Homework-2
Source Separation

HW2 TA：艾芯 (Ivy Ai)

Office hour: Tue. 14:00 ~ 15:30 @BL505

Outline

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

Rules

- Don’t cheat
- Don’t use pre-trained model for required tasks
- Don’t use extra data
- Don’t use the test data while training your model
- Can use public codes with citation in report

Timeline

- W5 – 10/03 (Thursday): Announcement of HW2
- W8 – 10/23 (Wednesday 23:59pm): Deadline
    - Late submission: 1 day (-20%), 2 days (-40%), after that (-60%)

Overview

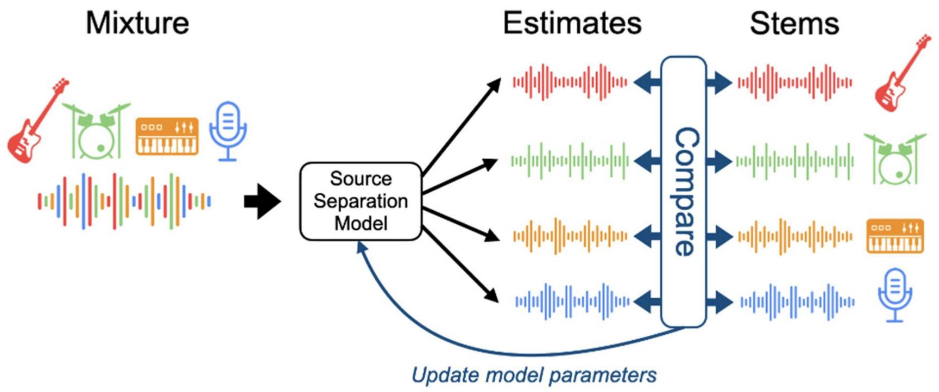
1. Train a source separation model from scratch
   - Two stems: vocal v.s. non-vocal
2. Predict phase with Griffin-Lim algorithm rather than using the mixture phase

Detailed Explanation

- Source Separation
- Dataset
- Required Task
- Optional Task
- Evaluation
- Training Tips
- TA Experience

Two-stems source separation

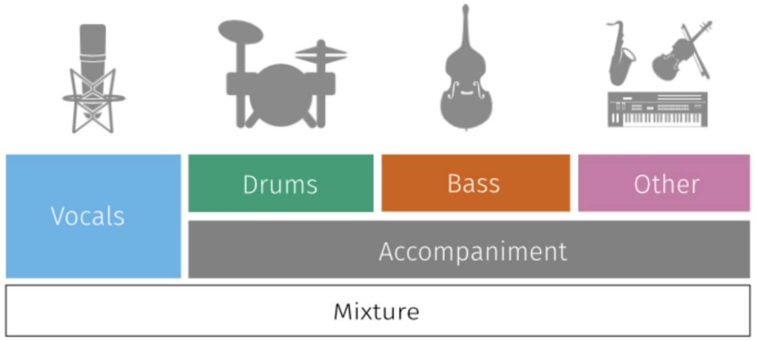
- In this homework, you need to train a source separation model to separate vocal/non-vocal
- Learn from paired data of {mixture, stems}



Dataset: MUSDB18

- 150 full lengths music tracks (10h duration)
  - Training: 100 tracks
    (Include 14 tracks for validation in Open-Unmix)
  - Testing: 50 tracks

- All signals are stereophonic and encoded at 44.1kHz
- For more detail please refer to the source



Dataset: MUSDB18

Download link: MUSDB18 MUSDB18-HQ

Version 1: MUSDB18

- Compressed MP4
- 4.7GB

- `0` - The mixture,
- `1` - The drums,
- `2` - The bass,
- `3` - The rest of the accompaniment,
- `4` - The vocals.

Version 2: MUSDB18-HQ

- Uncompressed WAV
- 22.7GB

- mixture.wav
- drums.wav
- bass.wav,
- other.wav,
- vocals.wav

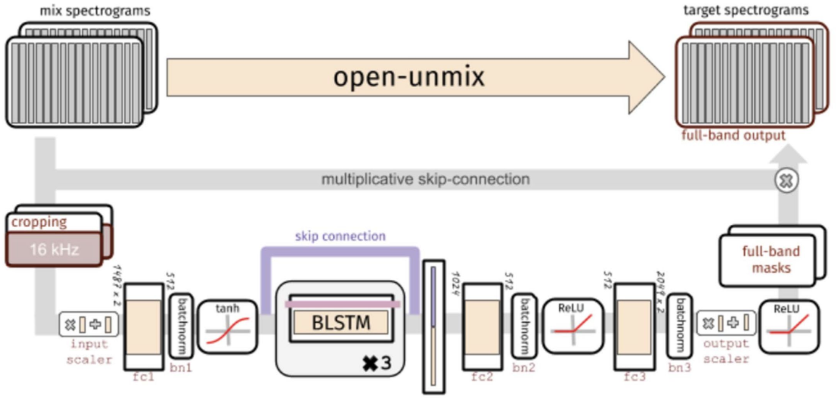
You can choose either MUSDB18 or MUSDB18-HQ in this homework.

```
@misc{musdb18, author = {Rafii, Zafar and Liutkus, Antoine and Fabian-Robert St{\"o}ter and Mimilakis, Stylianos Ioannis and Bittner, Rachel}, title = {The {MUSDB18} corpus for music separation}, month = dec, year = 2017, doi = {10.5281/zenodo.1117372}, url = {https://doi.org/10.5281/zenodo.1117372} }

@misc{musdb18-hq, author = {Rafii, Zafar and Liutkus, Antoine and Stöter, Fabian-Robert and Mimilakis, Stylianos Ioannis and Bittner, Rachel}, title = {MUSDB18-HQ - an uncompressed version of MUSDB18}, month = aug, year = 2019, doi = {10.5281/zenodo.3338373}, url = {https://doi.org/10.5281/zenodo.3338373} }
```

Task 1: Train a source separation model

- Train a source separation model from scratch to separate vocal/non-vocal stems.
- Model: You can use any source separation model, but your model parameter should be lower than 10M.
- Open-Unmix model is recommended. (Model parameter: 9M) [github] [paper]



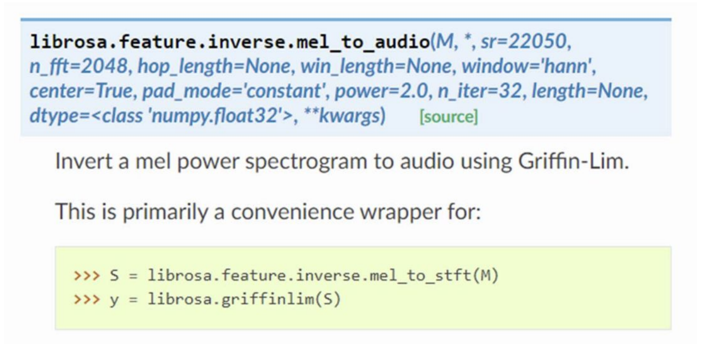
Stöter et al., (2019). Open-Unmix - A Reference Implementation for Music Source Separation. Journal of Open Source Software, 4(41), 1667, https://doi.org/10.21105/joss.01667

Task 1: Train a source separation model

- Need to report the model you choose and details of your implementation (model architecture, model parameters, data augmentation, …).
- Need to report the testing result with SDR matrix for [25, 50, 150] epoch.
- Need to generate some separated audio as listening samples.

Task 2: Griffin & Lim for phase estimation

- In Open-Unmix, we copy the phase from the mixture.
- Now, estimate phase using **Griffin & Lim algorithm** with your best epoch.
- Library: librosa.griffinlim. For more details, see lecture 5. slides.
- Need to report the testing result with SDR matrix on your best epoch in task 1.
- Need to generate some separated audio as listening samples.



librosa.feature.inverse.mel_to_audio(*M, *, sr=22050, n_fft=2048, hop_length=None, win_length=None, window='hann', center=True, pad_mode='constant', power=2.0, n_iter=32, length=None, dtype=<class 'numpy.float32'>, **kwargs*) [source]

Invert a mel power spectrogram to audio using Griffin-Lim.

This is primarily a convenience wrapper for:

```
>>> S = librosa.feature.inverse.mel_to_stft(M)
>>> y = librosa.griffinlim(S)
```

Optional Task

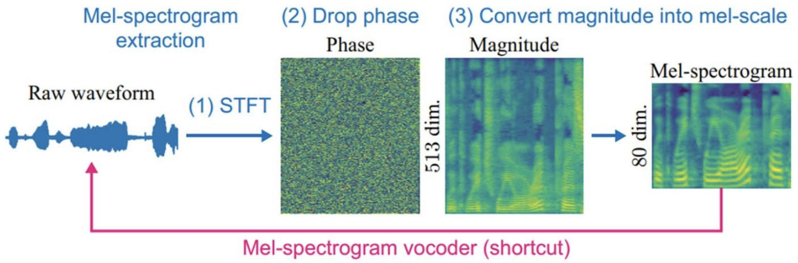
There are several optional tasks you can try. It's not required, but we look forward to see these in your report!

1. **Compare different models**
   - In this task, there's no restriction on model parameters, you can also use pre-trained models.
   - Remember to record: model parameters, whether using pre-trained model, using public code or write by yourself.
2. **Data augmentation**
   - Random swapping left/right channel for each instrument
   - Random scaling with uniform amplitudes from [0.25,1.25]
   - Random chunking into sequences for each instrument
   - Random mixing of instruments from different songs
   - Demucs auto-mix
   - …

Ref: Uhlich et al, "Improving music source separation based on deep neural networks through data augmentation and network blending," ICASSP 2017 92

Optional Task

3. **Estimate phase with Mel-Vocoder**
   - Given the magnitude, estimate the phase
   - **Be careful**: Some parameters in the Vocoder you choose must be the same as the source separation model you use (such as window size, hop len, sampling rate)
4. **Model structure modification**
   - remove/less LSTM
   - remove mask
   - …


Mel-spectrogram extraction
(2) Drop phase
(3) Convert magnitude into mel-scale
Phase
Magnitude
Mel-spectrogram
Raw waveform
(1) STFT
513 dim.
80 dim.
Mel-spectrogram vocoder (shortcut)

Ref: Figure from https://arxiv.org/pdf/2203.02395.pdf

Evaluation

- Matrix: **SDR** (Source-to-Distortion Ratio)
- SDR is usually considered to be an overall measure of how good a source sounds
- Compute in the time-domain

→ In this homework, we use **"median of median"** **SDR**.

→ Take the median SDR over all 1 second chunks in each song, then take the median of whole **test** set.

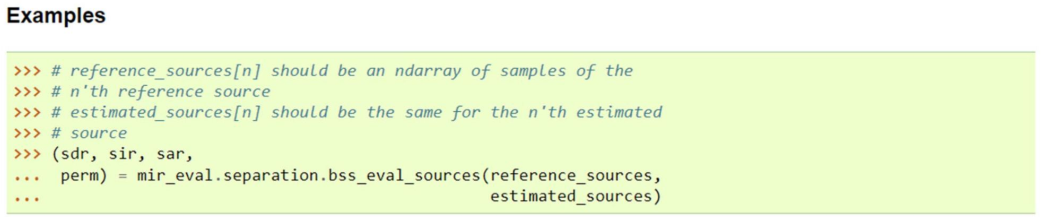
- Sometimes people report the **median** values, which can be more easily displayed in a **table**
  - e.g. "the median across the median SDR over all 1 second chunks in each song"
  - median values are more robust to outliers

(from lecture 4. slide)

Evaluation

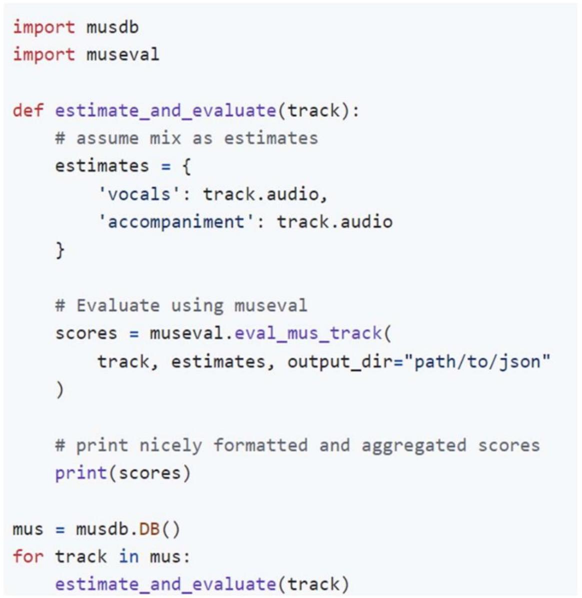
- Evaluation Library: mir_eval
    - mir_eval can be used in most MIR tasks as well

- Evaluation Library: museval
    - A python package to evaluate source separation results using the MUSDB18 dataset



Examples

```
>>> # reference_sources[n] should be an ndarray of samples of the
>>> # n'th reference source
>>> # estimated_sources[n] should be the same for the n'th estimated
>>> # source
>>> (sdr, sir, sar,
...  perm) = mir_eval.separation.bss_eval_sources(reference_sources,
...                                               estimated_sources)
```



```
import musdb
import museval

def estimate_and_evaluate(track):
    # assume mix as estimates
    estimates = {
        'vocals': track.audio,
        'accompaniment': track.audio
    }

    # Evaluate using museval
    scores = museval.eval_mus_track(
        track, estimates, output_dir="path/to/json"
    )

    # print nicely formatted and aggregated scores
    print(scores)

mus = musdb.DB()
for track in mus:
    estimate_and_evaluate(track)
```

Training Tips

- Convert dataset to **WAV** or use **MUSDB18HQ** which is already saved in WAV format, then use --is-wav configuration (for Open-Unmix). (Parser tools)
- When creating dataset, load audio and do some transformation (such as STFT) then save data to **Numpy file** first, so that it can reduce much more time when loading data during training (**highly recommended**).
- Consider increasing the number of workers using the --nb-workers k configuration.
  - Having a large number of workers does not always help though.
- Consider using pin_memory=True in DataLoader.

TA Experience (For Open-Unmix)

- MUSDB18
  - 18 mins / epoch (GTX 1080 Ti)
  - 3 hours / epoch (Colab T4)
- MUSDB18-HQ
  - 10 mins / epoch (GTX 1080 Ti)
  - 3 hours / epoch (Colab T4)

➔ Speed may be reduced much more when transformed data into Numpy file first
(within 1 min on 1080 Ti or few minutes on Colab T4)

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.

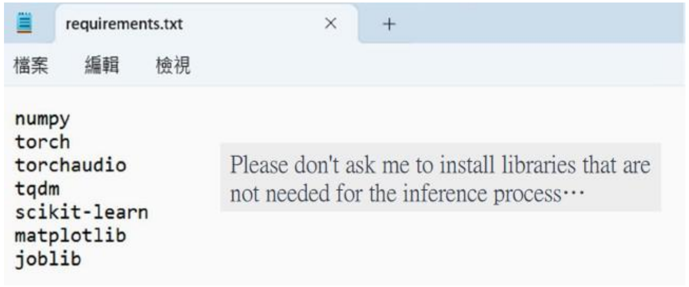
Report template

- **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 HW2_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**.



```
numpy
torch
torchaudio
tqdm
scikit-learn
matplotlib
joblib
```

Please don't ask me to install libraries that are not needed for the inference process…

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 MUSDB18/MUSDB18-HQ, ensure I can run your code with the test set path in my device.)
- The inference code should **print the SDR of test set and generate some listening samples.**

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

Scoring

- HW2 accounts for 15% of the total grade
- Report: 100%

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

- HW2_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 r12942156@ntu.edu.tw or come to office hour