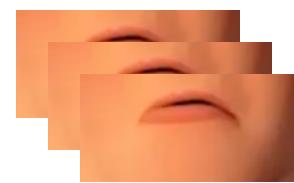


More Detailed steps for data preprocessing

(How to get face and lip in this work?)



Face region



Lip region

Some notes

- First of all, We sincerely appreciate Dr. Qi Chen (The University of Adelaide), and Dr. Chenxu Hu (Tsinghua University). We are very very grateful to the original authors for providing us with their excellent datasets .
- Numerous methods exist for extracting regions of lips and faces; in this instance, we share the more detailed preprocessing steps of our methods (HPMDubbing ) to obtain the mouth and lip region images.
- If you have more convenient and faster tools or methods (like Deep Visual Tracking or FaceTracking), you can also use them to process the original video.
- We will share the image areas of the mouth and face we extracted for readers to use conveniently. (If you want to extract features from other models, you can feed these images directly)
- Below we will take the chem dataset as an example to show you our processing steps in this work.

1. Download all list .txt

- Please note that we first removed five samples from the training set because we found the Montreal Forced Aligner (MFA) cannot obtain their TextGrid files for the five samples provided by the author, as follows:

```
Setting up corpus information...
Number of speakers in corpus: 1, average number of utterances per speaker: 6640.0
Creating dictionary information...
Setting up corpus_data directory...
Generating base features (mfcc)...
Calculating CMVN...
Done with setup.
There were 5 segments/files not aligned. Please see /data/conggaoxiang/Chem_wav16_feature/TextGrid/unaligned.txt for more details on why
alignment failed for these files.
Done! Everything took 712.3856081962585 seconds
(v2c) conggaoxiang@aa-ESC8000-G4:~/MFA_tools$ vim /data/conggaoxiang/Chem_wav16_feature/TextGrid/unaligned.txt
```

```
PyIiT0eXrA-003 Could not decode (beam too narrow)
X0nCo0X0LAG-022 Could not decode (beam too narrow)
ZwsWjelzqDA-099 Could not decode (beam too narrow)
hXLiubyd58Y-004 Could not decode (beam too narrow)
wFY9ZvUmM5o-039 Could not decode (beam too narrow)
```

all list number: 6635

```
All_list.txt x
All_list.txt
6633 myN3PqD38Ds-001
6634 myN3PqD38Ds-002
6635 myN3PqD38Ds-004
```

We share the all list .txt you can download it by Google Drive:
https://drive.google.com/file/d/13xJW7rRKl4uE0qZhZ0EgOvO_5VpbVzzG/view?usp=drive_link

2. Use FFmpeg to obtain each frame of the video, and set the FPS to 25

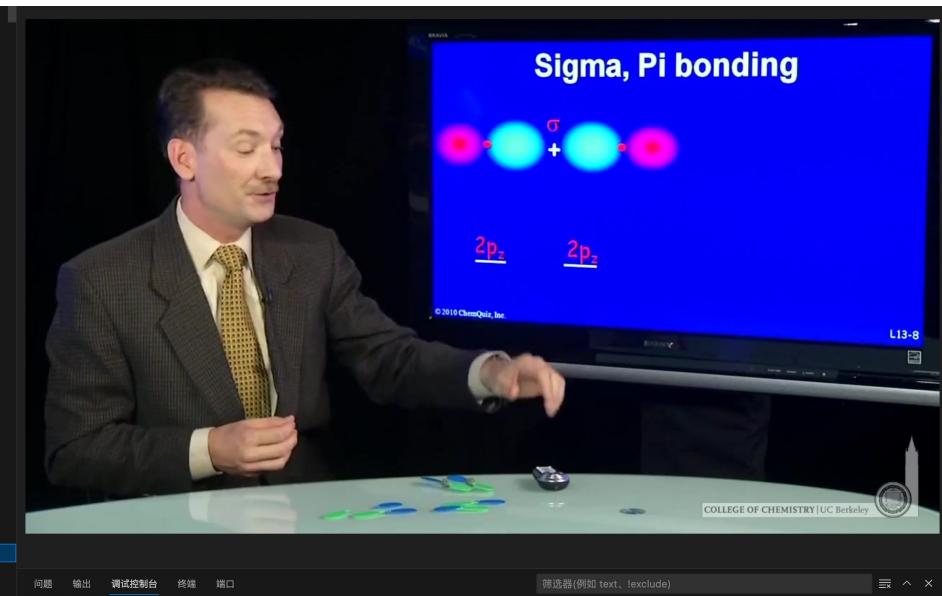
```
os.makedirs(args.pyframesPath, exist_ok=True) # Save all the video frames
command = ("ffmpeg -y -i %s -qscale:v 2 -threads %d -r %d -f image2 %s -loglevel panic" % \
           | | | (args.videoPath, args.nDataLoaderThread, 25, os.path.join(args.pyframesPath, '%06d.jpg')))

subprocess.call(command, shell=True, stdout=None)
```

Note that set **-r** to 25

```
(v2c) congaoxiang@aa-ESC8000-G4:~/TalkNet-ASD-main$ python demo_chem_last.py
 0%
=====NUM:1---ID:myN3PqD38Ds-005---SUCCESS=====
2023-12-01 16:11:59 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-005
 0%
=====NUM:2---ID:myN3PqD38Ds-014---SUCCESS=====
2023-12-01 16:12:00 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-014
 0%
=====NUM:3---ID:myN3PqD38Ds-015---SUCCESS=====
2023-12-01 16:12:01 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-015
 0%
=====NUM:4---ID:myN3PqD38Ds-016---SUCCESS=====
2023-12-01 16:12:02 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-016
 0%
=====NUM:5---ID:myN3PqD38Ds-017---SUCCESS=====
2023-12-01 16:12:03 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-017
 0%
=====NUM:6---ID:myN3PqD38Ds-022---SUCCESS=====
2023-12-01 16:12:04 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-022
 0%
=====NUM:7---ID:myN3PqD38Ds-024---SUCCESS=====
2023-12-01 16:12:04 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
-024
 0%
=====NUM:8---ID:myN3PqD38Ds-025---SUCCESS=====
2023-12-01 16:12:05 Extract the video and save in /data/congaoxiang/TalkNet-ASD-main/Chem_buhan/Chem_frame/myN3PqD38Ds-face-myN3PqD38Ds
```

```
> _JgKgx7Jn-g-face-_JgKgx7Jn-g-006
> _JgKgx7Jn-g-face-_JgKgx7Jn-g-007
> _JgKgx7Jn-g-face-_JgKgx7Jn-g-008
> _JgKgx7Jn-g-face-_JgKgx7Jn-g-010
> _JgKgx7Jn-g-face-_JgKgx7Jn-g-012
> -WxRxDp-luU-face--WxRxDp-luU-000
> -WxRxDp-luU-face--WxRxDp-luU-002
> -WxRxDp-luU-face--WxRxDp-luU-003
> -WxRxDp-luU-face--WxRxDp-luU-012
> -WxRxDp-luU-face--WxRxDp-luU-015
> -WxRxDp-luU-face--WxRxDp-luU-016
> -WxRxDp-luU-face--WxRxDp-luU-017
> -WxRxDp-luU-face--WxRxDp-luU-019
> -WxRxDp-luU-face--WxRxDp-luU-020
> -WxRxDp-luU-face--WxRxDp-luU-023
> -WxRxDp-luU-face--WxRxDp-luU-024
> -WxRxDp-luU-face--WxRxDp-luU-025
> -WxRxDp-luU-face--WxRxDp-luU-026
> -WxRxDp-luU-face--WxRxDp-luU-027
> -WxRxDp-luU-face--WxRxDp-luU-028
> -WxRxDp-luU-face--WxRxDp-luU-029
> -WxRxDp-luU-face--WxRxDp-luU-030
> 1qiwlqrG6VY-face-1qiwlqrG6VY-000
> 1qiwlqrG6VY-face-1qiwlqrG6VY-001
> 1qiwlqrG6VY-face-1qiwlqrG6VY-002
> 1qiwlqrG6VY-face-1qiwlqrG6VY-003
> 1qiwlqrG6VY-face-1qiwlqrG6VY-004
> 000001.jpg
> 000002.jpg
> 000003.jpg
> 000004.jpg
> 000005.jpg
```



Please run

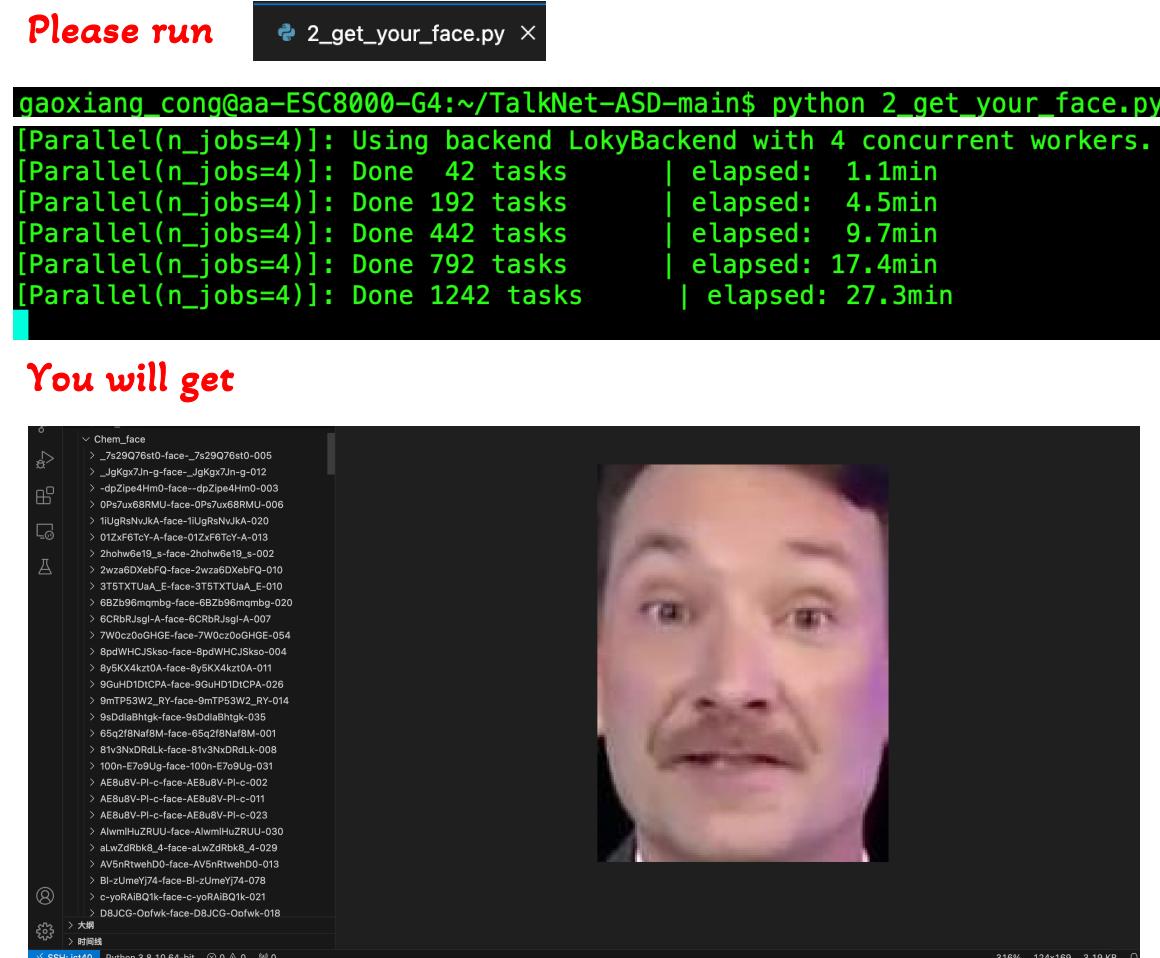
1_get_your_frames.py

You will get

3. Detect and crop the face from the video frames using S^3FD face detection model.

```
17 class S3FD():
18     def __init__(self, device='cuda'):
19         tstamp = time.time()
20         self.device = device
21         self.net = S3FDNet(device=self.device).to(self.device)
22         PATH = os.path.join(os.getcwd(), PATH_WEIGHT)
23         state_dict = torch.load(PATH, map_location=self.device)
24         self.net.load_state_dict(state_dict)
25         self.net.eval()
26     def detect_faces(self, image, conf_th=0.8, scales=[1]):
27         w, h = image.shape[1], image.shape[0]
28         bboxes = np.empty(shape=(0, 5))
29         with torch.no_grad():
30             for s in scales:
31                 scaled_img = cv2.resize(image, dsize=(0, 0), fx=
32
33                 scaled_img = np.swapaxes(scaled_img, 1, 2)
34                 scaled_img = np.swapaxes(scaled_img, 1, 0)
35                 scaled_img = scaled_img[[2, 1, 0], :, :]
36                 scaled_img = scaled_img.astype('float32')
37                 scaled_img -= img_mean
38                 scaled_img = scaled_img[[2, 1, 0], :, :]
39                 x = torch.from_numpy(scaled_img).unsqueeze(0).to(
40                     y = self.net(x)
```

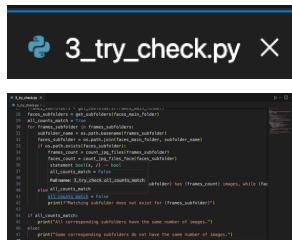
- This script is based on TalkNet-ASD :
<https://github.com/TaoRuijie/TalkNet-ASD>
- For the face process, we use Parallel to accelerate.
- We also save unrecognized images as JSON files for checking.



[Supplement] Why can't it be detected?

Script to check if faces are all aligned:

Please run



A screenshot of a terminal window titled "3_try_check.py". The window shows several lines of Python code. The code is used to compare the number of images in subfolders of a directory. It prints out discrepancies between different subfolders.

```
3_try_check.py
=====
# This script compares the number of images in subfolders of a directory.
# If two subfolders have the same name but different numbers of images, it prints a discrepancy message.
# Usage: python 3_try_check.py <directory>
# Example: python 3_try_check.py /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame

# Import required modules
import os
from collections import defaultdict

# Define the directory to check
dir_to_check = '/data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame'

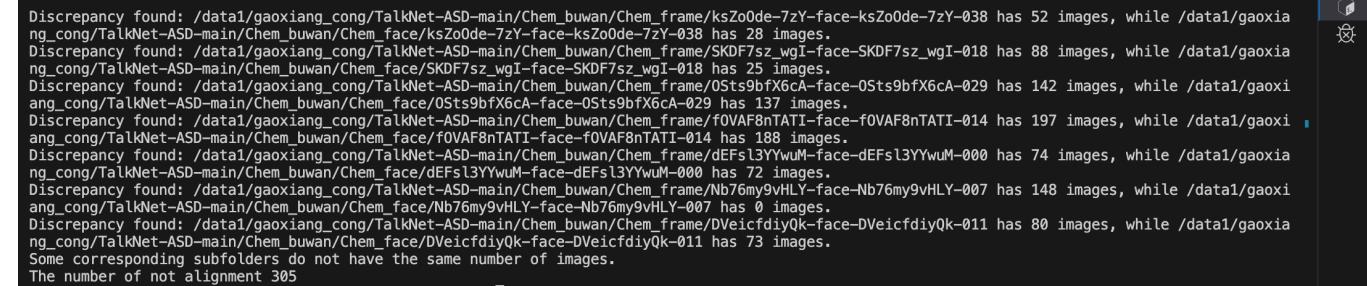
# Initialize a dictionary to store the count of images in each subfolder
image_count_dict = defaultdict(int)

# Loop through all subfolders in the main directory
for subfolder_name in os.listdir(dir_to_check):
    subfolder_path = os.path.join(dir_to_check, subfolder_name)
    if os.path.isdir(subfolder_path):
        # Count the number of images in the subfolder
        num_images = len(os.listdir(subfolder_path))
        image_count_dict[subfolder_name] = num_images

# Print discrepancies
for subfolder_name, count in image_count_dict.items():
    if count != 1:
        print(f"Discrepancy found: {subfolder_name} has {count} images, while the corresponding subfolders have the same name but different numbers of images: {image_count_dict}")
    else:
        print(f"Discrepancy found: {subfolder_name} has {count} images, while the corresponding subfolders have the same number of images: {image_count_dict}")

# Print the total number of discrepancies
print(f"The number of not alignment {len(image_count_dict)}")
```

You will get

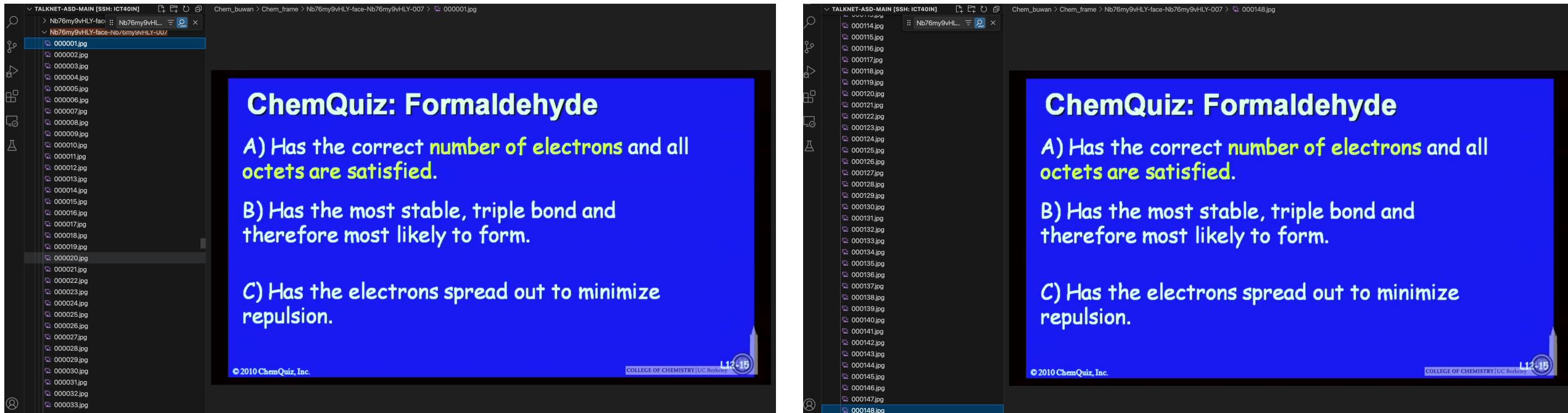


A screenshot of a terminal window showing the output of the script. The output lists several discrepancies where subfolders have different numbers of images, such as ksZo0de-7zY-face-ksZo0de-7zY-038 having 52 images while ksZo0de-7zY-face-ksZo0de-7zY-038 has 28 images. It also mentions some subfolders like dEFs13YYwuM-face-dEFs13YYwuM-000 having 74 images while dEFs13YYwuM-face-dEFs13YYwuM-000 has 0 images. The script concludes with a total count of 305 discrepancies.

```
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/ksZo0de-7zY-face-ksZo0de-7zY-038 has 52 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/ksZo0de-7zY-face-ksZo0de-7zY-038 has 28 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/SKDF7sz_wgI-face-SKDF7sz_wgI-018 has 88 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/SKDF7sz_wgI-face-SKDF7sz_wgI-018 has 25 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/0Sts9bfX6cA-face-0Sts9bfX6cA-029 has 142 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/0Sts9bfX6cA-face-0Sts9bfX6cA-029 has 137 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/f0VAF8nTATI-face-f0VAF8nTATI-014 has 197 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/f0VAF8nTATI-face-f0VAF8nTATI-014 has 188 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/dEFs13YYwuM-face-dEFs13YYwuM-000 has 74 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/dEFs13YYwuM-face-dEFs13YYwuM-000 has 0 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/Nb76my9vHLY-face-Nb76my9vHLY-007 has 148 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/Nb76my9vHLY-face-Nb76my9vHLY-007 has 0 images.
Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_frame/DVeicfdiyQk-face-DVeicfdiyQk-011 has 80 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_buhan/Chem_face/DVeicfdiyQk-face-DVeicfdiyQk-011 has 73 images.
Some corresponding subfolders do not have the same number of images.
The number of not alignment 305
```

- We share the face regions extracted by ourselves. You can download it to check:
https://drive.google.com/drive/folders/16Kl32oRMZ3_ft1SiqpJyPmOeJFL2om7c?usp=drive_link
- Besides, we give some failed examples (PPT on the following two pages) for you to check.
- We provide our failed list, i.e. JSON file of the previously mentioned, which you can download here:
https://drive.google.com/file/d/11TXnjkdVVdAPagXkSlgiuPrye5f3NEyS/view?usp=drive_link

For example, the sample **Nb76my9vHLY-face-Nb76my9vHLY-007** is all “Chemistry PPT”, and there is no “chemistry teacher”.



000001.jpg

000148.jpg

All frames are all “Chemistry PPT”, so the face detection is 0

Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_bewan/Chem_frame/Nb76my9vHLY-face-Nb76my9vHLY-007 has 148 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_bewan/Chem_face/Nb76my9vHLY-face-Nb76my9vHLY-007 has 0 images.

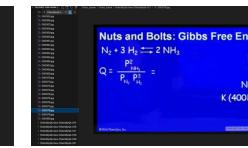
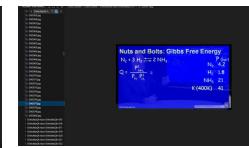
For example, in the later part, the screen gradually switches to “Chemistry PPT”, and the face of the chemistry teacher is no longer there.

Discrepancy found: /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_bewan/Chem_frame/DVeicfdiyQk-face-DVeicfdiyQk-011 has 80 images, while /data1/gaoxiang_cong/TalkNet-ASD-main/Chem_bewan/Chem_face/DVeicfdiyQk-face-DVeicfdiyQk-011 has 73 images.

000072.jpg 000073.jpg



000074.jpg

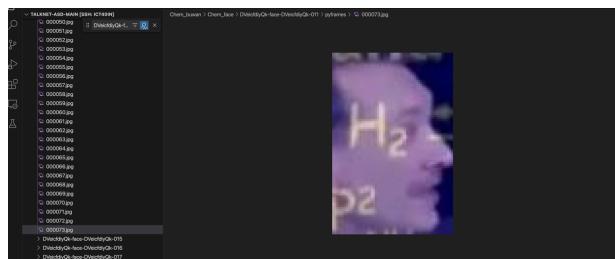
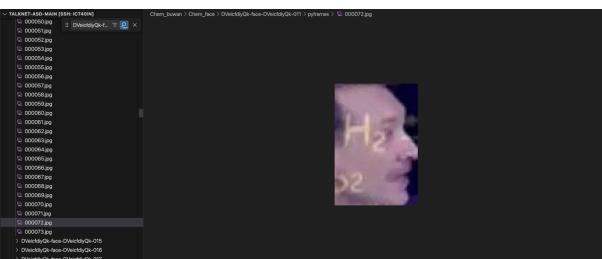


000080.jpg

000072.jpg

000073.jpg

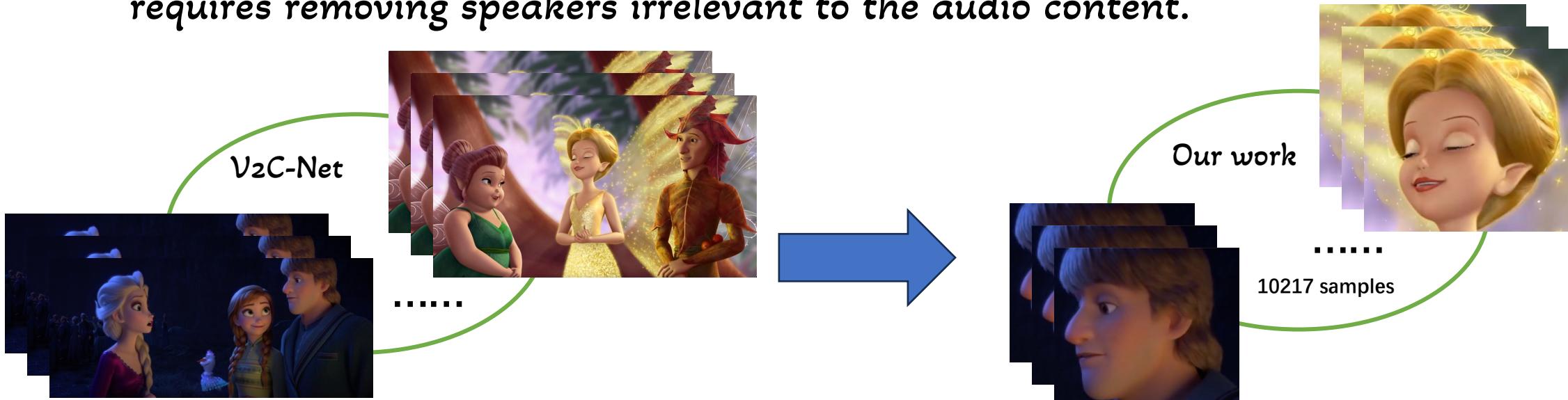
000074.jpg-000080.jpg



S3FD model cannot capture
human faces anymore

[More details] Face detection on V2C dataset

- ◆ The **difficulty** in scene processing of the V2C dataset is that unlike the **chem dataset**, which only has one speaker, there may be many speakers in a movie shot.
- ◆ The original V2C dataset (V2C: Visual Voice Clone [CVPR 2022]) did not take this into account did not differentiate between speakers, and used the entire scene as a visual cue. No corresponding annotation.
- ◆ One of the contributions of this work is the specification of movie speakers, which requires removing speakers irrelevant to the audio content.



4. Get the mouth ROIs from face frames

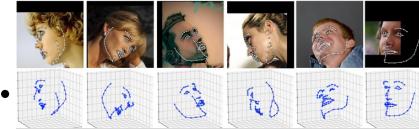
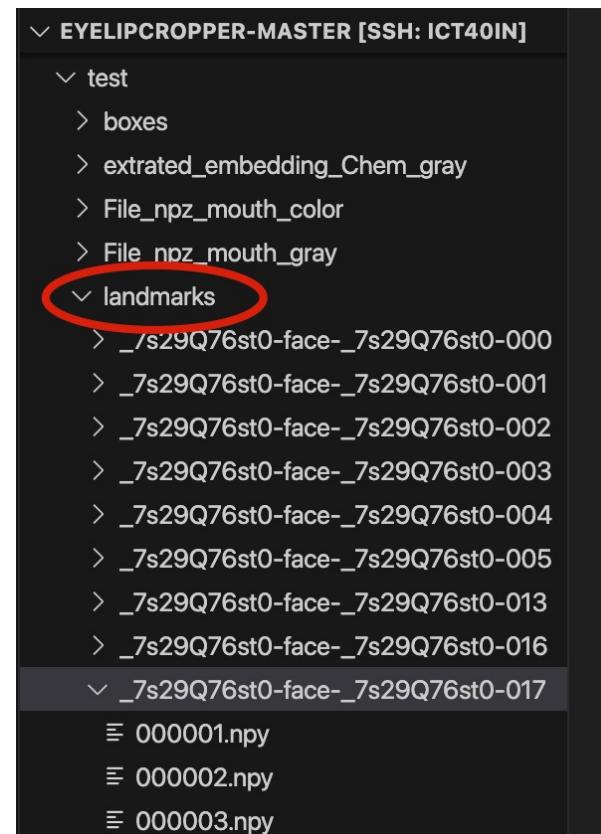
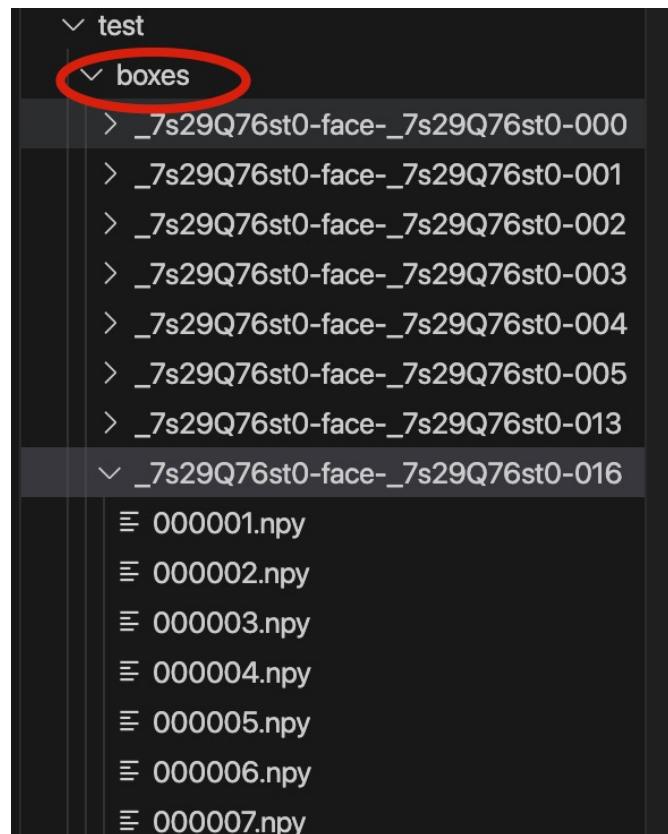
- 4.1) Align faces to generate 68 landmarks and bounding boxes.

Please run

4_face_alignment.py

You will get

- This script is based on EyeLipCropper:
<https://github.com/zhliuworks/EyeLipCrop>
- Facial landmarks refer to specific key points on the face that are used to help identify and analyze the structure of the face.
- A bounding box is a rectangular box used to define the location of a specific object in an image. In face recognition tasks, bounding boxes usually surround the detected face, indicating its position and size in the image.



4. Get the mouth ROIs from face frames

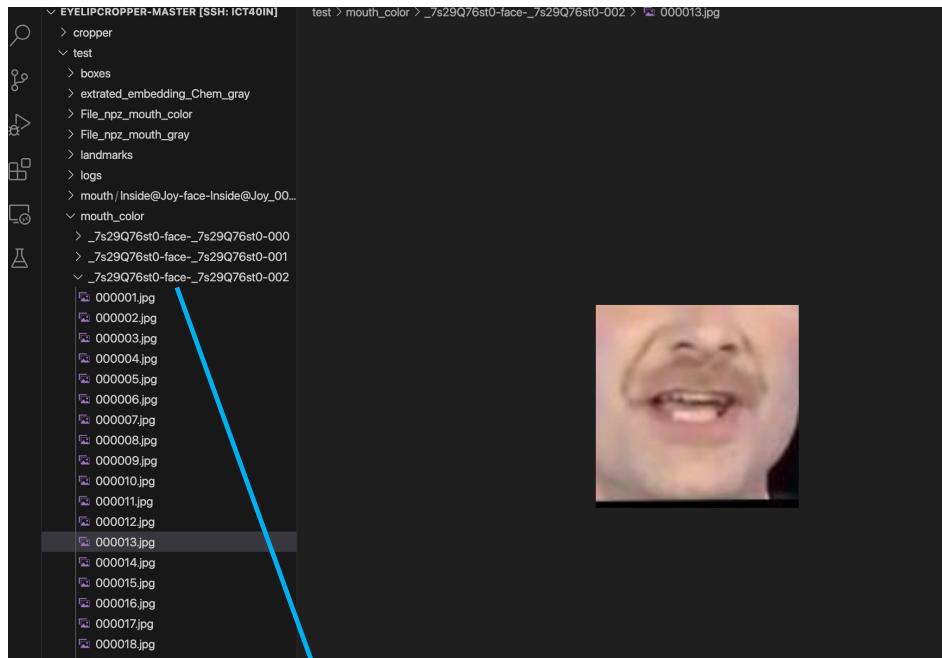
- 4.2) Next, we crop the mouth ROIs based on the bounding boxes

Please run

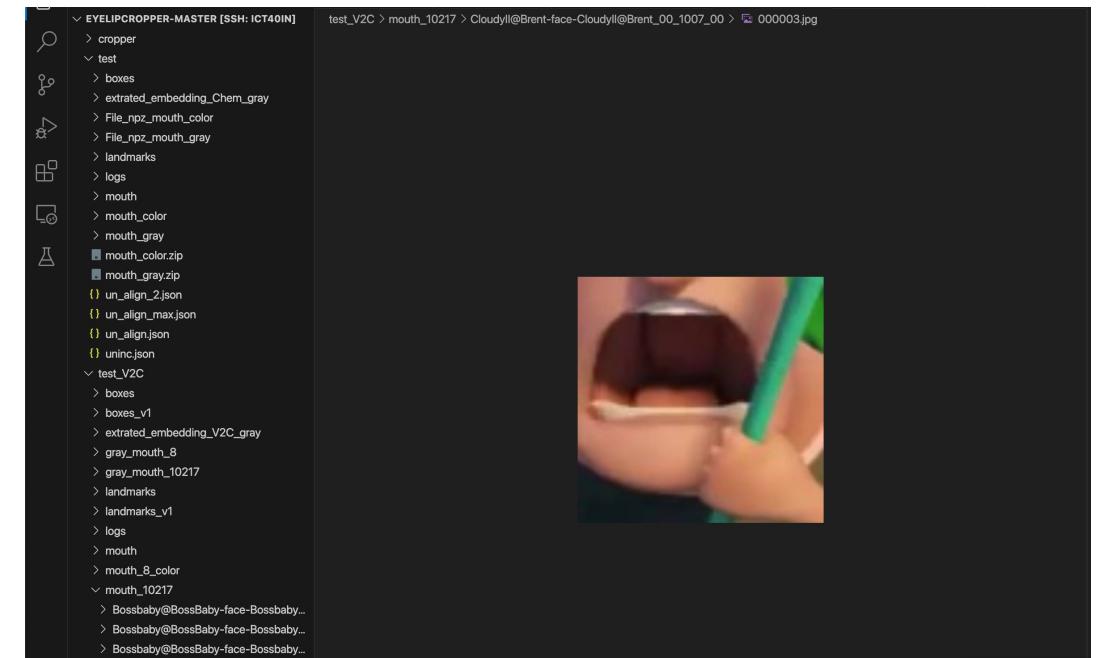
5_eye_mouth_crop.py

You will get

Lip region



Chem dataset

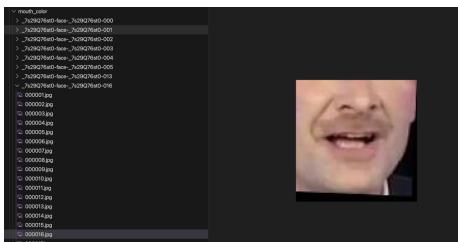


V2C dataset

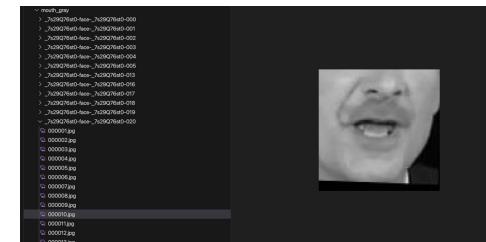
We accidentally named it wrong here. Writing "-face-" may be misleading, but the content inside is indeed the lip region (.jpg) we extracted.

[More details] Color or Gray?

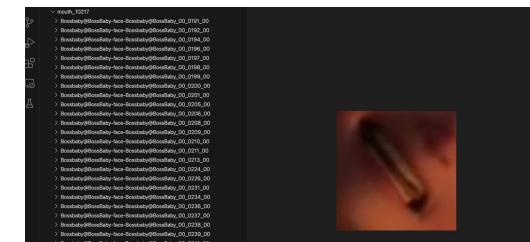
- For the convenience of readers, we provide grayscale and color versions for the two datasets respectively. You can directly download our extracted mouth area through the connection:



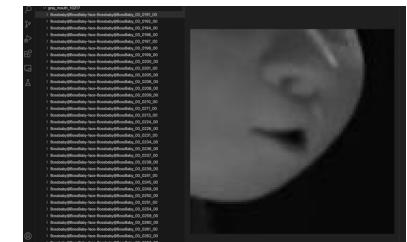
color.jpg



gray.jpg



color.jpg



gray.jpg

https://drive.google.com/drive/folders/1d9rQjM7L7sfmTorrgnsNHe6Z1qiYT4yz?usp=drive_link

- Why are there grayscale and color versions?

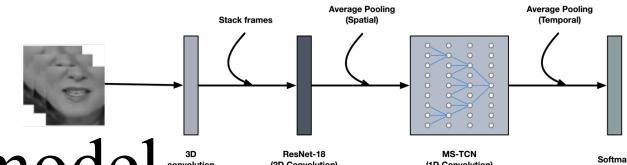
We first obtained the color version according to [cropper](#) (previous step), but we needed to follow the preprocessing process of Lipreading using Temporal Convolutional Networks

(https://github.com/mpc001/Lipreading_using_Temporal_Convolutional_Networks), so we converted it to gray level. In short, We just used the gray mouth and got the lip feature by feeding it into Lipreading using Temporal Convolutional Networks. [extracted_embedding_Chem_gray](#)

4. Get the mouth ROIs from face frames

- 4.3) Save as .npy (gray) or .npz(color) file to feed into model

Lipreading using Temporal Convolutional Networks

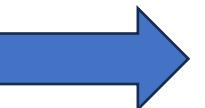


Please run

6_create_npz.py

Aim to convert
.jpg (previous step) to
"Frame.npz"

You will get



```

Please run
6_create_npz.py

Aim to convert
".jpg"(previous step) to
"Frame.npz"

You will get
How to extract embeddings
We assume you have cropped the mouth patches and put them into <MOUTH-PATCH-PATH>. The mouth
embeddings will be saved in the .npz format.

• To extract 512-D feature embeddings from the top of ResNet-18:
CUDA_VISIBLE_DEVICES=0 python main.py --modality video \
--extract-feats \
--config-path <MODEL-JSON-PATH> \
--model-path <MODEL-PATH> \
--mouth-patch-path <MOUTH-PATCH-PATH> \
--mouth-embedding-out-path <OUTPUT-PATH>
  
```

```

EYELIPCROPPER-MASTER [SSH: IC... ] 
cropper
test
boxes
extracted_embedding_Chem_gray
_7s29Q76st0-face-_7s29Q76st0-000.npy
_7s29Q76st0-face-_7s29Q76st0-001.npy
_7s29Q76st0-face-_7s29Q76st0-002.npy
_7s29Q76st0-face-_7s29Q76st0-003.npy
_7s29Q76st0-face-_7s29Q76st0-004.npy
_7s29Q76st0-face-_7s29Q76st0-005.npy
_7s29Q76st0-face-_7s29Q76st0-013.npy
_7s29Q76st0-face-_7s29Q76st0-016.npy
_7s29Q76st0-face-_7s29Q76st0-017.npy
_7s29Q76st0-face-_7s29Q76st0-018.npy
_7s29Q76st0-face-_7s29Q76st0-019.npy
_7s29Q76st0-face-_7s29Q76st0-020.npy
_7s29Q76st0-face-_7s29Q76st0-021.npy
_7s29Q76st0-face-_7s29Q76st0-022.npy
_7s29Q76st0-face-_7s29Q76st0-023.npy
_7s29Q76st0-face-_7s29Q76st0-024.npy
_7s29Q76st0-face-_7s29Q76st0-025.npy
_7s29Q76st0-face-_7s29Q76st0-027.npy
_7s29Q76st0-face-_7s29Q76st0-028.npy
_7s29Q76st0-face-_7s29Q76st0-030.npy
_7s29Q76st0-face-_7s29Q76st0-031.npy
_7s29Q76st0-face-_7s29Q76st0-032.npy
_7s29Q76st0-face-_7s29Q76st0-033.npy
_7s29Q76st0-face-_7s29Q76st0-034.npy
_7s29Q76st0-face-_7s29Q76st0-036.npy
_56-KoflBng-face-_56-KoflBng-000.npy
_56-KoflBng-face-_56-KoflBng-001.npy
_56-KoflBng-face-_56-KoflBng-002.npy
_56-KoflBng-face-_56-KoflBng-010.npy
_56-KoflBng-face-_56-KoflBng-011.npy
_56-KoflBng-face-_56-KoflBng-014.npy
_56-KoflBng-face-_56-KoflBng-010.npy
_56-KoflBng-face-_56-KoflBng-011.npy
JaKax7Jn-a-face- JaKax7Jn-a-000.npy
  
```

```

cropper
test_V2C
boxes
boxes_v1
extracted_embedding_V2C_gray
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0191_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0192_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0194_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0196_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0197_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0198_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0199_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0200_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0201_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0205_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0206_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0208_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0209_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0210_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0211_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0213_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0224_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0226_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0231_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0234_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0236_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0237_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0238_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0239_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0241_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0245_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0249_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0250_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0251_00.npy
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0254_00.npy
  
```

```

V2C_File_npz_mouth_color
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0191_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0192_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0194_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0196_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0197_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0198_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0199_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0200_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0201_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0205_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0206_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0208_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0209_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0210_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0211_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0213_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0224_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0226_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0231_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0234_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0236_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0237_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0238_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0239_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0241_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0245_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0249_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0250_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0251_00.npz
Bossbaby@BossBaby-face-Bossbaby@BossBaby_00_0254_00.npz
  
```

Acknowledge

- We sincerely appreciate the help from these open-source projects and tool packages. Thanks to these authors to open source their code!
- <https://github.com/TaoRuijie/TalkNet-ASD>
- <https://github.com/zhliuworks/EyeLipCropper>
- <https://github.com/face-analysis/emonet>
- https://github.com/mpc001/Lipreading_using_Temporal_Convolutional_Networks
- <https://montreal-forced-aligner.readthedocs.io/en/latest/>
- <https://github.com/FFmpeg/FFmpeg>
- <https://github.com/1adrianb/face-alignment>
- <https://github.com/yxlijun/S3FD.pytorch>