Acoustic Guitar Source Separation using Open-Unmix

Amélie Van Loock, Andrea Lorenzetti, Pietro Iavernaro
Barcelona School of Telecom Engineering (ETSETB), UPC

Abstract
This project addresses the challenge of Music Source Separation (MSS), with a particular focus on isolating the Acoustic Guitar. MSS is of great interest to professional music creators, as it facilitates the remixing and reworking of songs with increased flexibility. These systems process a song as input and produce individual tracks for each instrument, providing enhanced control and expanded creative possibilities.

This report aims to provide a comprehensive overview of the state of the art in MSS, along with a detailed account of the experiments conducted using the Open-Unmix model.

Contents

1 Introduction

Music Source Separation (MSS) is a challenging problem within the broader field of Blind Source Separation (BSS), which aims to separate mixed signals into their individual components. A classic example of BSS is the "cocktail party problem", where a listener focuses on a single conversation in a room full of overlapping voices. While the human brain is naturally skilled at solving such tasks, digital signal processing struggles to replicate this ability.

MSS is particularly complex because musical instruments are intricately intertwined in both the time and frequency domains. Instruments often repeat rhythmic patterns and play harmoniously related notes, making it difficult to separate them without additional information. For example, purely statistical methods like Independent Component Analysis (ICA)[9] and Non-Negative Matrix Factorization (NMF)[1] often fall short when applied to music.

To address this, some approaches incorporate musical scores, which provide useful information about the notes and instruments involved. By knowing which notes are played and which instruments are present, it becomes easier to train effective models and achieve better separation[2].

In recent years, MSS has gained momentum thanks to advancements in deep learning and initiatives like the Sound Demixing Challenge 2023, sponsored by Sony, Moises, and Mitsubishi Electric. * Modern deep

*https://www.aicrowd.com/challenges/sound-demixing-challenge-2023

learning methods, including Multi-Layer Perceptrons (MLP), Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), have largely replaced traditional techniques[4].

However, a recurring challenge is that many models focus solely on either the time or frequency domain, rather than leveraging both. Notable architectures for MSS include Band-Split RNN[5], Wave U-Net[8], TFC-TDF-UNet V3[3], and Open-Unmix[10]. For this project, Open-Unmix was selected as the primary model, and its architecture and functionality will be detailed in the next section.

Additionally, resources like the SigSep website † provide a wealth of publicly available datasets and open-source implementations for MSS research, supporting further advancements in the field.

The adoption of deep learning has transformed MSS, but incorporating domain-specific knowledge of audio and music remains critical for building effective systems[2].

2 Model Overview

A closer look at Open-Unmix

Open-Unmix[10] is a deep neural network designed for music source separation (MSS). It provides tools for isolating music into four stems: vocals, drums, bass, and other. The models within the framework are trained on the freely available MUSDB18 dataset[7].

2.1 MUSDB18 dataset

The MUSDB18 dataset consists of 150 full music tracks, amounting to approximately 10 hours of audio across various genres. Each track includes isolated stems for bass, drums, vocals, and other components. The dataset is organized into two subsets:

- Train: 100 tracks for model training.
- Test: 50 tracks for performance evaluation.

All tracks are stereo and encoded at a sampling rate of 44.1 kHz. For higher-quality data, MUSDB18-HQ offers uncompressed WAV files.

†https://sigsep.github.io/

2.2 Core model architecture

Open-Unmix employs a three-layer bidirectional Long Short-Term Memory (LSTM) network. The architecture learns to predict the spectrogram of a target source from the spectrogram of a mixed audio track. To improve accuracy, the framework uses the Mean Squared Error (MSE) as its loss function during training.
A key aspect of the model is its ability to compress the frequency and channel axes after initial normalization. This reduces redundancy in the input data, enhancing training efficiency and accelerating convergence. While the bidirectional LSTM allows Open-Unmix to consider both past and future temporal information, this design choice prevents real-time processing.
After the LSTM processes the signal, the model decodes the output back to its original input dimensions. A mask is then generated by multiplying the output with the input magnitude spectrogram. The model optimizes this mask for effective source separation.

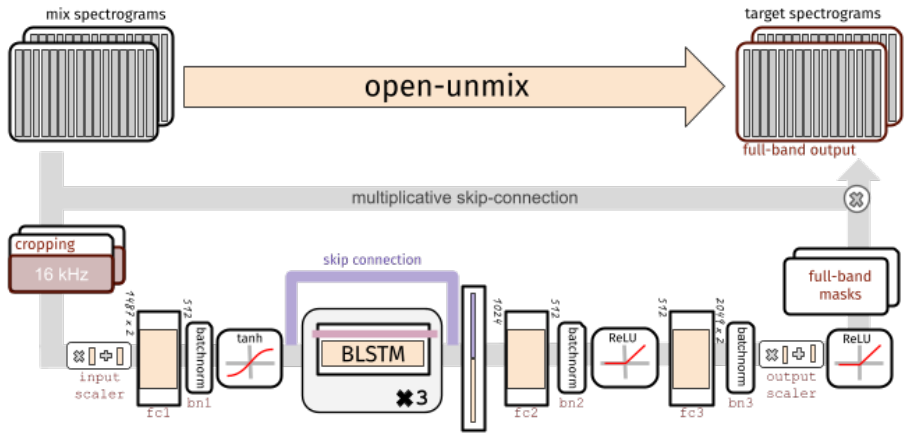

Figure 1: Structure of the Open-Unmix architecture.

2.3 Seperator integration and reconstruction

The Separator model combines the outputs from the individual Open-Unmix models (vocals, drums, bass and other) using a multichannel generalized Wiener filter. The filter refines the separation process by leveraging phase information. Finally, the inverse Short-Time Fourier Transform (STFT) is applied via Torchaudio to reconstruct the separated audio sources into the time domain.

2.4 Pre-trained separator models

Open-Unmix provides several pre-trained models, each designed for specific applications:

- umxhq (default): Trained on MUSDB18-HQ, providing full-bandwidth separation up to 22050 Hz.
- umx: Trained on the standard MUSDB18 dataset, with a bandwidth limit of 16 kHz due to AAC compression.
- umxse: A speech enhancement model trained on the Voicebank + DEMAND corpus, designed specifically for denoising speech.

2.5 Input handling

Open-Unmix supports two input methods:

- **Time-domain input** (via models.separator): Accepts a tensor with the shape (nb_samples, nb_channels, nb_timesteps), where:
  - nb_samples represents the batch size.
  - nb_channels can be 1 (mono) or 2 (stereo).
  - nb_timesteps is the number of audio samples in the time domain.

  In this mode, the model computes the STFT dynamically using Torch or Asteroid Filterbanks.
- **Magnitude spectrogram input** (via models.OpenUnmix): Accepts precomputed magnitude spectrograms with the shape (nb_frames, nb_samples, nb_channels, nb_bins), where nb_frames and nb_bins correspond to the time and frequency dimensions of the STFT. The spectrogram input undergoes normalization using the global mean and standard deviation for each frequency bin across all frames. Multi-stage normalization ensures robustness against variations in gain, improving the generalizability of the model.

3 Experiment and Discussion

Overview of the Fine-Tuning Process for Open-Unmix

3.1 Dataset

The dataset used in this experiment is BabySlakh[6], a subset of the Slakh2100 dataset. BabySlakh consists of the first 20 tracks of Slakh2100, with audio in WAV format sampled at 16 kHz.

The Synthesized Lakh (Slakh) dataset is designed for music source separation and multi-instrument transcription, featuring multitrack audio aligned with MIDI data. Tracks in Slakh are generated by synthesizing individual MIDI files from the Lakh MIDI Dataset v0.1 using professional-grade, sample-based virtual instruments. The full Slakh2100 dataset contains 2,100 tracks synthesized with 187 unique patches grouped into 34 instrument classes.

In this experiment, guitar tracks were grouped by type, such as acoustic, clean electric or overdriven electric. Each type has unique tonal qualities, shaped by the guitar's design and any effects used, like distortion or modulation. Due to the complexity of managing such diversity, this study focused exclusively on acoustic guitars.

3.2 Preprocessing

A Python script was used to preprocess the dataset, applying the following steps:

1. Filtering: Tracks containing both acoustic guitar and bass were retained.
2. File selection: Only mix.wav, acoustic_guitar.wav, and bass.wav files were kept; other stems were discarded.
3. Combining stems: Multiple acoustic guitar stems in a track were combined into a single file.
4. Segmentation: Tracks were divided into 20-second segments with a 10-second overlap.
5. Data augmentation: Gaussian noise, pitch shifting and time stretching were applied to diversify the dataset.
6. Content filtering: Segments with less than 15% acoustic guitar content were removed.
7. Organization: The processed data was split into Train (80%), Validation (10%), and Test (10%) sets.

3.3 Training

The model was fine-tuned using the following hyperparameters:

- Learning Rate: 0.001, with adaptive decay (factor 0.3) after epochs without improvement
- Batch Size: 32
- Epochs: 60
- Hidden Size: 512 units per layer

The training and validation losses are shown in Figure 2. The model exhibited rapid initial convergence, with the best performance observed at epoch 57, achieving a loss of 0.65. Validation loss closely tracked training loss, indicating minimal overfitting.

3.4 Evaluation metrics

Two key metrics were used to evaluate model performance:

1. **Signal-to-Distortion Ratio (SDR)** [11] [12]**:**

$$SDR = 10 \log_{10} \frac{\|s_{target}\|^2}{\|e_{interf} + e_{noise} + e_{artif}\|^2} \quad (1)$$

where S_{target} represents the true signal, and the denominator sums the errors

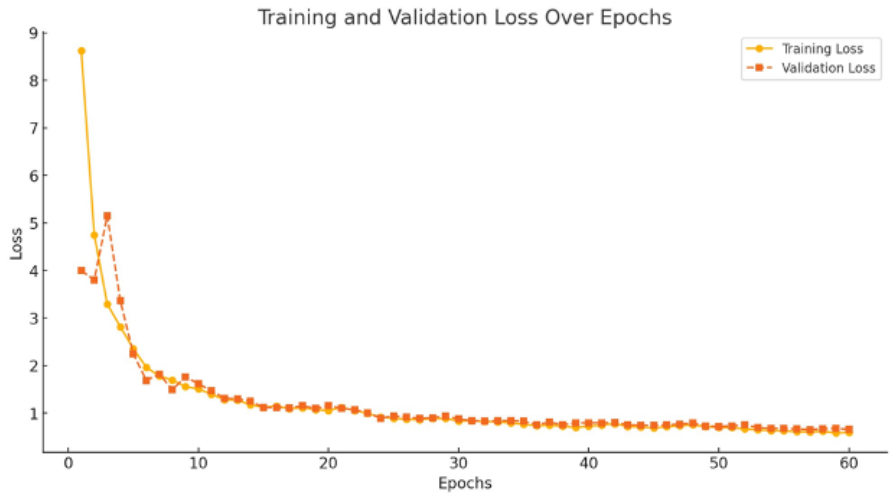

Figure 2: Training and Validation graphs.

from interference, noise, and artifacts. Higher SDR values indicate better separation quality.

2. **Signal-to-Artifacts Ratio (SAR)** [11]**:**

$$SAR = 10 \log_{10} \frac{\|s_{target} + e_{interf} + e_{noise}\|^2}{\|e_{artif}\|^2} \quad (2)$$

SAR measures the cleanliness of the separated signal by quantifying the presence of artifacts. Higher values denote reduced artifacts.

3.5 Results

The performance of the fine-tuned model was evaluated and compared against the original Open-Unmix model. Key distinctions include the datasets used: the original Open-Unmix was trained on the MUSDB18 dataset (real, recorded audio), while the fine-tuned model was trained on BabySlakh (synthetic, computer-generated audio). This difference likely influenced the observed results.

Aucoustic Guitar Stems

- SDR: Ranged from -5.05 to 10.35, with an average of 1.38.
- SAR: Ranged from -25.04 to 10.9, with an average of -6.06.

Bass Stems

- SDR: Ranged from -5.38 to 9.32, with an average of 1.42.
- SAR: Ranged from -25.77 to 9.32, with an average of -5.14.

These results suggest that the fine-tuned model effectively adapted to acoustic guitar stems, although some challenges remain, as indicated by the SAR scores.

4 Conclusions

The fine-tuning of Open-Unmix for acoustic guitar separation yielded promising results, with rapid convergence during training and evaluation metrics that demonstrated performance comparable to the baseline Open-Unmix model on bass. Notably, this was achieved using the newly adopted BabySlakh_16k dataset, showcasing the adaptability of Open-Unmix to handle new sources and synthetic data.

These findings highlight the potential of leveraging synthetic datasets for music source separation tasks, particularly in cases where real recordings are unavailable or limited. The successful adaptation of Open-Unmix to acoustic guitar separation opens the door to expanding its application to other instrument types or domains.

To further enhance the model's performance and robustness, several avenues for future work are recommended:

1. **Expanding the dataset:** Incorporating the complete Slakh2100 dataset, which provides a significantly larger and more diverse pool of audio tracks (approximately 100 GB), would allow for more robust training. Extending the number of training epochs in tandem with this larger dataset could yield better generalization and improved performance metrics.
2. **Integrating real recordings:** BabySlakh_16k consists exclusively of synthetic audio. Adding real-world recordings to the training set would create a more realistic training environment and enable the model to generalize better to real-world applications. This integration could also facilitate a more comprehensive evaluation of the model's effectiveness in practical scenarios.
3. **Exploring advanced data augmentation:** While the experiment already utilized techniques like Gaussian noise addition, pitch shifting and time stretching, further exploration of domain-specific augmentations (e.g., emulating room acoustics or microphone effects) could provide additional robustness.
4. **Adapting to real-time processing:** While Open-Unmix is inherently a non-real-time system due to its bidirectional LSTM architecture, exploring lightweight or causal adaptations could enable real-time source separation, broadening its usability in live applications such as concerts or broadcasts.
5. **Broader instrument coverage:** Following the success with acoustic guitars, extending the fine-tuning process to other underrepresented instrument categories, such as string ensembles or percussion, could further demonstrate the versatility of the approach.

By addressing these areas, the fine-tuned Open-Unmix model could evolve into a more comprehensive and practical tool for music source separation, with potential applications in music production, transcription, and audio restoration. These results also underscore the importance of synthetic datasets as a valuable resource in the continued development of source separation models.

References

[1] Rafal Zdunek. Anh Huy Phan Shun-ichi Amari Andrzej Cichocki. *Non- negative Matrix and Tensor Factorizations.* John Wiley Sons, Ltd, 2009.

[2] Sebastian Ewert, Bryan Pardo, Meinard Müller, and Mark D. Plumbley. Score-informed source separation for musical audio recordings: An overview. *IEEE Signal Processing Magazine*, 31:116–124, 2014.

[3] Minseok Kim, Jun Hyung Lee, and Soonyoung Jung. Sound demixing challenge 2023 music demixing track technical report: Tfc-tdf-unet v3, 2023.

[4] Ruolun Liu and Suping Li. A review on music source separation. In *2009 IEEE Youth Conference on Information, Computing and Telecommunication*, pages 343–346, 2009.

[5] Yi Luo and Jianwei Yu. Music source separation with band-split rnn, 2022.

[6] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux. Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019.

[7] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel M. Bittner. Musdb18 - a corpus for music separation. 2017.

[8] Daniel Stoller, Sebastian Ewert, and Simon Dixon. Wave-u-net: A multi-scale neural network for end-to-end audio source separation, 2018.

[9] James V Stone. *Independent component analysis: a tutorial introduction.* 2004.

[10] Fabian-Robert Stöter, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji. Open-unmix - a reference implementation for music source separation. *Journal of Open Source Software*, 4(41):1667, 2019.

[11] E. Vincent, R. Gribonval, and C. Fevotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469, 2006.

[12] Emmanuel Vincent, Shoko Araki, Fabian Theis, Guido Nolte, Pau Bofill, Hiroshi Sawada, Alexey Ozerov, Vikrham Gowreesunker, Dominik Lutter, and Ngoc Q.K. Duong. The signal separation evaluation campaign (20072010): Achievements and remaining challenges. *Signal Processing*, 92(8):1928–1936, August 2012.