Music Source Separation for PyTorch

CSCI 635 - Introduction to Machine Learning

Poonam Pawar, Kalpalathika Ramanujam, Mohammed Raeesul Irfan Riaz Ahmed, Sridhar Shenoy

1. Task Definition, Evaluation Protocol, and Data

1.1 Task Definition: The ‘what’

Music source separation involves decomposing a music mixture into its individual constituent components, such as vocals, drums, bass, and other instruments. The goal is to isolate each source stem from the mixture, allowing users to perform tasks like remixing, upmixing, creating karaoke versions, audio restoration, and sample creation [1].

1.2. Task Intent - The ‘why’

- Enables artists and creators to remix and rearrange music freely by providing access to individual stems.
- Aids in audio restoration and improves audio quality in post-production for various media.

1.3. Dataset

The implementation is based on the *MUSDB18 dataset* [2], which is a freely available dataset commonly used for music source separation research. MUSDB18 consists of music tracks with separate ground-truth stems for vocals, drums, bass, and other accompaniment instruments, making it suitable for training and evaluating source separation models.

For the convenience of training, *7 second preview version*[3] of the MUSDB18 dataset was used.

1.4. Regression Nature of Model

The "Open-Unmix" model is primarily a regression model which is designed to estimate the magnitudes of different audio sources present in a mixed audio recording and visually the process can be shown as followed

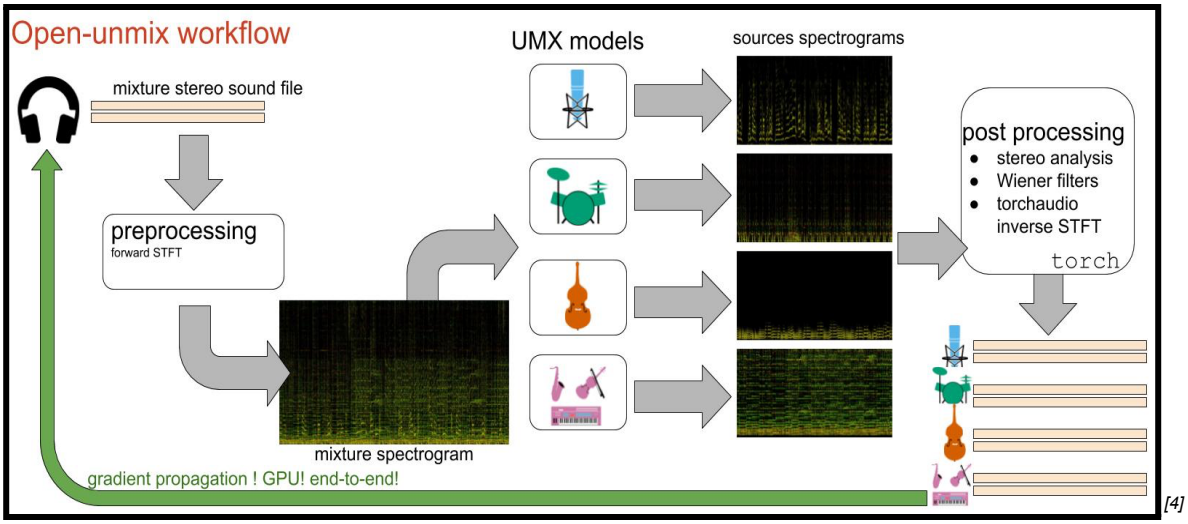


Fig 3. Open-unmix workflow

1.5. Evaluation Protocol and Metrics

- **Installation of museval Package:** To perform the evaluation, the *museval*[5] package is installed using the command "pip install museval". This package provides the necessary tools and metrics for assessing the quality of audio source separation.

- ***Objective Metrics***[6]: The evaluation metrics used in the assessment is Signal-to-Distortion Ratio (SDR). This quantifies the quality of the separated sources by measuring the similarity to the reference (ground-truth) sources.

1.6. Main and Associated References:

Main Paper: The 2018 Signal Separation Evaluation CampaignOpen-Unmix - A Reference Implementation for Music Source Separation[1]
Speech Enhancement Using Open-Unmix Music Source Separation Architecture[7]
Implementing Transformer Architectures for Audio Source Separation[8]
Addressing The Confounds Of Accompaniments In Singer Identification[9]

2. Neural Network Machine Learning Model

2.1 Overview

The proposed machine learning model in Open-Unmix is a deep neural network tailored for music source separation. Its architecture comprises several source models, each for a specific musical component like vocals or drums, built using a three-layer bidirectional deep LSTM. This enables it to process information from both past and future time frames.

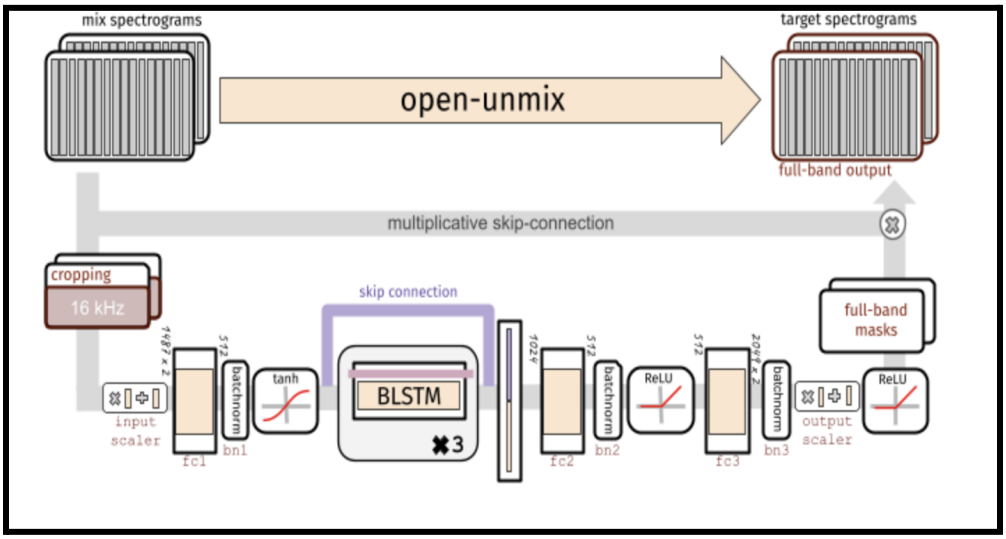
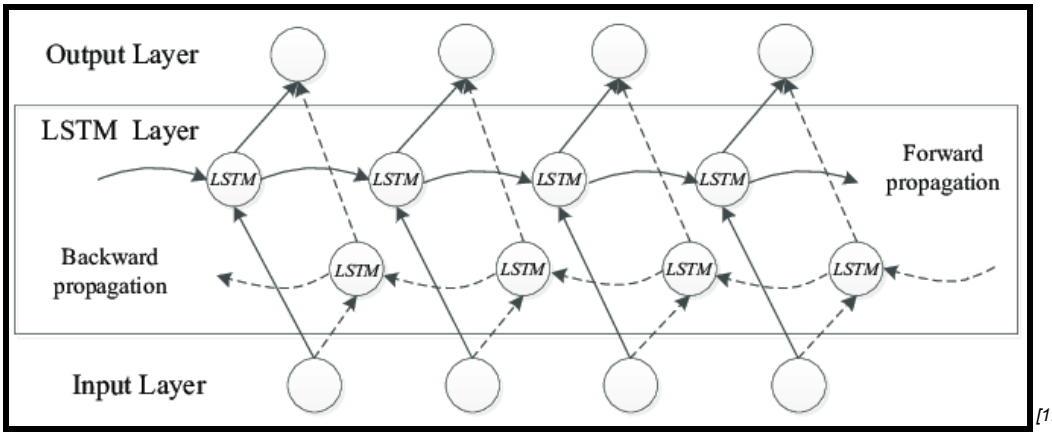


Fig 1. Representation of the Open-Unmix model architecture, showing the flow of input data through the LSTM layers and the mask application process to generate the target spectrograms. [10]

2.2 Long short-term memory network

- LSTM stands for Long-Short Term Memory network and is a type of recurrent neural network.[11]
- LSTMs can learn long-term dependencies in sequence prediction problems.[11]
- BLSTM (Bidirectional LSTM) uses data from both past and future simultaneously.
- Bidirectional LSTMs have information about the entire sequence, including points before and after any given point.[12]



[12]

Fig 2. The architecture of a BLSTM.

Working in the time-frequency domain, the model predicts the magnitude spectrogram of the target source based on the magnitude spectrogram of the mixed input. To achieve this, a mask is applied to the input, and the model's optimization is done in the magnitude domain using mean squared error.

2.2.3. Input Stage

Open-Unmix accepts either a time-domain signal tensor or a magnitude spectrogram as input. For time-domain signals, it performs an on-the-fly Short-Time Fourier Transform (STFT)[13]. The input spectrogram is standardized using the global mean and standard deviation for each frequency bin. Batch normalization[14] is used within the model to improve robustness against gain variations.

2.2.4. Dimensionality reduction

During the dimensionality reduction step, the LSTM reduces the frequency and channel axes of the model to reduce redundancy and facilitate faster convergence. The following image shows the process of dimensionality reduction in LSTM.[15]

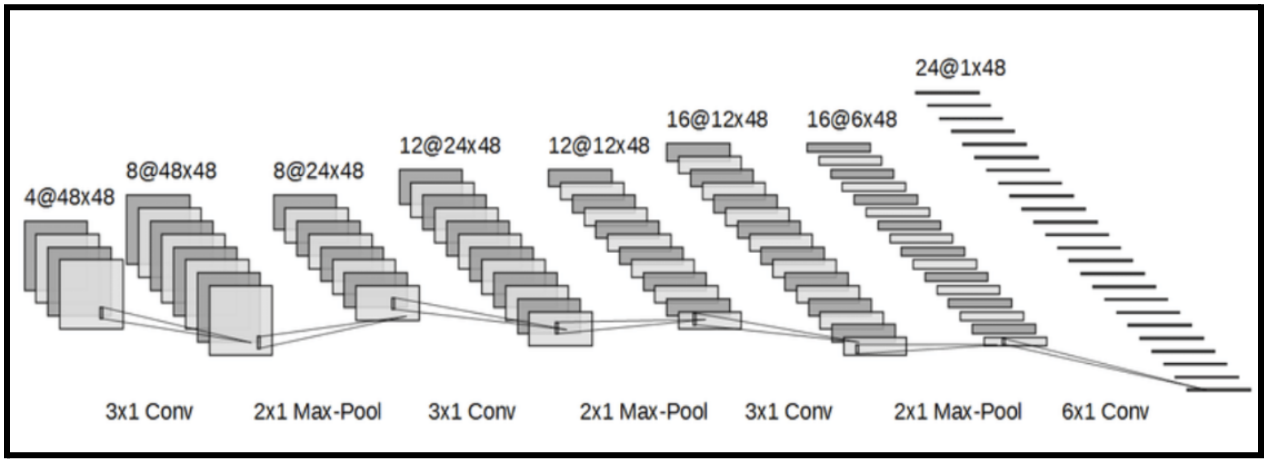


Fig 3. Reduce-LSTM. Principle of dimension reduction.[15]

2.2.5 Output stage

In the output stage, after the LSTM processing, the signal is restored to its original input dimensionality. The output is then multiplied with the input magnitude spectrogram, enabling the model to learn a mask that aids in separating the desired source from the mixture.

The Separator *`models.Separator`*[16] combines multiple Open-Unmix spectrogram models for each target source using a multichannel generalized Wiener filter. The filtering is a differentiable (but parameter-free) version of norbert[17]. It's important to note that due to its recurrent nature, the Open-Unmix model can handle audio signals of arbitrary lengths. *It is not suitable for real-time processing due to the bidirectional LSTM's computational requirements*.

2.2.6. Loss Function and Metrics

- Open-Unmix, using the L2 loss[18] ensures that the separated audio sources are as close as possible to the original sources in the mixture, by minimizing the difference between the predicted separated signals and the ground-truth separated signals.
- The squaring in the L2 loss function gives more weight to larger errors. This means that the model is penalized more for making larger mistakes, and this drives the model to make predictions that are close to the actual values.

3. Experiment

3.1 Research Questions:

1: "How does the batch size, hidden size, and number of samples per track influence the loss and training time of the open-unmix model?"
2: "Can modifications to the Open-Unmix model, specifically by altering or replacing the bidirectional LSTM, enable real-time audio source separation without significantly compromising the quality of separation?"

3.2 Design

3.2.1 Hypothesis

1. Increasing the batch size and hidden size will reduce the model's loss, but the relationship between these parameters and training time is non-linear
2. By replacing the bidirectional LSTM in the Open-Unmix model with a unidirectional LSTM or another less computationally intensive architecture, the model can achieve real-time audio source separation.
3. There is a direct relationship between hidden size and the rate at which the loss converges and the minimum loss that can be achieved up to a certain point after which the relationship becomes inversely proportional.

3.2.2. Independent variables (Experimental Settings):

These are the variables that you would manipulate or change to see their effect on the model's performance. For our experiment, these are:*[19]*

- --epochs: The total number of times the learning algorithm will work through the entire training dataset.
- --nb-workers: The number of parallel workers used for data loading.
- --batch-size: The number of training examples utilized in one iteration.
- --hidden-size: The size of the hidden layers in the neural network.
- --samples-per-track: The number of samples that are randomly drawn from each audio track.
- --unidirectional: A flag that changes the bidirectional LSTM to unidirectional, suitable for real-time applications.

3.2.3. Control variables (Biases and Modeling assumptions)

These are the variables that you would keep constant throughout the experiment to ensure that they don't influence the outcome. For our experiment, these are:*[19]*

- --patience: The number of epochs to wait before stopping the training process if no improvement is seen.
- --lr-decay-patience: The number of epochs with no improvement after which the learning rate will be reduced.
- --lr-decay-gamma: The factor by which the learning rate will be reduced.

- --bandwidth: The maximum frequency bandwidth in Hertz processed by the LSTM, while input and output remain full bandwidth.
- --nb-channels: The number of audio channels for the model, such as mono or stereo.

3.2.4 Dependent variables (Results Analysis)

These are the outcomes or results that you would measure after manipulating the independent variables. For our experiment, these are:[19]

- Training and validation loss: The measure of the model's error on the training and validation datasets.
- Time taken for training: The duration required to complete the training process.
- Model accuracy or performance on the validation set: The percentage of correct predictions made by the model on the validation dataset.

3.3 Methodology

Baseline for Model Evaluation: In any experimental study, establishing a baseline is crucial to provide a reference point against which the impact of modifications can be measured.
For our experiment, we have chosen two baselines:[19]

- Default Open-Unmix Model (UMX): The UMX model, provided by default, represents the original, unaltered state of the Open-Unmix framework. Using UMX allows us to measure the impact of any modifications directly against the original design.
- Open-Unmix Model Trained for 1 Epoch: Training a model for just one epoch provides a snapshot of its early learning phase. This baseline will help us understand the initial learning trajectory of the model and how subsequent training epochs, with or without modifications, deviate or build upon this trajectory.

Modifications:[19]

- Batch Size: Different batch sizes (e.g., 8, 16, 32) will be tested to observe their influence on the model's convergence speed, loss, and training time..
- Hidden Size: Different hidden sizes (e.g., 256, 512) will be experimented with to assess their impact on the model's performance and overfitting tendencies..
- Samples Per Track: Different numbers of samples per track (e.g., 16, 32) will be tested to determine their effect on model training and generalization.
- LSTM Modifications: The bidirectional LSTM will be replaced. The --unidirectional flag will be used to switch the model to a unidirectional LSTM.

Training:[19]
For each combination of batch size, hidden size, samples per track, and LSTM modifications, the model will be trained using the MUSDB18 dataset.

Results will be analyzed to:

- Determine the relationship between batch size, hidden size, samples per track, and the model's loss and training time.

- Assess the trade-offs between real-time processing capability and quality of separation for models with LSTM modifications compared to the original Open-Unmix model.

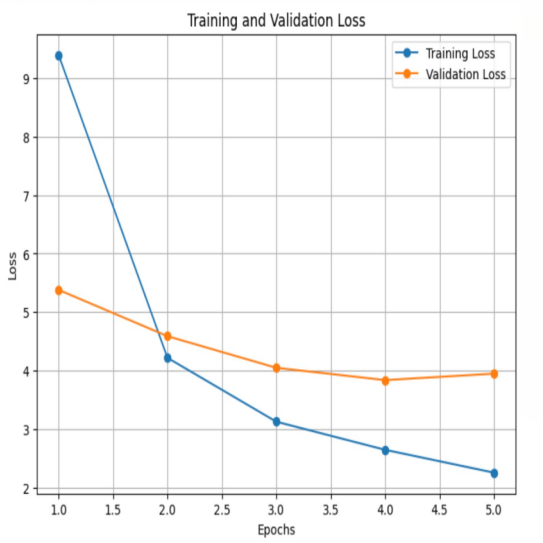
4. Experimental Results and Discussion

4.1 Evaluation metrics

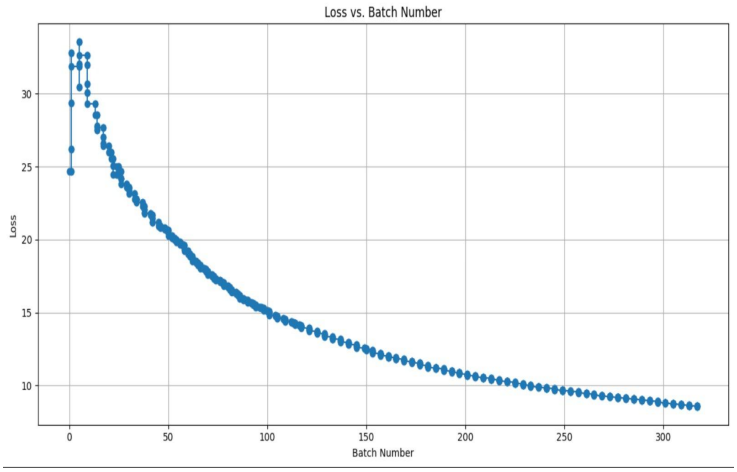
Model	SDR	SIR	ISR	SAR
UMX	7.537	12.091	14.586	8.679
open-unmix	2.326	3.028	6.588	5.347
open-unmix2	2.746	2.954	7.706	6.288
open-unmix3	1.934	2.849	5.579	4.07
open-unmix4	2.767	3.052	9.831	7.201
open-unmix5	2.898	3.628	9.272	6.63

4.2 Training Parameters and Performance:

Model name	Batch Size	Workers	Hidden Size	Samples Per track	Loss	Time	Epochs
open-unmix	8	4	256	16	9.536	7:50	1
open-unmix2	4	4	256	16	8.55	7:43	1
open-unmix3	16	4	512	16	11.2	7:42	1
open-unmix4	16	4	512	16	3.2	31:09:00	4
open-unmix 5	32	4	512	32	2.26	1:18:00	5
Open-unmix 6	32	4	512	32	9.56	20:00	1



Open-unmix 5 - Loss vs Epochs



open-unmix 3- Loss vs Batch Number

- **open-unmix2:** With the smallest batch size of 4 and a hidden size of 256, it managed to reduce the loss to 8.55 in a slightly shorter duration of 7:43 for one epoch.
- **open-unmix3:** Despite increasing the batch size to 16 and doubling the hidden size to 512, this model saw an increased loss of 11.2 but maintained a similar training time of 7:42 for one epoch.
- **open-unmix4:** Keeping the batch and hidden sizes the same as open-unmix3, this model drastically reduced its loss to 3.2 but required a significantly longer training time of over 31 minutes across 4 epochs.
- **open-unmix5:** By further increasing the batch size to 32 and adjusting the samples per track to 32, this model achieved the lowest loss of 2.26, taking a total of 1:18:00 for 5 epochs.

As the models progressed from open-unmix to open-unmix5, there were notable changes in batch sizes and hidden sizes, leading to varying losses and training durations.

4.3. Conclusion

- The relationship between batch size and loss reduction in training is not straightforward, as illustrated by the contrast between the open-unmix2 and open-unmix3 models for a single epoch. The training time does not relate to batch size since the number of batches is reduced accordingly so that each signal is processed a constant number of times.

- The unidirectional LSTM model does possess the ability to process the signals in real time but heavily depends on the characteristics of the machine in which it is executed. In order to achieve real time audio source separation, the model needs to process a batch of signals faster than the rate at which the audio is played. Since our model crops the frequency to 16 khz, it makes the processing of audio signals significantly faster. The presence of a GPU or multiple cores in the machine also significantly boosts audio separation time.

- The hidden size parameter controls the number of hidden layers in the LSTM. We have found that a hidden size of 256 is optimal for processing 16khz signals above which the performance begins to decrease due to the model capturing very subtle patterns in frequencies which cause overfitting to training data.The presence of Relu activation layers makes the model more robust to increase in hidden size.

4.4. Future Directions

- If we manage to enhance the efficiency of the unidirectional LSTM, it could be a candidate for real-time audio separation tasks like those employed in Apple Music. We can explore various cropping resolutions for frequencies other than 16kHz to potentially achieve improved performance and resolution in the resulting output signal.

7. References:

[1]Main Paper : The 2018 Signal Separation Evaluation Campaign Open-Unmix - A Reference Implementation for Music Source Separation- https://www.theoj.org/joss-papers/joss.01667/10.21105.joss.01667.pdf;
[2]musdb18 dataset - https://sigsep.github.io/datasets/musdb.html
[3]7 second preview version of musdb18 - https://zenodo.org/record/3270814
[4] Open-unmix regression model architecture -https://devpost.com/software/open-unmix
[5] museval - source separation evaluation tools for python -https://github.com/sigsep/sigsep-mus-eval
[6] Objective Measures and Metrics - https://source-separation.github.io/tutorial/basics/evaluation.html
[7]Speech Enhancement Using Open-Unmix Music Source Separation Architecture - https://ieeexplore.ieee.org/abstract/document/9753157?casa_token=nzT6PoUivkwAAAAA:dKxyvaD25JbB-88UT3pkxQPyF-pHVefo42Rmwvgyc5SwqwryQS7LySzshqMBWqZ40Rsz6jxjBA
[8]Implementing Transformer Architectures for Audio Source Separation - https://ieeexplore.ieee.org/abstract/document/10002207/authors#authors
[9]Addressing The Confounds Of Accompaniments In Singer Identification - https://ieeexplore.ieee.org/abstract/document/9054069?casa_token=Ok-9Q6MJ-h8AAAAA:bimAmGKqCQC4bfulRk06EM3wPmTzRBwTmuit0oCcMrJoOhI9MuX_yGVAO5T7cLJsp-4LHdHWCw
[10] Pytorch - https://pytorch.org/hub/sigsep_open-unmix-pytorch_umx/
[11] LSTM - https://en.wikipedia.org/wiki/Long_short-term_memory
[12] BLSTM Architecture https://www.researchgate.net/figure/The-architecture-of-BLSTM-model_fig2_339174926
[13]STFT Short-time Fourier transform https://en.wikipedia.org/wiki/Short-time_Fourier_transform
[14] Batch Normalization - https://en.wikipedia.org/wiki/Batch_normalization
[15]Principle of Dimension Reduction - https://www.researchgate.net/figure/Reduce-LSTM-Principle-of-the-dimension-reduction_fig3_344373314
[16]Separator class - https://github.com/sigsep/open-unmix-pytorch/blob/4318fb278e1863f4cf8556b513987faf14a15832/openunmix/model.py#L169
[17]Norbert - implementation of multichannel Wiener filter https://github.com/sigsep/norbert
[18]UMX Loss Functions - https://github.com/enricguso/UMX_loss_functions
[19]A Reference Implementation for Music Source Separation- https://github.com/sigsep/open-unmix-pytorch
[20]SiSEC Objective Evaluation Results - https://sisec18.unmix.app/#/results/vocals/SDR
[21]Magnitude spectrogram - https://www.researchgate.net/figure/Magnitude-spectrogram-of-four-example-music-signals-vocals-left-drums-mid-left_fig2_329464103
[22]Exploring Aligned Lyrics-Informed Singing Voice Separation - https://arxiv.org/abs/2008.04482