# Stereo Source Separation

Tom Sabag

March 2025

## 1 Abstract

## Introduction

Music source separation is a fundamental task in audio signal processing, aiming to isolate individual sound sources from a mixed audio signal. The goal is to separate components such as vocals, accompaniment, and instrumental tracks from a stereo or multichannel recording. This task has significant applications in various fields, including music production, music remixing, music analysis,speech denoising and enhancement, spatial audio, and remastering for individuals with hearing impairments.

Over the past few years, deep learning methods, particularly neural networks, have shown remarkable success in solving source separation problems. By leveraging large datasets of labeled audio, these models are able to learn complex patterns spatial features of mixed signals, allowing for more accurate separation.

In this paper, I investigate the effectiveness of the Multi-Channel U-Net for music source separation using the MUSDB18 dataset, focusing on the separation of vocals and accompaniment in mixed audio tracks in a supervised setting.

## 2 Related Works

Music source separation has been an active area of research, with numerous methods developed over the years, ranging from traditional signal processing techniques to modern deep learning approaches.

In recent years, deep learning-based approaches have gained significant traction due to their ability to model complex patterns in audio data. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been widely used for source separation tasks, particularly in the context of single-channel separation. However, they often require large amounts of labeled training data and extensive computational resources, as demonstrated in this paper.

U-Net, a deep learning architecture originally proposed for biomedical image segmentation, has also been successfully applied to music source separation. The U-Net architecture, with its encoder-decoder structure and skip connections, is well-suited for preserving high-frequency details during signal separation. Recent works have demonstrated that applying Multi-Channel U-Net to music source separation tasks, such as vocal and accompaniment separation, can significantly improve separation performance, especially when trained on large datasets like MUSDB18.

Several other deep learning-based models have been proposed for music source separation, including Spleeter, Demucs, and OpenUnmix, which have demonstrated strong performance in separating vocals and accompaniment.

Spleeter, developed by Deezer, leverages a deep neural network with a U-Net architecture and is trained on large datasets, achieving impressive results with high separation quality. Spleeter is known for its efficiency, allowing for real-time processing with relatively low computational requirements.

Demucs, developed by Facebook Research, utilizes a more advanced architecture based on convolutional and recurrent neural networks. It performs exceptionally well in separating complex audio mixtures, especially in the presence of overlapping sources, and has shown state-of-the-art results in some bench-

marks.

OpenUnmix, developed by the research group at the University of Saarland, is another deep learning-based model focused on music source separation. Based on convloutions and bidirectional LSTM, it is known for its strong performance in separating vocals and accompaniment while minimizing artifacts. Similarly to my implementation, OpenUnmix internally predict stem masks as well. However, it minimizes MSE, while my implementation minimizes L1 loss.

# 3 Dataset

The MUSDB18 dataset is a widely used benchmark for music source separation tasks. It consists of 150 tracks, each provided as a multi-track recording with four instrument stems: vocals, drums, bass, and other accompaniment. The dataset is divided into a training set of 84 songs, a validation set of 16 songs and a test set of 50 songs. The tracks span various musical genres, ensuring diversity and generalizability of models trained on the dataset. Each track is available in stereo at a sampling rate of 44.1 kHz. MUSDB18 is well-suited for supervised source separation tasks, as it provides individual stems separately, allowing models to learn to isolate specific sources from the full mix using ground truth labels.

# 4 Data Representation

The stereo signal was converted to a mono signal by averaging both channels. The resulting mono signal was then transformed into a log magnitude spectrogram using the following parameters:

- n_fft = 1024
- hop length = 512
- window size = 512
- duration = 6 seconds

# 5 Architecture, Loss Function

## 5.1 Architecture

The architecture is based on a Multi Chanel U-Net model, consisting of an encoder-decoder structure with skip connections. Both the encoder and decoder comprise eight convolutional blocks, each followed by batch normalization and ReLU activation. A bottleneck layer connects the two parts, capturing deep feature representations. Skip connections link corresponding encoder and decoder layers to preserve spatial information. The network outputs two channels: one for the vocal mask and another for the accompaniment mask, enabling source separation through a single model predicting both masks simultaneously.

## 5.2 Loss Function

L1 loss, is commonly used for predicting masks in source separation tasks due to its ability to encourage sparse and smooth predictions. By minimizing the absolute difference between the predicted and ground truth masks, L1 loss ensures that the network learns to produce accurate and well-structured masks while reducing the impact of outliers. This makes it particularly effective for music source separation, where preserving fine-grained details in the estimated vocal and accompaniment components is crucial. After experimenting with MSE, L1 loss yielded slightly better results.

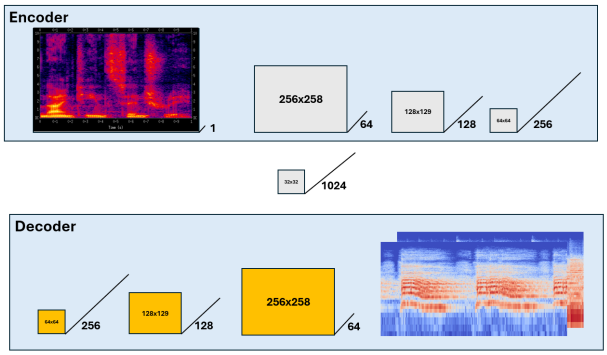


Figure 1: Vocal & Accompaniment Masks Example

# 6 Experiments

I conducted extensive hyperparameter tuning experiments, exploring various factors to optimize performance. Fitting the entire dataset was challenging, so I began by experimenting with different dataset sizes. Then i tested different learning rates, and the addition of a cross-entropy loss term. I also experimented with varying the encoder and decoder depth by adjusting the number of convolutional layers and kernel sizes. Additionally, I tested the impact of a sigmoid gate on the final layer and experimented with different weighting schemes for vocals and accompaniment to improve separation quality. In the following subsections i layout my experiments.

## 6.1 Dataset sizes

The model's performance across five different dataset sizes was evaluated to assess its ability to generalize across various songs.

Dataset sizes: $\in [1, 21, 41, 61, 81]$

**Loss:** As shown below, for datasets containing more than one song, the L1 loss converges at around 7. However, for the single-song dataset, the L1 loss drops below 7 after the first 30 epochs and continues to decrease over time.

**SDR:** Similarly, the SDR for data sets with multiple songs converges at approximately 20, which is lower than the SDR value observed for the data set with a single song after the initial few epochs.

This demonstrates the model's difficulty in generalizing when trained on more than one song, as it struggles to capture the diversity in the data. Additionally, as the size of the dataset increases, the computational bottleneck becomes much more dominant, requiring significantly more computational resources to train the model effectively.

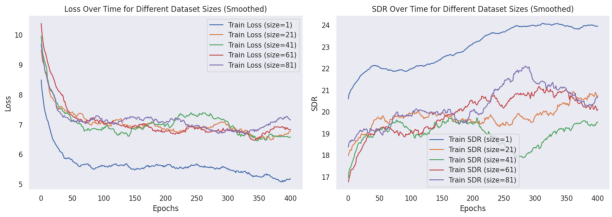


Figure 2: L1 Loss & SDR over different dataset sizes

## 6.2 Learning rates

I conducted a grid search for four different learning rates on a logarithmic scale to identify the optimal rate to effectively train the source separator network.

Learning rate: $\in [10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$

**Loss:** L1 loss decreases smoothly with learning rates of $10^{-2}$ or $10^{-3}$, while lower learning rates result in a slower reduction of loss.

**SDR:** Similarly, the SDR improves over time with higher learning rates ($10^{-2}$ or $10^{-3}$) and does not appear to converge within the first 100 epochs.


Figure 3: L1 Loss & SDR over different learning rates

## 6.3 Additional Cross Entropy Loss

The model initially performed poorly on the dataset. To verify its ability to learn meaningful representations, I added a source classification branch. This branch classifies spectrogram sources as either accompaniment or vocals by adding a linear layer with 2 neurons on top of the encoder and bottleneck layers, utilizing cross-entropy loss.

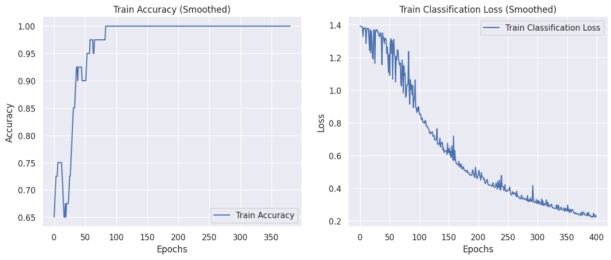


Figure 4: Source Classification Accuracy & Loss

## 6.4 Encoder - Decoder Depth

## 6.5 Network Depth

I evaluated the performance of the model in five different depths of network to investigate the impact of deeper encoders and decoders on source separation quality.

Network Depths: $\in$ [2, 4, 6, 8, 10]

**Loss:** The lowest L1 loss occurs at an intermediate depth, specifically with 8 or 10 convolution layers for the encoders and transpose convolution layers for the decoders. Using too many or too few layers leads to higher L1 losses, as the model either lacks the capacity to generalize or becomes too large to effectively learn.

**SDR:** Similarly, the highest SDR is achieved with 8 or 10 stacked layers, resulting in better source separation quality.


Figure 5: L1 Loss & SDR over Encoder & Decoder Depth

## 6.6 Convolution Kernel Sizes

The performance of the model was evaluated using five different kernel sizes. Since spectrograms rely heavily on the spatial structure of the data, kernel size plays a crucial role in capturing important spectral features.

Kernel Size: $\in$ [3, 5, 7, 9, 11]

**Loss:** The lowest L1 loss is achieved with an intermediate kernel size of 5 or 7. This is likely because smaller kernel sizes combined with deeper networks have fewer parameters but larger receptive fields, improving generalization.

**SDR:** Similarly, the highest SDR is observed with kernel sizes of 5 or 7, leading to better quality of source separation.


Figure 6: L1 Loss & SDR over Different Kernel Sizes

## 6.7 Importance of Sigmoid Gate

I investigated the role of a sigmoid gate at the end of the network to understand its impact on performance.

Sigmoid gate: $\in$ [True, False]

**Loss:** L1 loss is similar for both configurations, though it tends to be more unstable when a sigmoid gate is used.

**SDR:** Interestingly, the SDR appears to converge after 200 epochs without a sigmoid gate, while it continues to increase without converging when a sigmoid gate is applied.


Figure 7: L1 Loss & SDR With and Without Sigmoid Gate

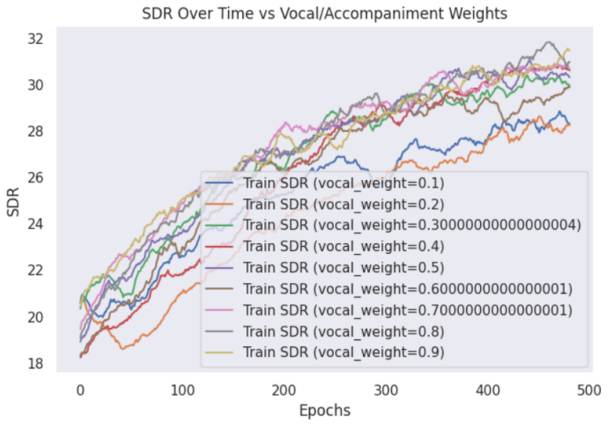


Figure 8: L1 Loss & SDR over Different Weighting Schemes

### 6.8 Weighting Schemas for Vocal and Accompaniment

I experimented with different weights for vocal and accompaniment source separation to assess their impact.

Vocal weight: $\in [0.1, 0.2, ..., 0.9]$

**Loss:** The lowest L1 loss occurs for smaller vocal weights (0.1, 0.2), as the L1 loss is generally lower for the separation of the accompaniment source compared to vocals.

**SDR:** On the other hand, the highest SDR is achieved with higher vocal weights (0.8, 0.9). Through experimentation with various weight combinations, I found that separating vocals is a more challenging task, requiring higher weights to be effectively learned.

## 7 Results

### 7.1 Overfitting a Single Song

The model effectively overfits the song A Classic Education - NightOwl, which has a duration of approximately three minutes. As shown in Figure 9, the SDR continues to improve over time without reaching a plateau. Similarly, the L1 loss steadily decreases, converging to a very low value for accompaniment source separation, demonstrating strong performance. Notably, the validation loss—computed on a broader dataset also decreases, suggesting that beyond overfitting a single song, the model retains some ability to generalize. For vocal source separation, the L1 loss appears to stabilize after around 3000 epochs, possibly due to the inherently greater difficulty of isolating vocals from the mix. As visualized in Figure 10, the reconstructed vocal spectrogram appears blurrier than the original, likely due to the model struggling to capture finer frequency details. This suggests that vocal separation is more challenging, as finer spectral components may be smeared or lost during reconstruction.

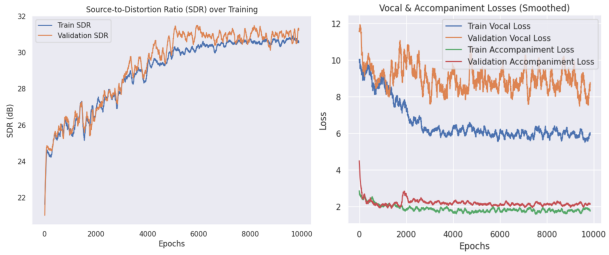


Figure 9: 1 Song L1 Losses & SDR over Time

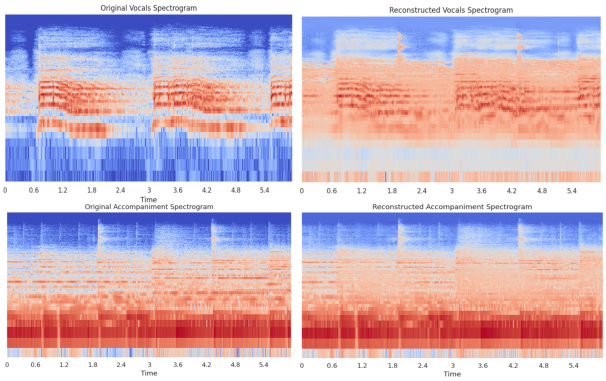


Figure 10: Original & Reconstructed Spectrograms

### 7.2 Overfitting a Small Dataset of 5 Songs

Overfitting a small dataset was more challenging. As visualized in figure 11, while the train SDR continues increasing, the validation SDR plateus after the first 2000 epochs. Moreover, it reaches a smaller value compared to the 1 song settings, while training on 5 times more data. However, the L1 loss of both the accompaniment and the vocals source seperation seem to converge to the same values, however needing 5 times more data. This emphasizes the computational resources bottleneck, as attempting to reach the same results of a single song dataset, trying to overfit a 100 songs training dataset could potentially take 100 times longer.

Overfitting a small dataset proved to be more challenging. As shown in Figure 11, while the training SDR continues to improve, the validation SDR plateaus after approximately 2,000 epochs. Notably, it reaches a lower value compared to the single-song setting, despite training on five times more data. However, the L1 loss for both accompaniment and vocal separation ultimately converges to similar values, albeit requiring significantly more data. This highlights the computational bottleneck—scaling from a single-song dataset to a 100-song dataset could potentially require 100 times more training time to achieve comparable results.


Figure 11: 5 Songs L1 Losses & SDR over Time

## 8 Discussion

n this paper, I developed a Multi-Channel U-Net to separate vocals and accompaniment from mixed audio tracks in a supervised setting. Through this process, I gained insights into the significance of robust representations, the application of computer vision techniques to audio and signal processing, the necessity of reliable evaluation metrics, and surprisingly, even how to effectively debug neural networks. During this work, I had some insights about potential ways to improve the quality of the algorithm

One potential improvement would be to enhance the spectrogram representation by increasing the number of frequency bins in the STFT used to generate it. Given the spectral-temporal tradeoff of the STFT, and the fact that spectral issues were more prominent, this adjustment could help the model achieve better generalization.

It might also be worth exploring alternative loss functions, such as Signal-to-Distortion Ratio (SDR) loss, which could provide more suitable evaluation criteria for the source separation task.

I also considered addressing the blurriness of the reconstructed vocals by incorporating a sharpening kernel, which could help reduce the blurriness in the spectrogram and improve the clarity of the separated sources.

Implementing this source separation algorithm, i came across two main challenges.

The first challenge was the significant computational bottleneck arose while attempting to implement the multi-channel U-Net, especially when trying to ensure effective source separation. This limitation highlights the need for more efficient architectures or optimizations.

The second challenge was the absence of an effective evaluation metric for source separation, complicating the assessment of the model's performance and its ability to generalize across different sources.

# 9 References

Kadandale, V. S., Montesinos, J. F., Haro, G., & Gómez, E. (2020). Multi-channel U-Net for music source separation. arXiv. https://doi.org/10.48550/arXiv.2003.10414

Romain Hennequin1, Anis Khlif1, Felix Voituret1, and Manuel Moussallam1(2020). SEPARATION: Spleeter - A fast and efficient music source separation tool with pre-trained models. arXiv preprint arXiv:1911.13281. https://www.theoj.org/joss-papers/joss.02154/10.21105.joss.02154

Alexandre Défossez, Nicolas Usunier, Léon Bottou, Francis Bach: Music Source Separation in the Waveform Domain (2021)https://arxiv.org/pdf/1911.13254

Stöter, F. R., & Ozerov, A. (2020). Open-Unmix: A reference implementation for music source separation. arXiv preprint arXiv:2003.03976. https://www.theoj.org/joss-papers/joss.01667/10.21105.joss.01667.pdf