.935129492077976e-05,-0.0006915362318977714,0.0353598222136
4975,0.003928466234356165,-0.999972403049469,-0.00016604764
095973223,-0.058480966836214066,0.9020557403564453,0.000230
876132263802,0.18530526757240295,-1.96458813661593e-06,-0.00
01270950451726094,-0.0002456571673974395,0.000462442694697
5291,-0.006134109105914831,2.7213154680794105e-05,-0.3034713
2682800293,-0.019075235351920128,-0.0001366795186186 2093,-0.
42970338463783264,3.3799959055613726e-05,0.000829884258564
5616,0.002682124963030219,5.2945442803320475e-06,-0.0059436
62021309137,-0.8987016677856445,0.9609013199806213,0.035220
712423324585,1.761554813128896e-05,0.0021499397698789835,0.
0002688801905605942,-0.0003323204873595387,0.0001828561071
306467,-0.3157676160335541,0.0015420590061694384,0.00131766
137201339,-0.01707097329199314,-0.0018446403555572033,3.065
4533475171775e-05,0.0002770031278487295,-0.001840127282775
9385,0.00512478593736887,-0.0002804587420541793,5.292212881
613523e-05,-0.00155169190838933,-0.0005928021273575723,-0.00
012880931899417192,0.0010238387621939182,0.001071811188012
3615,0.0017393792513757944,0.8301116824150085,0.97938913106
91833,-0.0023352152202278376,-6.500077142845839e-05,0.990756
2732696533,0.0015590167604386806,0.000690101645886898,-0.00
07636012160219252,0.7048553228378296,-4.4713393435813487e-
05,-0.06449655443429947,-0.07741118967533112,-0.004816698376
0893345,2.086353015329223e-05,-9.465600669500418e-06,0.00068
30165511928499,-0.0145809231325984,-0.03337850049138069,-7.4
66902025043964e-05,-0.12793982028961182,0.00015560787869617
343,5.664325362886302e-05,-0.9555206894874573,3.08244743791
8745e-05,0.0015334758209064603,3.970272155129351e-05,-0.0007
817916921339929,-0.0004188127350062132,-0.0098328199237585

07,-9.595917799742892e-05,4.789823287865147e-05,-3.118584936
60111e-05,-6.015351573296357e-06,0.0005138692795298994,-0.000
921993691008538,1.6700001651770435e-05,-0.9973419904708862,
-0.9882665276527405,-3.889846630045213e-05,0.00334721966646
6117,5.62912791792769e-05,-0.9690662026405334,-0.00061841052
95687914
/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_04.png,2
.04,0.0736208,0.0283564,7.71351478761062e-05,-0.0011207509087
398648,-0.10121594369411469,0.0027470351196825504,0.0002140
138385584578,0.5981841683387756,-0.003936385735869408,0.000
4648219037335366,4.833507045987062e-05,0.948082447052002,0.
0002503063296899195,-0.9956846833229065,0.515105128288269,
0.0029766298830509186,-4.288414857001044e-05,0.000288020964
94473517,-0.49031707644462585,-0.4452141225337982,-0.9993028
64074707,0.7616323828697205,0.00014176100376062095,-0.08119
379729032516,1.4326976270240266e-05,-5.61704182473477e-05,-0
.8316960334777832,0.0069367303512990475,0.0002443659759592
265,1.764145963534247e-05,0.0662611871957779,-0.000661476631
6488385,-0.004216812085360289,-0.011490555480122566,-1.62210
053531453e-05,-9.876208059722558e-05,-5.2927385695511475e-05
,0.0020939602982252836,-0.014973379671573639,-0.13823297619
81964,-2.5459723474341445e-05,-5.758433690061793e-05,-0.00063
47499438561499,-0.006641410756856203,-0.10865817964076996,0
.0004588663286995142 7,0.546251118183136,0.7473748922348022,
0.00027179220342077315,0.00040805444587022066,0.0005265942
309051752,-0.0001450455456506461,-0.9223670363426208,2.1699
819626519457e-05,0.0006689591682516038,0.00451615126803517
3,-0.4527413845062256,-5.322559081832878e-05,0.0001368392695
4399794,0.0001647298195166513 3,-0.0012810503831133246,-0.003
7980065681040287,-5.056470399722457e-05,-0.0006331938784569
502,0.0001234171650139615,0.8623743057250977,0.000169375023
68818969,-0.0030115193221718073,7.599550008308142e-05,0.0001
4811177970841527,-0.0005547169130295515,-9.399762348039076e
-05,4.6484819904435426e-05,8.293581777252257e-05,-3.09253700
8342333e-05,-7.680675480514765e-05,-0.008001931942999363,-0.0
0360839330824092 03,-0.0035748351365327835,0.0003644849057
3093295,-0.10240045189857483,0.9305885434150696,-0.00312183
8206425309,-0.004967025015503168,-0.0005425054114311934,0.00
035510846646502614,-0.0009745755232870579,0.00067887274781
24201,-0.001049402984790504,-0.0002084846782963723,0.001748
9265883341432,-0.023053839802742004,0.00010701285646064207,
-0.8268558382987976,-0.9624080061912537,-0.0736848786473274
2,-0.02899125963449478,-5.798719575977884e-05,0.002775915665
55202,-0.0017981083365157247,-0.2127171903848648,0.00014972
129429224879,-0.0031265923753380775,-0.9944584369659424,-0.3
184683918952942,0.00015722235548309982,0.00032888111309148
37,0.03986971825361252,-0.0013278444530442357,-0.00096971174
93487895,9.538554877508432e-05,-3.0564099233743036e-06,-0.00

08601606823503971,0.0002738270850386470 6,0.003502301406115
2935,-0.0008415366755798459,-6.989994290051982e-05,-0.874264
7171020508,5.409555888036266e-05,-0.5871296525001526,0.0016
578391660004854,-0.0002137144183507189 2,-0.000971373694483
1908,0.1951856017112732,3.882137389155105e-05,-0.00036565025
2206251,0.5118047595024109,0.00550199905410409,-0.003012416
884303093,0.0002913938369601965,-0.0003859482530970126 4,0.0
0014610996004194 02,-0.22086046636104584,0.0107525140047073
36,0.9890702366828918,-0.0004462547949515283,0.005585144273
9367485,-0.0023244081530719995,0.0001222122518811375,-0.0011
735664447769523,2.7580794267123565e-05,0.0002276098821312 1
89,-0.0011423461837694049,0.7111485004425049,-9.485982445767
15e-06,-1.1893998816958629e-06,-0.0009155413135886192,0.0038
089784793555737,-0.49217328429222107,0.8793644905090332,-0.
0002127206826116889 7,-0.0022458788007497787,-0.545851826667
7856,0.7067578434944153,-0.00029535655630752444,-0.000119297
28498216718,0.003255086485296488,-0.025400852784514427,-0.9
251227 37878845215,-5.4043153795646504e-05,-0.0155195081606507
3,0.8921995162963867,8.030949538806453e-05,-0.0065391906537
11557,0.9995057582855225,0.0001455976453144 1033,-0.00052325
34604147077,0.01671515963971615,0.003991701174527407,-0.999
9898672103882,-0.0001599721144884824 8,-0.09163244813680649,
0.8299311399459839,0.000219138921238482,0.1410105228424072
3,3.6982015444664285e-05,-0.0002283701614942401 6,-0.00020762
10075756535,0.0003895479603670537 5,-0.001714472658932209,1.
7238364307559095e-05,-0.17049574851989746,-0.01599297486245
632,-0.0001248079206561669 7,-0.6672534942626953,4.853609789
2792895e-05,0.0009492358658462763,0.002541068708524108,5.88
4397978661582e-06,-0.007444348651915789,-0.8823962211608887
,0.9566501379013062,0.060760434716939926,1.7009671864798293
e-05,0.0015732158208265 9,0.0002395315532339736 8,-0.00032819
854095578194,0.0003246333799324929 7,0.6120527982711792,0.00
148378713625930 2497,0.0014112988719716668,-0.012916255742311
478,-0.0016933055594563484,3.922565883840434e-05,0.00028877
120348624885,-0.0017154690576717257,0.003788077738136053,-0
.0002086744498228654 3,5.955526648904197e-05,-0.000312030140
7761872,-0.0005619070143438876,-0.0001764946937328 2045,0.00
08482660632580519,0.0012549491366371512,0.002049605129286 6
47,0.8856669664382935,0.9505980014801025,0.0029145211447030
306,-0.0001382905902573838 8,0.9698045253753662,0.0018052702
071145177,0.0006960382452234626,-0.0006022676243446767,0.67
82270669937134,-0.0001201002014568075 5,-0.0503715164959430
7,-0.37133198976516724,0.024325339123606682,2.1256199033814
482e-05,-1.3342359125090297e-05,0.000565420486964285 4,-0.015
69300889968872,-0.0390591509640216 8,-6.251609011087567e-05,
0.03661791235208511 4,0.0007767417700961232,4.8245685320580
38e-05,-0.9647300243377686,3.338125679874793e-05,0.001493934
541940689,1.9125598555547185e-05,-0.0007473729783669114,-0.0

007781252497807145,-0.011875340715050697,-9.037963900482282e-05,3.7242727557895705e-05,-3.4842625609599054e-05,-8.14145641925279e-06,0.0003941359755117446,-0.00032195745734497905,2.032137126661837e-05,-0.9948014616966248,-0.9734736680984497,-1.8062073650071397e-05,0.0029072677716612816,3.6435085348784924e-05,-0.9198500514030457,-0.0005124403396621346".
We see that it's the case only for timestamp = 1.7 . For the next time interval with start_interval = 1 sec, end_interval = 3 sec it would be the following time stamps (the 3 dots after the timestamp represent the rest of the features which i didn't include here, but they are of course there):
"/work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_1_7.png,1.7,.../work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_04.png,2.04,.../work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_38.png,2.38,.../work/home/dsu/Datasets/SEWA/preprocessed/SEW2117_2_72.png,2.72,...". Do it 5 times to find out the maximum amount of frames in one time interval for the window_length that has been set. Remember the maximal amount as a variable.

2) Now you again walk through the time interval for each video and decide which frames you keep or not based on the amount of frames in the time interval.
   Let's say the maximal amount of frames with timestamps that fit in the time interval is n=6.
   - If there are actually 6 frames in the time interval when you go through it, then keep it, along with the audio data that also fits in this time interval.
   - If there are n-2 (which would be in this example 4) frames in one of the time intervals, then duplicate the last line (last timestamp with the corresponding features) two times, to get the n amount of frames (here it would be 6) and then after that keep it along with the audio data that also fits in this time interval.
   - If there are n-1 (which would be in this example 5) frames in one of the time intervals, then duplicate the last timestamp with the corresponding features one time, to get the n amount of frames (here it would be 6) and then after that keep it along with the audio data that also fits in this time interval.
   - Everything that is smaller than n-1 (number_frames<n-2) you should ignore. Just ignore the whole line (timestamp with the corresponding features) and DON'T save it along with the audio data that also fits in this time interval.
   - so WHILE you are looking at video frames in each time interval, you are associating them with the corresponding audio data at the same time interval. That is the whole idea of synchronizing. Let me give

you an example: let’s say you only found one frame in video SEW2117 in time interval start_interval = 0 sec, end_interval = 2 sec for window_length = 2 sec and step = 1 sec. You look at the audio data for the same time interval
:”filename,start_timestep,end_timestep,arousal,valence,features,x_0,x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9,x_10,x_11,x_12,x_13,x_14,x_15,x_16,x_17,x_18,x_19,x_20,x_21,x_22,x_23,x_24,x_25,x_26,x_27,x_28,x_29,x_30,x_31,x_32,x_33,x_34,x_35,x_36,x_37,x_38,x_39,x_40,x_41,x_42,x_43,x_44,x_45,x_46,x_47,x_48,x_49,x_50,x_51,x_52,x_53,x_54,x_55,x_56,x_57,x_58,x_59,x_60,x_61,x_62,x_63,x_64,x_65,x_66,x_67,x_68,x_69,x_70,x_71,x_72,x_73,x_74,x_75,x_76,x_77,x_78,x_79,x_80,x_81,x_82,x_83,x_84,x_85,x_86,x_87,x_88,x_89,x_90,x_91,x_92,x_93,x_94,x_95,x_96,x_97,x_98,x_99,x_100,x_101,x_102,x_103,x_104,x_105,x_106,x_107,x_108,x_109,x_110,x_111,x_112,x_113,x_114,x_115,x_116,x_117,x_118,x_119,x_120,x_121,x_122,x_123,x_124,x_125,x_126,x_127,x_128,x_129,x_130,x_131,x_132,x_133,x_134,x_135,x_136,x_137,x_138,x_139,x_140,x_141,x_142,x_143,x_144,x_145,x_146,x_147,x_148,x_149,x_150,x_151,x_152,x_153,x_154,x_155,x_156,x_157,x_158,x_159,x_160,x_161,x_162,x_163,x_164,x_165,x_166,x_167,x_168,x_169,x_170,x_171,x_172,x_173,x_174,x_175,x_176,x_177,x_178,x_179,x_180,x_181,x_182,x_183,x_184,x_185,x_186,x_187,x_188,x_189,x_190,x_191,x_192,x_193,x_194,x_195,x_196,x_197,x_198,x_199,x_200,x_201,x_202,x_203,x_204,x_205,x_206,x_207,x_208,x_209,x_210,x_211,x_212,x_213,x_214,x_215,x_216,x_217,x_218,x_219,x_220,x_221,x_222,x_223,x_224,x_225,x_226,x_227,x_228,x_229,x_230,x_231,x_232,x_233,x_234,x_235,x_236,x_237,x_238,x_239,x_240,x_241,x_242,x_243,x_244,x_245,x_246,x_247,x_248,x_249,x_250,x_251,x_252,x_253,x_254,x_255,x_256,x_257,x_258,x_259,x_260,x_261,x_262,x_263,x_264,x_265,x_266,x_267,x_268,x_269,x_270,x_271,x_272,x_273,x_274,x_275,x_276,x_277,x_278,x_279,x_280,x_281,x_282,x_283,x_284,x_285,x_286,x_287,x_288,x_289,x_290,x_291,x_292,x_293,x_294,x_295,x_296,x_297,x_298,x_299,x_300,x_301,x_302,x_303,x_304,x_305,x_306,x_307,x_308,x_309,x_310,x_311,x_312,x_313,x_314,x_315,x_316,x_317,x_318,x_319,x_320,x_321,x_322,x_323,x_324,x_325,x_326,x_327,x_328,x_329,x_330,x_331,x_332,x_333,x_334,x_335,x_336,x_337,x_338,x_339,x_340,x_341,x_342,x_343,x_344,x_345,x_346,x_347,x_348,x_349,x_350,x_351,x_352,x_353,x_354,x_355,x_356,x_357,x_358,x_359,x_360,x_361,

x_362,x_363,x_364,x_365,x_366,x_367,x_368,x_369,x
_370,x_371,x_372,x_373,x_374,x_375,x_376,x_377,x_
378,x_379,x_380,x_381,x_382,x_383,x_384,x_385,x_3
86,x_387,x_388,x_389,x_390,x_391,x_392,x_393,x_39
4,x_395,x_396,x_397,x_398,x_399,x_400,x_401,x_402
,x_403,x_404,x_405,x_406,x_407,x_408,x_409,x_410,
x_411,x_412,x_413,x_414,x_415,x_416,x_417,x_418,x
_419,x_420,x_421,x_422,x_423,x_424,x_425,x_426,x_
427,x_428,x_429,x_430,x_431,x_432,x_433,x_434,x_4
35,x_436,x_437,x_438,x_439,x_440,x_441,x_442,x_44
3,x_444,x_445,x_446,x_447,x_448,x_449,x_450,x_451
,x_452,x_453,x_454,x_455,x_456,x_457,x_458,x_459,
x_460,x_461,x_462,x_463,x_464,x_465,x_466,x_467,x
_468,x_469,x_470,x_471,x_472,x_473,x_474,x_475,x_
476,x_477,x_478,x_479,x_480,x_481,x_482,x_483,x_4
84,x_485,x_486,x_487,x_488,x_489,x_490,x_491,x_49
2,x_493,x_494,x_495,x_496,x_497,x_498,x_499,x_500
,x_501,x_502,x_503,x_504,x_505,x_506,x_507,x_508,
x_509,x_510,x_511,x_512,x_513,x_514,x_515,x_516,x
_517,x_518,x_519,x_520,x_521,x_522,x_523,x_524,x_
525,x_526,x_527,x_528,x_529,x_530,x_531,x_532,x_5
33,x_534,x_535,x_536,x_537,x_538,x_539,x_540,x_54
1,x_542,x_543,x_544,x_545,x_546,x_547,x_548,x_549
,x_550,x_551,x_552,x_553,x_554,x_555,x_556,x_557,
x_558,x_559,x_560,x_561,x_562,x_563,x_564,x_565,x
_566,x_567,x_568,x_569,x_570,x_571,x_572,x_573,x_
574,x_575,x_576,x_577,x_578,x_579,x_580,x_581,x_5
82,x_583,x_584,x_585,x_586,x_587,x_588,x_589,x_59
0,x_591,x_592,x_593,x_594,x_595,x_596,x_597,x_598
,x_599,x_600,x_601,x_602,x_603,x_604,x_605,x_606,
x_607,x_608,x_609,x_610,x_611,x_612,x_613,x_614,x
_615,x_616,x_617,x_618,x_619,x_620,x_621,x_622,x_
623,x_624,x_625,x_626,x_627,x_628,x_629,x_630,x_6
31,x_632,x_633,x_634,x_635,x_636,x_637,x_638,x_63
9,x_640,x_641,x_642,x_643,x_644,x_645,x_646,x_647
,x_648,x_649,x_650,x_651,x_652,x_653,x_654,x_655,
x_656,x_657,x_658,x_659,x_660,x_661,x_662,x_663,x
_664,x_665,x_666,x_667,x_668,x_669,x_670,x_671,x_
672,x_673,x_674,x_675,x_676,x_677,x_678,x_679,x_6
80,x_681,x_682,x_683,x_684,x_685,x_686,x_687,x_68
8,x_689,x_690,x_691,x_692,x_693,x_694,x_695,x_696
,x_697,x_698,x_699,x_700,x_701,x_702,x_703,x_704,
x_705,x_706,x_707,x_708,x_709,x_710,x_711,x_712,x
_713,x_714,x_715,x_716,x_717,x_718,x_719,x_720,x_
721,x_722,x_723,x_724,x_725,x_726,x_727,x_728,x_7
29,x_730,x_731,x_732,x_733,x_734,x_735,x_736,x_73
7,x_738,x_739,x_740,x_741,x_742,x_743,x_744,x_745
,x_746,x_747,x_748,x_749,x_750,x_751,x_752,x_753,

x_754,x_755,x_756,x_757,x_758,x_759,x_760,x_761,x_762,x_763,x_764,x_765,x_766,x_767
/work/home/dsu/Datasets/SEWA/audio/SEW2117.wav,0.0,0.99,0.0,0.0,,-0.040896337,0.026374899,0.01298859
6,-0.02986379,-0.045906503,-0.1360833,0.052885786,-0.021498213,-0.004353988,-0.35344726,0.08795986,-0.024313012,0.050093617,0.0689269,-0.0066926954,0.016270893,-0.3718804,0.30839327,0.025095133,0.018364023,-0.18945871,0.12236829,0.18272236,0.007558954,0.18545088,0.023731688,-0.5093606,0.05451591,-0.021128634,-0.15978765,0.0965917,-0.009170476,-0.000807376,-0.07326856,-0.1774111,0.12054886,0.099158235,-0.23211241,-0.14533463,0.114760414,-0.12336798,-0.21725415,-0.09005656,0.27856675,-0.16004597,0.06169714,-0.032722566,0.012142515,0.00428443,0.0041760644,-0.11666851,-0.07436026,0.037105855,0.046537567,0.0199509,-0.013701948,-0.021371739,-0.4670346,-0.12670316,-0.12167124,0.020333512,-0.049968895,0.021250498,0.28267,-0.056057684,0.10819162,-0.0012569114,-0.058880303,-0.12440596,-0.036303975,0.003435968,-0.023229467,-0.22332713,-0.018964158,0.09341842,-0.02144246,-0.1542578,-0.038614947,-0.019796956,-0.055194646,-0.0893465,0.32712048,0.06555011,0.08462933,-0.02857156,-0.062423695,-0.06887032,-0.25653902,-0.03070706,0.04364387,0.3734293,-0.14936578,0.0911939,0.041056857,0.053487685,-0.06085786,0.0943971,0.06972217,-0.04191095,-0.094628066,-0.00044547534,-0.11805459,0.0104420325,-0.17173594,0.20628516,0.46304733,0.043256726,0.14229254,0.02536733,0.15230288,0.011983181,0.049990468,-0.020768637,-0.010112607,-0.16078171,0.05103956,0.034986075,0.026295602,-0.07078508,0.016228233,0.10700074,-0.013059988,0.08223293,-0.0083913,-0.04935442,-0.25897795,0.0374097,0.36388642,-0.10120151,0.04340003,0.10752246,-0.07104552,0.08419906,0.07041999,-0.13039461,-0.17517346,-0.14875683,-0.012655712,0.016584622,-0.59007144,-0.0012714337,0.0017199209,-0.01556444,0.07147523,0.08196373,-0.011279293,-0.6666314,0.15605178,-0.24749254,0.05487093,-0.12821436,0.13758755,0.08805018,-0.32042247,0.043961428,-0.029375577,0.044288207,-0.013617815,0.034443457,-0.069412015,-0.18987077,0.3087915,0.12944306,0.03256273,0.1275691,-0.03903755,-0.1261542,-0.056057367,0.37085482,0.14552443,0.048574656,0.0025647853,0.09585363,0.12091304,-0.0063643884,-0.05993584,-0.27481878,0.0048599443,0.051185742,-0.0057445224,0.003184533,0.01855264,-0.09241228,-0.21565606,-0.07

6281704,-0.036565293,-0.045665707,-0.15665112,-0.1
2743105,0.0072320295,-0.17754479,-0.045694802,-0.
06507642,-0.0067409384,-0.00282455,0.08749361,0.0
28982213,0.09929678,0.04045006,0.04042944,0.0201
11613,0.013911543,-0.29433626,0.10833347,-0.28750
116,0.26384535,-0.0011033552,-0.08426188,0.041339
207,0.32527065,-0.4479783,0.07894055,-0.006785398
,-0.07922606,0.030541731,-0.08975129,0.03099154,-0
.037281256,-0.0248004,0.018638818,-0.053975035,-0.
040110487,-0.26378307,0.009160995,-0.0924135,0.00
2446427,-0.0057712323,0.22722772,-0.05863003,-0.0
4980186,0.05212428,-0.021280434,0.029316802,-0.03
682708,-0.097779155,-0.0058947257,-0.011912827,-0.
0025038165,-0.098542154,-0.09606118,0.1863179,-0.
2818079,0.03428991,-0.055150572,-0.15433197,0.020
334106,-0.04233254,0.18939337,0.08693466,-0.45536
143,-0.026310503,0.33603403,0.016165212,0.066340
595,-0.03242722,0.13902014,-0.3836446,0.12056198,
0.09350746,-0.27488878,0.00046789314,0.07725479,-
0.024864275,0.026602808,-0.4364099,0.07739205,-0.
018513236,-0.04073153,-0.09088297,-0.3445324,0.00
08992414,0.20596533,-0.060749345,0.015963875,-0.2
2625154,-0.050271656,-0.03998827,0.058146533,-0.0
6824754,0.22830912,0.026543176,0.17447631,0.0135
8027,0.06988612,0.10777964,-0.10739178,-0.0357137
4,0.08441345,-0.30330762,0.21491328,-0.038966417,-
0.047211334,-0.0123214,0.007552682,0.0058911922,0
.13453509,-0.00092805224,0.026372848,0.12713777,
0.019814977,-0.042962097,-0.3316729,-0.2092072,0.1
16480365,0.1312475,-0.09856222,0.14255923,-0.0086
79841,0.048640005,-0.26592287,0.24321856,-0.10751
0954,-0.096044265,0.3197684,0.050148018,0.037003
238,-0.10284266,-0.08698866,-0.06710803,0.1007501
8,-0.035133146,0.16097723,0.08605814,-0.027325854
,0.015443576,-0.08440192,-0.10212165,-0.054764073,
0.17922175,-0.06384473,-0.013960337,-0.041084785,
0.03194371,0.08372127,0.11431034,0.050839666,-0.0
11476576,0.04676088,-0.07560954,-0.04258327,-0.09
6822165,0.119617976,-0.25691685,0.013456274,0.18
877165,-0.039925147,0.10015462,0.11773699,0.5964
703,0.0029087055,-0.11274416,0.024787577,0.70393
96,0.039464768,0.049129583,-0.1354808,0.02980573
7,-0.15859559,-0.038481757,-0.16676207,0.2188419,0
.1123096,-0.15251328,-0.009205478,0.07441747,-1.38
52111,0.052181624,0.067577265,0.0989299,-0.005006
294,0.043704018,-0.018776193,-0.00319578,-0.02277
8656,0.502462,0.5981354,0.10802231,0.012747081,0.
03018683,-0.036330495,-0.049766257,0.16185234,-0.

096006475,0.2088233,0.04455462,-0.00801439,-0.556
8917,-0.09867378,0.028259078,0.0069667343,-0.0793
9364,0.03317488,0.09159443,0.58842784,-0.0764882
2,0.64172894,-0.122268714,0.112593055,0.04873889
7,0.004853759,-0.0855071,-0.46023834,0.051585473,
0.122080885,0.0074865404,0.06970284,0.08133641,0
.0960021,0.11651924,-0.07619721,0.037006937,-0.09
862074,-0.031851575,-0.020317126,-0.23238303,-0.0
62396847,0.2880896,0.011522206,-0.027344707,0.02
4260554,-0.07853636,-0.12153345,0.11955588,-0.015
732259,0.024765808,0.0034121983,0.036024626,0.00
5361501,0.025387425,0.037502944,0.035679396,-0.0
14744148,0.62260747,0.11824905,-0.27338064,-0.113
76281,-0.033435937,-0.042815626,0.14951342,0.1729
2742,0.11041677,-0.15422693,0.060502328,0.009165
102,0.09924248,0.5396217,-0.2540672,-0.042164486,-
0.044172987,-0.31273553,-0.023917815,0.026161218,
-0.6491035,-0.013705008,0.0054544318,-0.03835923,
-0.3553638,-0.04324066,0.09851615,0.008332,0.0835
9513,0.044913847,-0.0054631294,0.071221046,-0.120
04603,-0.86574817,0.0068781916,0.028677419,0.021
595765,0.061218534,-0.07603636,-0.17542866,-0.069
54542,-0.21880971,0.15859033,-0.11035091,0.363547
2,0.063177094,-0.037057463,-0.08993099,0.11928335
6,-0.4068184,0.038283654,-0.051810097,0.29798067,-
0.097260684,-0.07306976,-0.1133876,-0.07967154,-0.
017711427,0.028369907,-0.11660614,0.002745861,-0.
13789044,-0.08988365,-0.029155388,0.02469655,-0.1
628228,-0.1644431,0.01463095,-0.020704238,0.03315
445,0.12532333,0.17331678,-0.2222325,-0.013561098
,-0.058632012,-0.31162903,-0.0697625,-0.10296723,0.
03374282,-0.21883933,-0.07553324,-0.083097816,-0.
08797563,0.1901883,0.14187825,-0.062817305,-0.034
577366,-0.06546838,-0.010862307,-0.016234687,0.30
086657,-0.028027046,0.076290324,0.015222089,-0.44
188797,-0.120731935,-0.10178856,-0.053054173,0.13
176242,0.13998464,-0.058369577,0.03881146,0.2014
7337,-0.0003414703,-0.05102841,0.0055448436,0.106
650814,0.0040996303,-0.3384382,0.0147782685,0.12
63047,-0.19515987,0.5302556,-0.018326277,0.088570
945,0.5588682,-0.13331816,-0.050434228,0.01702444
,-0.044767447,0.16262682,0.003824068,-0.003567478
4,-0.22550637,-0.10005487,-0.042484608,-0.0486376,
0.4226122,-0.0076028816,-0.08294846,-0.058252208,
-0.04952258,-0.042768184,0.022902567,0.047698613,
0.10906054,-0.01162768,0.096390866,0.079901814,-0
.046976034,-0.012507633,0.031118046,0.34423134,-0
.045268156,-0.036425095,0.31446737,0.26480928,0.0

39610103,0.008208308,0.35236683,-0.020851217,-0.0
46828788,-0.026155561,-0.1864967,-0.10144451,0.14
454934,0.10208928,-0.0496525,0.016115729,0.20249
277,-0.033322986,-0.0779726,0.118461706,0.0206829
63,0.024023192,0.007817912,-0.028824974,0.043815
233,0.06437551,-0.21689995,0.099074334,-0.2326491
9,-0.17976482,-0.014844355,-0.033601195,-0.1116353
5,-0.00043571473,-0.004549767,0.26876235,-0.13412
064,-0.07177666,-0.04095159,0.057031337,-0.069595
695,0.143284,0.06841056,-0.37584636,-0.3085362,-0.
047237024,0.030931398,0.0787339,0.09332033,0.029
096803,0.07443031,0.21315463,0.017964825,-0.2531
9657,0.013644004,0.0063413586,0.07547401,0.50446
635,-0.0008628224,-0.0044506676,0.05372777,0.0788
90935,-0.12473771,0.09672404,-0.02387691,0.119127
18,-0.094841845,-0.07771577,0.008621062,-0.085255
44,0.36218673,0.06299932,-0.114559874,-0.02166993
7,-0.02541307,0.08228664,-0.059385065,-0.07413223,
-0.08670861,0.00793988,0.041636642,-0.36668918,-0.
04696913,-0.08654195,0.008878796,0.1585377,-0.041
645516,-0.027797654,-0.17202605,0.018103177,0.107
08774,0.07103656,-0.13524342,-0.08670856,0.166017
93,-0.036363393,0.02881741,0.018363042,0.0693691
1,-0.066389255,-0.0019140333,0.08422327,0.0302071
52,0.14040963,-0.049068592,-0.07959952,0.08292159
4,0.17077388,-0.052035384,-0.09437072,1.1371422,0.
244537,0.1714013,-0.33402854,0.037860695,-0.12263
5715,-0.03345973,-0.086078085,-0.043356836,0.0153
24208,0.2509705,0.19132902,-0.08149983,0.1227958
7,0.0072303154,0.21676658,-0.0022587655,0.322878
4,0.07622053,0.17254108,0.09466797,0.052242376,-0
.008961978,-0.039232526,0.073868,-0.02116677,-0.11
833917,0.11017149,0.24211344,-0.3147815,0.3989369
6,0.04656121,0.36427042,0.06464534,-0.17979825,0.
15925708,0.10900148,0.0074112546,-0.07307722,-0.0
14357472,0.04578861,0.18140477,0.031758804,0.041
066613,-0.0036456173,0.32167068,-0.0837213,-0.443
48678,-0.04981048,0.07922944,0.094838,0.13855442,
-0.25744292,-0.0490936,-0.29510295,-0.058240023,-0
.089809746,-0.11226142,0.048874363,-0.012357657,-
0.042337425,-0.037825312,0.2094312,-0.095272236,0
.00851738,-0.09648082,0.014574472,0.03796379,-0.0
88964075,-0.023682097,0.06459891,-0.0037607283,0.
030457783,0.2011733,-0.057541545,-0.07718482,0.29
745978,-0.17873219,0.0013111127,-0.019296799,-0.0
98471016,0.0327052,0.023840906
/work/home/dsu/Datasets/SEWA/audio/SEW2117.wav,
1.0,1.99,0.0646728,0.02683,,-0.077449456,0.0527357

65,-0.0046621524,-0.044331424,-0.08720812,-0.1414
3021,0.03148162,-0.0019936073,-0.09991932,-0.3824
361,0.114354365,-0.046854965,0.06433814,0.087716
24,-0.014359951,0.02131485,-0.35426968,0.33032933
,0.017747227,0.025054853,-0.24174541,0.14367151,0
.033567093,0.037827555,0.13291353,0.048378173,-0.
4860091,0.016118206,-0.028265812,-0.24739397,0.11
713049,-0.006879245,-0.051712345,-0.12569785,-0.1
846363,-0.007718319,0.071335316,-0.23637156,-0.14
813867,0.06852997,-0.080188654,-0.18346815,-0.064
74636,0.33838436,-0.1433688,0.06249409,-0.0238941
27,-0.08226594,0.015075789,0.010629606,-0.1180639
4,-0.08713624,0.07318375,0.015390343,0.016012779,
0.06494637,-0.047547407,-0.49002638,-0.15969701,-
0.10952807,0.037795287,-0.048806015,0.022915477,
0.32939103,-0.054611944,0.062241003,-0.001307404,
-0.11462909,-0.12633617,-0.032320295,-0.0526544,-0.
020785905,-0.25075635,-0.03417602,0.12217818,0.03
4461696,-0.15118453,0.050335247,-0.017209047,-0.0
8872892,-0.0655074,0.2369259,0.068893485,0.09268
797,-0.031841967,-0.11340798,-0.05790644,-0.283535
87,-0.030228123,0.05683459,0.45517874,-0.23293364
,0.11357195,0.041023526,0.056189682,-0.08147317,0
.19998305,0.09205357,-0.057119846,-0.06229637,-0.0
3465407,-0.14319816,0.0026951686,-0.16433568,0.22
64167,0.452915,0.10083998,0.18017274,0.03645407,
0.20284338,0.0035164568,0.10580022,-0.04759555,-0
.004875172,-0.16096896,0.07457407,0.034246422,-0.
0547772,-0.06377729,0.019868853,0.07101529,-0.047
21115,0.060856644,-0.014574214,-0.13488449,-0.184
35535,0.029405976,0.37956592,-0.09741456,0.06264
1986,0.17120676,-0.11541444,0.11703431,0.06346405
,-0.14238785,-0.31694046,-0.1659194,0.01952305,0.0
3842668,-0.5466401,-0.0022507964,-0.02275391,-0.0
28490154,0.039425388,0.061285824,-0.01273141,-0.8
0334944,0.18726847,-0.21491706,0.12270244,-0.2284
8697,0.16200458,0.10519829,-0.3278909,0.08544639
5,-0.11540185,0.07616483,0.010743629,0.034872252,
-0.038867593,-0.23093028,0.33009034,0.163156,0.01
8755646,0.14581908,-0.037011098,-0.20508352,-0.01
3339488,0.28209063,0.12846191,0.04425982,-0.0006
6923903,0.09610446,0.101597704,-0.006000577,-0.00
5908343,-0.39202532,0.0022466106,0.10091497,-0.01
4762926,0.01327782,0.026660014,-0.09090456,-0.194
27076,-0.09217574,-0.036916707,-0.07832032,-0.125
52035,-0.15608934,-0.027288454,-0.18332905,-0.042
232007,-0.06253491,-0.012008867,-0.004714802,0.07
7848054,0.048045877,-0.081470564,0.11294877,0.02

2060238,0.015334551,0.015033038,-0.32939404,-0.02
3128688,-0.33090878,0.28825167,0.002062743,-0.102
40094,0.028226502,0.3174521,-0.51786864,0.041689
13,-0.00727017,-0.09443815,0.030922553,-0.0900075
9,0.010583948,-0.03463208,-0.025921186,0.04308016
2,-0.04619227,-0.037382253,-0.28219545,0.02027294
2,-0.06044965,0.03132053,-0.005796183,0.24887583,-
0.087374605,-0.030666359,-0.012041278,0.00511869
37,0.029460153,-0.032572344,-0.092463784,-0.01384
5674,-0.010646273,-0.002456267,-0.21118176,-0.0758
4882,0.20565295,-0.087489195,0.047139764,-0.06345
593,-0.18543465,0.020327063,-0.057898823,0.197044
94,0.09571889,-0.39025116,-0.02084777,0.3442194,-0
.036434658,0.09039747,-0.08149754,0.10772298,-0.4
1405803,0.10142855,0.107545026,-0.23676302,-0.021
55954,0.19316584,-0.06788564,0.04011274,-0.390067
96,0.0870187,-0.010373128,-0.06540405,-0.08335835
5,-0.34106982,0.018412935,0.22874291,-0.059837747
,0.01660314,-0.2894669,-0.04997165,-0.035417654,0.
036221165,-0.10540526,0.28082654,0.026407918,0.2
1345533,0.02204578,-0.08959373,0.11362957,-0.1266
1596,-0.059100293,0.11971399,-0.3141281,0.1379293
,0.085224755,-0.13923168,0.013688648,0.007878906,
0.04296568,0.17156398,-0.0010381193,-0.008301169,
0.15165211,0.02412031,-0.043458845,-0.337567,-0.17
941019,0.1574566,0.098451905,-0.12966329,0.05374
742,0.063844554,0.05453716,-0.21034285,0.1640504
4,-0.07079566,-0.11326327,0.31237888,0.1052738,0.0
9844167,-0.14573605,-0.03560851,-0.0711999,0.0970
1589,-0.16614188,0.17629445,0.14832608,-0.0320824
46,0.13622992,-0.091734886,-0.093017645,-0.048498
716,0.17731458,-0.06368216,-0.029912038,-0.059241
813,0.039413143,0.09049343,0.038009502,0.0715056
66,0.017382268,0.04209527,-0.07514348,-0.02138926
,-0.06780997,0.06415785,-0.315765,0.07192281,0.220
85756,0.054476,0.13124219,0.16222088,0.54915565,
0.0017511157,-0.1159737,-0.057404224,0.6812561,0.0
9622523,0.013424039,-0.14982036,0.030078063,-0.16
343454,-0.03966988,-0.14959155,0.20510739,0.10604
093,-0.17425664,-0.0065523107,0.08072817,-1.44747
84,0.069692075,0.036611438,0.113990664,-0.0083895
75,-0.04875961,-0.08881128,-0.039707445,-0.0127763
85,0.61791223,0.5592825,0.053049278,0.012607332,
0.017237043,-0.061125875,-0.02523377,0.094865285,
-0.18756054,0.22113654,0.0010294759,-0.023212656,
-0.69707185,-0.11678861,0.05953619,0.07659615,0.0
23463886,0.0713411,0.07057938,0.61321,-0.0750817
95,0.57004255,-0.12280633,0.07188852,0.06572377,0

.03260745,0.016676575,-0.24091418,0.096930966,0.1
411523,0.004340494,0.060082063,0.08003051,0.1169
4182,0.1444615,-0.12322097,0.021380696,-0.0962596
6,-0.022471491,-0.024147496,-0.11810331,-0.0633973
1,0.29728493,0.028007595,-0.027385617,0.03363475
2,-0.08524683,-0.18925157,0.11435291,-0.004028463
3,0.019647302,-0.032127853,0.07071174,0.10026787,
0.010409248,0.013960922,0.057442218,0.012654142,
0.59499186,0.12273257,-0.3228002,-0.08424257,0.20
966543,0.05287218,0.18749754,0.14357615,0.017279
724,-0.14261955,-0.004726135,0.008927188,0.116719
51,0.52268976,-0.23331073,-0.08508378,-0.06308558
6,-0.35463247,-0.0457023,0.006928838,-0.5167095,0.
06674347,0.005441601,-0.038343437,-0.25418624,-0.
043510254,0.090607174,-0.014976589,0.0869387,0.0
3725599,0.0018569586,0.13523014,-0.16463923,-0.94
0021,0.0034399177,0.01151644,0.038141567,-0.0165
8419,-0.059975896,-0.16620764,-0.070994884,-0.136
81708,0.001822282,-0.16873552,0.41779712,0.04286
9218,-0.013425466,-0.02850569,0.10683,-0.41885963,
0.13084692,-0.16921254,0.30320892,-0.14841053,-0.1
4394791,-0.09192233,-0.105509855,-0.007891417,0.0
29848604,-0.11560362,0.006594621,-0.13922146,-0.1
7213918,-0.0040649553,0.0044555366,-0.19291563,-0
.22550726,0.024141693,-0.0006699512,0.044601273,
0.14300564,0.22771706,-0.23818775,-0.05436208,-0.0
4742221,-0.33448374,-0.06868879,-0.10273698,-0.00
3485467,-0.23345669,-0.08425172,-0.13611083,-0.097
44066,0.22325861,0.094960526,-0.12205395,-0.06146
463,-0.060866587,-0.010757021,0.008373037,0.31281
33,-0.027997458,0.09201283,0.02347626,-0.41655073
,-0.066597834,-0.20911524,-0.07043331,0.13543147,0
.11914112,-0.08002824,0.043256186,0.19306451,-0.0
0043769536,-0.052985597,0.015455838,0.14332093,-
0.0021424117,-0.14982907,0.03763504,0.13154557,-0
.22466199,0.4862653,-0.06396737,0.07985167,0.4842
198,-0.032932006,-0.045011986,0.05335613,-0.04317
0817,0.14368759,0.0017699497,-0.0027043184,-0.223
44078,-0.120722264,-0.037718974,-0.048250314,0.44
11048,-0.0074395915,-0.08987771,-0.04715926,-0.077
21147,-0.03650398,0.01838622,0.08202795,0.025629
722,-0.019766675,0.06340336,0.104378425,-0.019469
675,-0.016993953,0.049148306,0.44774085,-0.036003
698,-0.08907662,0.5161471,0.23157008,0.03262429,0
.021788314,0.40660268,-0.002351788,-0.04519039,-0.
011461504,-0.13721567,-0.08578369,0.21514848,0.10
081394,-0.06518387,-0.24163961,0.20021234,-0.1538
6087,-0.098964415,0.16502446,0.03736894,0.043125

834,-0.015766056,-0.071704455,0.041412346,0.07256463,-0.14422178,0.1059639,-0.25689548,-0.25688565,-0.0007756981,0.029329922,-0.15358564,-0.00015648971,0.0008825208,0.24693802,-0.17792755,0.0009431814,-0.08150104,0.02268831,-0.13412698,0.15545738,0.078013584,-0.40473926,-0.33968797,-0.046332564,0.02662979,0.04515311,0.089249864,0.012545919,0.097552836,0.19872105,0.082239844,-0.2273558,-0.018249411,0.0028699085,0.08254186,0.5728817,0.043780454,-0.005936794,0.0021617997,0.03656074,-0.07329923,0.08745744,-0.023838667,0.18506187,-0.104239084,-0.30461115,0.010398393,-0.13160181,0.41199344,0.1295938,-0.09349966,-0.009748032,-0.02443513,0.13369945,-0.03851928,-0.13628483,-0.08967871,-0.025800062,0.05194298,-0.28133115,-0.039408483,-0.11538573,0.008737038,0.16391513,0.012687652,-0.09851262,-0.15476076,-0.12517758,0.100451194,0.08526045,-0.12942725,-0.08599794,0.20336191,-0.060500415,0.11718871,0.011480047,0.15664317,-0.08227475,-0.0015967463,0.13212928,0.037037756,0.17098415,-0.087073535,-0.14013797,0.09807435,0.20818853,-0.063403085,-0.063289754,1.0919275,-0.0008286827,0.058596157,-0.360828,0.052247398,-0.1629062,-0.0017065465,-0.08210442,-0.03078804,0.01561623,0.27506655,0.22585808,-0.081699096,0.10296504,0.0014660964,0.23504153,-0.07004909,0.5147599,0.087923765,0.13747269,0.11564262,0.07744169,-0.04457547,-0.05090585,0.058269296,0.011747996,-0.15673421,0.05497722,0.2682119,-0.38427186,0.5314941,0.05708898,0.40179417,0.06788717,-0.19178487,0.21844898,0.1696087,0.008687676,-0.073114336,0.026849486,0.055891942,0.1902524,0.04220431,0.031480953,-0.015966209,0.3643017,-0.1028374,-0.48265338,-0.041936267,0.069362156,0.18675967,0.15015589,-0.27170002,-0.0460517,-0.30298924,-0.05683496,-0.07605514,-0.123728804,0.047725,-0.04066562,-0.03121705,-0.01599063,0.13652727,-0.12615566,0.07205031,-0.164697,0.0258121,-0.017291354,-0.110880025,0.0037821245,0.1051419,0.0016697414,0.015444475,0.2174058,-0.05909589,-0.0309581,0.47576606,-0.110853136,-0.002690366,-0.014028912,-0.07007084,0.0065642726,-0.011737822". Here you see that for this time interval fit start_timestep=0.0,end_timestep=0.99 AND start_timestep=1.0,end_timestep=1.99. So since one video frame in this interval is not enough (it is smaller than n-2 like i explained earlier) then ignore this time interval start_interval = 0 sec, end_interval = 2 sec for

this specific video and DON'T save the corresponding audio and video data.

- Let's take a look at the next time interval start_interval = 1 sec, end_interval = 3 sec for the same video. We do the same procedure: looking at amount of video frames (amount of timestamps that fit in this time interval) first and see that it's four frames: 1.7,2.04,2.38,2.72, so you just duplicate the last line two times and then look at the corresponding audio data, where you find that for this time interval fit start_timestep=1.0,end_timestep=1.99 AND start_timestep=2.0,end_timestep=2.99. So you save BOTH the frames and the corresponding audio data/ features and go to the next time interval till the video is finished. You do this for EVERY video and save all the sliced time intervals/windows inside DataLoader in RAM. Give each window an index, which you will use in the __getitem__ method to retrieve a specific window of frames and its associated audio features ALONG with the labels. Labels are arousal and valence values associated with each frame of the video data. You always only look at the LAST video frame of the specific time interval/window and take the arousal and valence values as labels.
- You now output in the __getitem__ methoda tuple/list with the size=3 containing: (video features (path,timestamp,emb_0 till emb_255), audio features (filename,start_timestep,end_timestep,features,x_0 till x_767), labels). As you can see you extract everything from the audio and video features for each window except the arousal and valence values, because you have the "overall" arousal and valence as labels for each window which you find out by looking at the LAST video frame of the specific time interval/window. You have always several windows for each video, you do the windows till the video is done and then head to the next video and do it all over again.
- You have to do the synchronizing for train dataset (audio and video), test dataset (audio and video) and dev dataset (audio and video) and create three DataLoaders: train_dataloader, dev_dataloader, test_dataloader.

In the end, after doing all that I want to see the output of the dataloader of the first 10 lines to see how well it was synchronized.