STMAE - Music Source Separation

ABOUELAZM Youssef - 10960436
BINGLING Wu - 10975956
GARCÍA Adrián - 11105141
OUALI Ernest - 10984484

May 2025

Here are the GitHub repositories:

- Data Augmentation, U-Net Implementation, NMF Implementation
- Unofficial BandSplitRNN Implementation: Used baseline repo for on-top implemenation.
- Unofficial BandSplitRNN + SE-layer Implementation: Added se_layer.py module and integration on overall pipeline.

1 Introduction

Music Source Separation (MSS) represents a fundamental challenge in music information retrieval and audio signal processing, involving the decomposition of polyphonic musical mixtures into their constituent individual sources, such as *vocals*, *drums*, *bass*, and *other accompaniment instruments*. This separation capability serves as a cornerstone technology for numerous applications including music remixing, automatic transcription, karaoke systems, music education tools, and enhanced audio production workflows.

The evolution of MSS techniques has been marked by a significant paradigm shift from traditional signal processing approaches to sophisticated **deep learning methodologies**. Recent advances in neural network architectures have demonstrated remarkable improvements in separation quality, with models such as **BandSplitRNN (BSRNN)** achieving state-of-the-art performance by effectively exploiting both *temporal dynamics* and *spectral characteristics* inherent in audio signals. The BSRNN architecture's success stems from its ability to process frequency-domain representations through specialized recurrent structures that capture long-term dependencies while maintaining computational efficiency.

Despite these advances, existing models including the original BSRNN architecture exhibit limitations in their capacity to adaptively focus on the most informative **spectro-temporal features**. Specifically, these models lack sophisticated *attention mechanisms* that could enable selective emphasis on relevant frequency bands and temporal segments, potentially leaving room for improvement in separation accuracy and robustness across diverse musical content.

To address these limitations, this work proposes an enhanced version of the BSRNN architecture through the strategic integration of **Squeeze-and-Excitation (SE) attention layers**. The proposed **SE-enhanced BandSplitRNN** maintains the core structural advantages of the original framework while incorporating *channel-wise attention mechanisms* within residual blocks. These SE layers implement *soft attention strategies* that enable the model to dynamically recalibrate feature representations, thereby improving its ability to focus on salient information across both frequency and temporal dimensions, ultimately enhancing overall separation performance.

Complementing this attention-enhanced approach, we implement a computationally efficient alternative based on the **U-Net architecture family**. This implementation draws inspiration from the award-winning **TFC-TDF-UNet v3** architecture, which demonstrated exceptional performance in the Sound Demixing Challenge 2023 [2]. The TFC-TDF-UNet family has established its effectiveness in music source separation tasks through the innovative combination of **Time-Frequency Convolution (TFC)** blocks and **Temporal Dynamic Filtering (TDF)** modules, enabling comprehensive capture of both local spectro-temporal patterns and broader contextual information across time-frequency representations.

Our U-Net implementation focuses on adapting the fundamental principles underlying TFC-TDF-UNet v3 to create a more compact, **resource-efficient model architecture** suitable for local training environments without requiring access to high-performance computing clusters or extensive computational resources. Through this simplified yet strategically designed architecture, we aim to evaluate the practical viability of lightweight approaches and establish performance benchmarks against the more complex SE-enhanced BSRNN model.

While recent research has focused heavily on deep learning-based systems, it is important to consider traditional unsupervised methods as complementary baselines for comparison. Among these, **Non-negative Matrix Factorization (NMF)** has long served as a standard approach in music source separation tasks, relying on the decomposition of the magnitude spectrogram of the mixture into additive components associated with different sound sources. When combined with clustering algorithms such as *K-Means*, these components can be grouped and re-synthesized to estimate individual sources. Although these methods typically underperform compared to modern neural models in terms of perceptual quality and separation accuracy, they remain attractive due to their interpretability, low computational cost, and independence from labeled training data.

The **comparative evaluation** of these three complementary methodologies—the attention-enhanced BSRNN, the efficient U-Net variant, and the traditional NMF approach—provides valuable insights into the fundamental trade-offs between *architectural complexity*, *computational requirements*, and *separation performance* in contemporary music source separation systems. This triad investigation contributes to understanding both the potential for performance optimization through neural attention mechanisms and the practical considerations for deploying effective MSS systems under diverse resource constraints.

2 Dataset

2.1 Dataset Presentation

The **MUSDB18** dataset represents one of the most widely adopted benchmarks for music source separation research, originally developed for the annual Sound Separation Evaluation Campaign (**SiSEC**) and officially released in 2018. The primary objective of this dataset is to provide a standardized, real-world collection of multitrack recordings (stems) that enables fair and reproducible comparison of different source separation systems. Researchers utilize this dataset to train machine learning models—ranging from traditional signal processing techniques to modern deep neural networks—that are capable of separating *vocals*, *drums*, *bass*, and *other* instrumental components from mixed stereo audio tracks.

The dataset comprises a total of **150 songs** encoded in the **Native Instruments stems format** (.mp4), which is a specialized multitrack format consisting of 5 stereo streams, each encoded using **AAC compression at 256 kbps**. For applications requiring higher fidelity audio processing, the **MUSDB18HQ** variant provides uncompressed WAV files that support models designed to predict high-frequency content up to **22 kHz**. While this higher-quality version preserves superior signal information, it comes with significantly increased memory requirements and computational overhead.

Each track in the dataset is structured as a collection of **four separate source files** (*vocals*, *drums*, *bass*, *other*) alongside the complete **mixture file**, which represents the linear combination of all four individual sources. The songs exhibit considerable diversity in both duration (typically ranging from 2 to 6 minutes) and musical genre, encompassing rock, pop, electronic, hip-hop, and various other contemporary styles. This diversity ensures that trained models can generalize across different musical contexts and production techniques.

It should be noted that certain tracks within the dataset contain **stem imprecision artifacts** due to technical limitations in the original multitrack recording process. These corrupted files are documented and publicly available on the dataset's official webpage [1], allowing researchers to identify and appropriately handle these cases during preprocessing.

The **musdb library** provides a comprehensive interface for manipulating the dataset's fundamental data structure, the **Track object**. This object encapsulates all relevant information for each song, including:

- **Five audio files**: mixture, drums, bass, vocals, and other accompaniment
- **Metadata**: track title and artist name information
- **File system information**: absolute path of the mixture file to be processed
- **Technical specifications**: sample rate consistently maintained at **44.1 kHz**
- **Source dictionary**: comprehensive mapping of all sources used for the track
- **Audio tensor**: 3D numpy array containing all five stereo sources with shape $(5, n_{samples}, 2)$
  - The stems are consistently ordered as: ['mixture', 'drums', 'bass', 'other', 'vocals']
- **Target dictionary**: mapping of separation targets for the track, where sources and targets differ primarily in the treatment of accompaniment, defined as the sum of all sources excluding vocals

Additional technical details and implementation specifics can be found in the official documentation [1].

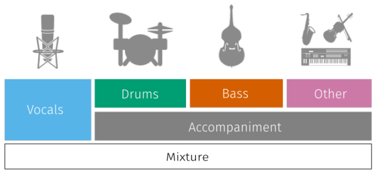

Figure 1: Audio source taxonomy in the MUSDB18 dataset

2.2 Data Augmentation Strategy

Given the relatively modest size of the MUSDB18 dataset for deep learning applications, we implemented a comprehensive data augmentation pipeline designed to generate **10-second audio segments** with applied signal modulation techniques to the four individual audio sources prior to mixture combination.

To contextualize our augmentation approach, it is essential to distinguish between **coherent mixing** and **incoherent mixing** methodologies. **Coherent mixing** applies random signal modulation effects uniformly to all stems derived from a single song, with all sources extracted at the same temporal offset t_0. This approach preserves the original musical relationships and harmonic structure between instruments. In contrast, **incoherent mixing** employs a more flexible strategy, selecting the four required stems from potentially different songs, each extracted at independent temporal offsets t_0^i, $\forall i \in \{1, 2, 3, 4\}$.

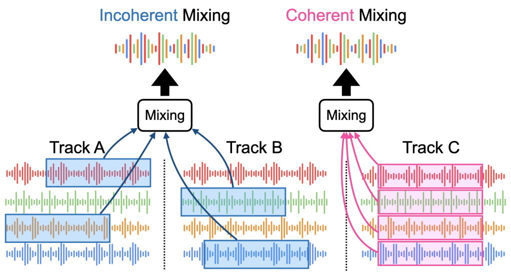

Figure 2: Comparison between incoherent and coherent mixing methodologies

Our implementation focuses on the **one-vs-all (OVA)** source separation paradigm, which trains specialized networks to isolate a single target source from the mixture while treating all other sources as background interference. This approach effectively teaches the network to "focus exclusively on the target source while ignoring all other acoustic content." While separating multiple sources using OVA requires training multiple independent networks—one per target source—this methodology offers the advantage of exposing each network to a more diverse and extensive training dataset compared to multisource approaches, albeit with increased computational overhead for training multiple models.

Considering the implementation efficiency and training speed advantages of OVA over multisource networks, we initially focused on **incoherent mixing** as our primary data augmentation technique. This method serves as a crucial augmentation strategy for MSS systems employing OVA architectures. While the resulting audio combinations may initially appear as *cacophonic mixtures*, this approach proves highly effective in generating valuable training data that significantly expands the available dataset.

The complete pipeline for data augmentation using incoherent mixing is illustrated below:

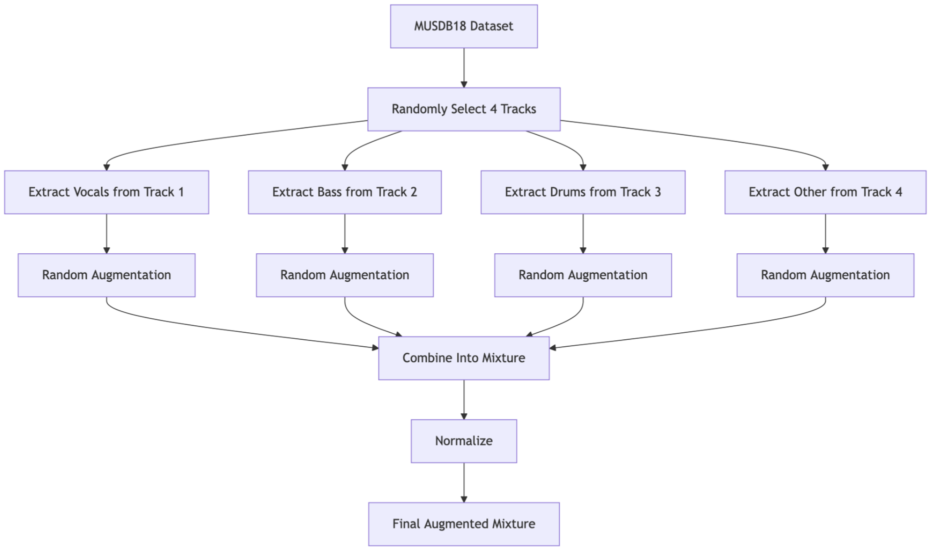

Figure 3: Comprehensive incoherent mixing pipeline for data augmentation

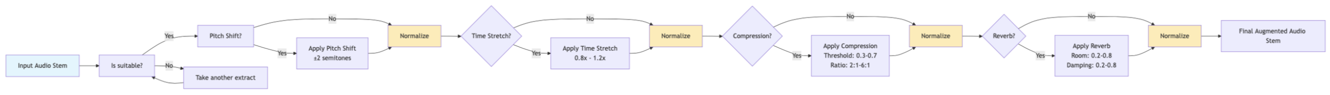
Input Audio Stem
Is suitable?
Yes
No
Take another extract
Pitch Shift?
No
Yes
Apply Pitch Shift ±2 semitones
Normalize
Time Stretch?
No
Yes
Apply Time Stretch 0.8x - 1.2x
Normalize
Compression?
No
Yes
Apply Compression Threshold: 0.3-0.7 Ratio: 2:1-6:1
Normalize
Reverb?
No
Yes
Apply Reverb Room: 0.2-0.8 Damping: 0.2-0.8
Normalize
Final Augmented Audio Stem

Figure 4: Random augmentation module implementing various signal modulation techniques

Additionally, we implemented a **coherent mixing pipeline** to support potential future expansion toward multi-source network architectures. The coherent mixing approach differs from incoherent mixing primarily in its stem extraction logic, which constrains all stems to originate from a single song and ensures extraction at the same temporal offset t_0, thereby preserving the original musical coherence and inter-instrument relationships.

Our data augmentation implementation builds upon the comprehensive methodologies presented in the open-source separation literature [4], adapting these techniques to our specific experimental requirements and computational constraints.

3 Method

In this work, we build upon the *BandSplitRNN* (BSRNN) model for music source separation (MSS) [3], proposing an enhanced version that incorporates Squeeze-and-Excitation (SE) layers to improve performance through channel-wise attention. The MSS task is framed as single-target source extraction, where one signal of interest is separated from a complex music mixture.

To provide a performance and efficiency baseline, we also implement a lightweight *U-Net architecture*, designed to operate under strict hardware constraints. This model serves as a low-resource benchmark to evaluate the relative gains introduced by more complex architectures like BSRNN and its SE-augmented variant. Furthermore, as a traditional and unsupervised point of comparison, we introduce a Non-negative Matrix Factorization (NMF)-based method, which operates entirely in the frequency domain and highlights the advancements offered by data-driven neural network approaches.

This section first introduces the original BSRNN model, then describes our proposed SE-enhanced variant, including a detailed discussion of architectural parameters, training details, and loss formulation. Next, we describe the design of the lightweight U-Net model, outlining its architecture and purpose as a comparative baseline in both performance and computational efficiency. Finally, we detail the implementation of the NMF-based source separation method, emphasizing its role as an unsupervised benchmark.

3.1 BandSplitRNN with Squeeze-and-Excitation (SE) Layers

3.1.1 Original BandSplitRNN Architecture

BandSplitRNN (BSRNN) is a frequency-domain model designed to leverage the structure of musical signals. It decomposes the input STFT spectrogram and explicitly models both temporal and spectral relationships through dual-path recurrent layers. The architecture (see Fig. 5) comprises three modules:

- **Band Split Module:** The complex-valued STFT input $X \in C^{F \times T}$ is split into K non-overlapping subbands based on predefined frequency bandwidths $\{G_i\}_{i=1}^{K}$, which vary per source type (e.g., 3-band split for bass, 6-band for vocals). Each subband is normalized and passed through a fully connected (FC) layer to obtain a shared feature space of dimension N.
- **Band and Sequence Modeling Module:** A series of 12 dual-path residual blocks, each containing two BLSTM layers, interleave temporal and spectral modeling. Specifically:
  - A BLSTM is first applied along the time axis (T), shared across all K bands.
  - A second BLSTM is then applied along the frequency axis (K), shared across all time steps.

  Residual connections stabilize training. Each BLSTM uses a hidden size of $2N = 256$ per direction.
- **Mask Estimation Module:** The output is processed by subband-specific MLPs with hidden size $4N = 512$ and a final GLU (Gated Linear Unit) output layer. A tanh activation is used internally. These MLPs estimate complex-valued masks $\{M_i\}_{i=1}^{K}$ for each subband, which are applied and concatenated to form the estimated spectrogram.

STFT Configuration. The input audio is transformed using an STFT with a window size of 2048 and hop size of 512.

Training Signal Filtering. A Source Activity Detector (SAD) is used to detect silent segments in the training data. Segments where the source has energy below a threshold are filtered out to avoid learning from silence.

On-the-fly Data Simulation. Instead of a static dataset, mixtures are dynamically generated during training using random segment sampling and data augmentation (gain, filtering, reverberation, etc.), enabling better generalization.

3.1.2 Extension: BSRNN with Squeeze-and-Excitation (SE) Layers

To enhance feature selectivity in BSRNN, we introduce *Squeeze-and-Excitation* (SE) layers after each RNN operation in the dual-path blocks. These layers apply channel-wise attention, helping the model emphasize relevant features based on context.

SE Layer Placement. We insert two SE layers per residual block:

- After the **temporal** BLSTM: reshape the tensor from $[B, K, T, N]$ to $[B \cdot K, N, T]$ and apply SE over time.
- After the **spectral** BLSTM: reshape from $[B, T, K, N]$ to $[B \cdot T, N, K]$ and apply SE over subbands.

SE Mechanism. Each SE layer performs:

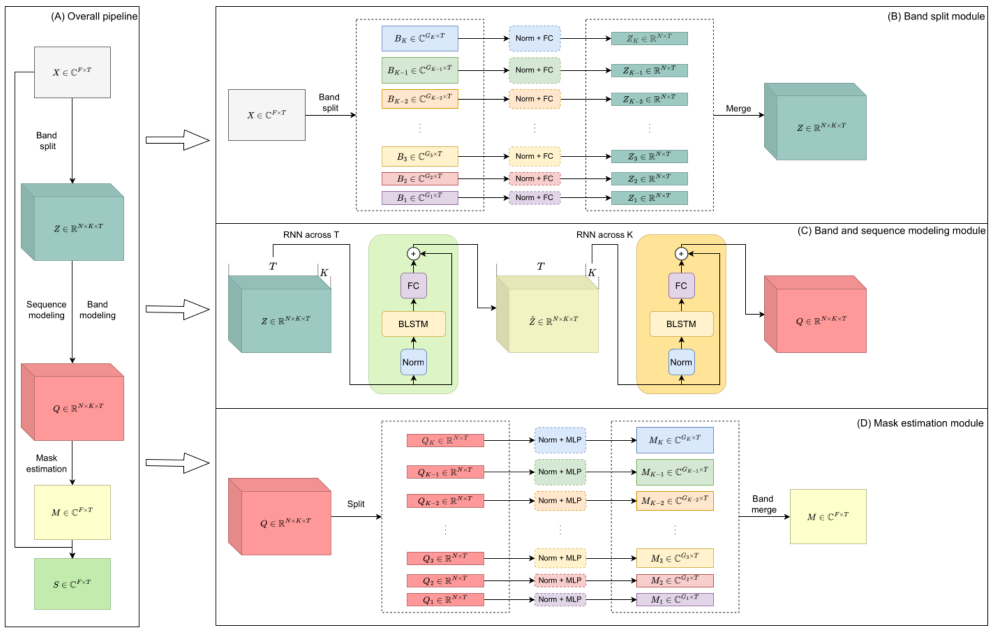

Figure 5: Original BandSplitRNN architecture. The model processes input spectrograms through a subband split, dual-path BLSTM blocks, and subband-specific mask estimation.

1. **Squeeze:** Apply global average pooling along the last dimension, yielding a summary vector $z \in R^{B \cdot D, C}$ (with $D = K$ or T, and $C = N$).
2. **Excitation:** Pass z through two fully connected layers:
$$s = \sigma(W_2 \cdot \text{ReLU}(W_1 \cdot z))$$
with $W_1 \in R^{C \times C/r}$ and $W_2 \in R^{C/r \times C}$. The result $s \in R^{B \cdot D, C}$ are the attention weights.
3. **Scale:** Broadcast s along the time/subband axis and multiply with the input features.

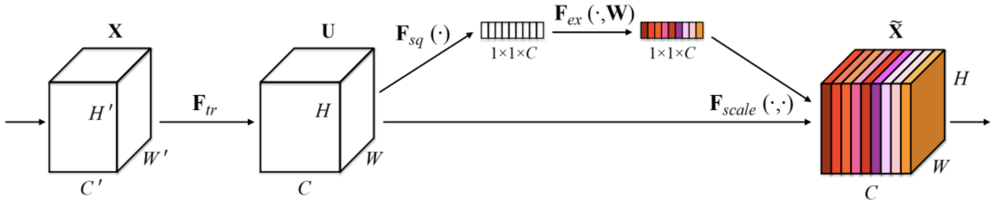

Figure 6: Illustration of the SE layer logic. A feature map is globally pooled, passed through an MLP with bottleneck, and used to generate per-channel attention weights.

Model Size Impact. Despite adding 24 SE layers (2 per block), the total number of parameters only increases from 32.4M to 32.5M — a negligible cost relative to the model size.

3.1.3 Loss Function

Let X be the input mixture and $\hat{S}, S$ be the estimated and ground truth STFTs of the target source. The model estimates a complex mask $\hat{M}$, and the reconstruction is $\hat{S} = \hat{M} \odot X$. The loss is a weighted sum of magnitude and phase-sensitive L1 losses:

$$\mathcal{L} = \alpha \cdot \||\hat{S}| - |S|\|_1 + (1-\alpha) \cdot \left(\|\Re(\hat{S}) - \Re(S)\|_1 + \|\Im(\hat{S}) - \Im(S)\|_1 \right)$$

where α balances magnitude and phase terms (e.g., $\alpha = 0.5$).

3.1.4 Training and Optimization

The model is trained using the Adam optimizer with:

- Initial learning rate: 10^{-3}, decayed by 0.98 every 2 epochs.
- Batch size: 2, with 10,000 batches per epoch.
- Gradient clipping: max norm of 5.
- Early stopping: triggered if no improvement is seen in 10 epochs.

3.2 U-Net Architecture (Low-Resource Benchmark)

To provide a point of comparison under limited computational resources, we implemented a lightweight *U-Net architecture* inspired by TFC-TDF-UNet v3 [2]. The model is designed to be trainable on a single NVIDIA GTX 1650 GPU and is therefore constrained in both depth and capacity. As such, it is not expected to yield competitive results in terms of separation quality, but instead serves as a benchmark for evaluating architectural improvements under realistic constraints.

3.2.1 Data Preprocessing

During training, 5-second audio clips are randomly sampled from the dataset, in addition to augmented data, and resampled to 16 kHz. The input mixture and source signals are converted to magnitude STFT spectrograms using a Hann window with $n_{\text{fft}} = 1024$ and hop size of $h = 512$. The phase of the mixture is saved and reused during audio reconstruction.

Extracting phase this early in the processing pipeline allows us to consolidate the simple magnitude spectrogram analysis performed by the U-Net Model. It is referred to as *noisy phase*, as it contains phase information for all sources at the same time. By doing so, we somewhat assume that phase distortion is negligible, which simplifies the problem but can introduce artifacts. Models like Channel-wise Subband Phase-aware ResUNet (CWS-PResUNet) or PhaseNet incorporate phase modeling to overcome the limitation of the basic U-Net. We won't deal with these models here as we chose to dig deeper with the BSRNN.

3.2.2 Architecture Overview

The model follows a standard encoder-bottleneck-decoder U-Net design with skip connections:

- **Encoder:** Three convolutional blocks, each consisting of Conv2D $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPooling. The number of filters increases progressively (8, 16, 32).
- **Bottleneck:** A convolutional block with 64 filters and dropout (rate = 0.3), serving as the model's core representation.
- **Decoder:** A mirrored architecture using Conv2DTranspose layers and skip connections from the encoder. Dropout (rate = 0.2) is applied after each skip connection.

3.2.3 Output Layer and Spectrogram Prediction

A final 1×1 convolution outputs a real-valued magnitude spectrogram, representing the estimated source. This predicted magnitude is combined with the phase of the input mixture to reconstruct the time-domain waveform using the inverse STFT.

3.2.4 Audio Reconstruction

To reconstruct the time-domain audio, the estimated magnitude spectrogram is combined with the mixture phase (saved during preprocessing) to form a complex STFT. An inverse STFT is then performed to produce the waveform output.

3.2.5 Regularization and Optimization

L2 weight regularization ($\lambda = 10^{-5}$) is applied to all convolutional layers. The model is trained using the Adam optimizer with an initial learning rate of 10^{-3} and batch size of 4. Training is performed from scratch.

3.2.6 Padding and Cropping

To maintain spatial alignment, spectrogram inputs are zero-padded to make their time and frequency dimensions divisible by 8. The final output is cropped back to match the original input shape.

3.3 Non-negative Matrix Factorization (NMF) Baseline

As a traditional and unsupervised baseline, we implemented a music source separation method based on **Non-negative Matrix Factorization (NMF)** followed by clustering and signal reconstruction. This approach operates entirely in the frequency domain and does not rely on supervised learning, offering a complementary perspective to deep learning-based systems.

STFT Preprocessing. The input mixture is first converted to a magnitude spectrogram using the Short-Time Fourier Transform (STFT) with a window size of 2048 and hop size of 512. The phase is stored for use in the final signal reconstruction step.

NMF Decomposition. The magnitude spectrogram $V \in R^{F \times T}$ is factorized into two non-negative matrices: $W \in R^{F \times K}$ and $H \in R^{K \times T}$, where $K = 8$ is the number of components. This decomposition is achieved using the *multiplicative update rule* with a Kullback-Leibler divergence loss.

Component Clustering. The K components (rows of H) are clustered into 4 groups using **K-Means**, aiming to associate each cluster with one of the four MUSDB18 source types: vocals, drums, bass, and other. Each cluster is assigned to a source using a greedy matching strategy based on comparing the estimated signal from each cluster to the ground-truth sources.

Signal Reconstruction. The magnitude spectrogram of each estimated source is reconstructed by selecting and summing the relevant components based on cluster assignments. The original phase of the mixture is reused to compute the complex spectrogram, and the inverse STFT is applied to obtain the time-domain waveform.

This NMF-based method provides a fast, interpretable, and fully unsupervised alternative to neural architectures. Although its performance is limited—particularly in complex polyphonic contexts—it serves as a valuable baseline for benchmarking the relative gains achieved by deep learning-based approaches such as BSRNN and U-Net.

4 Evaluation Metrics & Results

To evaluate the effectiveness of our source separation models, we rely on metrics that quantify the fidelity of the separated signals with respect to the ground truth sources.

4.1 BandSplitRNN and SE-enhanced BandSplitRNN

To evaluate our BandSplitRNN-based models, we use two variations of the Signal-to-Distortion Ratio (SDR) metric, both widely used in music source separation:

- **Conservative SDR (cSDR):** Penalizes both interference from other sources and artifacts introduced by the separation process. It provides a strict measure of how well the model isolates the target source while maintaining signal fidelity.
- **Unconservative SDR (uSDR):** Focuses mainly on suppressing interference from other sources, making it a more lenient measure that highlights the model's ability to extract the desired component without necessarily penalizing minor artifacts.

We evaluate performance across the four standard target sources defined in MUSDB18—*vocals*, *drums*, *bass*, and *other*. An overall average score is also reported to summarize general performance across all sources.

The results are shown in Table 1. The inclusion of Squeeze-and-Excitation (SE) layers enables the model to focus more selectively on relevant features, leading to consistent improvements in both cSDR and uSDR across most sources.

Table 1: Performance comparison of BandSplitRNN and SE-enhanced BandSplitRNN on MUSDB18 using cSDR and uSDR (in dB). We report results from the best-performing checkpoint on training, selected based on validation uSDR.

Source	**BandSplitRNN**		**BandSplitRNN + SE**	
	cSDR	**uSDR**	**cSDR**	**uSDR**
Vocals	6.298	6.540	6.827	6.985
Drums	7.285	7.182	7.666	7.579
Bass	4.956	4.932	5.578	5.510
Other	4.102	4.055	4.336	4.154
Average	5.660	5.677	6.102	6.057

To complement the quantitative evaluation based on uSDR and cSDR, we also analyze the training and validation dynamics of the BandSplitRNN models. In particular, we compare the original BandSplitRNN and the SE-enhanced variant in terms of both source reconstruction loss and signal quality (uSDR).

The following plots show the evolution of training and validation metrics during learning. For each metric, the top plot corresponds to the original BandSplitRNN, while the bottom plot shows the SE-enhanced version.

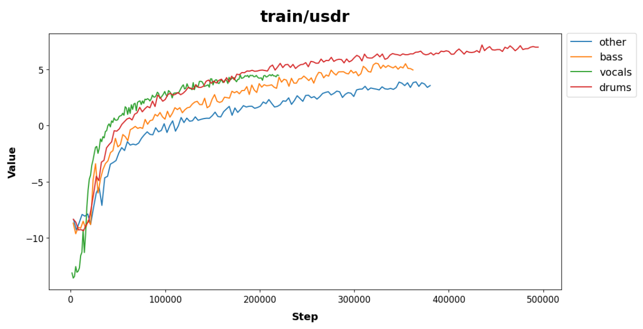

Figure 7: *
Original BandSplitRNN

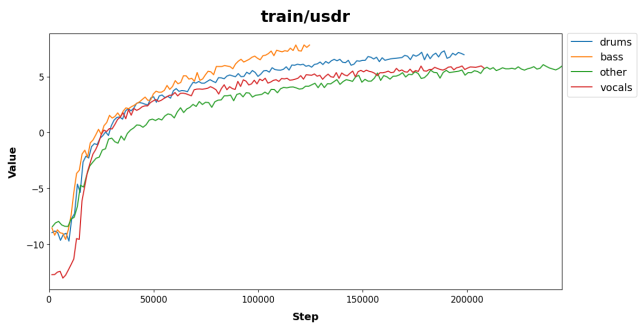

Figure 8: *
SE-enhanced BandSplitRNN

Figure 9: Comparison of training uSDR over time. SE layers lead to faster and more stable quality improvements.

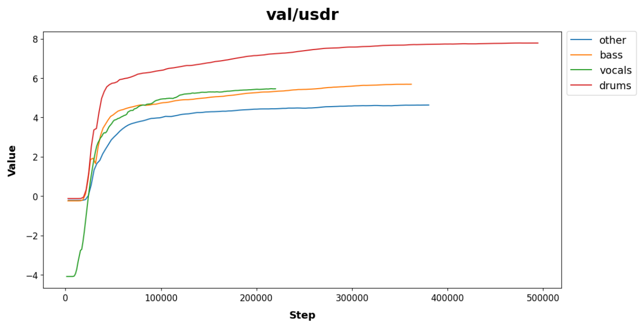

Figure 10: *
Original BandSplitRNN

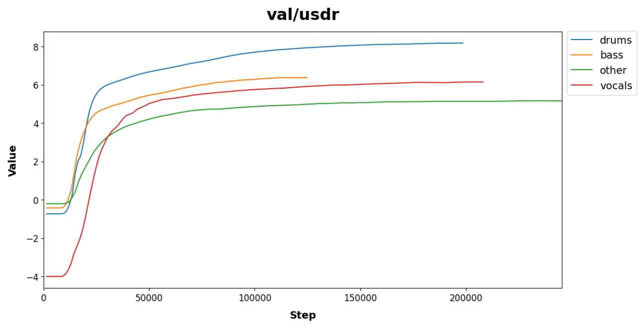

Figure 11: *
SE-enhanced BandSplitRNN

Figure 12: Validation uSDR comparison. The SE-enhanced model reaches better overall separation quality.

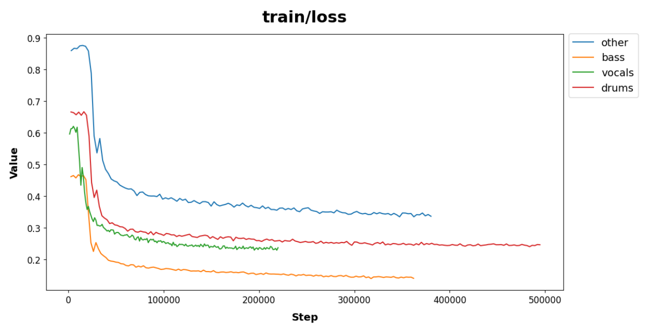

Figure 13: *
Original BandSplitRNN

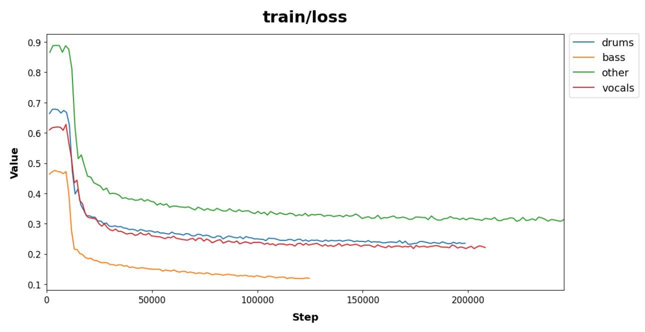

Figure 14: *
SE-enhanced BandSplitRNN

Figure 15: Training loss comparison. The SE-enhanced model converges slightly faster and smoother.

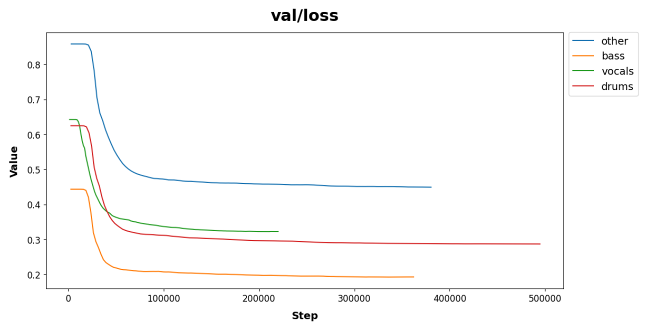

Figure 16: *
Original BandSplitRNN

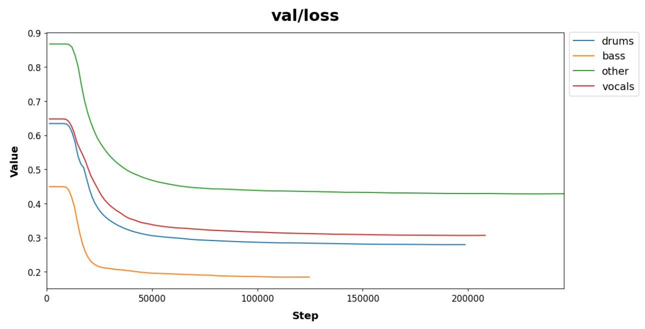

Figure 17: *
SE-enhanced BandSplitRNN

Figure 18: Validation loss curves. The SE-enhanced version shows better generalization with lower final loss.

4.2 U-Net Model (Low-Resource Benchmark)

To provide a point of comparison under limited computational resources, we implemented a lightweight U-Net architecture inspired by TFC-TDF-UNet v3 [2]. The model is designed to be trainable on a single NVIDIA GTX 1650 GPU and is therefore constrained in both depth and capacity. As such, it is not expected to yield competitive results in terms of separation quality, but instead serves as a benchmark for evaluating architectural improvements under realistic constraints.

4.2.1 Data Preprocessing

During training, 5-second audio clips are randomly sampled from the dataset, in addition to augmented data, and resampled to 16 kHz. The input mixture and source signals are converted to magnitude STFT spectrograms using a Hann window with $n_{\text{fft}} = 1024$ and hop size of $h = 512$. The phase of the mixture is saved and reused during audio reconstruction.

Extracting phase this early in the processing pipeline allows us to consolidate the simple magnitude spectrogram analysis performed by the U-Net Model. It is referred to as noisy phase, as it contains phase information for all sources at the same time. By doing so, we somewhat assume that phase distortion is negligible, which simplifies the problem but can introduce artifacts. Models like Channel-wise Subband Phase-aware ResUNet (CWS-PResUNet) or PhaseNet incorporate phase modeling to overcome the limitation of the basic U-Net. We won't deal with these models here as we chose to dig deeper with the BSRNN.

4.2.2 Architecture Overview

The model follows a standard encoder-bottleneck-decoder U-Net design with skip connections:

- **Encoder:** Three convolutional blocks, each consisting of Conv2D → BatchNorm → ReLU → MaxPooling. The number of filters increases progressively (8, 16, 32).
- **Bottleneck:** A convolutional block with 64 filters and dropout (rate = 0.3), serving as the model's core representation.
- **Decoder:** A mirrored architecture using Conv2DTranspose layers and skip connections from the encoder. Dropout (rate = 0.2) is applied after each skip connection.

4.2.3 Output Layer and Spectrogram Prediction

A final 1×1 convolution outputs a real-valued magnitude spectrogram, representing the estimated source. This predicted magnitude is combined with the phase of the input mixture to reconstruct the time-domain waveform using the inverse STFT.

4.2.4 Audio Reconstruction

To reconstruct the time-domain audio, the estimated magnitude spectrogram is combined with the mixture phase (saved during preprocessing) to form a complex STFT. An inverse STFT is then performed to produce the waveform output.

4.2.5 Regularization and Optimization

L2 weight regularization ($\lambda = 10^{-5}$) is applied to all convolutional layers. The model is trained using the Adam optimizer with an initial learning rate of 10^{-3} and a batch size of 4. Training is performed from scratch.

4.2.6 Padding and Cropping

To maintain spatial alignment, spectrogram inputs are zero-padded to make their time and frequency dimensions divisible by 8. The final output is cropped back to match the original input shape.

We evaluate the U-Net's performance on the same MUSDB18 dataset using mean and median SDR for each target source. While its performance is significantly lower than the BandSplitRNN variants, it provides a useful point of comparison for assessing the trade-off between model complexity and separation quality.

Table 2: Mean and median SDR (in dB) for the lightweight U-Net across MUSDB18 source categories.

Source	**Mean SDR**	**Median SDR**
Vocals	-1.7040	1.9151
Bass	-1.6274	0.3885
Drums	0.9718	2.3487
Other	-1.5745	-0.7003

To further illustrate the behavior of the U-Net model, we visualize its predicted spectrogram output for a vocal track in Figure 19. The subplot compares the predicted spectrogram with the ground truth source, highlighting the qualitative differences in separation performance.

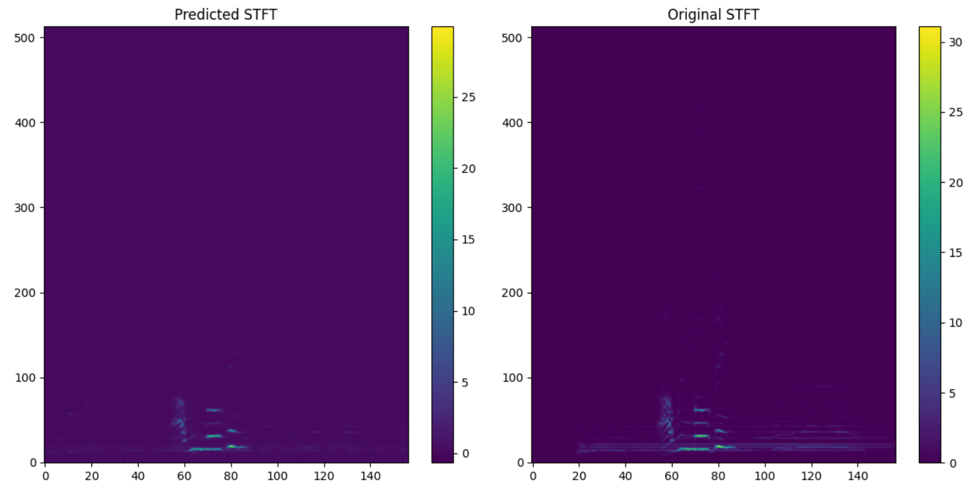

Figure 19: Predicted vs. ground truth spectrogram for the vocals source using the lightweight U-Net. While the overall structure is preserved, the prediction suffers from notable interference and loss of detail.

4.3 Traditional Method: NMF + K-Means Clustering

To provide a traditional, unsupervised baseline, we implemented a classic **Non-negative Matrix Factorization (NMF)** approach followed by **K-Means clustering** to group spectral components into source categories. This method does not require model training and relies purely on decomposing the magnitude spectrogram of the mixture signal.

The evaluation uses the same cSDR and uSDR metrics as the neural models, allowing a direct performance comparison. Results are averaged over the MUSDB18 test set.

Table 3: Mean cSDR and uSDR (in dB) for each source using the traditional NMF + K-Means baseline.

Source	**cSDR**	**uSDR**
Vocals	-0.04	-0.03
Drums	1.60	1.61
Bass	0.66	0.66
Other	-0.25	-0.25
Average	0.49	0.50

While the performance is significantly lower than that of the deep learning models, the NMF method still succeeds in extracting coarse approximations of the dominant sources, particularly drums and vocals. Its advantages lie in interpretability, fast inference, and lack of need for labeled training data.

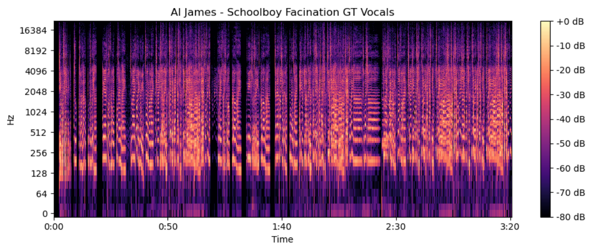

Figure 20: *
(GT) Al James - Schoolboy Facination vocals
Predicted Spectrogram (NMF)

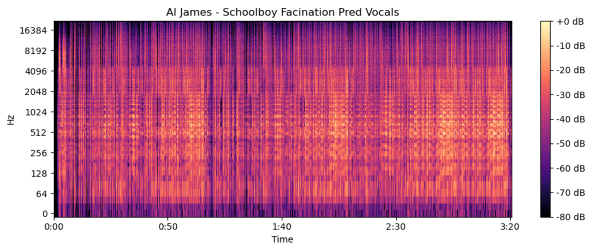

Figure 21: *
(PRED) Al James - Schoolboy Facination vocals
Predicted Spectrogram (NMF)

Figure 22: Comparison between ground truth and predicted vocal spectrograms using the NMF baseline. The estimate captures some vocal energy but lacks harmonic detail and shows interference from other instruments.

5 Conclusion

In this project, we examined three different approaches to Music Source Separation (MSS): the recurrent BandSplitRNN model, a lightweight U-Net convolutional model, and a traditional baseline using Non-negative Matrix Factorization (NMF) followed by K-Means clustering. These approaches span a wide spectrum in terms of complexity, interpretability, and computational requirements.

The BandSplitRNN model, particularly when enhanced with Squeeze-and-Excitation (SE) layers, delivered the highest performance. The SE modules enabled channel-wise attention mechanisms that improved the model's capacity to focus on the most informative spectral and temporal components. This resulted in consistent improvements across conservative and unconservative SDR metrics, faster convergence during training, and better generalization on the test set—all with a minimal increase in model size (from 32.4M to 32.5M parameters). These results underscore the effectiveness of combining frequency band decomposition with recurrent attention mechanisms in deep learning-based MSS systems.

The U-Net model, while significantly more constrained in depth and complexity, served as a useful baseline under hardware limitations. Its performance was predictably lower, but it demonstrated that even simplified architectures can recover rough estimates of target sources. However, it lacked robustness in the presence of phase distortion and showed greater difficulty in handling overlapping spectral components.

To provide a baseline grounded in traditional signal processing, we also evaluated a classic NMF + K-Means pipeline. This unsupervised method, although significantly underperforming compared to the learned models (e.g., average uSDR of 0.50 dB vs. 6.06 dB for the SE-BSRNN), still managed to extract basic structural elements, particularly for rhythmic or percussive sources like drums. Its advantages lie in its simplicity, interpretability, and independence from labeled training data, making it suitable for rapid prototyping or settings with limited annotations.

In summary, the comparative evaluation revealed the following:

- **SE-enhanced BSRNN** achieves the best separation quality, particularly suited for high-fidelity applications.
- **Lightweight U-Net** offers a viable low-resource alternative but suffers from reduced accuracy and detail retention.
- **NMF + K-Means** provides a fast, interpretable baseline with limited precision but no training cost.

Future work could explore dynamic sub-band partitioning, hybrid models combining convolutional and recurrent elements, and lightweight models with explicit phase modeling. Moreover, integrating unsupervised techniques like NMF with deep learning pipelines (e.g., for pre-initialization or regularization) may also be a promising avenue for improving both performance and interpretability.

References

[1] SiSep Datasets. Musdb18 compressed stems. `https://sigsep.github.io/datasets/musdb.html#musdb18-compressed-stems`, 2022. Last updated: September 29, 2022. Accessed: 2025-06-06.

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

[3] Qiuqiang Kong, Yin Cao, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Band-split rnn for music source separation. In *ICASSP 2022 - IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 926–930. IEEE, 2022.

[4] Ethan Manilow, Prem Seetharaman, and Justin Salamon. *Open Source Tools & Data for Music Source Separation.* https://source-separation.github.io/tutorial, Oct. 2020.