Homework-1
Instrument activity detection

HW1 TA：李維釗 (Lonian Lee)

Office hour: Thu. 13:20 ~ 14:05 @BL505

Outline

- Rules
- Overview
- Timeline
- Detailed Explanation
- Scoring
- Submission

Rules

- Don’t cheat
- Don’t use extra data
- Can use public codes with citation in report

Overview

1. **Multi-label music tagging transfer learning**:
    - Self-supervise representation learning
    - Implementation of transfer learning downstream task
    - (Huggingface practice)

Timeline

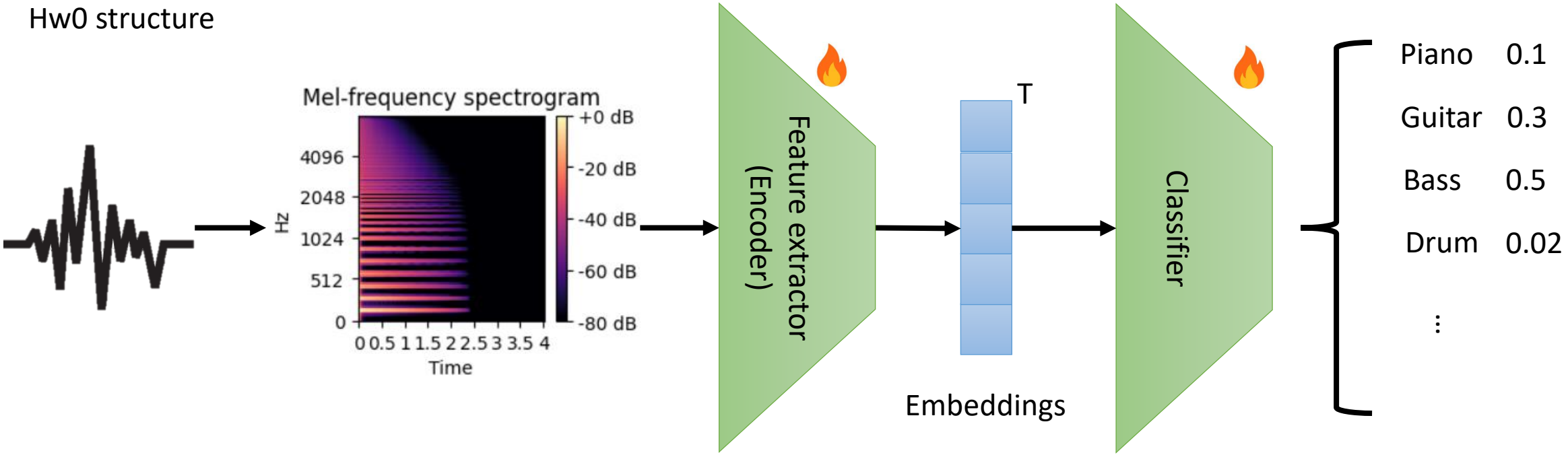
- W3 – 09/19 (Thursday): Announcement of HW1
- W5 – 10/02 (Wednesday 23:59pm): Deadline
  - Late submission: 1 day (-20%), 2 days (-40%), after that (-60%)

Detailed Explanation

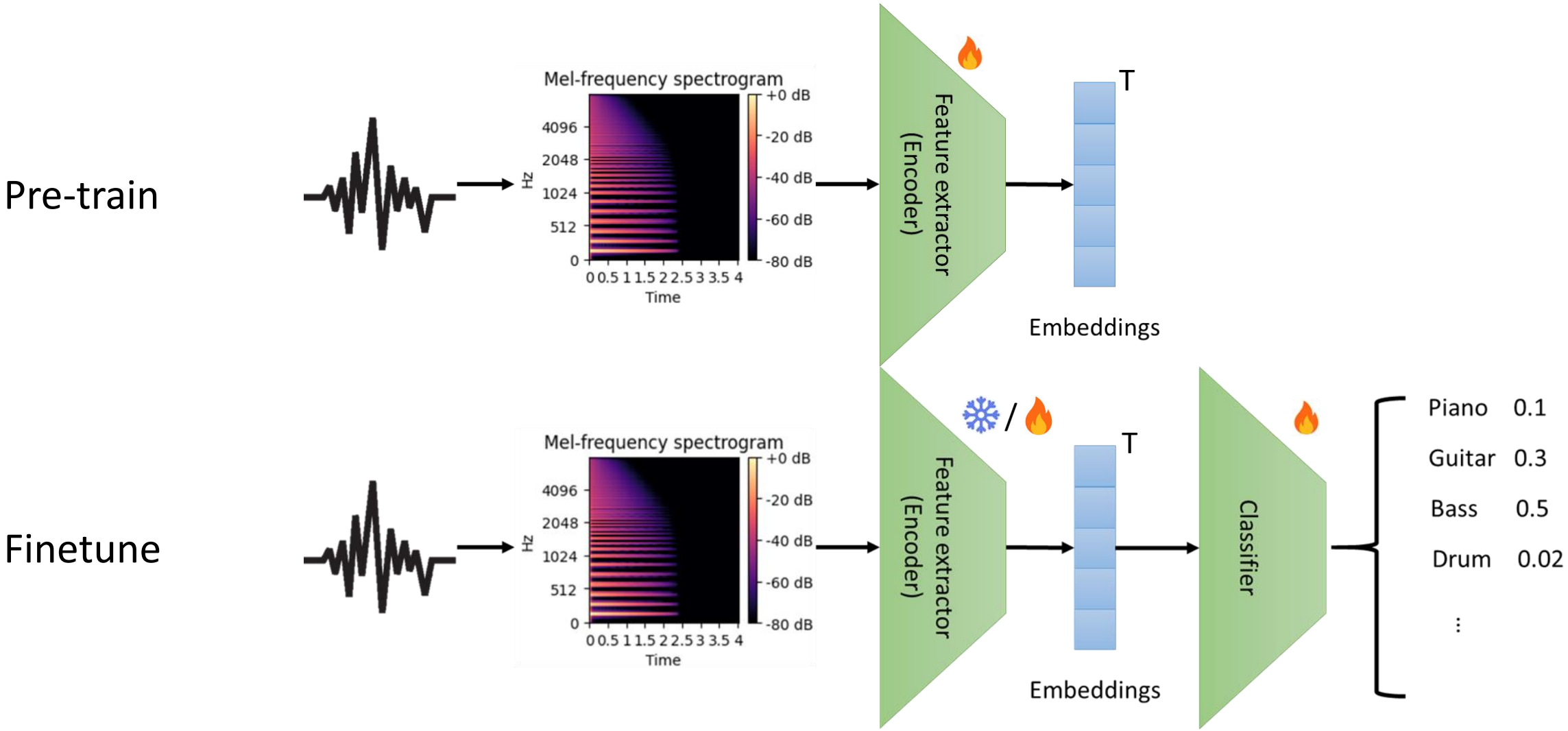
- Transfer learning
- Dataset
- Task
- Huggingface
- Evaluation

Transfer learning

Hw0 structure

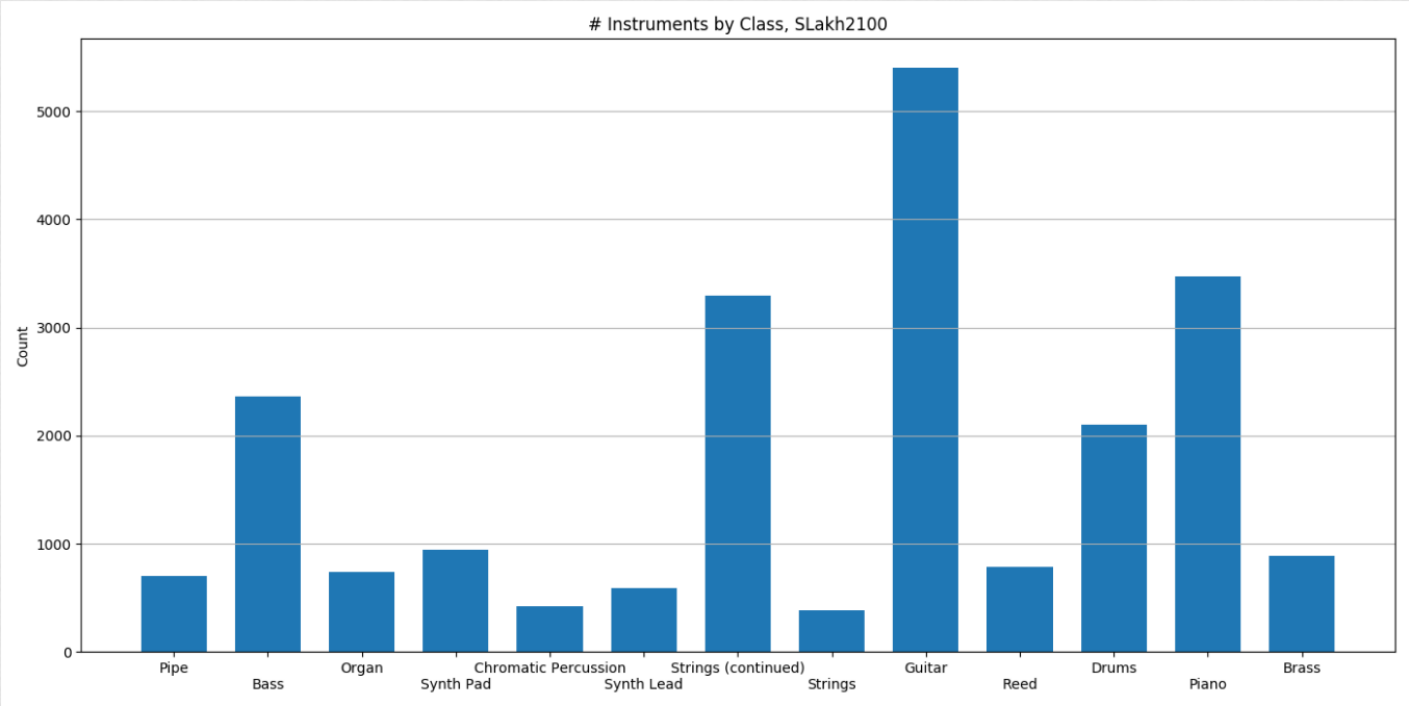


Transfer learning



Dataset: Slakh

- 1500 tracks for training
  375 tracks for validation
  225 tracks for testing
- Total length is about 145 hours (101G)
- More detailed information please refer the source




```
@inproceedings{manilow2019cutting,
  title={Cutting Music Source Separation Some {Slakh}: A Dataset to Study the Impact of Training Data Quality and Quantity},
  author={Manilow, Ethan and Wichern, Gordon and Seetharaman, Prem and Le Roux, Jonathan},
  booktitle={Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)},
  year={2019},
  organization={IEEE}
}
```

Dataset: Slakh (released version)

- Input:
  5 seconds audio saved as numpy array in .npy file
- Label:
  Dictionary data: {''File_name.npy'': multi-hot label list}

```
Track00001
    └── all_src.mid
    └── metadata.yaml
    └── MIDI
    │    └── S01.mid
    │    │    ...
    │    └── SXX.mid
    └── mix.flac
    └── stems
         └── S01.flac
         │    ...
         └── SXX.flac
```

Original dataset structure

```
"Track01950_1.npy": [
    0,
    0,
    0,
    1,
    0,
    0,
    0,
    0,
    0
],
```

Labels data

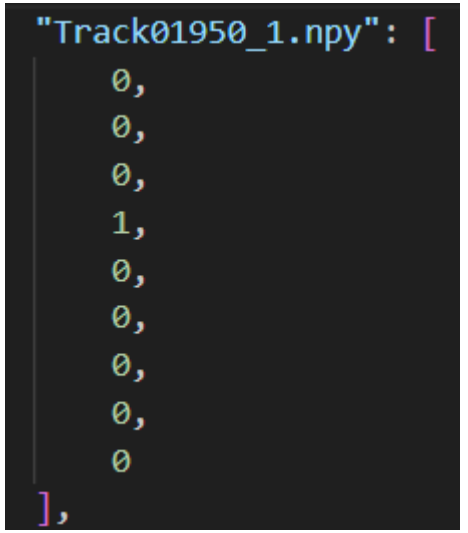
```
slakh
├── train
│    ├── Tracks_ids_0.npy
│    ├── Tracks_ids_1.npy
│    ├── Tracks_ids_2.npy
│    └── ...
├── validation
│    ├── Tracks_ids_0.npy
│    ├── Tracks_ids_1.npy
│    ├── Tracks_ids_2.npy
│    └── ...
├── test
│    ├── Tracks_ids_0.npy
│    ├── Tracks_ids_1.npy
│    ├── Tracks_ids_2.npy
│    └── ...
├── train_labels.json
├── validation_labels.json
└── test_labels.json
```

Released sub-dataset structure

Dataset: Slakh (released version)

- In this homework you need to train models to predict the instruments activity
- Label categories:

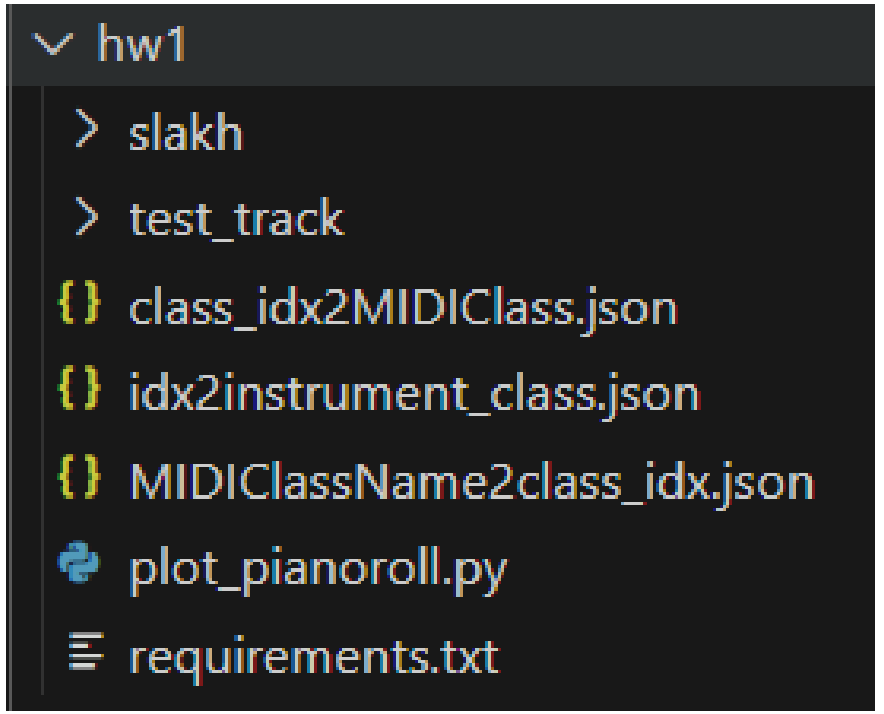
```
[
        'Piano', 'Percussion', 'Organ', 'Guitar', 'Bass',
        'Strings', 'Voice', 'Wind Instruments', 'Synth'
]
```



Labels data

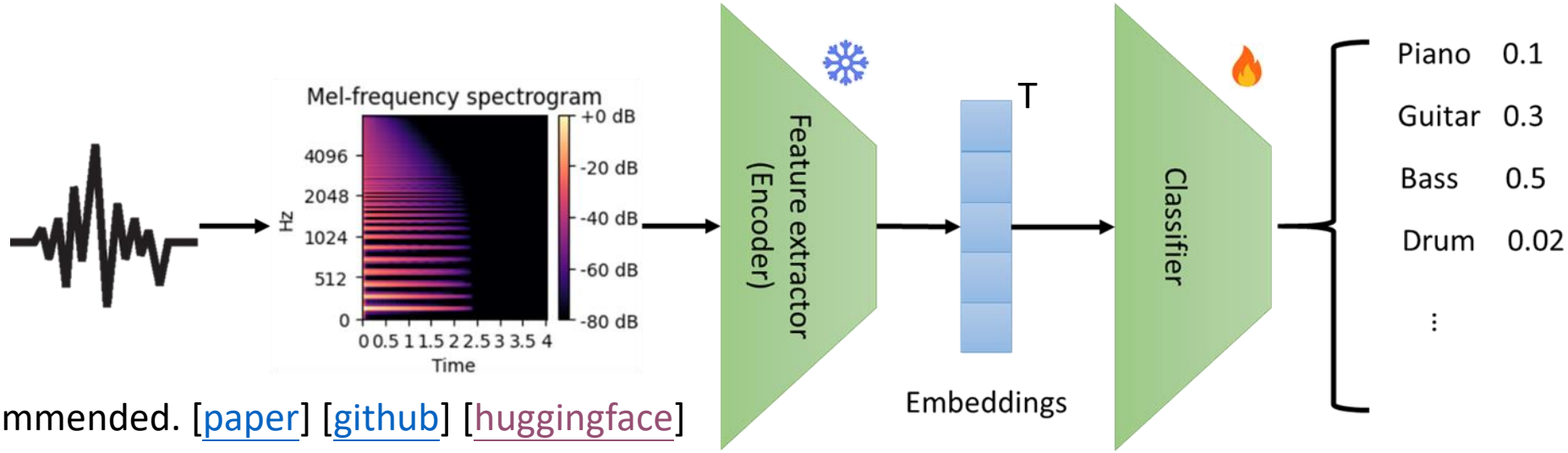
TA supports

- Download the dataset and code here [link]
- Train: 14994 clips
  Validation: 3747 clips
  Test: 2247 clips
- 5 seconds / clip with sampling rate = 24000



Task

- Use a self-supervise audio encoder model and a classifier to implement **multi-label instruments classification**
- Please predict the instruments activity every **5 seconds**

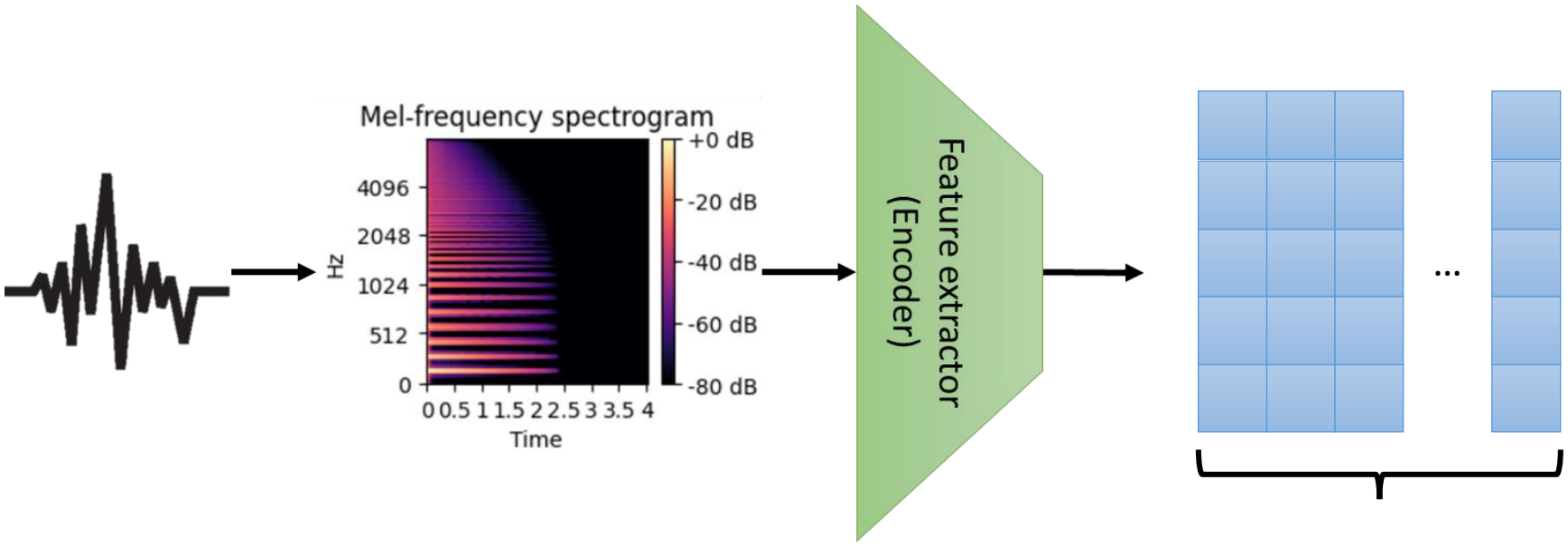


MERT is recommended. [paper] [github] [huggingface]

Yizhi Li and Ruibin Yuan and Ge Zhang and Yinghao Ma and Xingran Chen and Hanzhi Yin and Chenghua Lin and Anton Ragni and Emmanouil Benetos and Norbert Gyenge and Roger Dannenberg and Ruibo Liu and Wenhu Chen and Gus Xia and Yemin Shi and Wenhao Huang and Yike Guo and Jie Fu.
“MERT: Acoustic Music Understanding Model with Large-Scale Self-supervised Training”, 2023

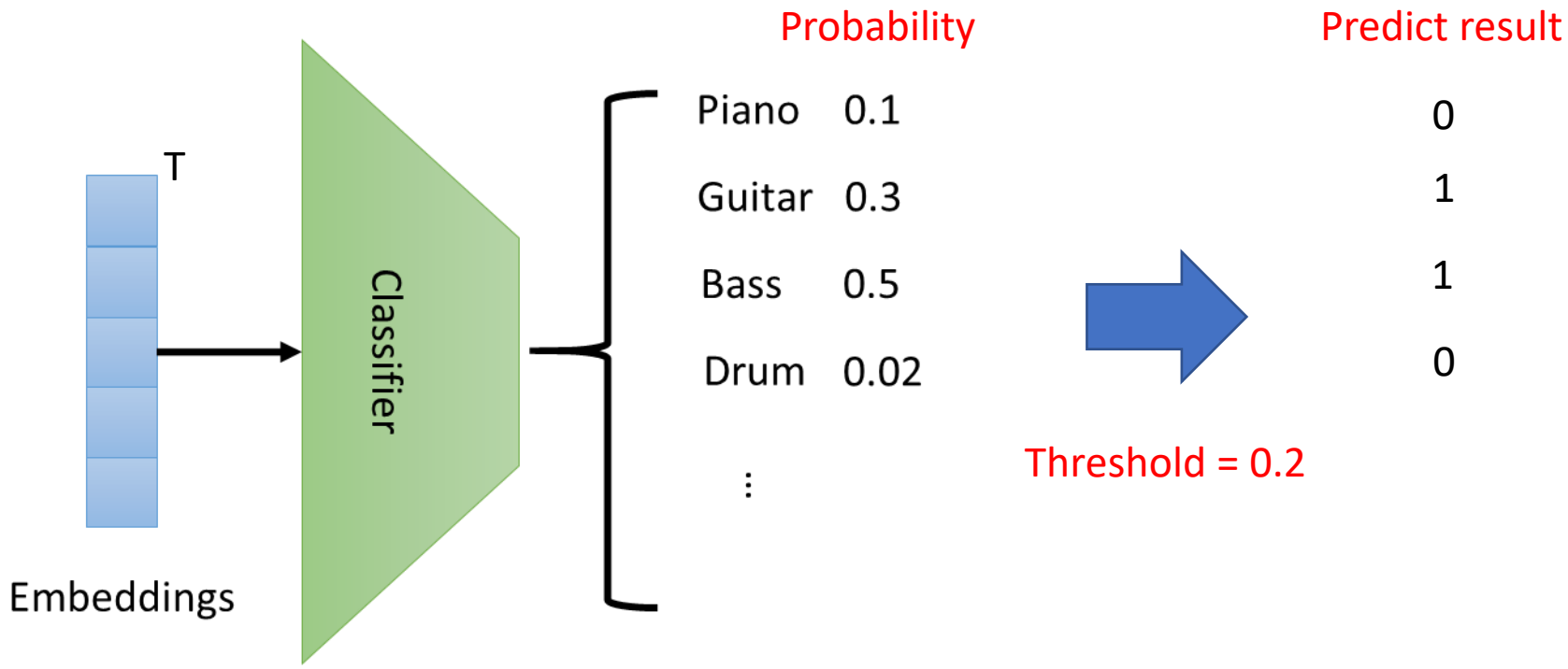
Hint 1: Pooling

- Need to do pooling along the time axis



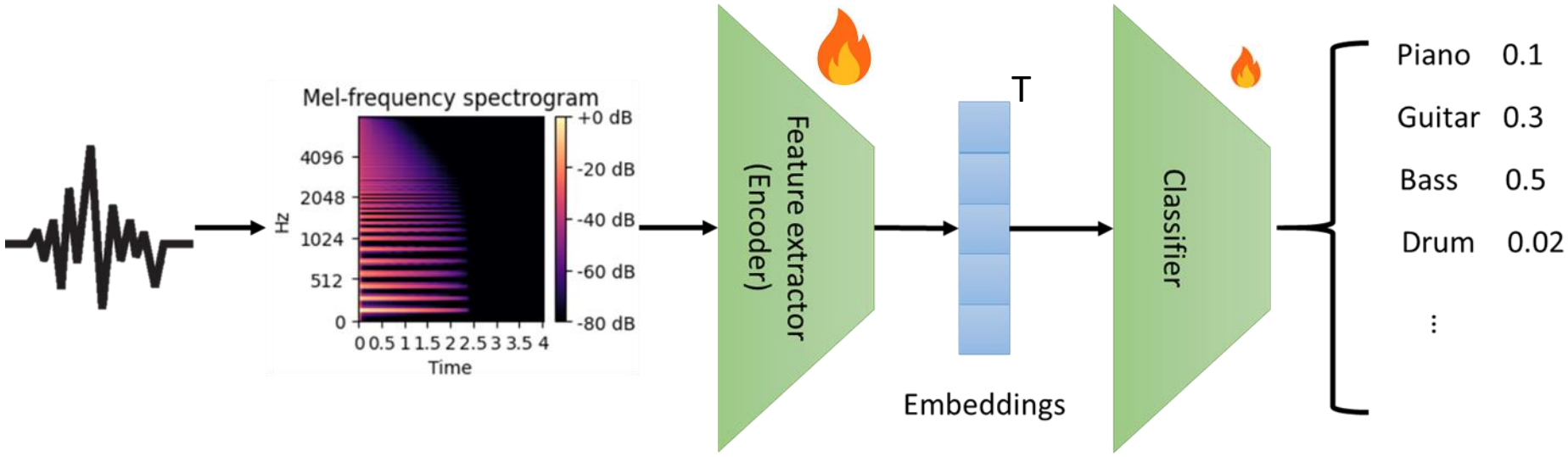
Hint 2: Probability Threshold

- Need to set a threshold to map the probability to {0, 1}



Hint 3: Fine-tune strategy (advanced)

- You can try to set your encoder trainable when using slakh dataset training your classifier.



Huggingface

- You can click here to get more info
- 3 main components
  - Model config & model
  - Preprocessing
  - Output
- For Dummies
  1. Choose a model
  2. Copy the **Model Usage** part in the page

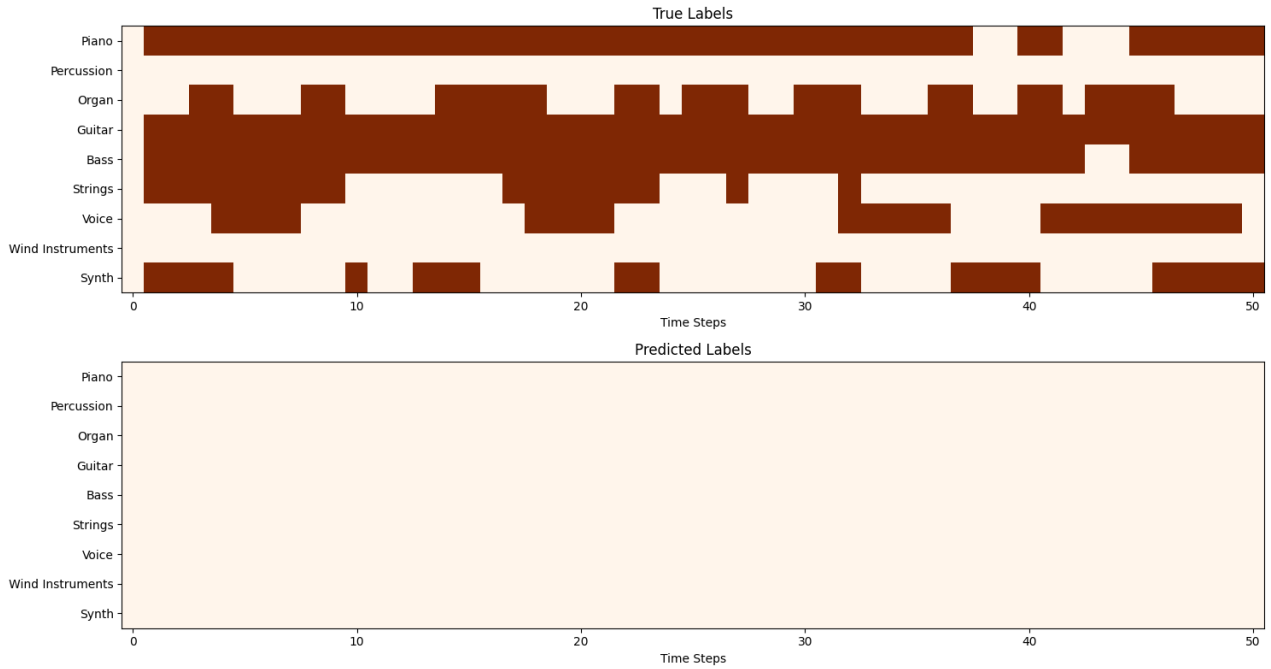
Name	Pre-train Paradigm	Training Data (hour)	Pre-train Context (second)	Model Size	Transformer Layer-Dimension	Feature Rate	Sample Rate	Release Date
MERT-v1-330M	MLM	160K	5	330M	24-1024	75 Hz	24K Hz	17/03/2023
MERT-v1-95M	MLM	20K	5	95M	12-768	75 Hz	24K Hz	17/03/2023
MERT-v0-public	MLM	900	5	95M	12-768	50 Hz	16K Hz	14/03/2023
MERT-v0	MLM	1000	5	95 M	12-768	50 Hz	16K Hz	29/12/2022
music2vec-v1	BYOL	1000	30	95 M	12-768	50 Hz	16K Hz	30/10/2022

```
# loading our model weights
model = AutoModel.from_pretrained("m-a-p/MERT-v1-330M", trust_remote_code=True)
# loading the corresponding preprocessor config
processor = Wav2Vec2FeatureExtractor.from_pretrained("m-a-p/MERT-v1-330M",
trust_remote_code=True)
```

Evaluation

1. Classification report [ref]
   A table that summarizes the performance of the results by precision, recall, f1-score.
2. Instruments activity Comparison (5) [TA support code]

	precision	recall	f1-score	support
1	1.00	0.67	0.80	3
2	0.00	0.00	0.00	0
3	0.00	0.00	0.00	0
micro avg	1.00	0.67	0.80	3
macro avg	0.33	0.22	0.27	3
weighted avg	1.00	0.67	0.80	3



Scoring

- HW1 accounts for 15% of the total grade.
  - Report: 100%

Submission file and details

1. Report (to NTU cool)
2. Readme file and requirements.txt (to your cloud drive)
3. Code and one model checkpoint for inference. (to your cloud drive)

- We will randomly select several classmates' code to run inference on your model and run the score on your results, so please ensure that the files you upload include trained model which can successfully execute the entire inference process.
- Don’t upload: training data, testing data, preprocessed data, others model, cache file

Report

- Write with PPT or PPT-like format (16:9)
- Upload studentID_report.pdf (ex: r12345678_report.pdf)
- Please create a report that is clear and can be understood without the need for oral explanations.
- There is no specific length requirement, but it should clearly communicate the experiments conducted and their results. Approximately 10 pages is a suggested standard, but not a strict limitation.

Report template

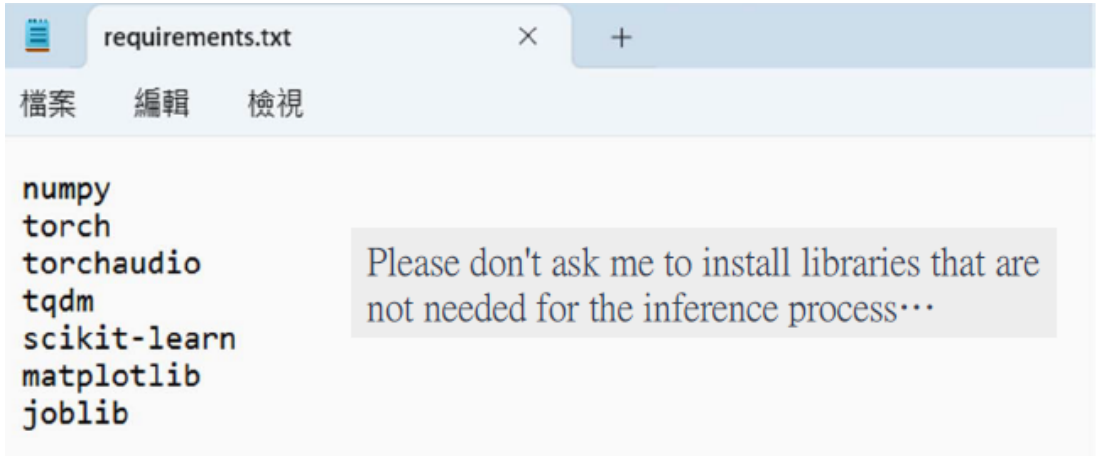
- **Cover page**: your name, student ID etc
- **Novelty highlight** (one page; optional): what's special about your work?
- **Methodology highlight** (one page): how did you make it?
  Or, list the attempts you have made
- **Result highlight** (one page): result on your test set
- **Findings highlight** (one page): main takeaways of your study
- **Details of your approach** (multi-pages): If you use open source code, you may want to read some of the associated paper(s) and summarize your understanding of the paper(s) (e.g., why it works)
- **Result analysis & discussion** (multi-pages)

Code

- Upload all your **source code and model** to a cloud drive, **open access permissions**, and then **upload the link to the NTU Cool assignment HW1_report in comments**, as well as include it on the first page of the report.
- You will need to upload requirements.txt
- I'll run :

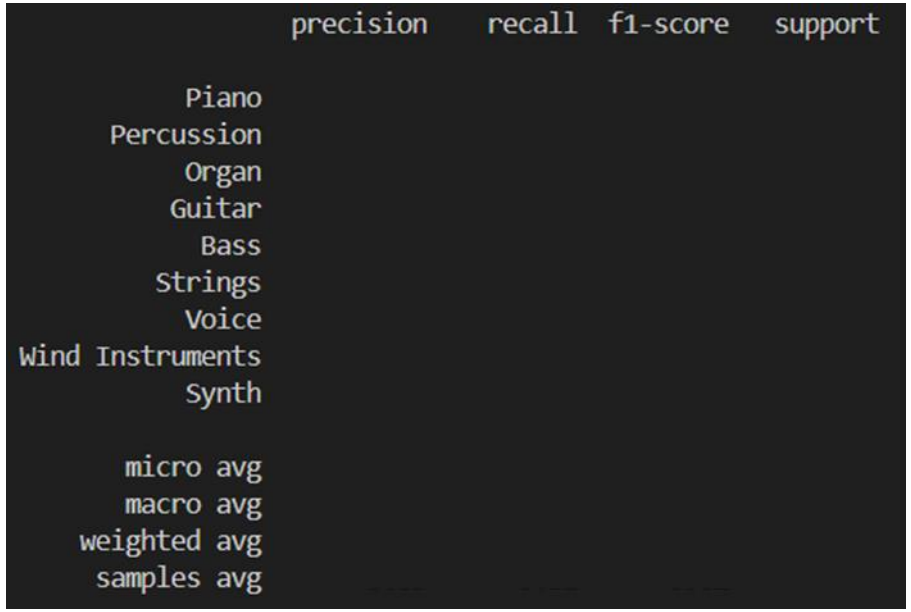
```
pip install -r requirements.txt
```

If you have used third-party programs that cannot be installed directly via 'pip install,' please write the URL and install method command by command on **your readme file**.



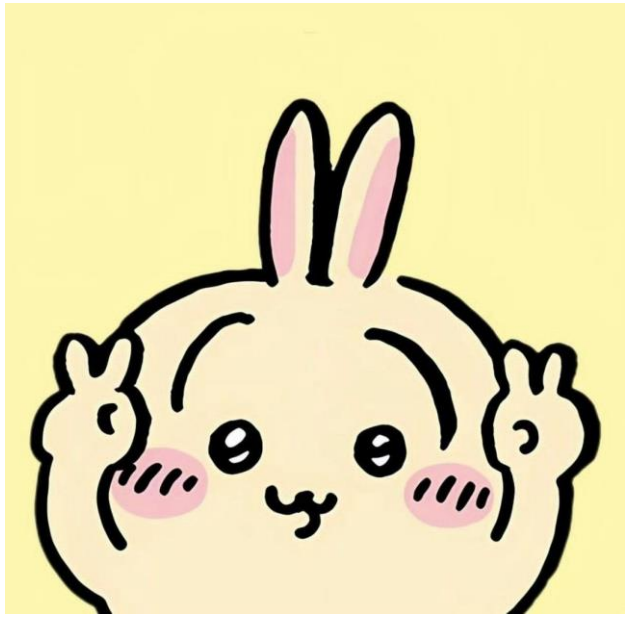
Code

- You will also need to upload README.txt or README.pdf to guide me on how to perform inference on your model. (I'll use the same test set in released Slakh sub-dataset, ensure I can run your code with the test set path in my device.) (denote how to modify the path in readme file is okay!)
- The inference code should **print the prediction result on test set.**



ALL things you need to do before 10/02 23:59

- HW1_report
  - StudentID_report.pdf
  - Cloud drive link
    - README.txt or README.pdf
    - Requirements.txt
    - Codes and model to run inference
    - Others codes



When you encounter problem:

1. Check out all course materials and announcement documents
2. Use the power of the internet and AI
3. Use **Discussions** on NTU COOL
4. Email me weijaw2000@gmail.com or come to office hour