Final Project

DLCV Fall 2022

Update

- 12/6
  - Challenge 2 submission zip file should contain .txt file directly
- 12/10
  - Challenge 2's dataset label mapping should only consider these 200 labels (in VALID_CLASS_IDS_200, CLASS_LABELS_200, SCANNET_COLOR_MAP_200) https://github.com/RozDavid/LanguageGroundedSemseg/blob/master/lib/constants/scannet_constants.py?fbclid=IwAR0PeVzJgClUu7Td6XXi3r5UHbkc7JksxvPmfqNP2efHgd21aUQS20dPWow
- 12/17
  - Final presentation schedule & rules (see page 9, 10)

Timeline & Deadlines (GMT+8)

Teaming-up Form Completion	2022/12/02 23:59
Poster Submission	2022/12/26 11:59
Kaggle/CodaLab Submission	2022/12/29 07:59
On-Site **Presentation**	2022/12/29 13:00-17:00
GitHub Code Commit	2022/12/29 23:59

Outline

- General Rules
  - Teaming up
  - GitHub / Kaggle or Codalab / Poster / Presentation
  - Grading
- Challenges
  - **Challenge 1** - Talking to Me (TTM)
  - **Challenge 2 -** 3D Indoor Scene Long Tail Segmentation

Teaming Up and Challenge Selection

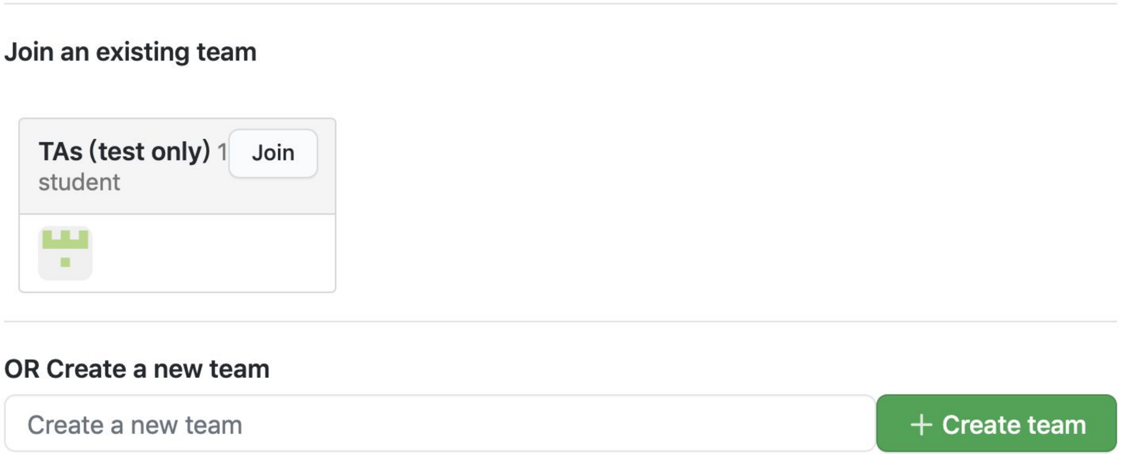
- Please fill in this form before **2022/12/02 23:59**
  - Each team should have **3-5** members (strongly recommended: 4+)
  - Team name
    - English letters (lowercase and uppercase) and numbers only; no spaces
    - **You must use the same team name for GitHub/Kaggle/CodaLab**
  - Team leader
    - Responsible for GitHub team creation and poster/code submission
- We will split the teams equally between the two challenges
  - Your topic choice will be determined by the order of form submissions

GitHub

- Join the GitHub group assignments (for each challenge) with your **team name**
  - You must use the same team name for GitHub/Kaggle/CodaLab
  - **The team leader creates the team first, and the team members join afterwards**

If you are not the team leader ⟶

If you are the team leader ⟶



Join an existing team

TAs (test only) 1 student

OR Create a new team

Create a new team

Kaggle / CodaLab

- You need to participate in the Kaggle / CodaLab challenge with your **team name**
  - Kaggle - for Challenge 1
  - CodaLab - for Challenge 2
- Maximum Daily Submissions: 5 times (for each team)
  - Kaggle and CodaLab - reset at 8am (GMT+8) every day
- **Submission Deadline: 2022/12/29 07:59**

Poster for On-Site Presentation

- **PDF format of size A1 (Portrait, 84.1 cm x 59.4 cm)**
- TAs will print it out for your on-site presentation only if you submit it before the deadline
- **Submission Deadline: 2022/12/26 11:59**
    - Submitted to the root directory of the team’s GitHub repository (format: **poster.pdf**)
    - You can leave some blank areas on your poster for further experiment results and fill them up right before the final presentation
- If you do not submit your poster before the above deadline,
  you will need to print it out on your own

On-Site Presentation

- **Schedule: 2022/12/29 13:00-17:00**
- Location: 學新館(MI Building) 2F SPACE M

13:00 - 13:20	**Challenge #1 (Talking to Me) - Poster Readying**
13:20 - 14:40	**Challenge #1 - Presentation**
14:40 - 15:00	**Tea Break / Challenge #2 (3D Indoor Scene Long Tail Segmentation) - Poster Readying**
15:00 - 15:10	**Challenge #1 - Awarding Ceremony**
15:10 - 16:30	**Challenge #2 - Presentation**
16:30 - 16:50	**Tea Break**
16:50 - 17:00	**Challenge #2 - Awarding Ceremony**

On-Site Presentation

- **Poster Readying**
    - 13:00-13:20 for Challenge-1 and 14:40-15:00 for Challenge-2
    - Prepare your posters (i.e., pasting them onto to the boards) in the given time slots
- **Presentation**
    - Proceed team-by-team according to the **Team ID** for each challenge (Find your Team ID here)
    - **Time Limit - 5 mins per team**
        - Each team will be given a maximum of 4 minutes for presentation
        - An additional 1 minute will be reserved for Q&A from the lecturer and the TAs
        - As we have a tight schedule, we will control your time strictly!
    - For each team, if no members show up for the final presentation, all team members will receive 0 points for this part (0 out of 25 points)

Code Submission

- **Code Submission Deadline: 2022/12/29 23:59**
- Submit all the training/testing code to your team's Github repository
- Provide a detailed **README.md** file with example scripts for TAs to reproduce your results (including model training and inference)
- If TAs cannot reproduce your results, you will receive 0 points in the code part (unless minor errors)

Grading

- Model Performance - Kaggle / CodaLab
  - Baseline
  - Relative ranking
- Approach & Presentation
  - Novelty and Technical Contributions
  - Completeness of Experiments
  - Poster & Oral Presentation
  - Bonus - Intra / Inter-Team Evaluation

Grading - Intra/Inter-Team Evaluation

- Intra-Team Evaluation
  - You must participate and work with your team member
  - We might adjust your final scores based on the evaluation
- Inter-Team Evaluation
  - The top 3 teams selected by (lecturer, guest, & TA) judges will receive cash prizes
  - The most-voted teams for each challenge will receive bonus points (or gifts)

Challenge 1 - Talking to Me (TTM)

GitHub Classroom Link

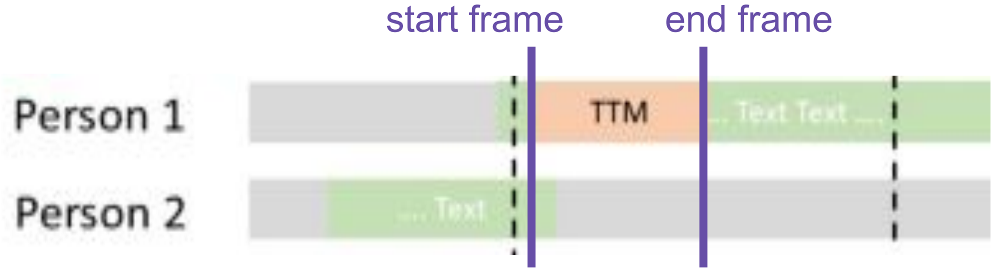
Kaggle Competition Link

(Do not join them until we announce the final topics for your teams)

Talking to Me - Task (1/2)

- Identify whether and when each visible face in a **video** is talking to the camera wearer
    - **Input**: **video** + **audio**

      + **target person** (face bounding box) + **target time period** (start/end frame id)
    - **Output**: binary prediction (whether the **specified person** is talking to me during **this period**)



start frame

end frame

Talking to Me - Task (2/2)

- This task may be challenging due to
  - Cross-modal inputs - you might need to consider both **vision** & **audio** information
  - Video data - you might need to consider **temporal** information or relations
- We provide some hints about data preprocessing and audio feature extraction in the following slides

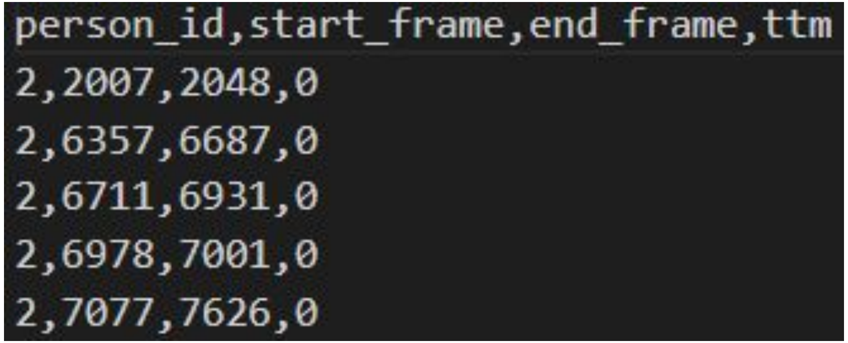
Talking to Me - Dataset (1/5)

- Following the Ego4D Challenge 2023, we use the Ego4D dataset for our challenge
- **You can directly download the dataset from our Kaggle competition**
- **Data Format**

```
student_data/
        videos/ (～25GB)   # Total 433 clips of raw video data; each clip is 5 minutes long
        train/
               /seg
               /bbox
        test/
              /seg  (without GT)
              /bbox
```

Talking to Me - Dataset (2/5)

- Format of **seg/{video_hashcode}_seg.csv**
    - person_id: which person we focus on
    - start_frame: the first frame of the time segment
    - end_frame: the last frame of the time segment
    - ttm: 1/0 indicates the person is/isn’t talking to the camera wearer (not provided in testing data)

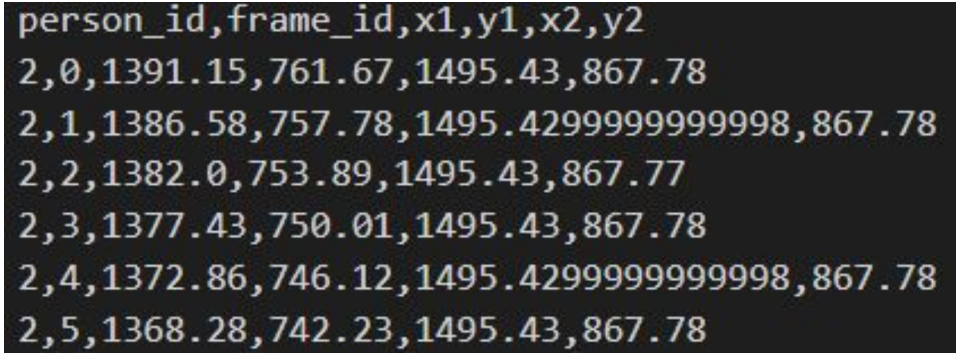


```
person_id,start_frame,end_frame,ttm
2,2007,2048,0
2,6357,6687,0
2,6711,6931,0
2,6978,7001,0
2,7077,7626,0
```

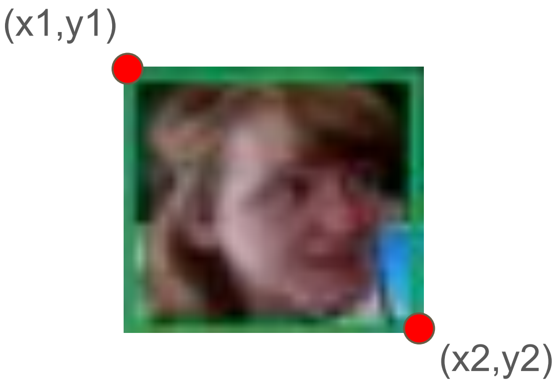
Talking to Me - Dataset (3/5)

- Format of **bbox/{video_hashcode}_bbox.csv**
  - person_id: which person we focus on
  - frame_id: which frame the bounding box at
  - x1/y1: the coordinate of the top-left point of the bounding box
  - x2/y2: the coordinate of the bottom-right point of the bounding box

    ⚠️ **If (x1,y1,x2,y2) is (-1,-1,-1,-1), this person is not detected in this frame.**

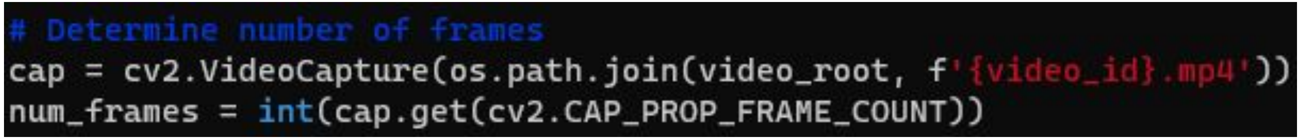


```
person_id,frame_id,x1,y1,x2,y2
2,0,1391.15,761.67,1495.43,867.78
2,1,1386.58,757.78,1495.4299999999998,867.78
2,2,1382.0,753.89,1495.43,867.77
2,3,1377.43,750.01,1495.43,867.78
2,4,1372.86,746.12,1495.4299999999998,867.78
2,5,1368.28,742.23,1495.43,867.78
```



Talking to Me - Dataset (4/5)

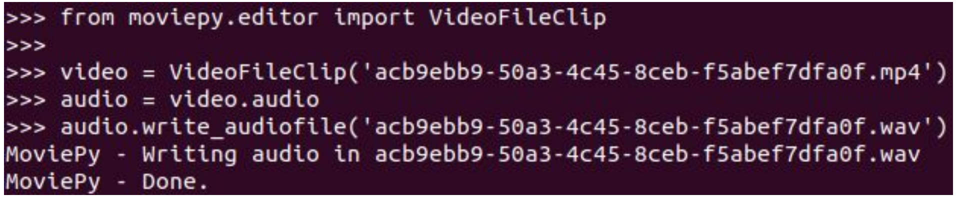
- **Video Preprocessing** - You should prepocess the video data based on the below code
  - The original frame rate of the video data is 30 frames per second
  - Please use the original video frame rate so that you can match the correct frame ids (do not down-sample the video inputs)



```
# Determine number of frames
cap = cv2.VideoCapture(os.path.join(video_root, f'{video_id}.mp4'))
num_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
```

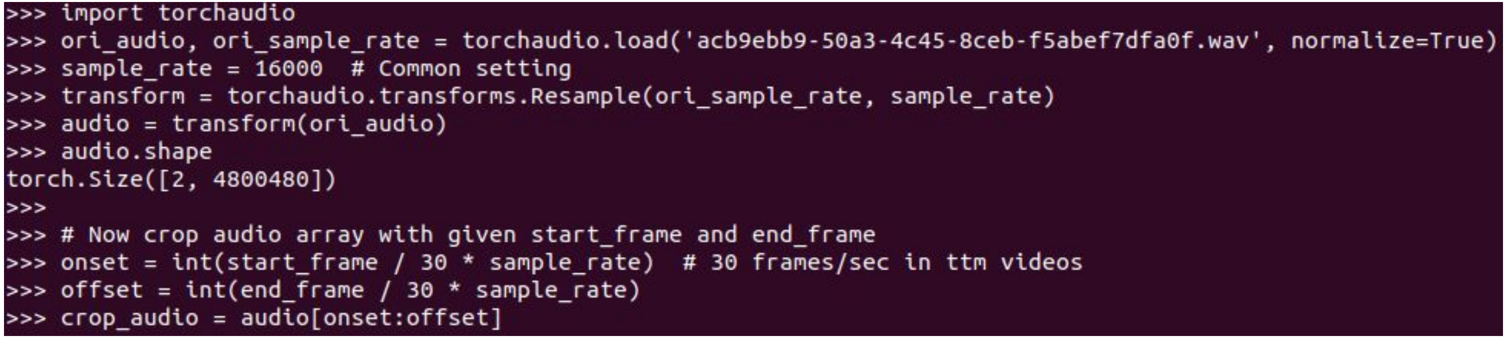
Talking to Me - Dataset (5/5)

- **Audio Preprocessing & Feature Extraction**
  - First, convert mp4 to wav



```
>>> from moviepy.editor import VideoFileClip
>>>
>>> video = VideoFileClip('acb9ebb9-50a3-4c45-8ceb-f5abef7dfa0f.mp4')
>>> audio = video.audio
>>> audio.write_audiofile('acb9ebb9-50a3-4c45-8ceb-f5abef7dfa0f.wav')
MoviePy - Writing audio in acb9ebb9-50a3-4c45-8ceb-f5abef7dfa0f.wav
MoviePy - Done.
```

  - Second, obtain the audio segment with the start/end frame id from wav

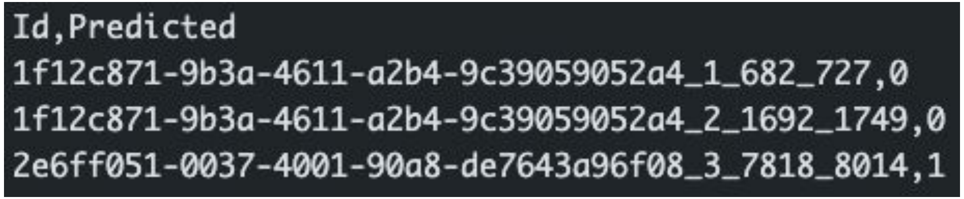


```
>>> import torchaudio
>>> ori_audio, ori_sample_rate = torchaudio.load('acb9ebb9-50a3-4c45-8ceb-f5abef7dfa0f.wav', normalize=True)
>>> sample_rate = 16000  # Common setting
>>> transform = torchaudio.transforms.Resample(ori_sample_rate, sample_rate)
>>> audio = transform(ori_audio)
>>> audio.shape
torch.Size([2, 4800480])
>>>
>>> # Now crop audio array with given start_frame and end_frame
>>> onset = int(start_frame / 30 * sample_rate)  # 30 frames/sec in ttm videos
>>> offset = int(end_frame / 30 * sample_rate)
>>> crop_audio = audio[onset:offset]
```

  - Finally, you can extract audio features with some widely-used methods such as **MFCC**

Talking to Me - Evaluation

- Quantitative Metric Evaluation
  - You should submit your csv file to the Kaggle competition (with your team name)
  - Max submissions per day: 5 (per team)
  - We will use the **classification accuracy** to quantify your model performance
- **Submission format** (same as **sample_submission.csv** on Kaggle)
  - **Id**: {video hashcode}_{which person}_{start frame}_{end frame}
  - **Predicted**: 1/0 if the person is/isn’t talking to the camera wearer



```
Id,Predicted
1f12c871-9b3a-4611-a2b4-9c39059052a4_1_682_727,0
1f12c871-9b3a-4611-a2b4-9c39059052a4_2_1692_1749,0
2e6ff051-0037-4001-90a8-de7643a96f08_3_7818_8014,1
```

Talking to Me - Grading

- **Final 34%** (Bonus up to **3%**)
  - **Model Performance - Kaggle 9%**
    - Baseline **4%**
    - Relative ranking in class **5%**
  - **Approach & Presentation 25% + 3%**
    - Novelty and technical contributions **10%**
    - Completeness of experiments **10%**
      (e.g., ablation study, visualization, etc.)
    - Poster & Oral Presentation **5%**
    - Bonus (intra / inter-team evaluation) up to **3%**

Points	**Team Ranking**
5	top 0% - 20%
4.5	top 20% - 40%
4	top 40% - 60%
3.5	top 60% - 80%
3	top 80% - 100%

Talking to Me - Rules

- Do not download any train/validation/test data or use any pretrained weight related to the TTM challenge from Ego4D
  - However, the well known public pretrained models or external datasets (e.g., COCO or ImageNet) are allowed to use. If you want to use some pretrained weights or datasets but not sure its legitimacy, please feel free to ask TAs
- Do not disclose the dataset
- Your results need to be reproducible with your submitted code and models
- Please use **python3** instead of python for your scripts
- Any violation would result in 0 score for your final project

Challenge 2 -
3D Indoor Scene Long Tail Segmentation

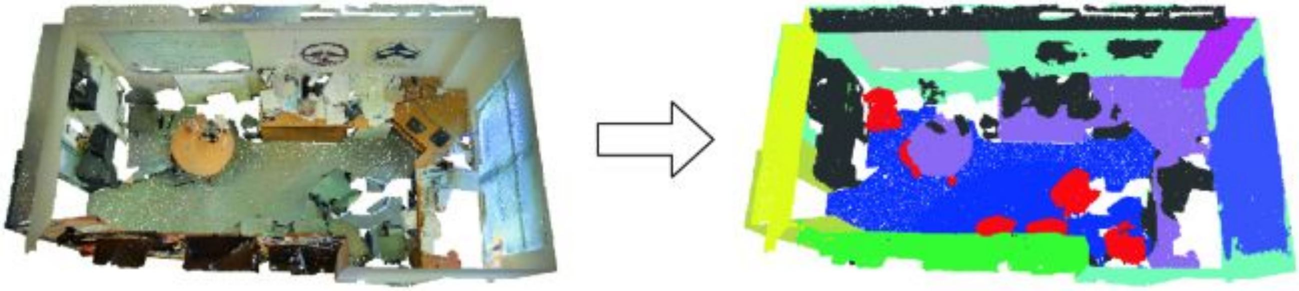
GitHub Classroom Link

CodaLab Competition Link

(Do not join them until we announce the final topics for your teams)

ScanNet200 - Task

- Goal
    - Train a neural network to conduct **3D indoor scene semantic segmentation.**
    - **Input:** 3D point cloud scene including **XYZ position** and **RGB color (optional)** for each point
    - **Output:** Semantic class label for each point

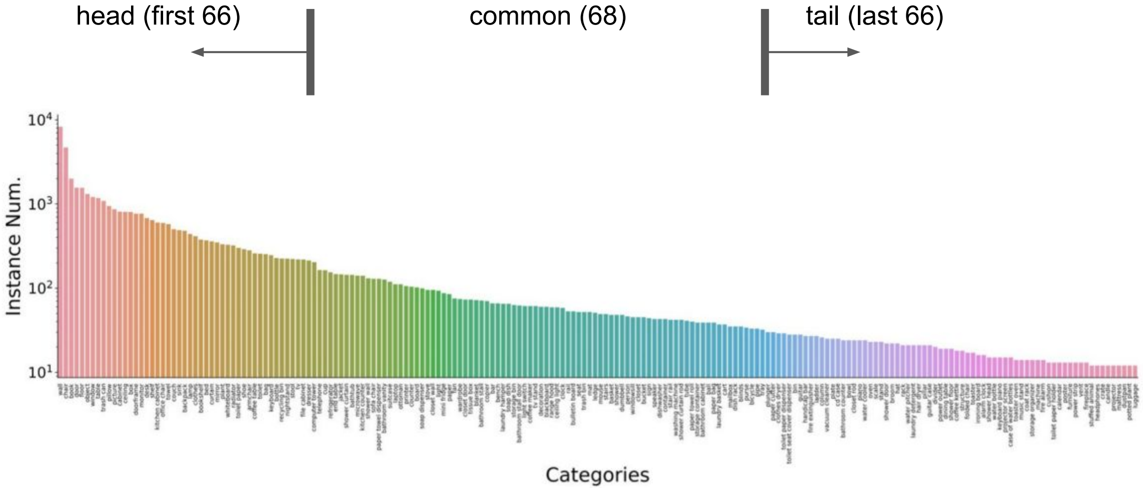


Input: 3D Point Cloud

Output: Semantic Segmentation

ScanNet200 - Dataset (1/4)

- ScanNet200 Benchmark
  - Fine-grained 200-class 3D semantic segmentation
  - Imbalanced data distribution (e.g. floor points are seen much more than fire extinguisher points)



head (first 66)

common (68)

tail (last 66)

ScanNet200 - Dataset (2/4)

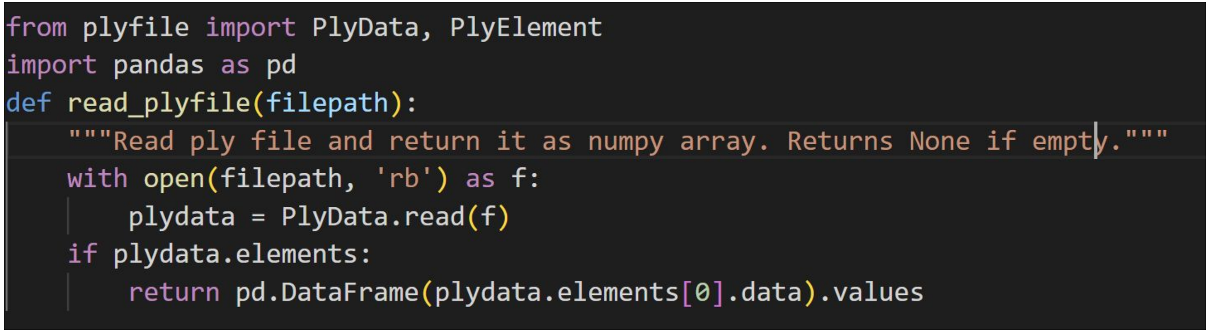
- You can directly download dataset on the CodaLab competition.

- **Data Format**

```
challenge2_data/
        scannet200/ (～3.6GB)   # Total 1059 scenes for training, 112 scenes for testing
        train/
               /scene0XXX_XX.ply  # Every .ply file contains a scene
              …
        test/
              /scene0XXX_XX.ply
             …
        sample_submission.zip   # Contains the sample submission to Codalab
        train.txt               # Training scene filenames
        test.txt                # Test scene filenames
```

ScanNet200 - Dataset (3/4)

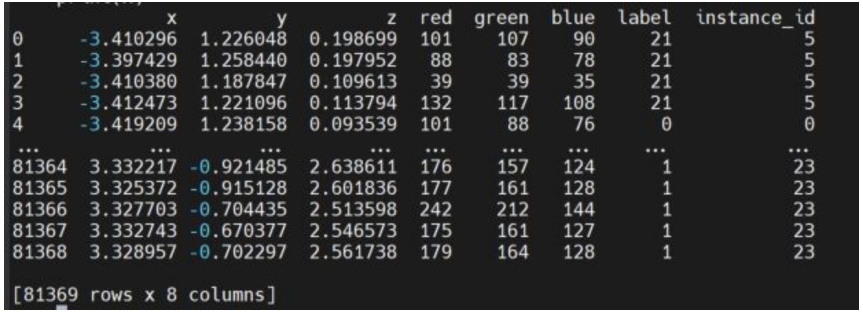
- You can read the .ply file as below (with plyfile, pandas package)



```python
from plyfile import PlyData, PlyElement
import pandas as pd
def read_plyfile(filepath):
    """Read ply file and return it as numpy array. Returns None if empty."""
    with open(filepath, 'rb') as f:
        plydata = PlyData.read(f)
    if plydata.elements:
        return pd.DataFrame(plydata.elements[0].data).values
```

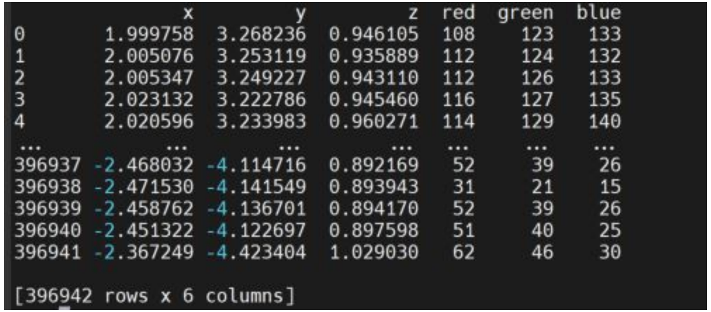
ScanNet200 - Dataset (4/4)

- The .ply file in train/ folder have 8 columns
  - each row contains a point’s xyz value and it’s class label (which you should predict)
  - feel free to use rgb value as feature input
  - instance_id indicates different instances in a same object class, you don’t necessary need to use that
- The .ply file in test/ folder have 6 columns (without label)
  - each row contains xyz value without class label



```
              x         y         z  red  green  blue  label  instance_id
0     -3.410296  1.226048  0.198699  101    107    90     21            5
1     -3.397429  1.258440  0.197952   88     83    78     21            5
2     -3.410380  1.187847  0.109613   39     39    35     21            5
3     -3.412473  1.221096  0.113794  132    117   108     21            5
4     -3.419209  1.238158  0.093539  101     88    76      0            0
...         ...       ...       ...  ...    ...   ...    ...          ...
81364  3.332217 -0.921485  2.638611  176    157   124      1           23
81365  3.325372 -0.915128  2.601836  177    161   128      1           23
81366  3.327703 -0.704435  2.513598  242    212   144      1           23
81367  3.332743 -0.670377  2.546573  175    161   127      1           23
81368  3.328957 -0.702297  2.561738  179    164   128      1           23

[81369 rows x 8 columns]
```

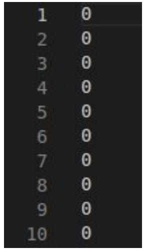


```
               x         y         z  red  green  blue
0       1.999758  3.268236  0.946105  108    123   133
1       2.005076  3.253119  0.935889  112    124   132
2       2.005347  3.249227  0.943110  112    126   133
3       2.023132  3.222786  0.945460  116    127   135
4       2.020596  3.233983  0.960271  114    129   140
...          ...       ...       ...  ...    ...   ...
396937 -2.468032 -4.114716  0.892169   52     39    26
396938 -2.471530 -4.141549  0.893943   31     21    15
396939 -2.458762 -4.136701  0.894170   52     39    26
396940 -2.451322 -4.122697  0.897598   51     40    25
396941 -2.367249 -4.423404  1.029030   62     46    30

[396942 rows x 6 columns]
```

ScanNet200 - Evaluation (1/4)

- Quantitative Metric Evaluation
  - We will use the overall **Intersection over Union (IoU)** to quantify your model performance, i.e., we calculate the average of mIoU from head, common, and tail classes
  - You should submit your **zip** file to the CodaLab competition
  - Please note that the **zip** file should **ONLY** contain .txt files
- **Submission format** (same as **sample_submission.zip** on CodaLab)
  - sample_submission.zip
    - scene0500_00.txt
    - scene0500_01.txt
    - ……



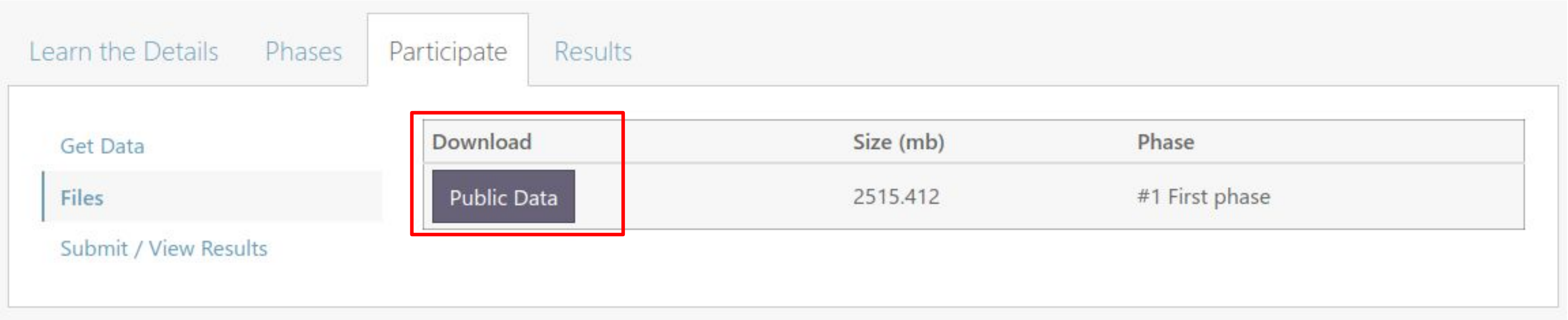
```
0
0
0
0
0
0
0
0
0
0
```

Each .txt contain one scene, one line for a point (with points order as the corresponding .ply file)

Each line containing a integer label of the predicted class.

ScanNet200 - Evaluation (2/4)

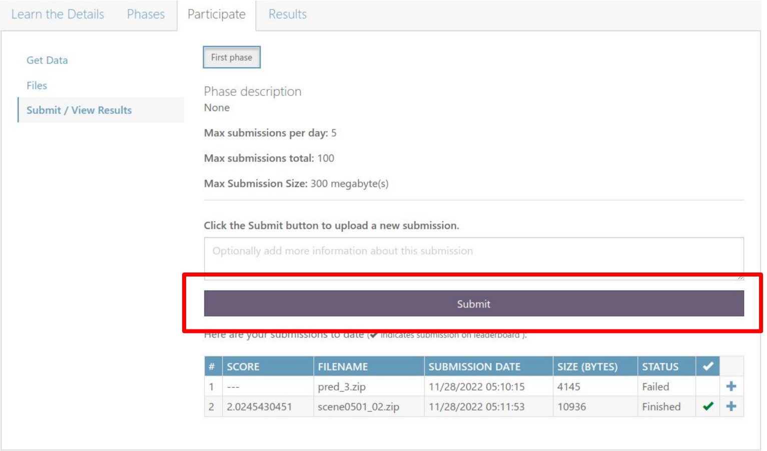
- Create an account and participate the competition (with your team name)
  - CodaLab Competition
- Download the dataset from the link or the **Files** page.
  - Download dataset



Download	Size (mb)	Phase
Public Data	2515.412	#1 First phase

ScanNet200 - Evaluation (3/4)

- Submit you .zip prediction file in the Submit/View Results page.
- You can check your score and ranking in the Results page.
- After you upload the zip file to the "Submit/View Results" page. You can wait for the evaluation.
    - The submission status should go as "Submitting" -> "Submitted" -> "Running" -> "Finished".
    - Click "Refresh status" too many times may cause the submission stuck at "Submitted"!
    - If it has been stuck at the "Submitted" status for a long time (more than 30 minutes), you can e-mail TAs to re-run your submission.



Phase description
None

Max submissions per day: 5

Max submissions total: 100

Max Submission Size: 300 megabyte(s)

Click the Submit button to upload a new submission.

Optionally add more information about this submission

Here are your submissions to date (✓ indicates submission on leaderboard).

#	SCORE	FILENAME	SUBMISSION DATE	SIZE (BYTES)	STATUS	✓	
1	---	pred_3.zip	11/28/2022 05:10:15	4145	Failed		+
2	2.0245430451	scene0501_02.zip	11/28/2022 05:11:53	10936	Finished	✓	+

ScanNet200 - Evaluation (4/4)

- Submission deadline: Dec. 29, 2022, 07:59 a.m. (**UTC+8**)
- Max submissions per day: 5 (per team)
    - **IMPORTANT:** Please note that failed submission is also counted as one submission

ScanNet200 - Grading

- **Final 34%** (Bonus up to **3%**)
    - **Model Performance - CodaLab 9%**
        - Baseline **4%** (overall IoU of 10%)
        - Relative ranking in class **5%**
    - **Approach & Presentation 25% + 3%**
        - Novelty and technical contributions **10%**
          (e.g., how to deal with long-tailed class distribution)
        - Completeness of experiments **10%**
          (e.g., ablation study, visualization, etc.)
        - Poster & Oral Presentation **5%**
        - Bonus (intra / inter-team evaluation) up to **3%**

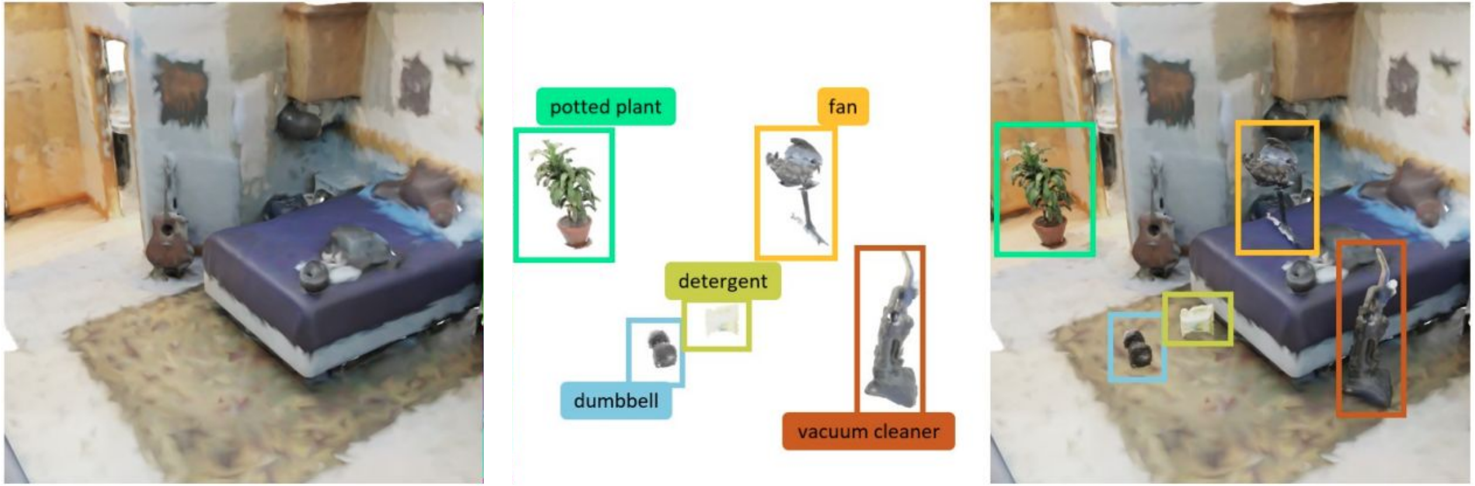
Points	**Team Ranking**
5	top 0% - 20%
4.5	top 20% - 40%
4	top 40% - 60%
3.5	top 60% - 80%
3	top 80% - 100%

ScanNet200 - Rules

- Do not use any external data and pretrained models related to 3D semantic segmentation
- Do not disclose the dataset
- Your results need to be reproducible with your submitted code and models
- Please use **python3** instead of python for your scripts
- Any violation would result in 0 score for your final project

ScanNet200 - Hints

- You can consider different loss functions to handle imbalanced data
  - E.g., focal loss
- You can try different data augmentation techniques



Original scan

Sampled instances

Augmented scan with sampled instances