Vocal from instrumental separation - Deep Machine Learning

Nicholas Granlund, Martin Ducros

Chalmers University of Technology
Electrical Engineering Department

CHALMERS
UNIVERSITY OF TECHNOLOGY

Introduction

The purpose of this work is to be able to automatically split a vocal from an instrumental in any music

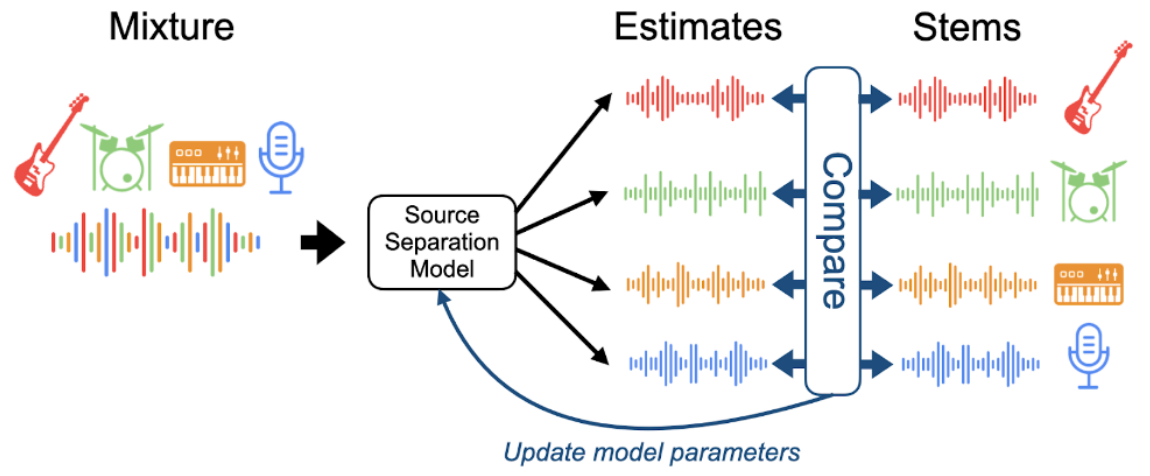

Figure 1: Explanation of audio separation [1]

- A sound can be represented by a spectrogram if we use a time-frequency representation.
- For all frequency $f \in [0, 44100/2]Hz$ of the input at time t, we need to determine if f belongs to a voice, and how much.
- This problem can be seen as a semantic segmentation task.

Model

Here are the main constraints we had to build our network:

- We focused on CNN instead of Transformers to save time and computational resources.
- We wanted to use transfer learning not to start from scratch.

From this paper [2], we have found this network which is able to split instruments from a video.

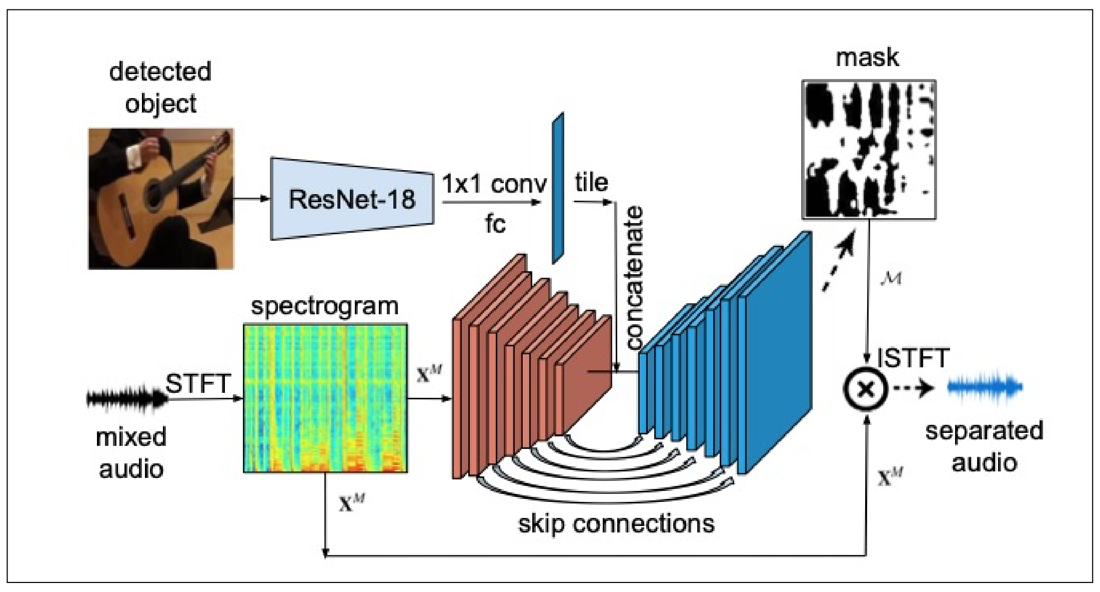

Figure 2: A network found in a paper

Then we have taken the encoder (brown part) and train a new decoder for our network.

- The output of this network is a **real-valued ratio mask** $\mathcal{M}_{vocals}$. This is multiplied with the input to get the desired signal. $X_{Filtered} = X_{Mixture} \times \mathcal{M}_{vocals}$
- We used a MSE loss for the training. Computed with regards to the difference between desired filtered spectrogram and actual vocals track.
- The network has $\approx$ 42M trainable parameters

Here is a Unet architecture example, (Our model is similar in structure, but has input size of 256×256):

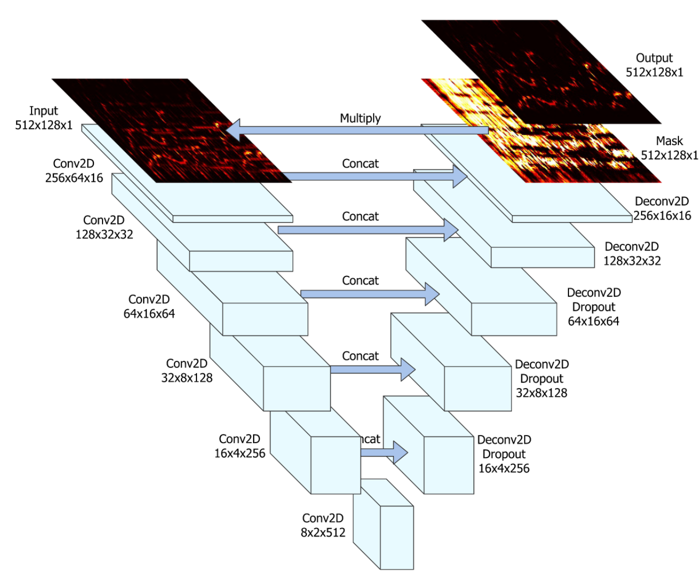

Figure 3: The U-Net architecture by Rachel Bittner[1]

Pre-processing

- How do we go from a .wav file to usable numerical representation? Spectrograms!

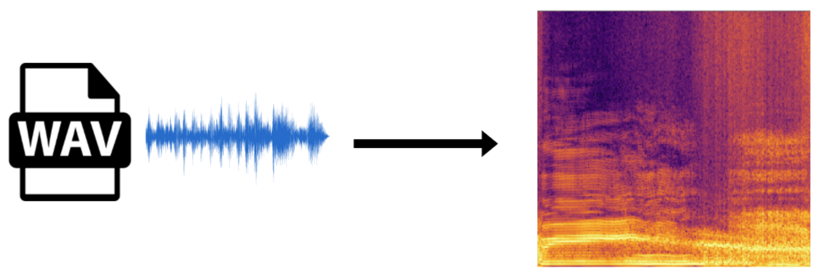

Figure 4: Turning .wav files into spectrograms

- We turn the .wav files into a magnitude spectrogram $X_{Mixture} \in \mathbb{R}_+^{F \times N}$ using STFT (short-time fourier transform)The resolution is dictated by the number of Frequency bins ($F = 256$) and the number of STFT ($N = 256$)
- Larger $F, N \rightarrow$ Larger spectrograms $\rightarrow$ Higher sampling accuracy but more expensive computations
- Smaller $F, N \rightarrow$ Smaller spectrograms $\rightarrow$ Lower sampling accuracy but less computationally demanding
- **Spectrograms $\neq$ Image**, and cannot be preprocessed as such. Normalization, image flipping (etc.) are not fitting in this context
- One spectrogram has the size 256×256 which is $\approx$1s of audio.

The original paper [2] used raw unnormalized magnitude spectrograms, and so did we.

Training and result

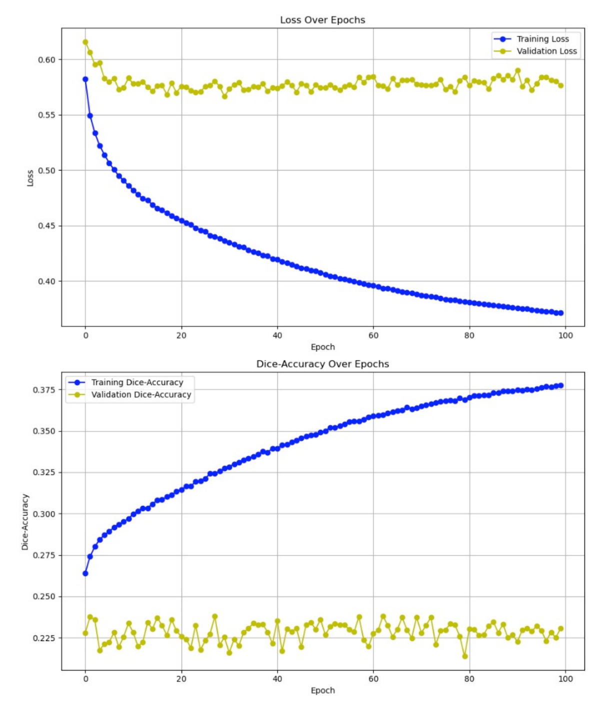

Figure 5: Loss and accuracy during training

Here are the information about the training method:

- Model trained on 100 epochs with batch of 8 ($\approx$ 13k samples per epoch)
- Use of batch normalization
- Adam as optimization method with a learning rate of 0.00001 and a weight decay of 0.000001
- Accuracy value of each epoch represent the DICE value

Here is an example of prediction our network is able to do:

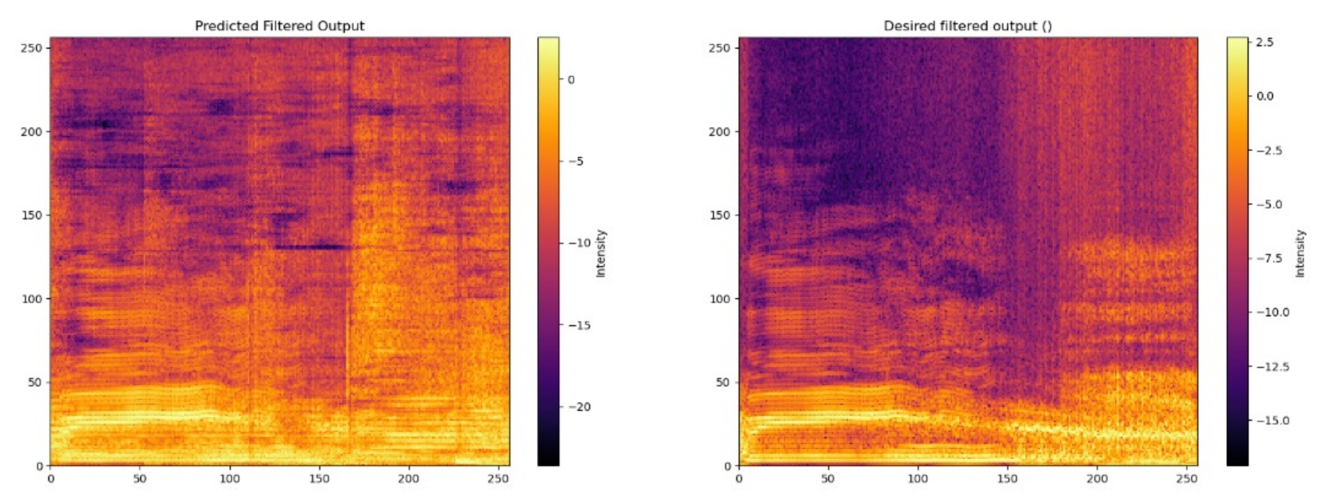

Figure 6: Example of prediction

According to the loss, **we detect a huge bias on our model**. Then we may ask:

- **Was transfer learning with this encoder a good choice?** No, transfer learning in this context did not generate great results.
- **Does this model fit the problem?** Yes, the U-Net has an overall effectiveness in dense prediction tasks.

References

[1] J. S. Ethan Manilow, Prem Seetharaman.
Open-source tools data for music source separation.
https://source-separation.github.io/tutorial, 2020.

[2] R. Gao and K. Grauman.
Co-separating sounds of visual objects.
2019.

nicgra@student.chalmers.se ducros@student.chalmers.se