

Music Source Separation

201724101 강감찬



목차

01

목표 및 필요성

02

이론 및 개요

03

코드 실행 결과

04

참고 문헌

01

목표 및 필요성

I Goals

1

효율적인 모델

••••

Colab 환경에서
작동하는 모델

2

양질의 stem 분리

••••

Loss, SDR
→ high quality

3

Stem to Midi

••••

mp3/wav 파일에서
mid 파일로 변경

Necessity

Stem to Midi

악기 별 음원에서
멜로디 추출

Auto-composing

멜로디, sound
design 자동화



Source Separation

음원 파일
→ 악기 별 구분

Melody vs Inst

멜로디와 Stem의
관계

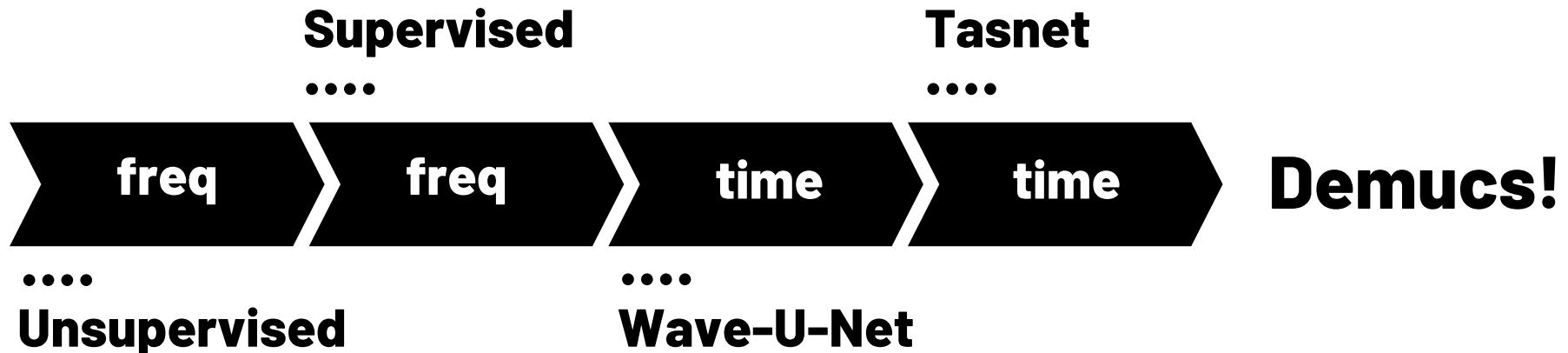
02

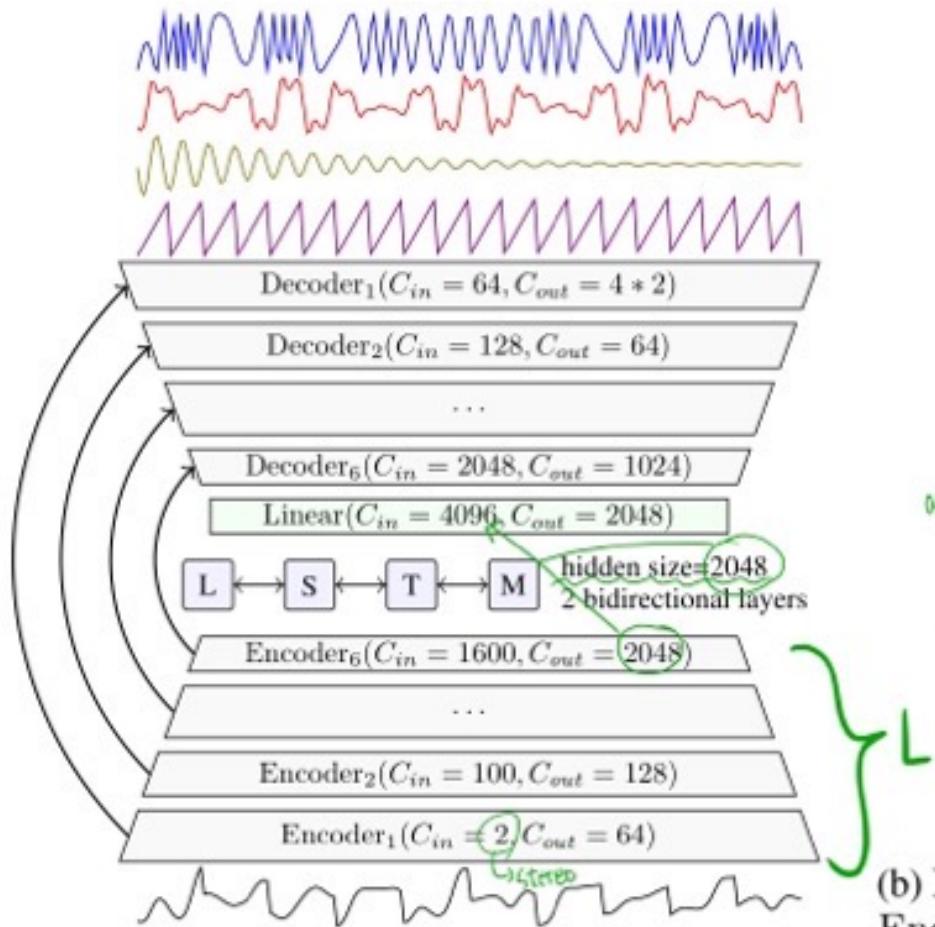
이론 및 개요

Demucs (Facebook AI Research. 2019)

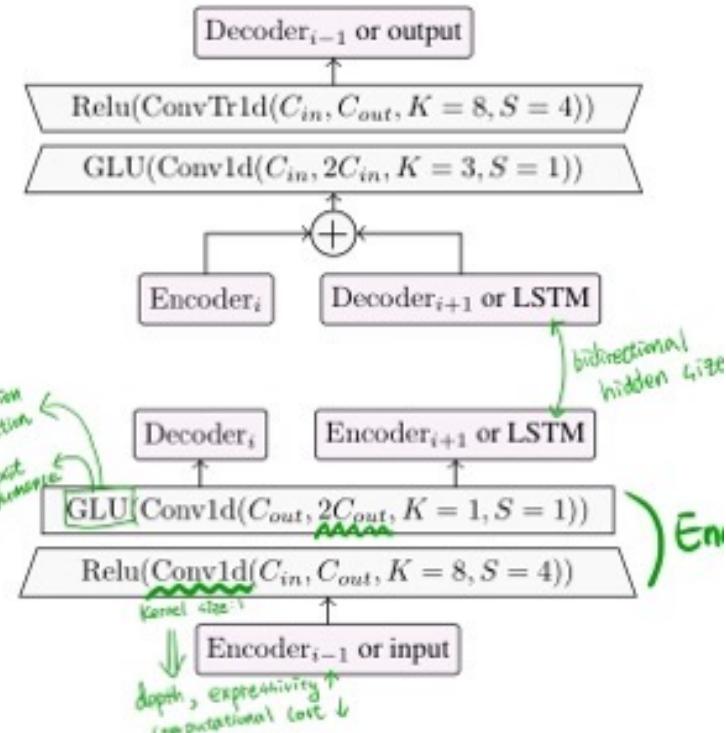
- ❖ Time domain에서 동작
- ❖ U-Net 구조
- ❖ 기존 모델보다 우수한 동작

Time domain





(a) Demucs architecture with the mixture waveform as



(b) Detailed view of the layers Decoder_i on the top and Encoder_i on the bottom. Arrows represent connections to other parts of the model. For convolutions, C_{in} (input channels), C_{out} (output channels), K (kernel size), and S (stride).

| Dataset & Epoch

MusDB

....

- ❖ 44.1kHz sampled / Stereo / 150 곡
- ❖ Drums, Bass, Others, vocals
- ❖ Train set : 84곡 Valid set : 16곡 Test set : 50곡

360 epoch

I SDR

Reusing the notations from Vincent et al. [2006], let us take a source $j \in \{1, 2, 3, 4\}$ and introduce P_{s_j} (resp P_s) the orthogonal projection on s_j (resp on $\text{Span}(s_1, \dots, s_4)$). We then take with \hat{s}_j the estimate of source s_j

$$s_{\text{target}} := P_{s_j}(\hat{s}_j) \quad e_{\text{interf}} := P_s(\hat{s}_j) - P_{s_j}(\hat{s}_j) \quad e_{\text{artif}} := \hat{s}_j - P_s(\hat{s}_j)$$

We can now define various signal to noise ratio, expressed in decibels (dB): the source to distortion ratio

$$\text{SDR} := 10 \log_{10} \frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}} + e_{\text{artif}}\|^2},$$

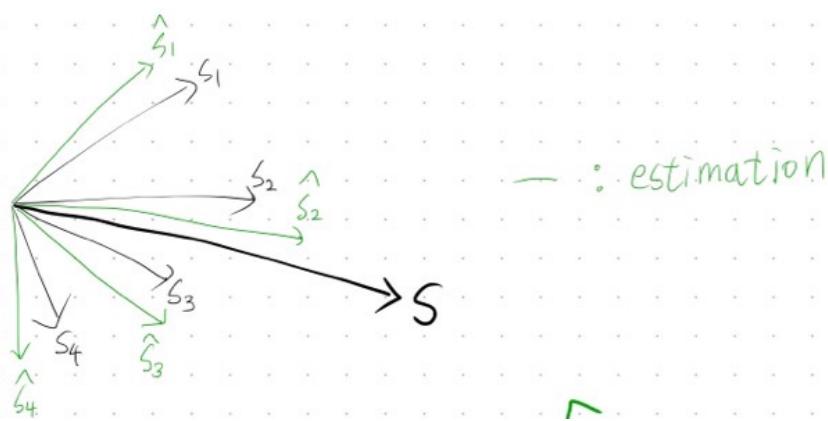
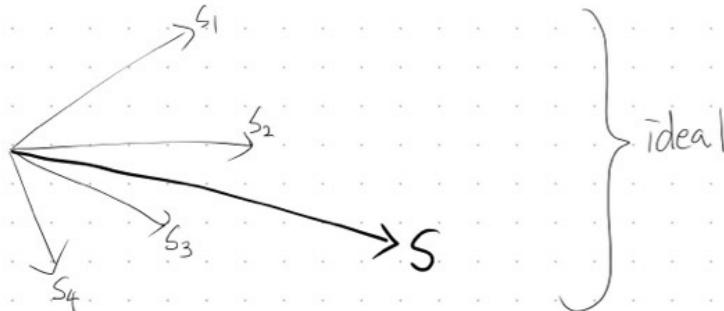
the source to interference ratio

$$\text{SIR} := 10 \log_{10} \frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}}\|^2}$$

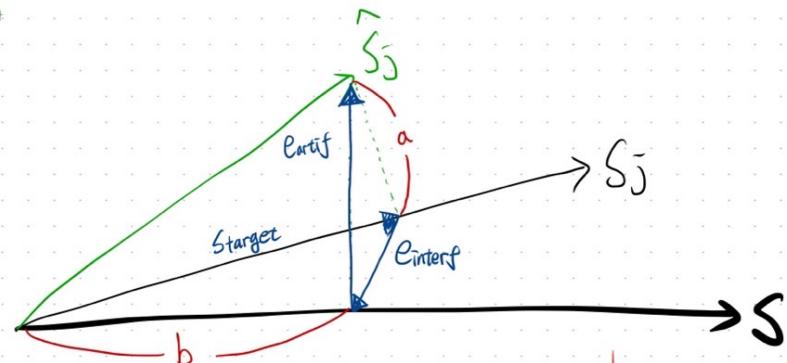
and the sources to artifacts ratio

$$\text{SAR} := 10 \log_{10} \frac{\|s_{\text{target}} + e_{\text{interf}}\|^2}{\|e_{\text{artif}}\|^2}.$$

SDR



— : estimation



$$\text{SDR} : \frac{s_{\text{target}}}{e_{\text{artif}}}$$

$$\text{SIR} : \frac{s_{\text{target}}}{e_{\text{interf}}}$$

$$\text{SAR} : \frac{b}{e_{\text{artif}}}$$

```
[ ] !pip install demucs
```

```
[ ] # Please BE VERY CAREFUL, this will link your entire drive.  
# So don't edit code, except the one that says 'Customize the following options',  
# or you might mess up your files.  
# IF YOU DO NOT WANT TO LINK DRIVE, please see below for an alternative!  
from google.colab import drive  
drive.mount('/content/drive')
```

wav source file: <https://www.ee.columbia.edu/~dpwe/sounds/music/>

```
[ ] # Customize the following options!  
model = "mdx_extra"  
extensions = ["mp3", "wav", "ogg", "flac"] # we will look for all those file types.  
two_stems = None # only separate one stems from the rest, for instance  
# two_stems = "vocals"  
  
# Options for the output audio.  
mp3 = True  
mp3_rate = 320  
float32 = False # output as float 32 wavs, unused if 'mp3' is True.  
int24 = False # output as int24 wavs, unused if 'mp3' is True.  
# You cannot set both 'float32 = True' and 'int24 = True' !!  
  
in_path = '/content/drive/MyDrive/demucs/'  
out_path = '/content/drive/MyDrive/demucs_separated/'
```

Useful functions, don't forget to execute



코드 표시

```
[ ] # This can be quite slow, in particular the loading, and saving from GDrive. Please be patient!  
# This is from google drive! Also, this will separate all the files inside the MyDrive/demucs folder,  
# so when you are happy with the results, remove the songs from there.  
separate()
```

madmom (S Böck. 2016)

- ❖ 객체 지향 programming
- ❖ Machine Learning 기반
- ❖ 의존성이 낫다
 - <https://www.ofoct.com/audio-converter/convert-wav-or-mp3-ogg-aac-wma-to-midi.html>

Run Terminal Help ← → ⌂ graduation_projects

Release Notes: 1.74.2 audio_comparison.ipynb madmom_main.ipynb

audio_to_midi > madmom_main.ipynb > Wav file analysis > D = np.abs(librosa.stft(y, n_fft=2048, hop_length=512))

- Code + Markdown | Run All Clear Outputs of All Cells Restart Variables Outline ...

```
proc = RNNPianoNoteProcessor()
abs_path = 'C:/Users/Admin/graduation_projects/Audio_to_midi/'
act = proc(abs_path+'wav/vocals.wav')

print("(frame #, piano keys) = (%d, %d)" %act.shape)
print(act)
```

```
[ ]
```



```
proc = NoteOnsetPeakPickingProcessor(fps=100, pitch_offset=21)
act = RNNPianoNoteProcessor()(abs_path+'wav/vocals.wav')
print(proc(act).shape)
print(proc(act))
```

```
[ ]
```



```
proc = CNNPianoNoteProcessor()
adsr = ADSRNoteTrackingProcessor(onset_threshold=0.55, pitch_offset=21)

# long runtime: 2m 20s
act = proc(abs_path+'wav/vocals.wav')

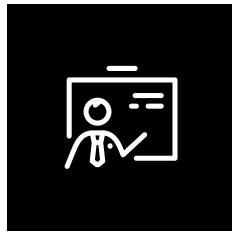
# midi matrix
MIDI_notes = adsr(act)

print(MIDI_notes)

# Requirements for midi file
# 1. midiutil library
# 2. track # : 1
# 3. beginning time(sec) : 0
# 4. volume : 100
```

onset_threshold : float, optional
Require notes to have an onset activation greater or equal this threshold.

Midi vs Audio



Midi

Piano roll (keys)에 따라
위치만 표현



Audio

실제로 들을 수 있는 파일

This image shows a screenshot of PreSonus Studio One Pro software interface, specifically the MIDI Editor and Project View.

MIDI Editor (Bottom Left):

- Toolbar:** Includes tools for Parameter selection, zoom, edit, and playback.
- Header:** Shows Quantize (1/16), Timebase (Bars), Snap (Adaptive), and various transport controls.
- Instrument Panel:** Shows the current track is "Vocals Track" (MIDI, Sustain, Velocity). It includes dropdowns for None, All Inputs, and Velocity.
- Performance Panel:** Shows M (MIDI), S (Sustain), and Velocity (Medium).
- Event List:** Shows a list of notes with columns for Action, Note Color, Part, AQ, Quantize, Timebase, Snap, and Quantize.
- Properties Panel:** Shows settings for the "Vocals Track":
 - Instrument: Instrument
 - Audition Notes: checked
 - Default Velocity: 80%
 - Scale: Chromatic
 - Length: 1/16, Straight
 - Sound Variations: None selected
 - No event selected
 - Input Chord: C2
 - Current Chord: C2
- Velocity Controller:** A slider and buttons for Velocity, Note Controller, Sound Variations, Musical Symbols, Modulation, Pitch Bend, and After Touch.

Project View (Top Right):

- Header:** Shows Instruments, Effects, Loops, Files, Cloud, Shop, and Pool.
- Instruments:** Show Instruments and Note FX.
- Effects:** Show installed Effects.
- Loops:** Browse Studio One loop libraries.
- Files:** Browse files and folders.
- Cloud:** PreSonus Exchange and other services.
- Shop:** Get add-ons from PreSonus Shop.
- Pool:** Show files used in this Song.
- Buttons at the bottom:** Re-Index Presets and Plug-In Manager.

Transport Bar (Bottom):

- Shows 5:03 days, 00030.01.01.00, Bars, and Record Max.
- Transport controls: Backward, Forward, Play, Stop, and Metronome.
- Session Info: L 00001.01.01.00, R 00001.01.01.00, Off Sync, Metronome, Timing, Key, Tempo (172.27).
- Mode Buttons: Edit, Mix, and Browse (Korean).

The screenshot shows the PreSonus Studio One Pro 5 software interface. The main window displays the XT PRESENCE instrument editor. On the left, the 'Parameter' panel shows a 'Vocals Track' selected with various controls like LFOs, Mod/FX, and a keyboard. The central workspace features a piano-roll style interface for programming notes, with sections for 'Filter' (Cutoff, Drive, Punch, Resonance), 'Amp Env' (Attack, Decay, Sustain, Release), and 'Env 2'. Below these are 'Modulation' (FX A/B, Chorus, Flanger, Phaser) and 'Reverb' (Delay, Reverb) sections. The bottom of the screen includes transport controls (MIDI, Performance, Record Max), a timeline (Bars: 00030.01.01.00), and metronome/timing controls (Key: 4/4, Tempo: 172.27). The top right corner shows the 'Song', 'Project', and 'Show' tabs, along with a search bar and a file browser.

Parameter

+ Instruments

1 - Presence

Auto: Off

Comparing

Vocals Track

M S Presence All Inputs None

LFO 1 Rate

LFO 2 Rate

Artist Instruments

- Brass
- Guitar
- Keyboards
- Organ
- Percussion
- Piano
- Strings
- Vox
- Winds

Grand Piano

Filter

Drive

Vel

Punch

Key

Cutoff

Res

Amp Env

Global Volume

Velocity

Poly Mono Glide

Glide

Env 2

Grand Piano

Program: Grand Piano

Voices: 0

64 3.69 MB

MOD/FX

XT PRESENCE

Modulation

FX A FX B

Mod A Mod B

Chorus Flanger Phaser

Mono Delay Speed Width FB Sync Depth

Delay Off Reverb

Low High 1/4 FB Mix

Reverb

Pre Damp Size Low High Mix

Bend Mod

2 Bend 2

Bend Mod

5:03 days 00030.01.01.00 Bars: 00030.01.01.00

L 00001.01.01.00 Off Sync Metronome

R 00001.01.01.00

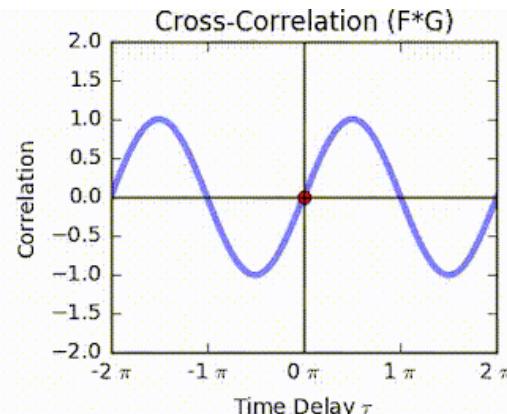
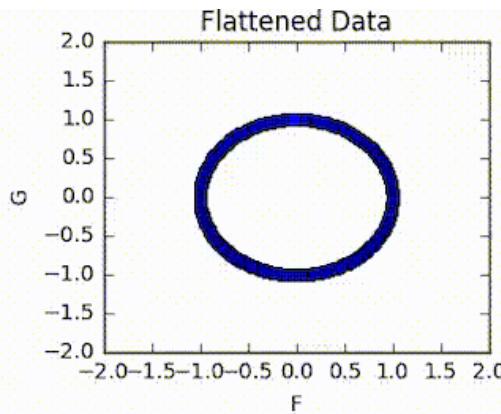
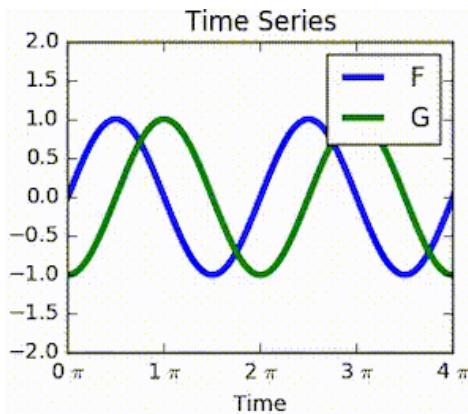
4 / 4 Timing Key: 4/4

172.27 Tempo

MIDI Performance Record Max

Edit Mix Browse

Cross correlation



$$(f \star g)(\tau) \triangleq \int_{-\infty}^{\infty} \overline{f(t)} g(t + \tau) dt$$

which is equivalent to

$$(f \star g)(\tau) \triangleq \int_{-\infty}^{\infty} \overline{f(t - \tau)} g(t) dt$$

audio_comparison.m × +

```
1 [x,fs]=audioread("vocal_test.wav");
2 [y,fs]=audioread("vocal_07.wav");
3 lx = length(x);
4 ly = length(y);
5 samples = 1:min(lx,ly);
6
7 subplot(3,1,1), plot (x(samples));
8 subplot(3,1,2), plot (y(samples));
9 [C1, lag1] = xcorr(x(samples),y(samples));
10 subplot(3,1,3), plot(lag1/fs,C1);
11 ylabel("Amplitude"); grid on
12 title("Cross-correlation ")
```

03

코드 실행 결과

Source Separation



Runtime : 8m 30s ~ 9m (in Google Colab pro env)

Original



vocal



bass



drum



other

Source Separation

Architecture	Wav?	Extra?	Test SDR in dB				
			All	Drums	Bass	Other	Vocals
IRM oracle	✗	N/A	8.22	8.45	7.12	7.85	9.43
Wave-U-Net	✓	✗	3.23	4.22	3.21	2.25	3.25
Open-Unmix	✗	✗	5.33	5.73	5.23	4.02	6.32
Meta-Tasnet	✓	✗	5.52	5.91	5.58	4.19	6.40
Conv-Tasnet [†]	✓	✗	5.73 ± .10	6.02 ± .08	6.20 ± .15	4.27 ± .03	6.43 ± .16
DPRNN	✓	✗	5.82	6.15	5.88	4.32	6.92
D3Net	✗	✗	6.01	7.01	5.25	4.53	7.24
Demucs [†]	✓	✗	6.28 ± .03	6.86 ± .05	7.01 ± .19	4.42 ± .06	6.84 ± .10
Spleeter	✗	~ 25k*	5.91	6.71	5.51	4.55	6.86
TasNet	✓	~ 2.5k	6.01	7.01	5.25	4.53	7.24
MMDenseLSTM	✗	804	6.04	6.81	5.40	4.80	7.16
Conv-Tasnet ^{††}	✓	150	6.32 ± .04	7.11 ± .13	7.00 ± .05	4.44 ± .03	6.74 ± .06
D3Net	✗	1.5k	6.68	7.36	6.20	5.37	7.80
Demucs [†]	✓	150	6.79 ± .02	7.58 ± .02	7.60 ± .13	4.69 ± .04	7.29 ± .06

*: each track is only 30 seconds, †: from current work, ††: trained without pitch/tempo augmentation, as it deteriorates performance.

Audio to Midi



Runtime : 4m ~ 4m 30s (in GTX 1060)

stem



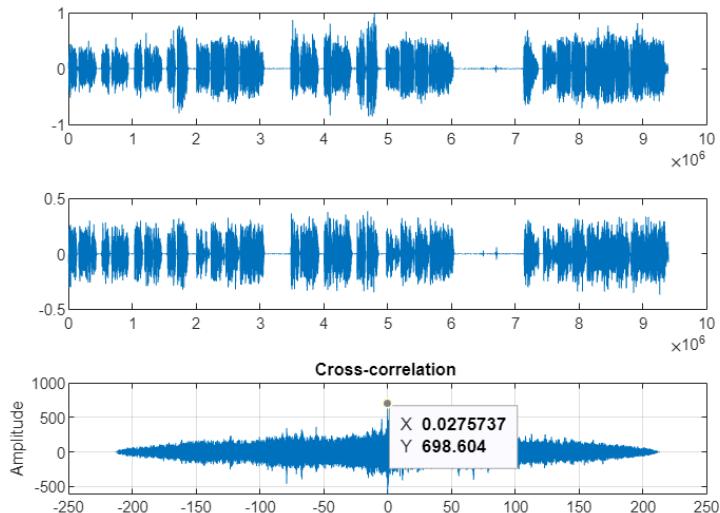
vocal test

0.55

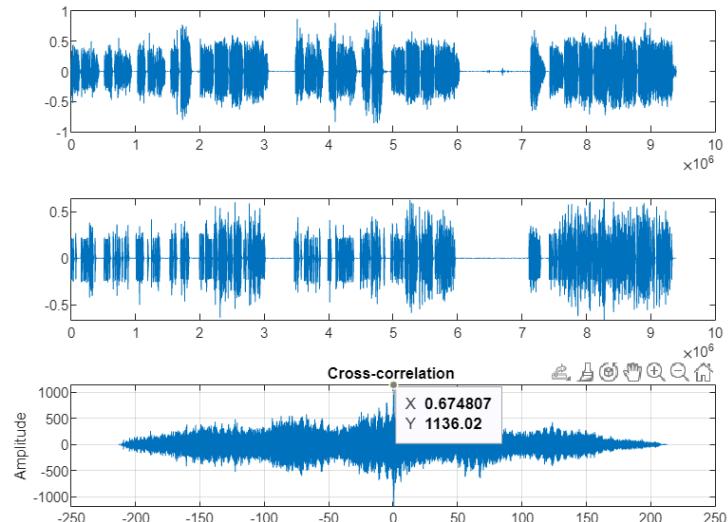
0.7

0.9

Cross Correlation

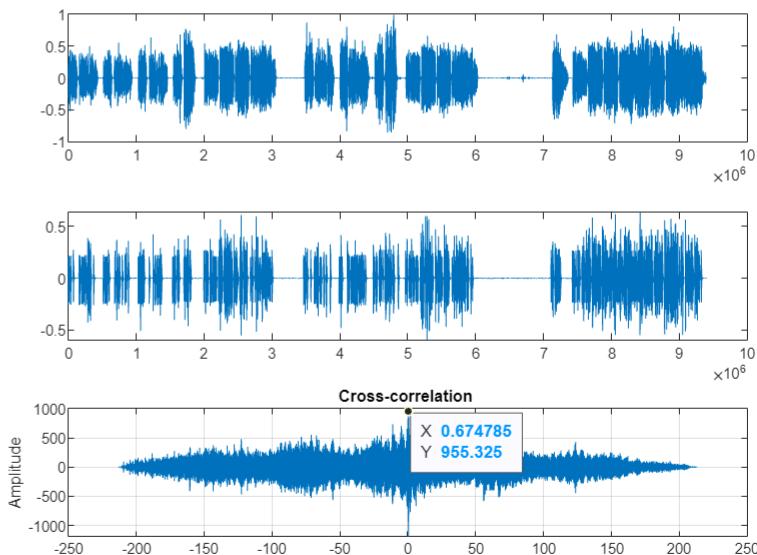


stem vs test

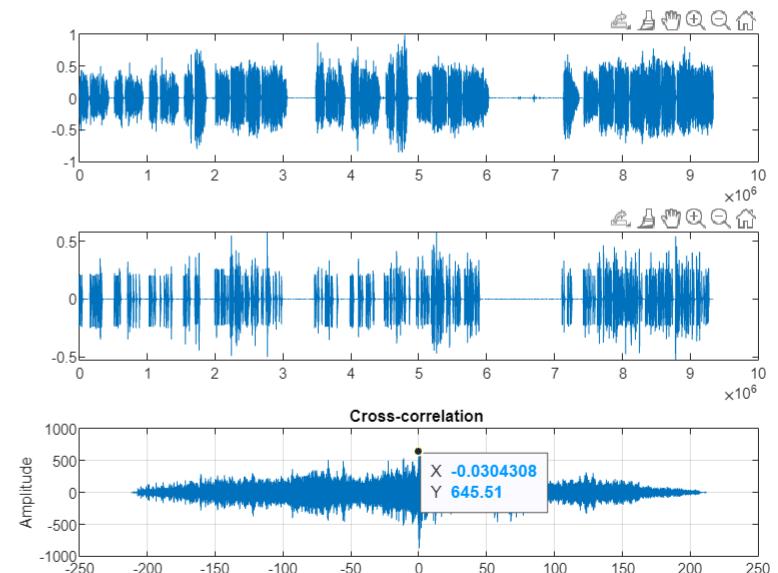


stem vs 0.55

Cross Correlation

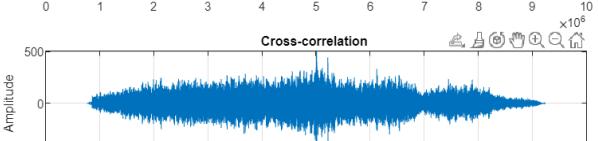
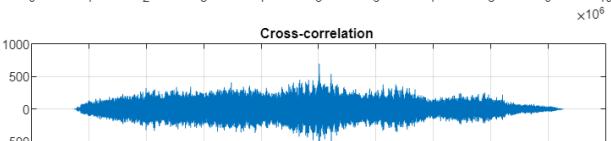
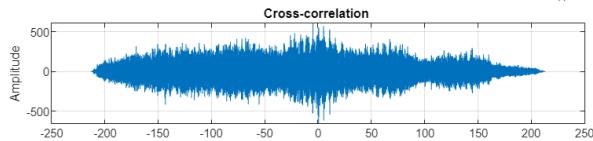
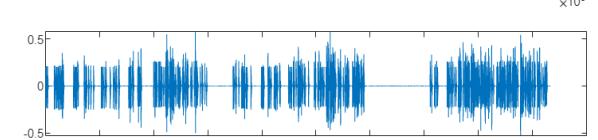
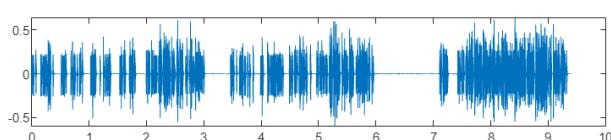
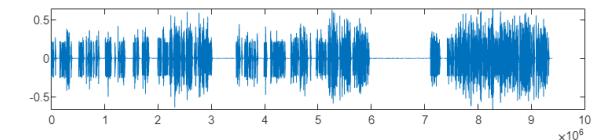
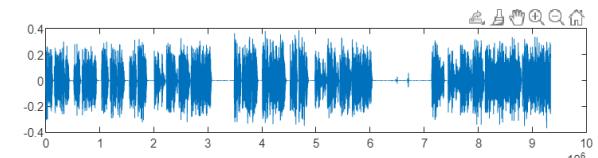
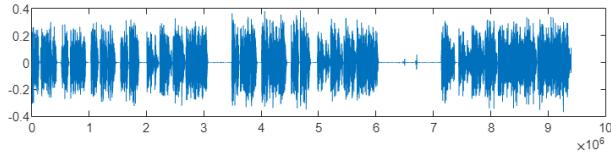
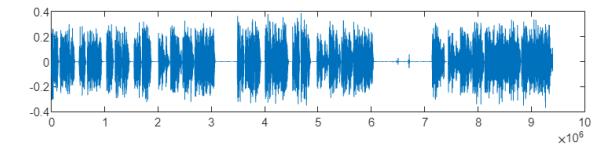


stem vs 0.7



stem vs 0.9

Cross Correlation



test vs 0.55

test vs 0.7

test vs 0.9

Cross Corr Table

vs stem

	Test	0.55	0.7	0.9
Correlation max	698.60	1.1360e+03	955.32	672.08
vs test	0.55	0.7	0.9	

Correlation max

```
>> audio_comparison  
698.6040|
```

```
>> audio_comparison  
1.1360e+03
```

```
>> audio_comparison  
955.3251
```

```
>> audio_comparison  
672.0814
```

vs test

	0.55	0.7	0.9	
Correlation max	606.58	695.43	501.62	
vs stem	0.55	0.7	0.9	

Correlation max

```
>> audio_comparison  
606.5845
```

```
>> audio_comparison  
695.4330
```

```
>> audio_comparison  
501.6190
```

Conclusions

1

효율적인 모델

- ❖ Runtime이 각각 8~8.5m / 4~4.5m
- ❖ Demucs : 150MB / madmom : 90MB
- ❖ 객체 지향적 → 사용하기 쉬움

I Conclusions

2

양질의 stem 분리

- ❖ SDR 값이 대체적으로 높다
- ❖ 다양한 file format (wav)

I Conclusions

3

Stem to Midi

- ❖ Original stem과 가장 근접한 것 → 0.55
- ❖ Test와 가장 근접한 것 → 0.7
- ❖ 0.55 ~ 0.7 사이 값이 적정

References

Demucs

- ❖ <https://github.com/facebookresearch/demucs>
- ❖ Music Source Separation in the Waveform Domain (2019)

Madmom

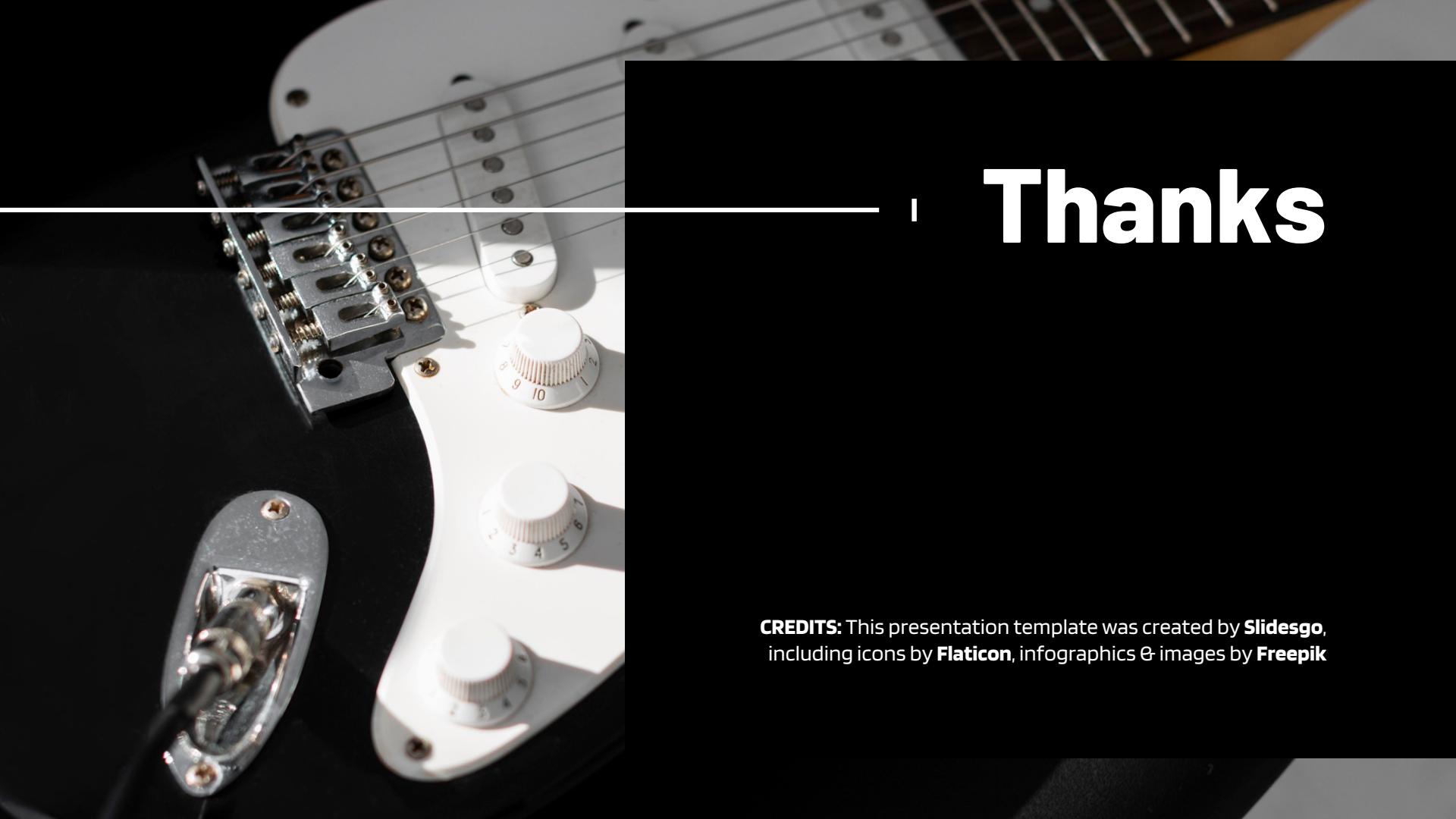
- ❖ <https://github.com/CPJKU/madmom>
- ❖ madmom: A New Python Audio and Music Signal Processing Library (2016)
- ❖ <https://madmom.readthedocs.io/en/latest/index.html>

SDR

- ❖ Performance Measurement in Blind Audio Source Separation (2006)

Correlation

- ❖ <https://en.wikipedia.org/wiki/Cross-correlation>

A close-up photograph of a white electric guitar's neck and bridge area. The guitar has six strings and a white pickguard. The bridge is made of metal with four saddles. The neck is light-colored wood with dark frets. A black rectangular overlay covers the right side of the image, containing the word "Thanks".

Thanks

CREDITS: This presentation template was created by [Slidesgo](#),
including icons by [Flaticon](#), infographics & images by [Freepik](#)