Machine Learning and Big Data Processing

Voice isolation in Songs

MAKHOUL Rayan
OGUNGBURE Semilogo
PRIEELS Lucas
TROUILLEZ Benoît

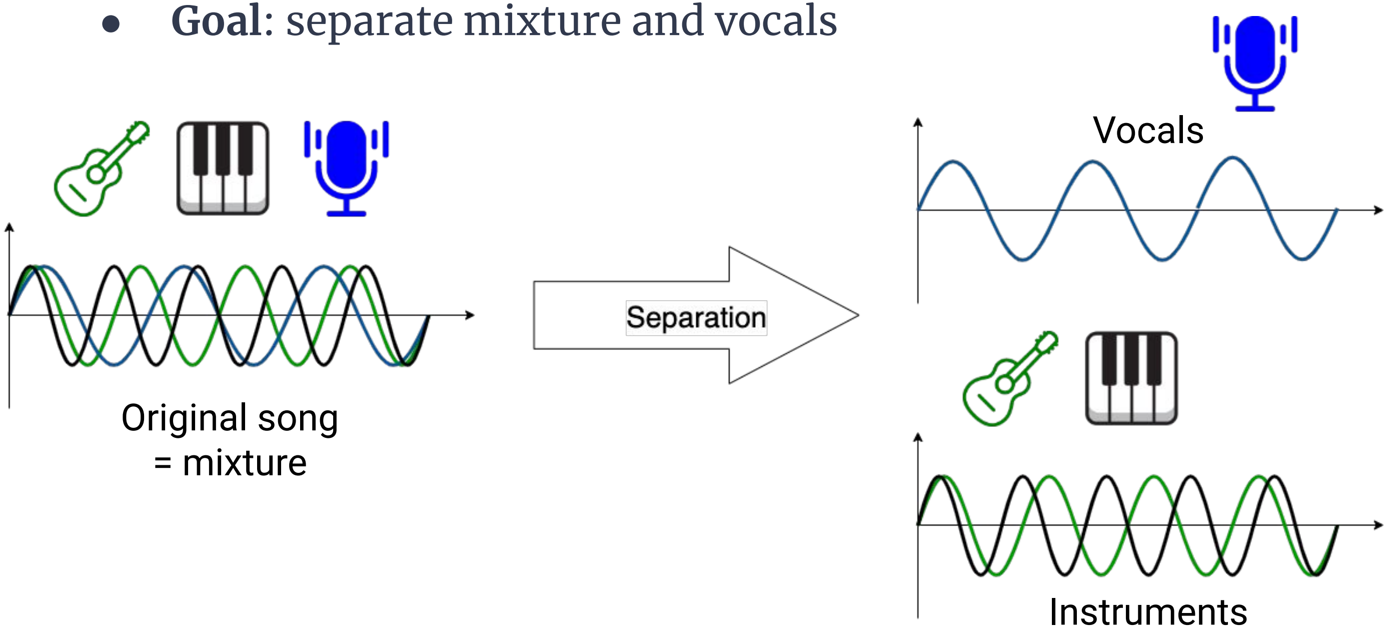
19 June 2023

Contents

Contents

Introduction

- **Goal**: separate mixture and vocals



- Deep learning approach to learn to **extract vocals from song**
- 3 architectures considered: CNN, U-Net, Wave-U-Net

Contents

- Introduction
- **Convolutional Deep Neural Network**
- U-Net
- Wave-U-Net
- Implementation
- Results
- Conclusion

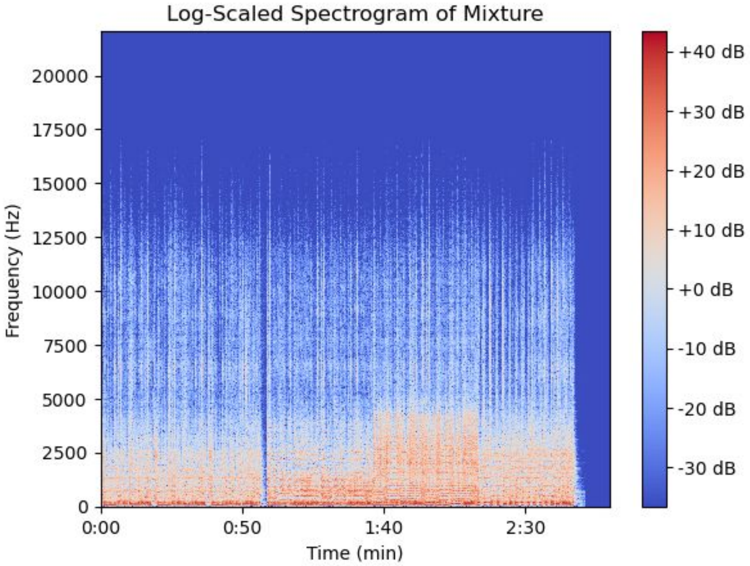
Convolutional Deep Neural Network

A. **Pre- and post-processing**

B. Network architecture

C. Training and testing strategy

- **Spectrogram**: graph showing the **frequency content** of the song at different points in time



- Spectrogram = plot of the square of the modulus of the Short-Term Fourier Transform (**STFT**) in log scale
- STFT: Fourier transform on many short, overlapping sequences

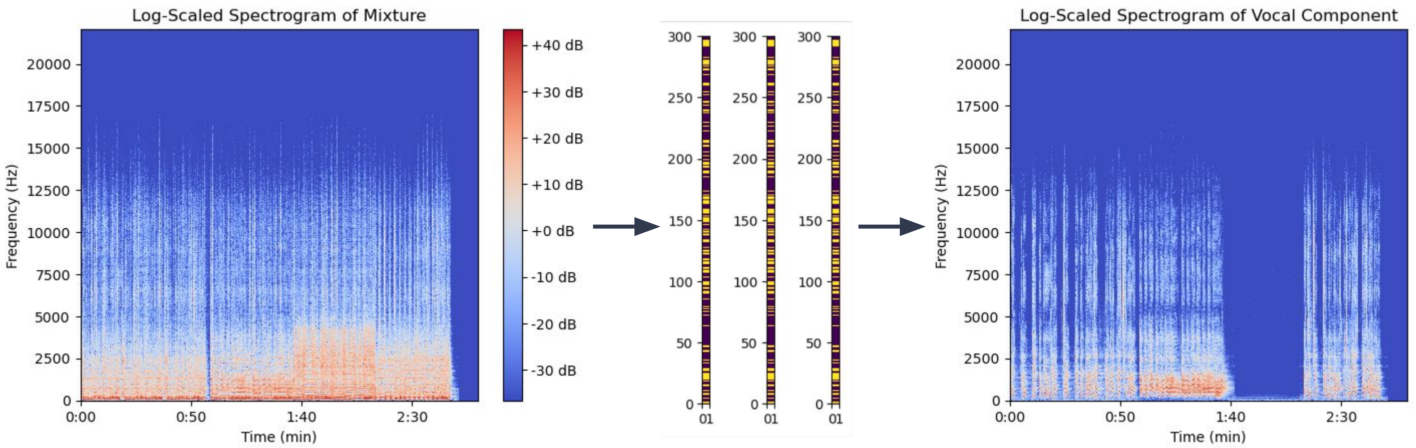
Convolutional Deep Neural Network

A. **Pre- and post-processing**

B. Network architecture

C. Training and testing strategy

- Neural network **input**: **spectrogram**
- Neural network **output**: frequency **binary mask**



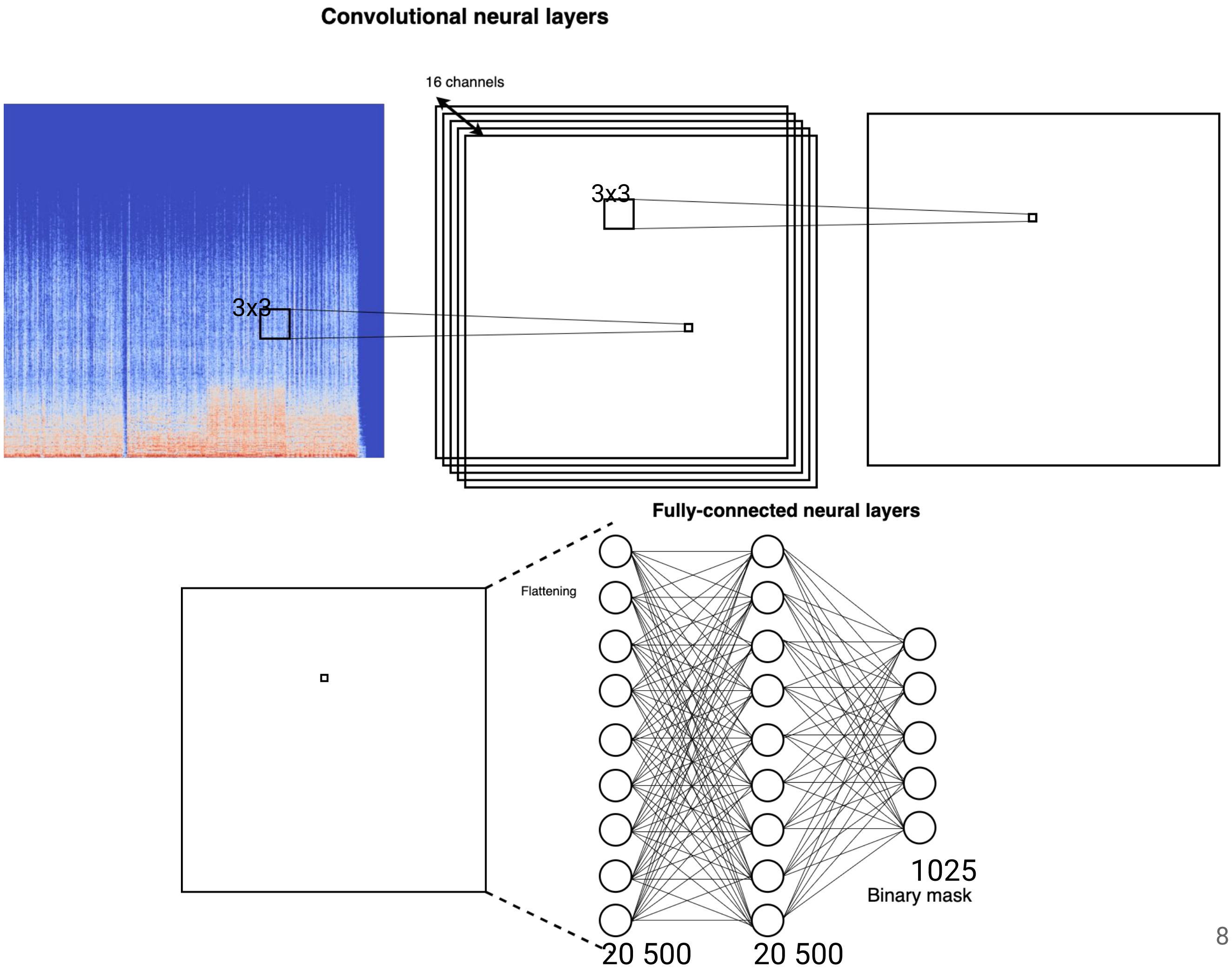
- **Pre-processing**: computing input spectrogram from original song with STFT transform
- **Post-processing**:
  - **Computing vocal spectrogram** from frequency mask and mixture spectrogram
  - **Reconstructing vocals** from output spectrogram with Inverse STFT transform

Convolutional Deep Neural Network

A. Pre- and post-processing

B. Network architecture

C. Training and testing strategy

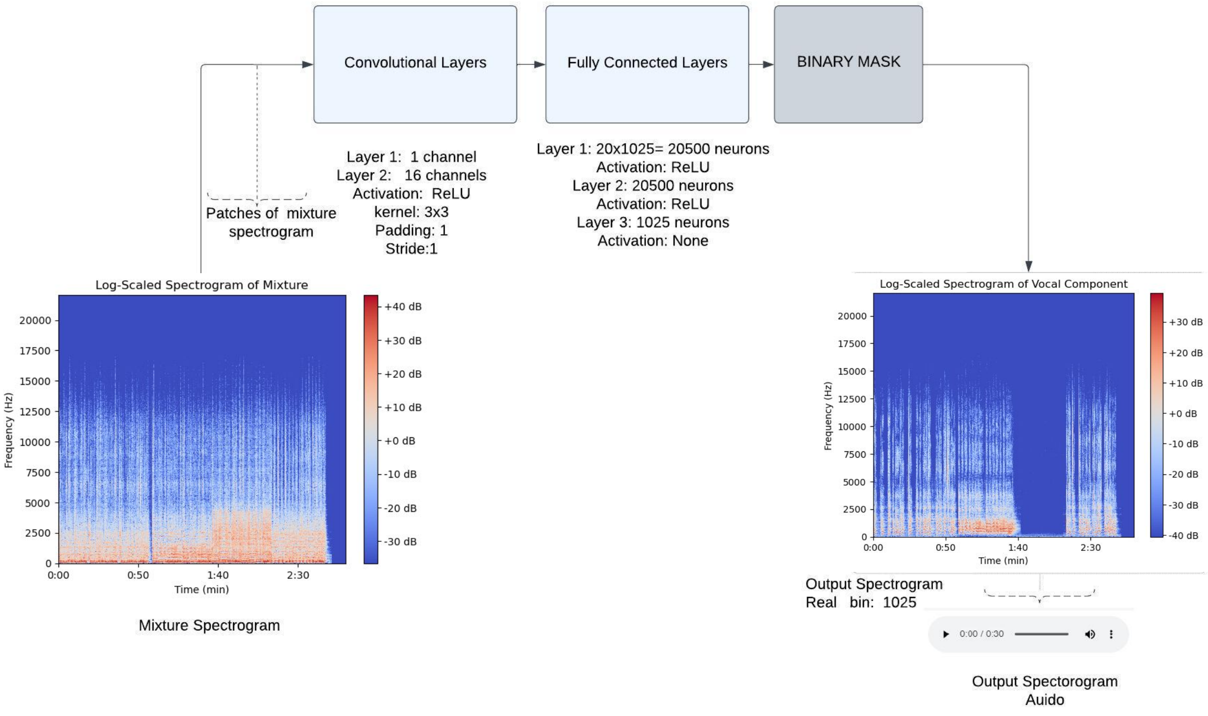


Convolutional Deep Neural Network

A. Pre- and post-processing

B. Network architecture

C. Training and testing strategy

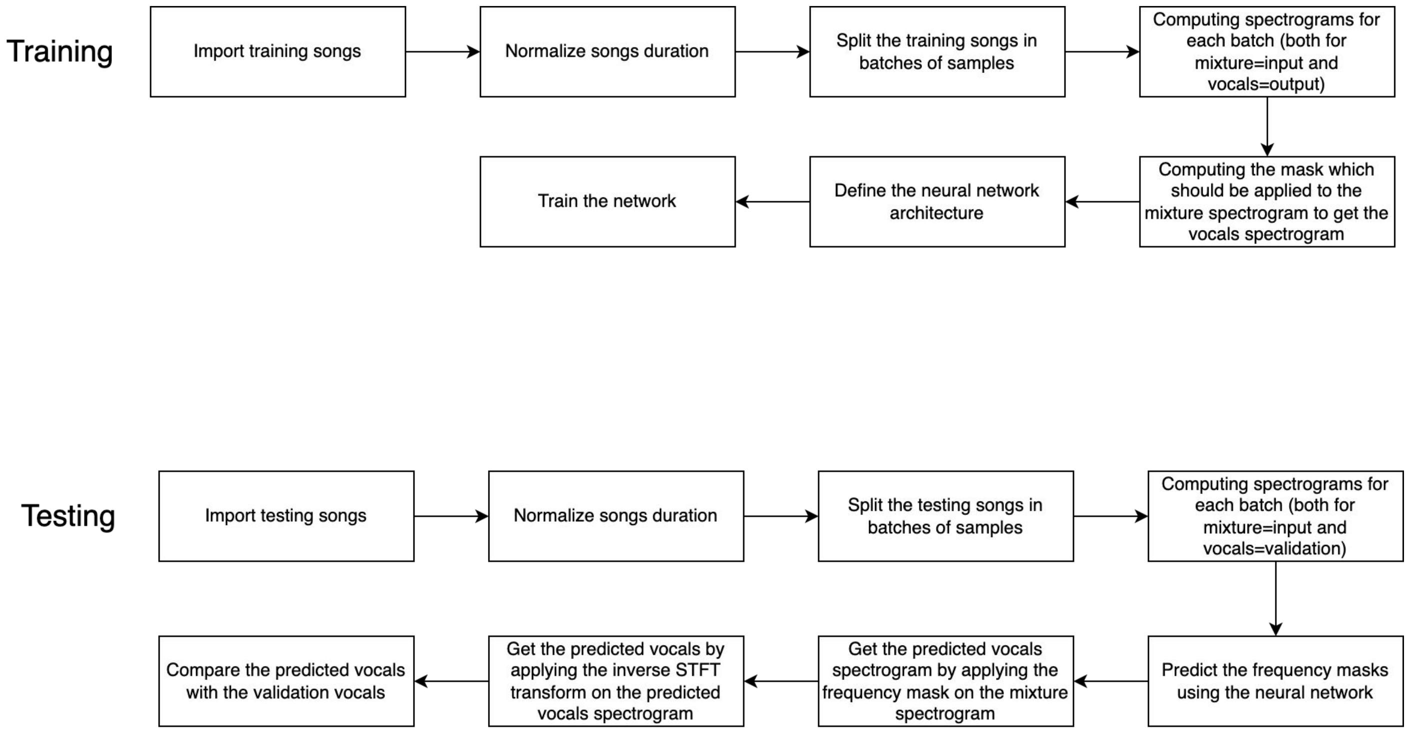


Convolutional Deep Neural Network

A. Pre- and post-processing

B. Network architecture

C. **Training and testing strategy**



Contents

- Introduction
- Convolutional Deep Neural Network
- **U-Net**
- Wave-U-Net
- Implementation
- Results
- Conclusion

U-Net

A. **Overview**

B. Pre- and post-processing

U-Net

Based on the **U-Net** architecture, initially developed for medical imaging

Possible since proven capacity for recreating the **fine, low-level detail** required for high-quality audio reproduction.

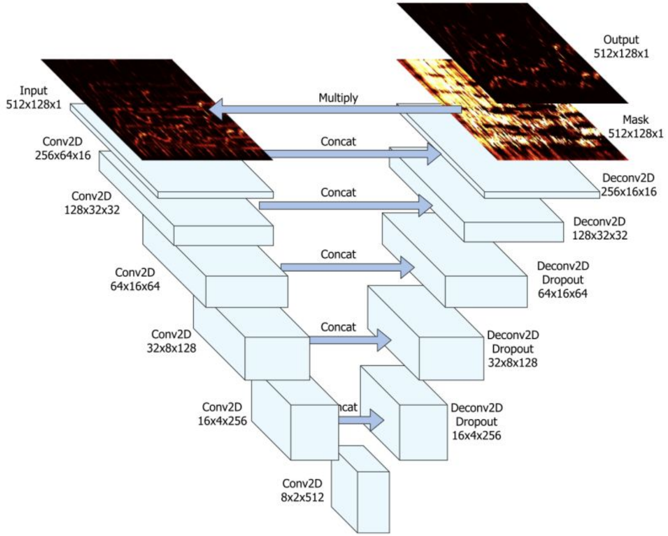
Application proposed in 2017

Architecture principle

<u>Input</u> : **Mixture** (Spectrogram)

<u>Encoding-Decoding layers</u> :
- 6 serial **encoding layers** : extract features
- 6 serial **decoding layers** : upsample feature maps and recover spatial dimensions
- Concat **skip connections** between encoder and decoder

<u>Output</u> : **Masked mixture** (Spectrogram)
- Last decoder output (mask) **multiplied** with input mixture
- Input **phase** added



U-Net

A. **Overview**

B. Pre- and post-processing

U-Net

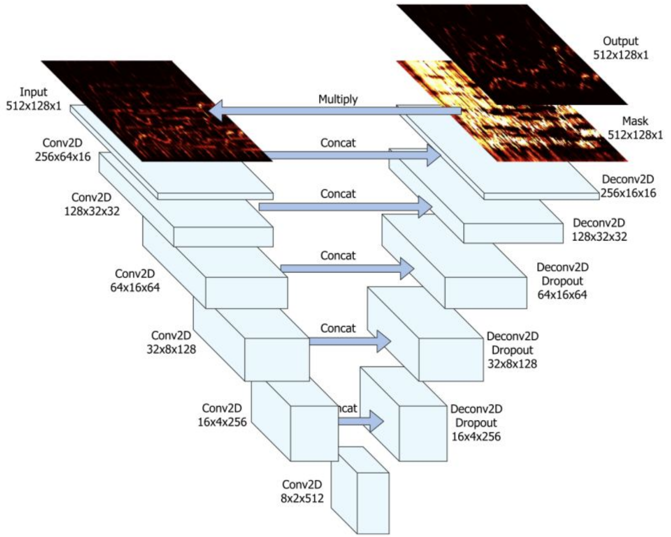
Detailed architecture

Encoder :

- 2D Convolutions
  - Kernel = 5x5
  - Stride = 2
- Batch normalization
- Leaky ReLU, leakiness = 0.2
- Skip connections:
  - **Preservation** of spatial details
  - Path for **gradient flow**
  - Localization of **features**

Decoder :

- Transposed 2D Convolutions
  - Kernel = 5x5
  - Stride = 2
- Batch normalization
- Plain ReLU
- First three layers: 50% dropout
  - Prevent overfitting (**regularization**)
- Last module output: **mask**
  - Multiplied with input



U-Net

A. Overview

B. Pre- and post-processing

U-Net – Data type

Dataset

MusDB18:

- 100 training songs
- 50 testing songs
- Mixture, drums, bass and vocals
- Stereo signals, encoded at 44.1kHz

Pre-processing:

- Signal downsampled in time domain (to 8192Hz)
- Spectrograms obtained from STFT:
  - Window size = 1024
  - Hop length = 768
- Patched each 128 frames

Post-processing:

- Spectrogram reconstruction from patches
- Input phase added
- Inverse STFT

Contents

- Introduction
- Convolutional Deep Neural Network
- U-Net
- **Wave-U-Net**
- Implementation
- Results
- Conclusion

Wave-U-Net

A. **Architecture**

B. Data pre- and post-processing

C. Computational flow

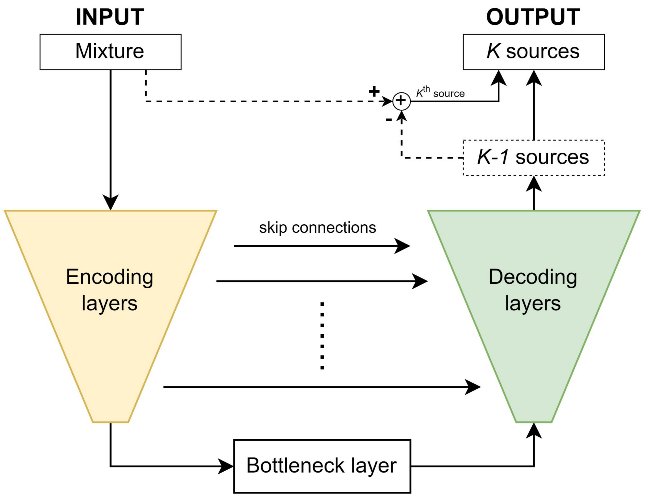
Wave-U-Net

Inspired by the **U-Net** architecture → **Encoding-Decoding** architecture

“Wave” : It uses the **time domain** representation of the audio instead of its spectrogram representation → **Waveform** representation of the signal

New architecture presented in 2018

Architecture principle



Input : **Mixture** (TD)

Encoding-Decoding layers :

- *L* serial **encoding layers** : extract features
- *L* serial **decoding layers** : reconstruct the separated sources
- **Skip connections** between encoding and decoding layers
- **Bottleneck** layer

Output : ***K* sources** (TD)

- The **NN** outputs ***K-1* sources**
- The **last source** is **deduced** from the others and the given mixture

Wave-U-Net

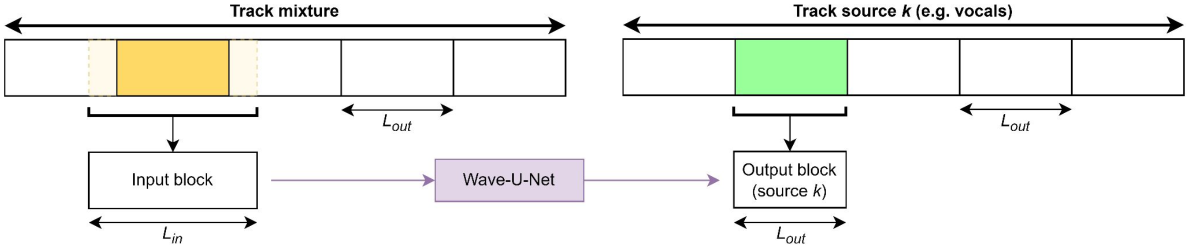
A. Architecture

B. Data pre- and post-processing

C. Computational flow

Input and output data of the Wave-U-Net

Time domain waveform representation of the audio (f_s = 44.1KHz, x(t) between [-1, 1])
Block of size $\boldsymbol{L_{in}}$ for the input mixture and size $\boldsymbol{L_{out}}$ for output sources



Context in input data

The idea is to consider some **context around** the input data **block**
→ Improvement compared to non-context case

Without context : $L_{in} = L_{out}$, With context : $L_{in} > L_{out}$

<u>**Pre-processing**</u> : **Split** input mixture **into blocks** of size L_{in}

Reconstruction of the separated sources audio

At the output of the Wave-U-Net, one has blocks for the K sources

<u>**Post-processing**</u> : **Gather blocks** of same source together to get the **reconstructed audio** for this source

Wave-U-Net

A. Architecture

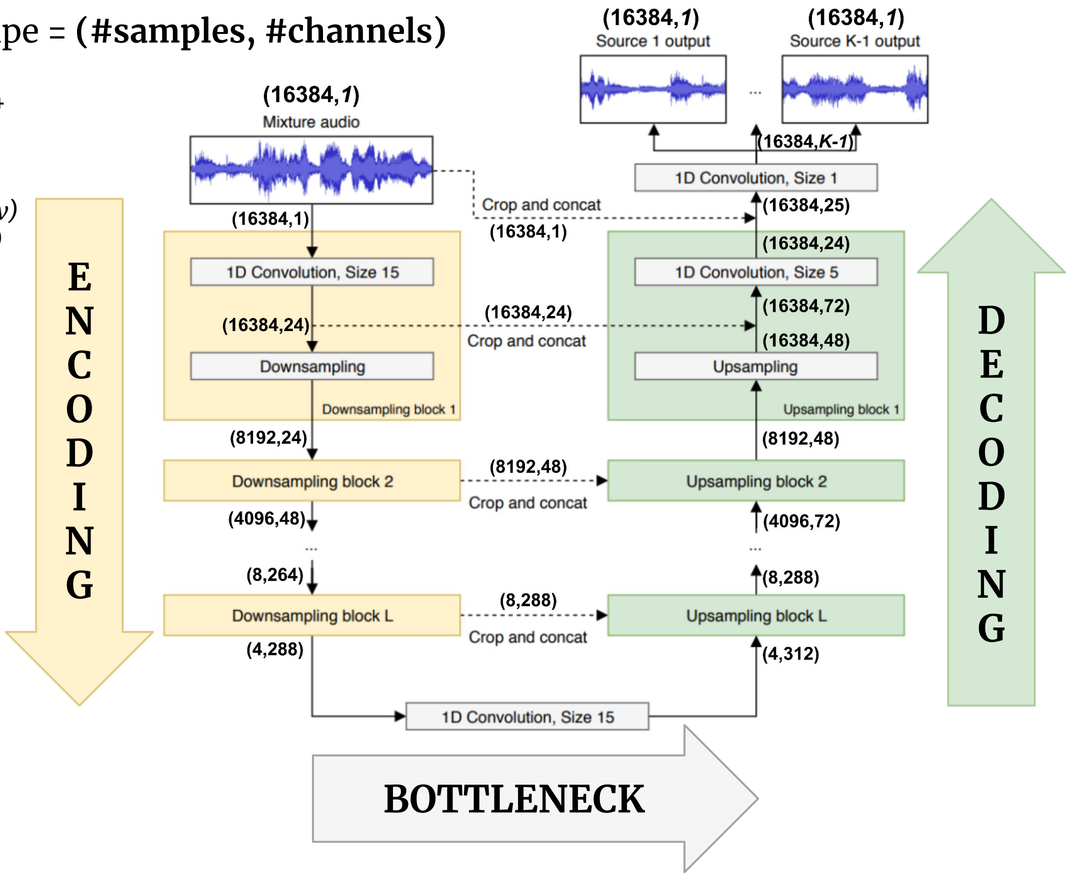
B. Data pre- and post-processing

C. **Computational flow**

What happens in the Wave-U-Net?

Let data shape = (#samples, #channels)

Example :
$L_{in} = L_{out} = 16384$
$L = 12$
$F_c = 24$
$f_d = 15, f_u = 5$
LeakyReLU (Conv)
tanh (final Conv)



Contents

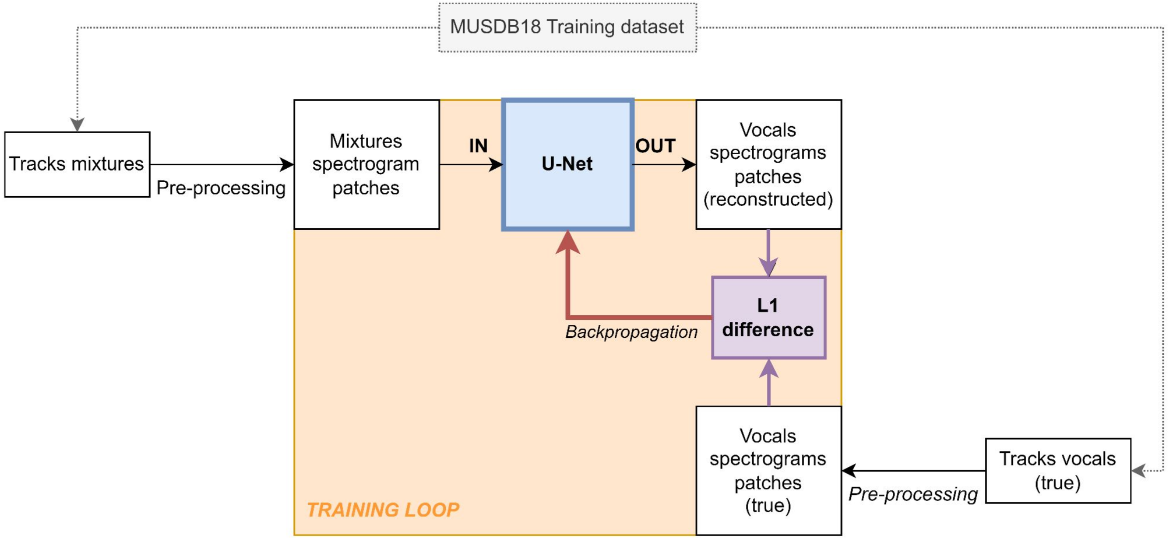
Implementation

A. Training

B. Testing

After implementing and exploring the **three architectures**, we decided to focus on the **U-Net** since the others gave unsatisfactory results and we had a **time constraint**

<u>Training parameters</u> : *Number of tracks* : 50/100 from MUSDB18/train (5580 input/target samples)
Training algorithm : ADAM optimizer with L1-norm as loss
Leaning rate : 0.01
Number of epochs : 5
Batch size : 1



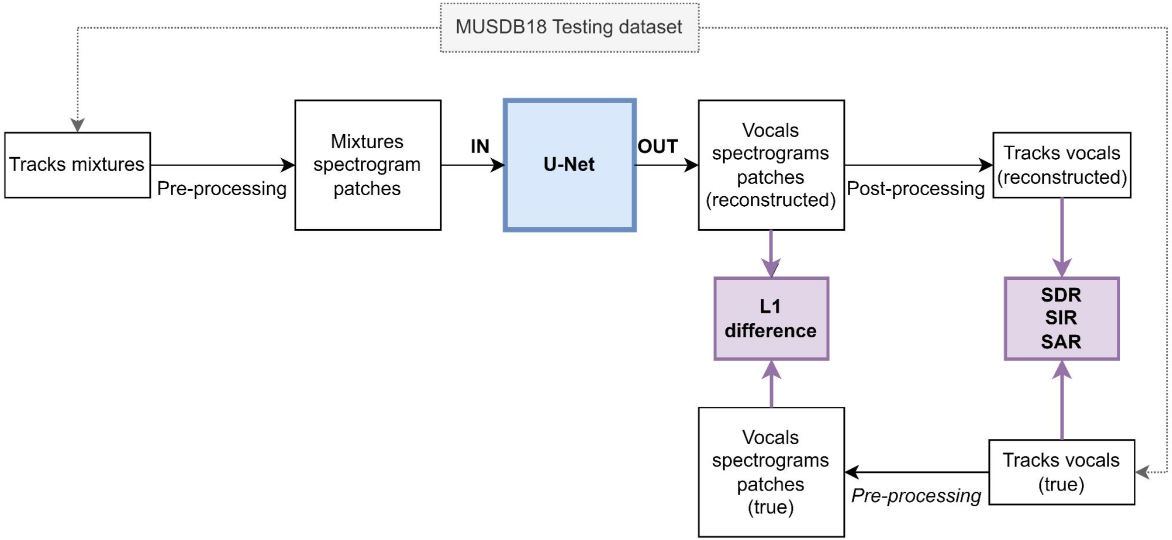
Implementation

A. Training

B. **Testing**

The **trained U-Net** is then **tested** on tracks from the testing dataset of the MUSDB18.
→ Testing tracks ≠ Training tracks

Testing parameters : *Number of tracks* : 20/50 from MUSDB18/test (2232 input/target samples)
Evaluation : - output/target samples average L1-difference
- reconstructed/true vocals SDR/SIR/SAR



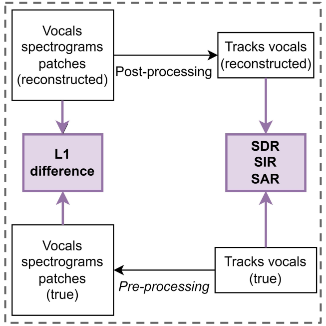
Contents

- Introduction
- Convolutional Deep Neural Network
- U-Net
- Wave-U-Net
- Implementation
- **Results**
- Conclusion

Results

A. **Performance assessment**

B. Testing results



How to assess voice isolation performance ?

→ SDR/SIR/SAR (Audio comparison)

$$s_p = s_t + e_n + e_i + e_a$$

- Signal-to-Distortion Ratio (SDR) $SDR = 10 \log 10 \frac{||s_t||^2}{||e_n + e_i + e_a||^2}$
- Signal-to-Interference Ratio (SIR) $SIR = 10 \log 10 \frac{||s_t||^2}{||e_i||^2}$
- Signal-to-Artefact Ratio (SAR) $SAR = 10 \log 10 \frac{||s_t + e_i + e_n||^2}{||e_a||^2}$

→ To be **maximized**

→ L1-difference (Spectrograms comparison)

Predicted spectrogram patch *s* : $\tilde{S}_{s,p} \in \mathbb{R}^{N_F \times N_T}$

Target spectrogram patch *s* : $\tilde{S}_{s,t} \in \mathbb{R}^{N_F \times N_T}$

L1-difference for N_S **samples** (= spectrogram patches) :

$$error_{L1} = \frac{1}{N_S} \sum_{s=1}^{N_S} \sum_{i=1}^{N_F} \sum_{j=1}^{N_T} |\tilde{S}_{s,t}[i,j] - \tilde{S}_{s,p}[i,j]|$$

→ To be **minimized**

Results

A. **Performance assessment**

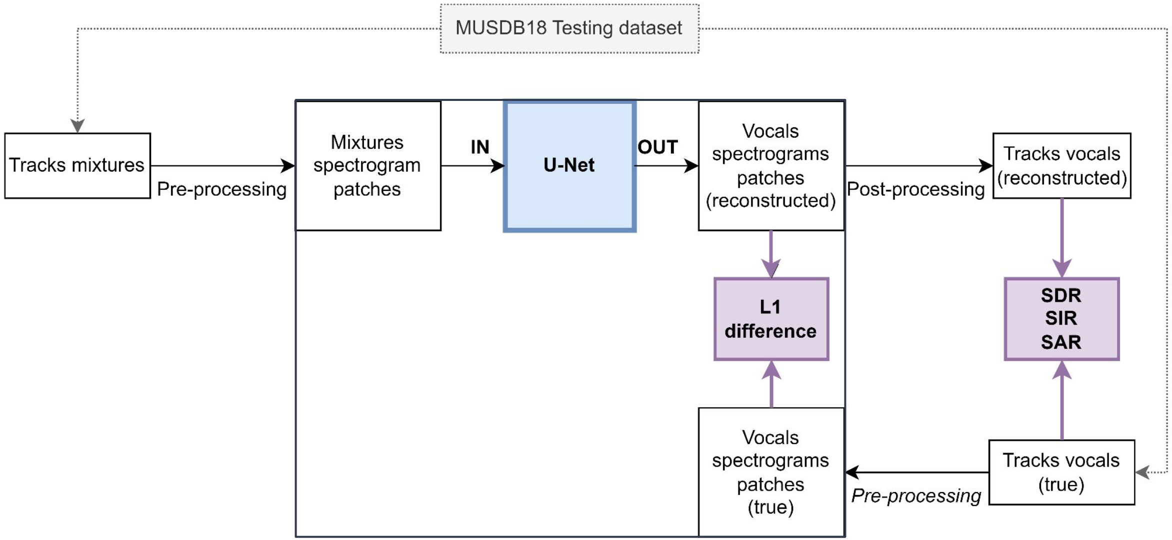
B. Testing results

Audio reconstruction problem

Probably error in post-processing step ...

→ SDR/SIR/SAR : Difficult to assess properly

→ **Spectrograms L1-difference and visualization :** Evaluates the U-Net performance directly



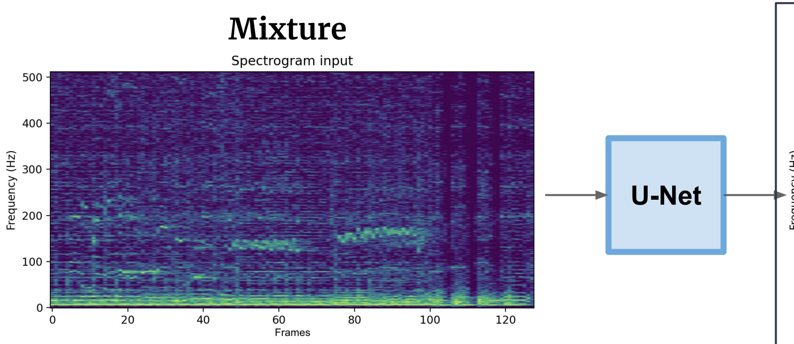
Results

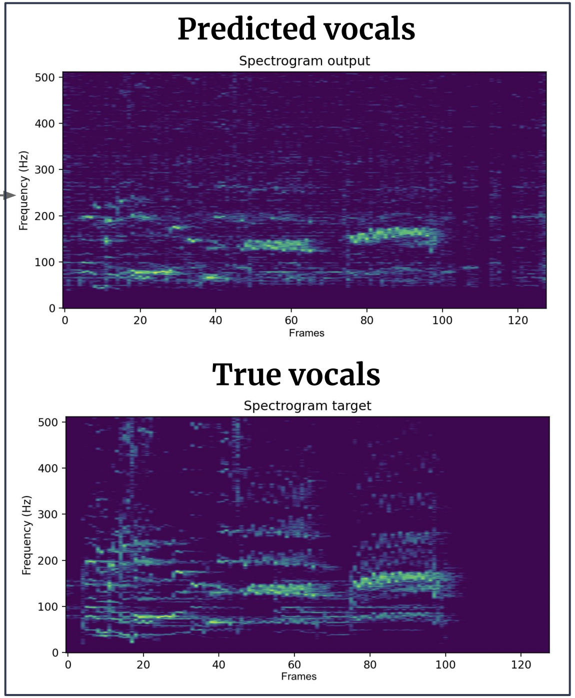
A. Performance assessment

B. Testing results

Spectrograms comparison

As a first evaluation, we can observe spectrogram patches and compare visually (L=5)





➡ Mathematical comparison is done through **L1-Loss**

Results

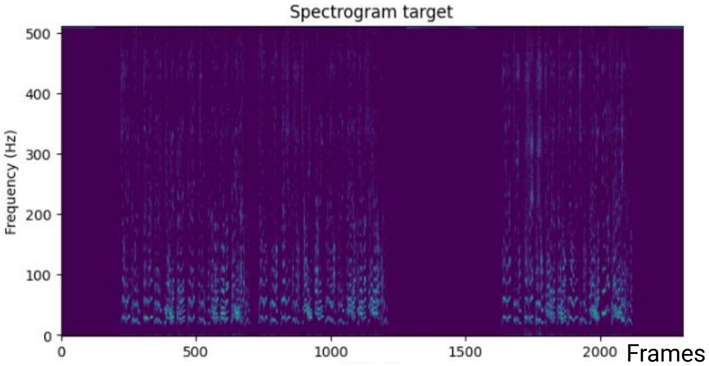
A. Performance assessment

B. Testing results

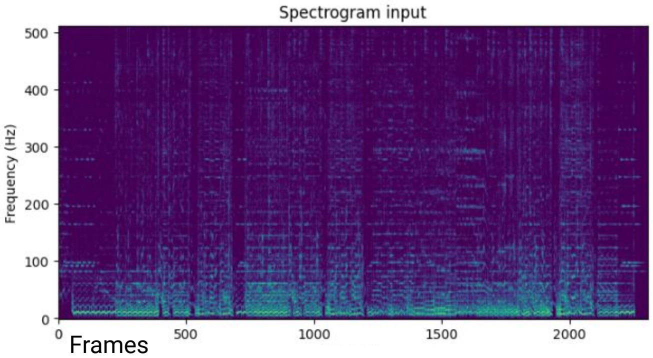
Spectrograms comparison

Other example (*L*=5) :

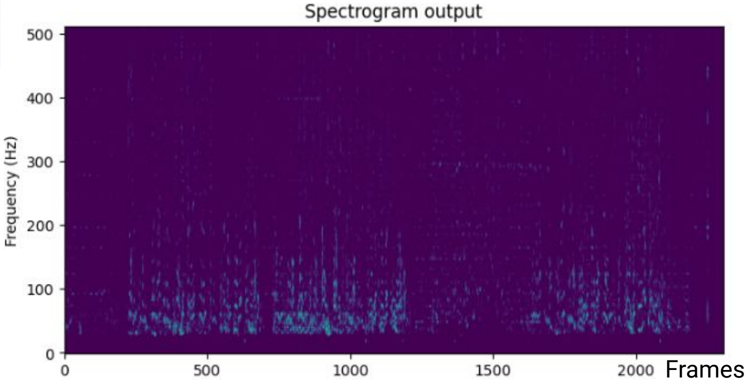
True vocals



Mixture



Predicted vocals



Results

A. Performance assessment

B. Testing results

Effect of parameters

Varying the number of encoding and decoding layers in the U-Net architecture

Number of layers	L1 (matrix) loss
3	1.516×10^{-3}
4	1.304×10^{-3}
5	$\mathbf{9.800 \times 10^{-4}}$
6	1.176×10^{-3}

→ Implementing ***L*=5 layers** seems to be **appropriate** for this architecture and input/output shape (as made in the U-Net paper)

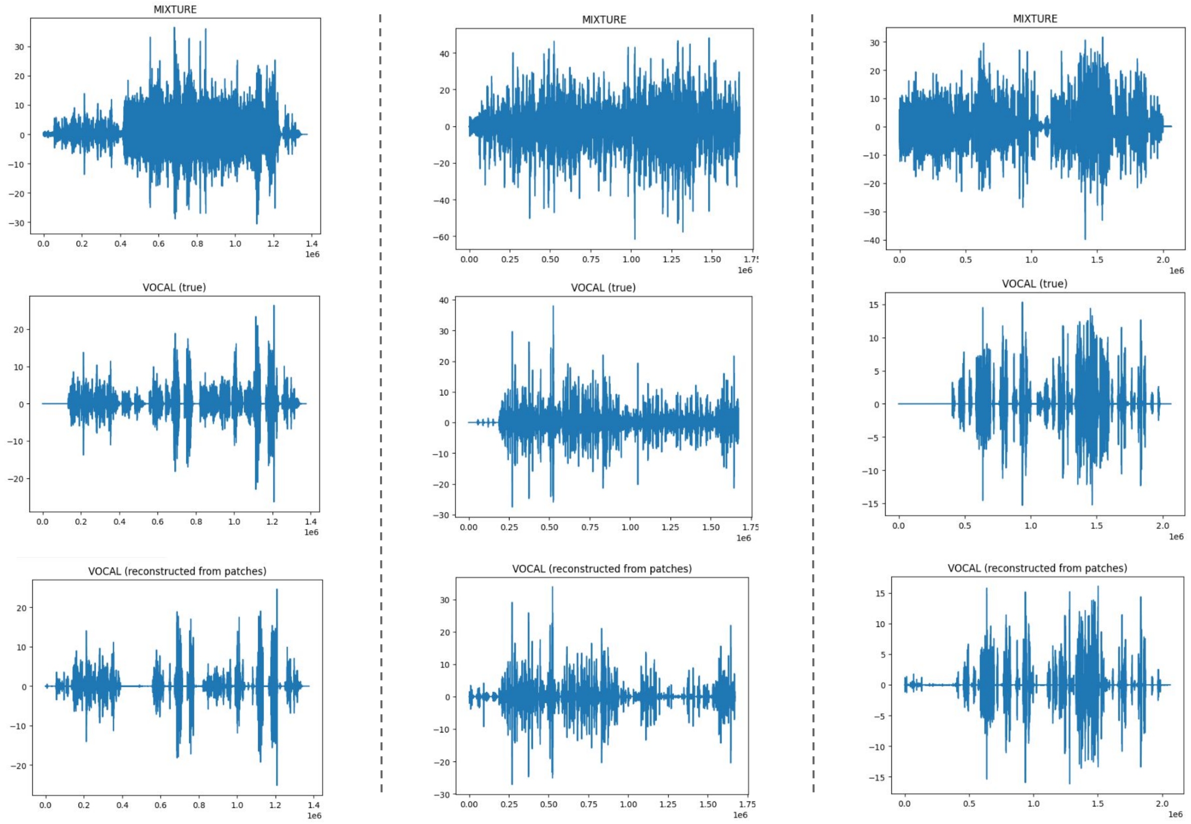
Results

A. Performance assessment

B. **Testing results**

Audio comparison

We gave a try to waveforms comparison (L=5) (the audios are quite highly “wavy” distorted)



Contents

- Introduction
- Convolutional Deep Neural Network
- U-Net
- Wave-U-Net
- Implementation
- Results
- **Conclusion**

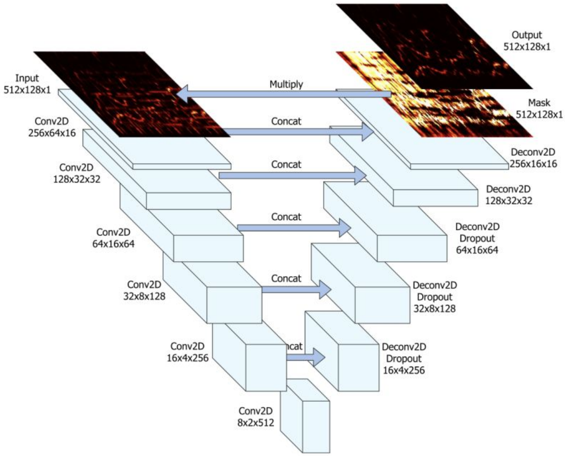
Conclusion

- **U-Net results**

We observe **good results** for the **U-Net** but there is probably a problem in pre/post-processing that makes audio highly “wavy” distorted

- **Comparison between U-Net complexity**

We **varied** the **complexity** (i.e. the number of encoding/decoding layers in our case) of the U-Net in order to observe its **impact on performance**



Do you have any question ?