Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2023) 31: 1276 – 1293
© TÜBİTAK
doi:10.55730/1300-0632.4047

A comparative study of blind source separation methods

Burak BAYSAL[1,*], **Mehmet Önder EFE**[2]
[1]Graduate School of Science and Engineering, Faculty of Engineering, Hacettepe University, Ankara, Turkiye
[2]Department of Computer Engineering, Faculty of Engineering, Hacettepe University, Ankara, Turkiye

Received: 14.04.2023 • **Accepted/Published Online:** 17.10.2023 • **Final Version:** 30.11.2023

Abstract: Blind source separation is a popular research topic used for decomposing mixed signals, particularly in the field of music. In addition to exploring machine learning-based approaches, this study aims to examine the performance of classical algorithms in separating audio signal sources. The evaluation of different genres is a significant aspect of this study as the performance of the methods may vary across various musical genres and different audio components. This consideration provides a novel perspective and contributes to a comprehensive analysis of the algorithms. Using the MusDB-HQ dataset, we conducted experimental studies comparing classical algorithms, including FastICA, NMF, and DUET, with implemented architectures such as Hybrid Demucs, Spleeter, Open Unmix, and Wave-U-Net. The audio components were assessed based on several factors, including time, genre, and signal-to-distortion ratio (SDR) scores, after artificially mixing the signals. The results demonstrated the superior performance of machine learning models over classical methods. Specifically, Wave-U-Net achieved the highest SDR scores for drums, other, and mixture components (2.041, –2.087, and 0.941, respectively), while Spleeter showed the highest SDR scores for vocals and bass components (3.145 and 0.066, respectively). Additionally, this study highlights the influence of different genres on algorithm performance, providing valuable insights for music production and related applications. Overall, this study contributes to the existing knowledge in the field of audio source separation by comparing classical algorithms and machine learning models, considering genre variations, and evaluating performance across different audio components. The findings have implications for the development of improved algorithms and their application in various musical genres.

Key words: Blind source separation, music information retrieval

1. Introduction

Blind source separation (BSS) is a significant field of study in digital signal processing, with many applications in various fields such as speech recognition, image processing, and music information retrieval (MIR). The primary goal of BSS is to separate mixed signals into their individual source components without any prior knowledge of the sources, such as their frequency characteristics and location or the way the sources are mixed, This is a challenging problem due to the limited available information formed by the observed mixture of signals from different input channels. BSS is a useful approach to solving this problem [1]. The cocktail party problem is depicted in Figure 1 as one of the well-known typical BSS problems.

Several algorithms have been proposed to address BSS problems, many of which are based on unsupervised learning and prior knowledge of the signal structure. To extract reliable and meaningful components from the

[*]Correspondence: bbaysal@outlook.com

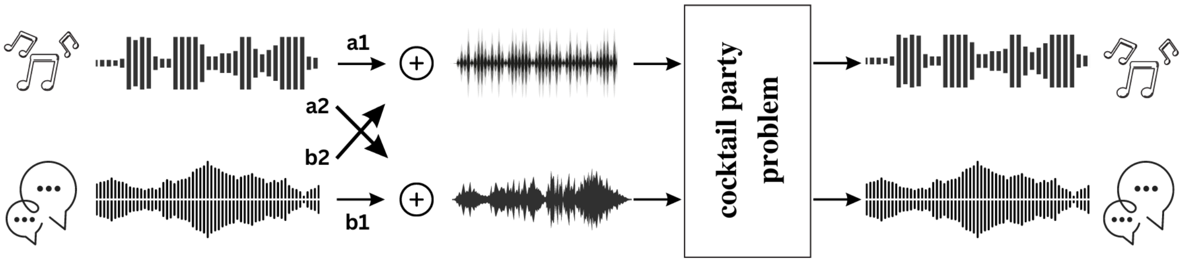

Figure 1. Example of the cocktail party problem. The two sounds $s1$ and $s2$ produced by music and voice are superposed and recorded by the microphones. Both microphones receive the unique weighted sum of the two sources. The weight of each microphone ($a1$, $b1$ and $a2$, $b2$) indicates the proximity of the microphone to the sound source [2].

data, preprocessing and postprocessing techniques are essential [3]. Classical algorithms such as nonnegative matrix factorization (NMF) [4] and fast and robust fixed-point algorithms for independent component analysis (FastICA)[5] have been widely used in BSS. However, recent advances in machine learning have led to the development of more sophisticated and versatile models that show promising results in various applications, including MIR.

In this paper, we provide an extensive treatment of well-known classical and contemporary implemented architectures for BSS in the MIR field. We compare the performance of seven different algorithms, including FastICA, NMF, and the degenerate unmixing estimation technique (DUET) as classical approaches and Spleeter [6], OpenUnmix [7], Hybrid Demucs [8], and Wave-U-Net [9] as implemented architectures. We evaluate their performances in terms of time efficiency, genre specificity, and signal-to-distortion ratio (SDR) scores.

The remainder of this paper is organized as follows: Section 2 presents a review of comparative studies in the field of BSS, Section 3 provides detailed explanations of the approaches used in this study, Section 4 describes the methodology employed in our experiments, Section 5 presents and discusses the experimental results, and, finally, Section 6 concludes the paper with a summary of the findings and potential avenues for future research. Additionally, we include Appendix A, which contains tables of the experimental results by genre.

Overall, this paper contributes to the existing literature by providing a comprehensive comparison of classical and contemporary BSS methods in the MIR field. By providing a detailed analysis of the strengths and weaknesses of each approach, we aim to help researchers and practitioners choose the most appropriate method for their specific applications.

2. Related work

Comparative studies of BSS methods have been carried out in different fields and have guided researchers from different disciplines. In particular, the blind separation of signals from health data (e.g., electroencephalograms) has been a popular topic. In [10], the authors compared the use of BSS methods on functional magnetic resonance imaging (fMRI) signals. They used Infomax (maximum likelihood) [11], FastICA, joint approximate diagonalization of eigenmatrices (JADE) [12], and eigenvalue decomposition (EVD) algorithms for separation.

In [13], matrix factorization-based BSS methods were compared for audio signals. The authors conducted an experimental study utilizing FastICA, principal component analysis, and NMF algorithms. Their results showed that FastICA was superior in separating mixed signals with the negentropy technique for finding the maximum non-Gaussianity. In [14], on the other hand, the authors presented a BSS study using the kernel additive model (KAM) [15] approach apart from conventional methods. In addition, the authors developed a customized kernel model called KAM-CUST, which showed promising results.

However, despite these previous studies, it is still challenging to achieve reliable and accurate source separation in music signals. Classical approaches such as FastICA and NMF are insufficient in the MIR literature, and recent studies have shown that deep learning-based approaches have the potential to outperform classical approaches. This study aims to contribute to the field by providing a comparative analysis of well-known classical and contemporary machine learning models for music source separation.

3. Blind source separation methods

3.1. Classical methods

3.1.1. FastICA

Independent component analysis (ICA) is one of the most widely known algorithms for solving the cocktail party problem. ICA is a dimensionality reduction algorithm and also a filtering operation that allows a particular source to be kept or removed [2]. Two major types of approaches are used to solve ICA problems: statistical approaches and neural network approaches. Statistical ICA approaches include many popular algorithms, such as FastICA and Infomax [11]. The main goal of these algorithms is to extract independent components by (i) maximizing the non-Gaussianity, (ii) minimizing the mutual information, or (iii) using the maximum likelihood (ML) estimation method [5].

The ICA method performs a matrix multiplication to obtain the mixed output by $x = As$, where x is an observed signal vector, s is a source signal vector, and A is a constant invertible mixing matrix to be estimated. The main goal of ICA algorithms is to estimate the invertible mixing matrix A and extract the predicted source vector denoted by $\hat{s}$ with the unmixing matrix W, which is an approximation of A^{-1} such that $\hat{s} = Wx$.

Before applying the ICA algorithm, performing some preprocessing on the observed data is useful. Among these steps, the most frequently used are centering and whitening. The centering step aims to center signals by subtracting the mean values ($E\{x\}$) from signal data. Given an observed vector signal denoted by x, centered observed vector x_i can be obtained by $x_i = x - E\{x\}$ [13].

The whitening step aims to transform signal data into uncorrelated components and rescale them with unit variance [16]. It is known that for a whitened x vector, the associated covariance matrix equals the identity matrix that can be obtained by $E\{\hat{x}\hat{x}^T\} = I$.

One way of performing whitening transformation is to exploit the eigenvalue decomposition (EVD) of the covariance matrix $E\{\hat{x}\hat{x}^T\} = VDV^T$, where V is the eigenvectors of the covariance matrix and D is the diagonal matrix of eigenvalues. The transformation $\hat{x} = VD^{-1/2}V^Tx$ can whiten the observed vector [5], where D can be obtained by a component-wise operation as $D^{-1/2} = diag(d_1^{-1/2}, d_2^{-1/2}, ..., d_n^{-1/2})$. The equation transforms into a new form as $\hat{x} = VD^{-1/2}V^TAs = \hat{A}s$ [5]. Hence, $E\{\hat{x}\hat{x}^T\} = \hat{A}E\{ss^T\}\hat{A}^T = \hat{A}\hat{A}^T = I$.

Let w denote a row of the unmixing matrix W; the FastICA algorithm uses a fixed-point scheme for finding the maximum of the non-Gaussianity of w^Tx, which can be done by maximizing the negentropy [5]. FastICA is much faster than gradient-based algorithms, which display linear convergence, whereas FastICA displays cubic or quadratic convergence speeds. In addition, FastICA does not have any adjustable parameters such as learning rate [5]. Therefore, the following contrast function $g(u)$ and its first derivative $g'(u)$ were used as a contrast function in this paper:

$$g(u) = \tanh(u) \text{ and } g'(u) = 1 - \tanh^2(u)$$

With this approach, it is assumed that data have been preprocessed, as mentioned above. The FastICA algorithm has the steps given in Algorithm 1. Let k denote the total step number of the algorithm, which is used to stop the algorithm if the algorithm does not converge below the given threshold value. The time complexity of the FastICA algorithm can be approximated as $O(ck(pnN+np))$, where c represents the number of components, k represents the maximum number of iterations, and N represents the number of samples.

Algorithm 1 FastICA algorithm.

```
for 1 to number of components $c$ do
    $w_p \equiv random\ initialization$
    for 0 to $k$ do
        $w_p \equiv \frac{1}{n}(Xg(W^TX) - g'(W^TX)W)$
        $w_p \equiv w_p - \sum_{j=1}^{p-1}(w_p^T w_j)w_j$
        $w_p \equiv w_p / \|w_p\|_2$
        if $w_p <\ threshold$ then
            break;
        end if
    end for
    $W \equiv [w_1, w_2, ..., w_c]$
end for
```

3.1.2. NMF

Matrix factorization is a helpful way to extract meaningful properties of the matrix, which results from the products of matrices with specific properties. Nonnegative matrix factorization is the process of decomposing a matrix V without negative entries into matrix W and H. The nonnegativity constraint helps in decomposing meaningful parts of the data [4].

Let V be a nonnegative matrix. NMF aims to find nonnegative matrix factors W and H as follows:

$$V \approx W \cdot H, \quad V \in \mathbb{R}^{N\times M}, W \in \mathbb{R}^{N\times R}, H \in \mathbb{R}^{R\times M} \tag{1}$$

Here, V is composed of N observed data vectors with M-dimensional data samples. This matrix can be approximately factorized into the ($N \times R$ dimensional) W matrix called the basis vectors or template vectors and the ($R \times M$ dimensional) H matrix called the activations. The dimension parameter R here represents the *rank* of the factorization and is chosen to be smaller than N and M. Considering the model column by column, $v \approx Wh$, each data vector in the V matrix consists of the linear combination of the columns of W containing the basis vectors using the activation vector h. W represents a large number of data vectors of low dimensions. In this respect, it is crucial to correctly estimate W to find the observed data's latent structure and make a good source separation approximation [4].

For measuring the approximation quality, some functions such as Euclidean distance and Kullback–Leibler divergence are used in different NMF implementations, and NMF aims to minimize the $||V - WH||_2^2$ joint distance function. One of the methods applied to overcome the joint optimization problem is to optimize the W and H factors sequentially. In each iteration, first, the W factor is fixed, and the H factor is optimized regarding W; then the optimized H factor is fixed, and the W factor is optimized regarding H. In these optimization steps, multiplicative update rules are used to guarantee nonnegativity. Thus, in the next step, the factors are updated as follows [17]:

$$H_{rn}^{(l+1)} = H_{rn}^{(l)} \frac{(W^T V)_{rn}}{(W^T W H^{(l)})_{rn}} \text{ and } W_{kr}^{(l+1)} = W_{kr}^{(l)} \frac{(V H^T)_{kr}}{(W^{(l)} H H^T)_{kr}}$$

where l is the iteration number and subscripts indicate row and column numbers. The NMF algorithm has the steps given in Algorithm 2. Similar to Algorithm 1, the hyperparameter k indicates the total step number of the algorithm. In addition, the NMF algorithm has a time complexity of approximately $O(km)$, where k is the maximum number of iterations, M is the number of features, N is the number of samples, and R is the rank of the factorization.

Algorithm 2 NMF algorithm.

```
$W^{(0)}, H^{(0)} \equiv random\ initialization$
$l = 0$
**for** l to $k$ **do**
    $H^{(l+1)} \leftarrow H^{(l)} \frac{(W^{(l)})^T V}{(W^{(l)})^T W^{(l)} H^{(l)}}$
    $W^{(l+1)} \leftarrow W^{(l)} \frac{V (H^{(l+1)})^T}{W^{(l)} H^{(l)} (H^{(l+1)})^T}$
    **if** $\left\| H^{(l)} - H^{(l-1)} \right\|_2 \leq \varepsilon$ and $\left\| W^{(l)} - W^{(l-1)} \right\|_2 \leq \varepsilon$ **then**
        $H \leftarrow H^{(l)}$
        $W \leftarrow W^{(l)}$
        break;
    **end if**
    $l = l + 1$
**end for**
```

3.1.3. DUET

"Degenerate" refers to more sources than mixtures in the environment, and it is a challenging phenomenon in blind source separation because the mixing matrix is not invertible. Thus, methods that separate sources by inverting the mixing matrix can not solve underdetermined problems. DUET, the degenerate unmixing estimation technique, uses time-frequency representations of the mixture to obtain attenuation and delay values for estimating independent components [18].

DUET achieves good results in separating mixtures with delays, but it has five assumptions for the mixture: anechoic mixing, W-disjoint orthogonality, local stationarity, close microphones, and different spatial signatures [19]. The anechoic mixture can be expressed as follows:

$$x_1(t) = \sum_{j=1}^{N} s_j(t) \text{ and } x_2(t) = \sum_{j=1}^{N} a_j s_j(t - \delta_j)$$

Here, N is the number of sources, a_j is the relative attenuation between sensors and sources, and δ_j is the arrival delay between the sensors. Δ represents the maximum possible delay between sensors; hence, $\delta_j \leq \Delta, \forall j$. Another assumption is W-disjoint orthogonality. For a given $W(t)$ windowing function, two functions, $s_j(t)$ and $s_k(t)$, are W-disjoint orthogonal if the supports of the windowed Fourier transform of $s_j(t)$ and $s_k(t)$ are disjoint. Considering that $\hat{s}_j(\tau,\omega)$ is the windowed Fourier transform of $s_j(t)$, W-disjoint orthogonality can be stated as $\hat{s}_j(\tau,\omega)\hat{s}_k(\tau,\omega) = 0, \forall\tau,\omega,\forall j \neq k$ [18].

In the DUET algorithm, first, local symmetric attenuation and local delay maximum likelihood estimators are extracted from the time-frequency representation of the data. Then a 2D smooth histogram is created based on these estimators. The histogram is crucial for localization and discrimination, and the number of peaks indicates the number of sources. The locations of peaks in the histogram represent the source's actual attenuation and delay values, namely mixing parameters. Each source's mixing parameter is assigned to the closest time-frequency bin to separate sources. The maximum likelihood function defined in [20] is used to define the closeness. Time-frequency masks related to the mixing parameters are constructed and applied to mixtures to obtain each component's time-frequency representation. The steps for the DUET algorithm are as follows [18]: (i) Construct time-frequency representations from mixtures $\hat{x}_1(\tau,\omega) \leftarrow x_1(t)$, $\hat{x}_2(\tau,\omega) \leftarrow x_2(t)$; (ii) $\alpha \leftarrow \left|\frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)}\right| - \left|\frac{\hat{x}_1(\tau,\omega)}{\hat{x}_2(\tau,\omega)}\right|$, $\delta \leftarrow \frac{-1}{\omega}\angle\left|\frac{\hat{x}_2(\tau,\omega)}{\hat{x}_1(\tau,\omega)}\right|$; (iii) build two-dimensional smoothed weighted histogram $H(\alpha,\delta)$; (iv) find peaks and peak centers and determine the actual mixing parameters; (v) for each peak in the histogram, build a time-frequency binary mask; (vi) apply masks to mixtures and get each source's time-frequency representation; (vii) convert the time-frequency representations of the sources to the time domain.

3.2. Implemented architectures

3.2.1. Open-Unmix

In addition to obtaining state-of-the-art results in music source separation studies, it is also aimed that Open-Unmix will be a basis for these studies. With this structure, which is similar to the modified National Institute of Standards and Technology (MNIST) database, the dataset can be downloaded instantly and training can be started immediately [7].

It is necessary to train the model separately for each source to separate sources with the Open-Unmix model. Because there is a separate model output for each source, this creates flexibility on the one hand and a disadvantage on the other. Figure 2 depicts the Open-Unmix model, and the explanation of the model is as follows: Open-Unmix starts with a fully connected layer, batch normalization, and hyperbolic tangent activation functions, followed by three bidirectional long short-term memory (Bi-LSTM) layers, and it continues with two fully connected layers with batch normalizations and rectified linear unit (ReLu) activation functions. At the end of the model, the output of ReLU activation is multiplied by spectrogram values and the predicted source is obtained. Some important notes about the model are as follows [21]:

- Fully connected layers before Bi-LSTM layers are helpful for dimensionality reduction to provide more distilled input data to the Bi-LSTM layers while the time dimension stays the same.
- Skip connections aid convergence and allow the network to learn without the help of layers. At the same time, multiplying the input spectrogram by the output of the last ReLU activation shows that this output is the weight of the predicted source on the original spectrogram, not the spectrogram data of the predicted source.

3.2.2. Spleeter

Spleeter is an implementation of the U-Net [22] architecture. U-Net was established for biomedical image segmentation [23] and later used in audio source separation studies. After successful results, it has become one

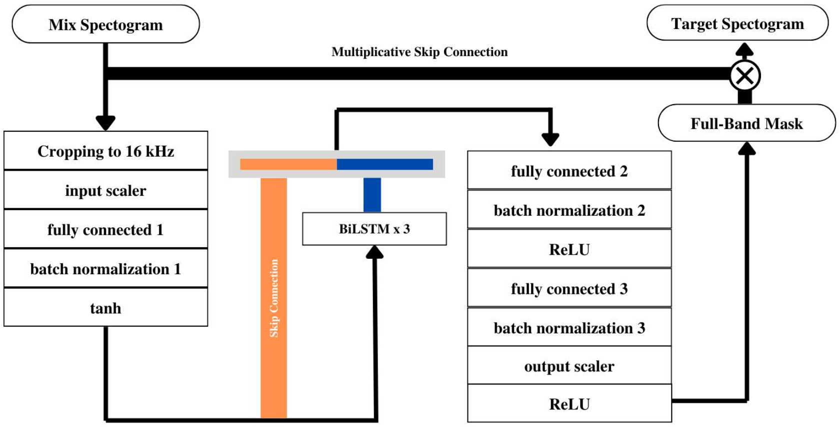

Figure 2. Open-Unmix model.

of the popular architectures in this field.

The input of the network is the spectrogram of audio. The network first processes this spectrogram with two-dimensional convolutions, and in each convolution, a smaller encoded output of the previous input emerges, and finally, latent data are obtained. The latent data are then reconstructed by decoding with two-dimensional deconvolution layers. Each deconvolution layer has the same shape as its corresponding convolution layer, and its output is concatenated with the data encoded in the corresponding convolution layer. Finally, the latent data are rescaled, and the mask applied to the spectrogram is created to predict the source [21]. The U-Net model is depicted in Figure 3.

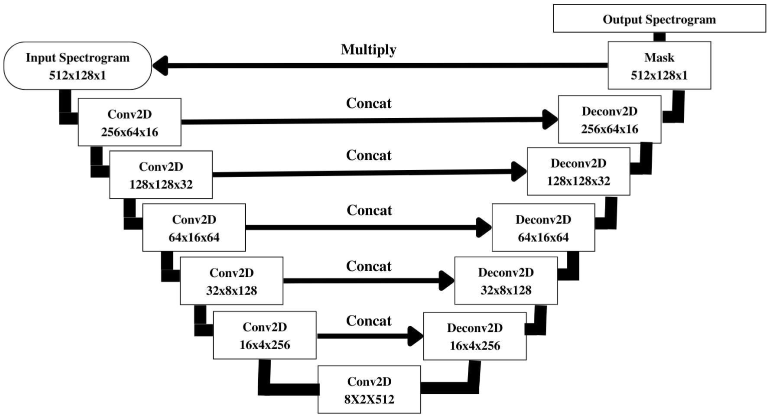

Figure 3. U-Net architecture.

U-Net first uses six convolution layers for encoding and five deconvolution layers for decoding. These layers are strides of two and have a 5×5 kernel size. Batch normalization and ReLU activation functions are used between all encoder and decoder layers. However, the output layer has softmax activation as it builds a mask for the spectrogram.

The difference between Spleeter and the original U-Net is that the audio is processed here in stereo, not mono. It works with two-channel audio as input and output (channel, time steps, frequency bins), and the architecture remains identical [24].

3.2.3. Wave-U-Net

Wave-U-Net, a modification of U-Net, separates audio sources directly using waveforms instead of the spectrogram [21]. Figure 4 depicts the Wave-U-Net architecture. Wave-U-Net uses a similar convolution/deconvolution sequence; however, it processes on waveforms instead of the spectrogram. It performs source separation by applying one-dimensional convolution/deconvolution operations on audio signals. Again, similar to U-Net, it concatenates the corresponding encoded and decoded data.

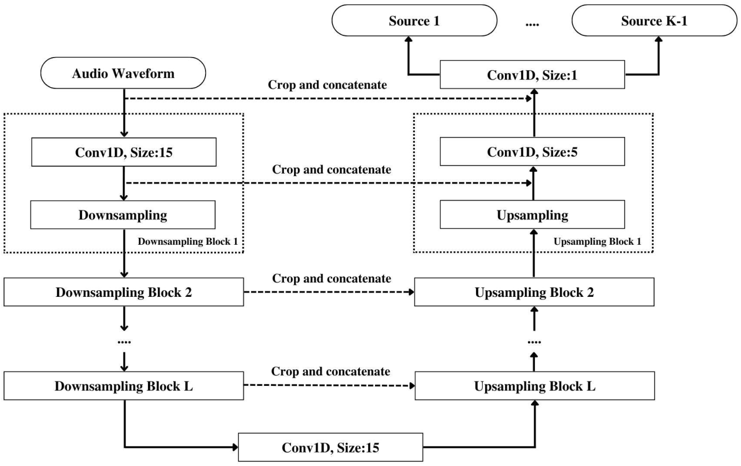

Figure 4. Wave-U-Net architecture.

Another dissimilarity from the U-Net model is the output layer. The model generates output for each source by applying convolution filters and hyperbolic tangent activations to the last feature map [9].

3.2.4. Hybrid Demucs

In audio source separation studies, the methods work over waveforms or spectrograms. Similarly, the original Demucs [25] only performed audio source separation with waveform input. Hybrid Demucs, on the other hand, adds the spectral domain to the original architecture and provides analysis in multiple domains [8]. The overall architecture is depicted in Figure 5. The original Demucs has the U-Net structure but, unlike U-Net, it consists of two Bi-LSTM layers in the middle of sequential encoders and decoders to better handle long-term context. It comprises six encoder and decoder layers with a skip connection between corresponding layers. The encoder layer includes a convolution layer with a kernel size of 8 and a stride size of 4, followed by ReLU activation. The encoder ends with the 1×1 convolution layer with gated linear unit (GLU) activation. The decoder layer sums the corresponding skip connection with the input from the previous decoder and applies GLU with 1×1 convolution. The decoder layer ends with a convolution with a kernel size of 8 and a stride of 4, followed by a ReLU activation [8].

With Hybrid Demucs, Demucs was extended with the spectral domain and multidomain analysis was provided. Besides the spectral domain, Hybrid Demucs has compressed residual branches, local attention, and

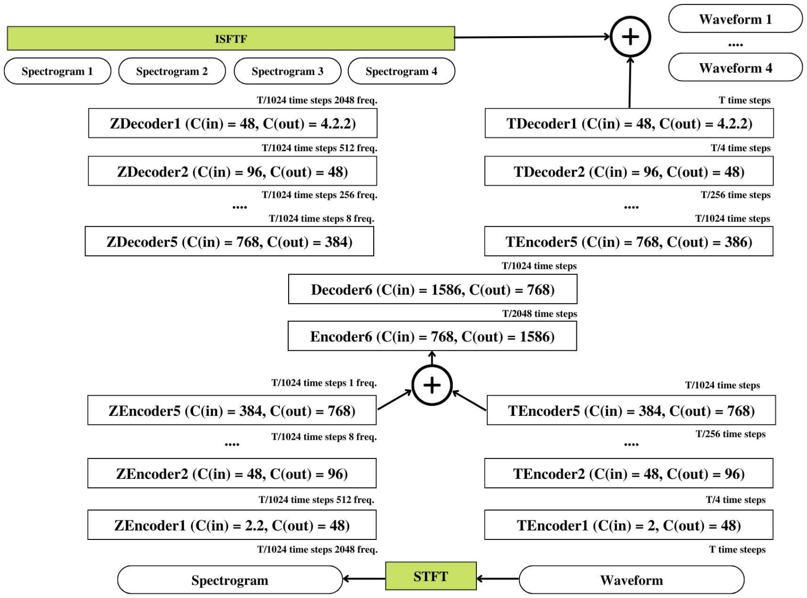
ISFTF
Waveform 1
....
Waveform 4
Spectrogram 1
Spectrogram 2
Spectrogram 3
Spectrogram 4
T/1024 time steps 2048 freq.
ZDecoder1 (C(in) = 48, C(out) = 4.2.2)
T time steps
TDecoder1 (C(in) = 48, C(out) = 4.2.2)
T/1024 time steps 512 freq.
ZDecoder2 (C(in) = 96, C(out) = 48)
T/4 time steps
TDecoder2 (C(in) = 96, C(out) = 48)
T/1024 time steps 256 freq.
....
T/1024 time steps 8 freq.
ZDecoder5 (C(in) = 768, C(out) = 384)
T/256 time steps
....
T/1024 time steps
TEncoder5 (C(in) = 768, C(out) = 386)
T/1024 time steps
Decoder6 (C(in) = 1586, C(out) = 768)
T/2048 time steps
Encoder6 (C(in) = 768, C(out) = 1586)
T/1024 time steps 1 freq.
ZEncoder5 (C(in) = 384, C(out) = 768)
T/1024 time steps
TEncoder5 (C(in) = 384, C(out) = 768)
T/1024 time steps 8 freq.
....
T/256 time steps
....
ZEncoder2 (C(in) = 48, C(out) = 96)
TEncoder2 (C(in) = 48, C(out) = 96)
T/1024 time steps 512 freq.
ZEncoder1 (C(in) = 2.2, C(out) = 48)
T/4 time steps
TEncoder1 (C(in) = 2, C(out) = 48)
T/1024 time steps 2048 freq.
T time steeps
Spectrogram
STFT
Waveform

Figure 5. Hybrid Demucs architecture.

singular value regularization improvements. Another distinction is that ReLU activations in the model change with Gaussian error linear units. Each encoder layer has compressed residual branches that are composed of dilated convolutions. Inside, convolutions are processed separately for both the time dimension and frequency bins. However, in the fifth and sixth layers, there are additional two-layer Bi-LSTM and local attention. Thus, Bi-LSTM, between encoders and decoders in the original Demucs, is moved into the encoder in Hybrid Demucs. After decoding, the output spectrogram is inversed with the inverse short-time Fourier transform (ISTFT) and summed with the temporal output. This way, the model creates the final prediction output [8].

4. Methodology

4.1. Data preparation

The MUSDB18-HQ [26] dataset was used in the experimental studies. It is a popular dataset in audio source separation studies. MUSDB18-HQ is an uncompressed version of the MUSDB18 dataset, and it consists of 150 tracks separated as training and test subsets. Signals of songs are stereophonic and encoded at 44.1 kHz.

Combining the training and testing subsets of the dataset, 150 songs were used for the audio source separation experiment. Drums, bass, vocals, and other components were mixed artificially instead of using the mixture in the dataset and the song itself. After that, the separation experiment was carried out. The mixing algorithm is shown below.

The overall time complexity of the source mixing algorithm can be approximated as $O(c * n)$, where c is the number of components and n is the average size of each component. The matrix factorization-based approaches, FastICA and NMF, cannot solve underdetermined BSS problems as the mixing matrix is not invertible. Hence, the $R \times N$ matrix was given as an input to these algorithms, in which R is the number of components, and N is the number of samples of the song. $R = 4$ as components are vocal, bass, drums, and others.

Algorithm 3 Source mixing algorithm.

```
for 1 to $number\ of\ components$ do
    $min \leftarrow minimum\ of\ component$
    $max \leftarrow maximum\ of\ component$
    if $max < 1$ or $min > 1$ then
        $component \leftarrow \frac{component}{\frac{max}{2}} - 0.5$
    end if
end for
```

Other methods, on the other hand, separate sources using two channels. The input data of these methods were prepared by applying the mixing algorithm for two channels. The sources were mixed with each channel's mixing algorithm, and then the mean of the four components was taken. The mean values represented one channel. Thus, the $C \times N$ matrix created with two channels became the input of the algorithms.

4.2. Method implementation details

Pretrained models trained by the authors of the architectures were used in experiments for the implemented architectures. Relevant application-specific subtleties are as follows:

- FastICA: The threshold value (separation error) and iteration count are crucial parameters. In our implementation, we used $1e$-8 and 5000, respectively.
- NMF [27]: NMF was implemented with threshold and iteration parameters set to $1e$-5 and 1000, respectively, chosen for faster runtime compared to FastICA.
- DUET: The implementation was done in the experimental studies as in the library, which includes several separation methods as mentioned in [28].
- Spleeter: The pretrained model called "spleeter:4stems" was used in the experimental studies and there were no additional parameters.
- Wave-U-Net: The pretrained model of [29] was used in the experimental studies.
- Hybrid Demucs: The pretrained model called "mdx" was used in the experimental studies and there were no additional parameters.
- Open Unmix: The pretrained model called "umxl" was used in the experimental studies and there were no additional parameters.

4.3. Metric

The source-to-distortion ratio (SDR) [30], a frequently used metric in the field of blind source separation, was used to evaluate the separated sources.

Evaluating the effectiveness of separation using performance measurements has limitations. In some cases, allowing for more distortions may be acceptable, or in musical applications, the measure of interference is crucial. To address these challenges, orthogonal projection is proposed for decomposition. Assume that $\prod\{y_1, ..., y_k\}$ is the orthogonal projector onto the subspace spanned by the vectors $y_1, ..., y_k$. The three projections can be considered in this way:

$$P_{s_j} := \prod\{s_j\} \text{ and } P_{\mathbf{s}} := \prod\{\left(s_{j'}\right)_{1<j'<n}\} \text{ and } P_{\mathbf{s.n}} := \prod\{\left(s_{j'}\right)_{1<j'<n}, (n_i)_{1<i<m}\}$$

Furthermore, $\hat{s}_j$ can be decomposed as follows:

$$\begin{aligned} s_{target} &:= P_{s_j}\hat{s}_j \\ e_{interf} &:= P_{\mathrm{s}}\hat{s}_j - P_{s_j}\hat{s}_j \\ e_{noise} &:= P_{\mathrm{s,n}}\hat{s}_j - P_{s_j}\hat{s}_j \\ e_{artif} &:= \hat{s}_j - P_{\mathrm{s,n}}\hat{s}_j \end{aligned}$$

The s_{target} signal is the desired output, while the three terms can be identified as error terms. The initial error term, e_{interf}, denotes the presence of unwanted signals from external sources like electronic devices. The second error term, e_{noise}, portrays the unwanted signals triggered by various factors such as atmospheric signals and thermal noises. Finally, the term e_{artif} refers to other unwanted signals such as “burbling” artifacts [30].

SDR, expressed in decibels (dB), is a comprehensive metric used to evaluate the efficacy of source separation and can be expressed as follows:

$$SDR := 10\log_{10}\frac{\|s_{target}\|^2}{\|e_{interf} + e_{noise} + e_{artif}\|^2}$$

The Museval library [31] was used to measure these values.

4.4. Experimental environment

Experiments were conducted on a computer operating system with the following hardware specifications: Ubuntu 20.04, Intel Core i7-1065G7, 24 GB RAM, and 2 GB NVidia MX330 GPU. All experiments were run with the CPU due to a lack of GPU memory and a primary processor on which all methods could run.

5. Evaluation

The results are presented here within the three subsections of time evaluation, genre evaluation, and score evaluation.

5.1. Time evaluation

The measure of time efficiency is indicative of an algorithm’s capacity to effectively isolate signals within a reasonable timeframe, thereby rendering it practical for real-world applications. The duration expended during the process of source separation serves as a critical determinant of the method’s complexity and time efficacy. The average time spent per piece for each method is shown in Table 1. The total time taken in minutes for separation is given in Table 2. The Hybrid Demucs architecture works with the spectral and time domains, and two Bi-LSTM layers are located in the center of the architecture. Hence, the complexity is increased and the separation time is increased considerably. Since the experiments are done with the CPU, it is expected that the machine learning models will take longer, but it can be concluded that the GPU memory requirement is higher for Hybrid Demucs or Wave-U-Net.

Table 1. Average time spent per track in seconds.

FastICA	NMF	DUET	Hybrid Demucs	Wave-U-Net	Open Unmix	Spleeter
9.21	296.20	29.21	574.17	357.20	148.29	23.21

Table 2. Total separation time in minutes.

FastICA	NMF	DUET	Hybrid Demucs	Wave-U-Net	Open Unmix	Spleeter
23.22	673	73.02	1435.43	893	370.71	58.04

5.2. Score evaluation

The median values of the SDR scores of the blind source separation methods after the artificial mixing and separation of 150 tracks in the MUSDB18-HQ dataset are shown in Table 3. The mean, standard deviation, and 95% confidence interval (CI) of the SDR scores are shown in Table 4.

Several valuable insights can be drawn from Table 4. First, it is worth mentioning that different methods have varying mean SDR scores. Spleeter and Wave-U-Net achieve positive scores, indicating that they perform better than the other methods. Nonetheless, it is important to highlight that the mean scores are still quite low, which emphasizes the difficulties involved in separating audio sources, especially in mixed music found in real-world situations.

Analyzing standard deviation figures provides valuable information about the distribution of SDR scores. A larger standard deviation suggests that the method produces varied performance results when separating components. It is important to take note of standard deviation results to gain insights. For instance, Hybrid Demucs and NMF exhibit smaller standard deviations, indicating consistent separation performance. Conversely, FastICA and Open Unmix have higher standard deviations, indicating a wider range of outcomes.

One crucial measure of the SDR score distribution is the 95% confidence interval, which provides insight into the actual distribution. A wider confidence interval indicates greater uncertainty regarding the true average. Analysis of Table 4 shows that the confidence interval values are rather wide, highlighting the difficulty in separating audio source signals. Nonetheless, it is worth noting that examining narrower interval values for other components may afford us some indication of consistency.

Upon closer examination of the three statistical concepts, intriguing patterns and trends emerge from the experimental data. Notably, the bass component exhibited a lower mean value and a wider confidence interval range, suggesting insufficient overall performance in separating this component using all methods. Additionally, classical methods showed lower mean values and wider confidence intervals, indicating that these methods encountered difficulties in separating audio source signals.

5.3. Genre evaluation

When thinking of rock music, drums come to mind, along with guitars. Different genres of music give weight to different instruments. In this regard, it is significant to explain how the audio source separation methods perform. The number of tracks according to genres in the dataset is shown in Table 5.

The SDR scores of each method's performance in separating different components can be found in Appendix A. Specifically, vocal scores are provided in Table 6, bass scores in Table 7, drums scores in Table 8, and other scores in Table 9. Moreover, the resulting mixture file, obtained by taking the mean value over the first axis of these components after stacking, is compared with the ground truth mixture file. The corresponding

Table 3. SDR scores of methods by components.

	Methods						
Components	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Vocals	-22.575	-20.178	-4.808	-11.650	-0.035	**3.145**	-0.567
Bass	-21.989	-20.639	-18.129	-18.686	-0.043	0.066	-9.303
Drums	-21.270	-20.031	-5.224	-13.655	**1.530**	1.035	**2.041**
Other	-22.582	-20.621	-5.273	**-6.855**	-16.052	-18.132	-2.087
Mixture	**-14.345**	**-13.692**	**-2.140**	-7.439	-1.754	-1.768	0.941

Table 4. Mean, standard deviation, and 95% confidence interval of the SDR (dB) scores.

	Components		Vocals	Bass	Drum	Other	Mixture
Methods	FastICA	Mean	-25.158	-23.769	-22.527	-23.249	-14.360
		Std. Dev.	11.551	10.194	8.107	3.787	2.105
		95% CI	-27,007 / -23,310	-25,391 / -22,147	-23,824 / -21,229	-23,855 / -22,643	-14,697 / -14,023
	NMF	Mean	-22.938	-22.158	-20.860	-21.582	-13.629
		Std. Dev.	12.074	10.078	8.133	4.400	2.373
		95% CI	-24,870 / -21,005	-23,771 / -20,546	-22,161 / -19,558	-21,906 / -20,497	-14,009 / -13,250
	DUET	Mean	-8.724	-18.294	-6.285	-5.967	-2.340
		Std. Dev.	13.702	11.101	6.035	5.175	2.105
		95% CI	-10,917 / -6,531	-20,071 / -16,518	-7,251 / -5,320	-6,795 / -5,138	-2,616 / -2,065
	Hybrid Demucs	Mean	-14.080	-20.540	-14.983	-7.664	-7.517
		Std. Dev.	10.324	9.303	7.513	3.655	1.729
		95% CI	-15.732 / -12.428	-22,028 / -19,051	-16,185 / -13,780	-8,249 / -7,079	-7,794 / -7,241
	Open Unmix	Mean	-5.674	-1.115	2.160	-15.374	-1.862
		Std. Dev.	10.821	5.661	2.390	5.718	1.291
		95% CI	-7,406 / -3,942	-2,021 / -0,209	1,778 / 2,543	-16,289 / -14,459	-2,069 / -1,655
	Spleeter	Mean	2.868	-0.762	0.927	-18.864	-1.863
		Std. Dev.	3.285	5.435	2.283	3.679	1.295
		95% CI	2,342 / 3,394	-1,632 / 0,108	0,475 / 1,379	-19,453 / -18,275	-2,071 / -1,656
	Wave-U-Net	Mean	-2.429	-10.643	1.649	-2.877	0.901
		Std. Dev.	6.893	8.864	4.692	2.647	0.528
		95% CI	-3,532 / -1,326	-12,061 / -9,224	0,898 / 2,400	-3,300 / -2,453	0,816 / 0,985

SDR scores are presented in Table 10.

The vocals component scores reveal that the country music genre, known for its clear vocal melodies and distinct instrumental arrangements, yields favorable results in separating vocals from other elements. Spleeter and Wave-U-Net demonstrate superior performance in isolating vocals. However, the methods' overall effectiveness in the electronic and rap genres, except for Spleeter, is comparatively weaker, possibly due to the

Table 5. Numbers of tracks in the dataset by genre.

Singer/Songwriter	Pop/Rock	Country	Rock	Rap	Reggae	Electronic	Heavy Metal	Pop	Jazz
14	72	3	17	8	2	8	12	11	3

complex and layered nature of arrangements in these genres.

The bass component analysis reveals several key findings. The methods excel in the reggae and electronic genres, leveraging the unique bass characteristics of these genres. Spleeter and Open Unmix stand out as top performers, effectively isolating bass elements. However, the jazz genre poses challenges for the methods, resulting in comparatively lower scores, likely due to the complex and improvisational nature of jazz music. It is worth noting that Wave-U-Net performs well for vocals but shows room for improvement in separating the bass component.

When it comes to the drum component results, the methods show promising results in the electronic genre, which is characterized by distinct drum patterns and repetitive rhythms, making it relatively easier to separate drums from other components. Spleeter, Open Unmix, and Wave-U-Net emerge as top performers in isolating drums. However, when it comes to the jazz genre, which often involves complex and improvised drum patterns, the methods have trouble getting good results. This suggests the need for further exploration and refinement of the methods to effectively handle the intricacies of jazz drumming.

In the analysis of the other components, first, there is a noticeable decrease in scores compared to the preceding components, indicating the challenges related to isolating these specific elements from the mixture. In terms of genre-specific performance, the methods produce slightly better results for the heavy metal genre, which often features distinct and prominent instrumentation. Wave-U-Net emerges as a slightly more effective method, although the scores remain relatively modest. However, when examining the rap genre, the methods struggle to achieve satisfactory results, suggesting the complexity of layered arrangements and intricate rhythms in this genre.

The results indicate variations in the performances of the methods across genres, with some methods consistently achieving higher scores in certain genres. These findings suggest that the choice of method should be tailored to the specific genre to achieve optimal component separation results.

6. Conclusion

This study investigated various audio source separation algorithms and evaluated their performances on different datasets. Through this analysis, we uncovered several important features of these algorithms. First, certain methods exhibited high RAM requirements due to their long runtimes or computational complexity, necessitating careful consideration of computational costs when selecting an appropriate algorithm. Second, the performance of algorithms such as Hybrid Demucs, which achieved high scores in the Sony Music Demixing Challenge, varied when applied to artificially mixed components, indicating the influence of input data characteristics on algorithm performance.

Additionally, we observed a shift in popularity from classical methods to machine learning models, as evidenced by the number of stars received by their official GitHub repositories. Spleeter emerged as the most popular method, followed by Hybrid Demucs, Open Unmix, and Wave-U-Net. Our findings align well with these popularity metrics.

Importantly, we found that the Spleeter method consistently delivered the best overall performance in terms of both working time and score values. However, it is worth noting that matrix factorization-based algorithms like NMF and FastICA may face limitations in solving underdetermined problems, which should be considered as a potential drawback.

Furthermore, our study shed light on the influence of genre-specific characteristics on the performance of audio source separation algorithms. Different methods demonstrated varying degrees of effectiveness across genres and specific components. This underscores the importance of genre considerations when selecting an algorithm, as certain components, such as vocals and bass, play crucial roles in specific music genres. By taking genre-related insights into account, future advancements in audio source separation can be tailored to better address the intricacies and variations within different music genres, ultimately enhancing the accuracy and quality of the separation process.

In summary, our study provides valuable insights for researchers and practitioners in the field of audio source separation, aiding in the selection of appropriate algorithms for specific tasks. We believe that our findings will contribute to the ongoing development of more accurate and efficient source separation algorithms, further advancing the field and facilitating enhanced music analysis and processing capabilities.

References

[1] Makino S, Lee TW, Sawada H. Blind Speech Separation. Berlin, Germany: Springer, 2007.

[2] Shlens J. A Tutorial on Independent Component Analysis. arXiv. 2014:1404.2986.

[3] Yu X, Hu D, Xu J. Blind Source Separation - Theory and Applications. New York, NY, USA: John Wiley & Sons, 2014.

[4] Lee D, Seung HS. Algorithms for non-negative matrix factorization. In: Leen TK, Dietterich TG, Tresp V (editors). Advances in Neural Information Processing Systems. Cambridge: MIT Press, 2000, pp. 535-541.

[5] Hyvärinen A, Oja E. Independent component analysis: algorithms and applications. Neural Networks. 2000; 13 (4): 411-430. http://doi.org/10.1016/S0893-6080(00)00026-5

[6] Hennequin R, Khlif A, Voituret F, Moussallam M. Spleeter: a fast and efficient music source separation tool with pre-trained models. Journal of Open Source Software 2020; 5 (50): 2154. http://doi.org/10.21105/joss.02154

[7] Stöter FR, Uhlich S, Liutkus A, Mitsufuji Y. Open-Unmix - A reference implementation for music source separation. Journal of Open Source Software 2019; 4 (41):1667. http://doi.org/10.21105/joss.01667

[8] D'efossez A. Hybrid Spectrogram and Waveform Source Separation. arXiv. 2111.03600 [cs, eess].

[9] Stoller D, Ewert S, Dixon S. Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation. arXiv. 2018:1806.03185.

[10] Correa N, Adali T, Calhoun VD. Performance of blind source separation algorithms for fMRI analysis using a group ICA method. Magnetic Resonance Imaging 2006; 25 (5): 684-694. http://doi.org/10.1016/j.mri.2006.10.017

[11] Bell AJ, Sejnowski TJ. An information-maximization approach to blind separation and blind deconvolution. Neural Computation 1995; 7 (6): 1129-1159. http://doi.org/10.1162/neco.1995.7.6.1129

[12] Cardoso JF, Souloumiac A. Blind beamforming for non Gaussian signals. IEE Proceedings F (Radar and Signal Processing) 1994; 140: 362-370. http://doi.org/10.1049/ip-f-2.1993.0054

[13] Hassan N, Ramli DA. A comparative study of blind source separation for bioacoustics sounds based on FastICA, PCA and NMF. Procedia Computer Science 2018; 126: 363-372. http://doi.org/10.1016/j.procs.2018.07.270

[14] Fourer D, Peeters G. Single-Channel Blind Source Separation for Singing Voice Detection: A Comparative Study. arXiv preprint. arXiv: 1805.01201. 2018.

[15] Liutkus A, Fitzgerald D, Rafii Z, Pardo B, Daudet L. Kernel additive models for source separation. IEEE Transactions on Signal Processing. 2014; 62 (16):4298-4310. http://doi.org/10.1109/TSP.2014.2332434

[16] Tharwat A. Independent component analysis: an introduction. Applied Computing and Informatics 2021; 17 (2): 222-249. https://doi.org/10.1016/j.aci.2018.08.006

[17] Müller M. Fundamentals of Music Processing: Using Python and Jupyter Notebooks. Cham, Switzerland: Springer Nature AG, 2021. https://doi.org/10.1007/978-3-030-69808-9

[18] Scott R. The DUET blind source separation algorithm. In: Naik GR, Wang W (editors). Blind Source Separation (Signals and Communication Technology). Berlin, Germany: Springer, 2007, pp. 217-241.

[19] Gunawan AAS, Stevelino A, Ngarianto H, Budiharto W, Wongso R. Implementation of blind speech separation for intelligent humanoid robot using DUET method. Procedia Computer Science 2017; 116: 87-98. https://doi.org/10.1016/j.procs.2017.10.014

[20] Yilmaz O, Rickard S. Blind separation of speech mixtures via time–frequency masking. IEEE Transactions on Signal Processing 2004; 52 (7): 1830-1847. https://doi.org/10.1109/TSP.2004.828896

[21] Manilow E, Seetharaman P, Salamon J. Open Source Tools & Data for Music Source Separation. 2020. https://source-separation.github.io/tutorial/landing.html

[22] Jansson A, Humphrey EJ, Montecchio N, Bittner RM, Kumar A et al. Singing voice separation with deep U-Net convolutional networks. In: Proceedings of the 18th International Society for Music Information Retrieval Conference; Suzhou, China; 2017.

[23] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional networks for biomedical image segmentation. Lecture Notes in Computer Science 2015; 9351: 234-241. https://doi.org/10.1007/978-3-319-24574-4_28

[24] Prétet L, Hennequin R, Royo-Letelier J, Vaglio A. Singing voice separation: a study on training data. arXiv. 1906.02618.

[25] D'efossez A, Usunier N, Bottou L, Bach F. Music source separation in the waveform domain. arXiv. 1911.13254 [cs, eess].

[26] Rafii Z, Liutkus A, Stöer FR, Mimilakis SI, Bittner R. Data from: MUSDB18-HQ - an uncompressed version of MUSDB18. Zenodo Repository. https://doi.org/10.5281/zenodo.3338373

[27] López-Serrano P, Dittmar C, Özer Y, Müller M. NMF Toolbox: Music processing applications of nonnegative matrix factorization. In: Proceedings of the International Conference on Digital Audio Effects (DAFx); Birmingham, UK; 2019. pp. 1-8.

[28] Manilow E, Seetharaman P, Pardo B. The Northwestern University Source Separation Library. In: Proceedings of the 19th International Society for Music Information Retrieval Conference; Paris, France; 2018. pp. 297-305.

[29] Stoller D. Implementation of the Wave-U-Net for audio source separation. Github. https://github.com/f90/Wave-U-Net

[30] Vincent E, Gribonval R, Fevotte C. Performance measurement in blind audio source separation. EEE/ACM Transactions on Audio, Speech, and Language Processing 2006; 14 (4): 1462-1469. https://doi.org/10.1109/TSA.2005.858005

[31] Stöter FR, Liutkus A, Ito N. The 2018 Signal Separation Evaluation Campaign. In: Deville Y, Gannot S, Mason R, Plumbley MD, Ward D (editors). Latent Variable Analysis and Signal Separation, LVA/ICA 2018. Berlin, Germany: Springer, 2018, pp. 293-305.

A. Experimental Results

Table 6. Vocals component SDR (dB) scores by genre.

	Methods						
Genre	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Singer/Songwriter	-25.512	-22.935	-5.678	-14.759	-16.341	0.641	-1.922
Pop/Rock	-22.445	-19.357	-4.678	-11.176	-0.031	3.886	-0.642
Country	**-13.824**	**-16.960**	**-1.071**	**-6.811**	-1.159	-1.168	**1.607**
Rock	-20.635	-20.598	-3.641	-10.107	**1.860**	**4.955**	0.234
Rap	-22.304	-20.905	-7.076	-11.642	-6.518	2.612	-1.479
Reggae	-21.645	-19.599	-6.388	-10.472	-12.639	4.800	0.271
Electronic	-30.544	-28.465	-18.262	-13.682	-4.602	-0.203	-1.547
Heavy Metal	-22.724	-20.380	-5.826	-13.030	0.627	2.077	-0.280
Pop	-21.194	-20.394	-5.539	-10.467	-2.927	2.958	-1.970
Jazz	-23.729	-23.064	-10.550	-12.651	-18.917	3.182	-1.506

Table 7. Bass component SDR (dB) scores by genre.

	Methods						
Genre	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Singer/Songwriter	-20.654	-19.179	-16.351	-17.084	1.313	0.251	-7.680
Pop/Rock	-22.496	-20.486	-19.221	-19.516	-0.716	0.063	-9.945
Country	-21.873	-24.113	-17.742	-18.260	**4.813**	0.293	-8.754
Rock	-20.748	-21.160	-16.009	-17.435	0.183	1.153	**-7.011**
Rap	-24.149	-22.359	-18.020	-20.260	0.673	0.742	-9.582
Reggae	**-18.213**	-17.724	-15.161	**-16.042**	2.380	0.623	-6.549
Electronic	-19.342	**-17.026**	**-13.941**	-17.365	0.308	**1.324**	-7.012
Heavy Metal	-21.995	-19.920	-17.617	-18.920	-0.543	0.018	-9.930
Pop	-21.842	-22.497	-17.373	-18.336	-1.665	-0.001	-9.282
Jazz	-57.417	-59.130	-42.362	-48.966	0.971	-5.992	-36.074

Table 8. Drums component SDR (dB) scores by genre.

	Methods						
Genre	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Singer/Songwriter	-26,323	-22,618	-10,473	-18,898	0,191	0,101	0,946
Pop/Rock	-20,957	-19,336	-5,145	-13,273	1,771	1,383	2,169
Country	-18,487	-22,147	-4,202	-12,962	**6,210**	1,155	3,220
Rock	-19,675	-20,269	-4,793	-11,521	2,616	1,165	2,068
Rap	-22,402	-20,615	-6,005	-14,098	1,939	0,628	2,248
Reggae	-20,938	-18,961	-5,383	-13,306	0,059	-0,282	0,242
Electronic	**-17,892**	**-15,269**	-4,205	**-10,259**	3,234	**2,266**	**3,259**
Heavy Metal	-20,561	-18,625	**-3,719**	-13,096	0,483	0,555	1,505
Pop	-23,190	-21,218	-7,902	-15,209	0,966	1,158	2,519
Jazz	-58,338	-59,447	-25,711	-45,379	0,203	-8,450	-17,671

Table 9. Other component SDR (dB) scores by genre.

	Methods						
Genre	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Singer/Songwriter	-21.368	-20.025	-3.425	-5.632	-12.462	-17.221	-1.334
Pop/Rock	-22.740	-20.221	-6.078	-7.119	-15.948	-18.552	-2.246
Country	-23.794	-26.680	**-0.738**	-8.511	-11.130	-20.174	-3.639
Rock	-22.388	-22.543	-3.392	-6.664	-18.281	-17.973	-2.205
Rap	-26.756	-24.496	-15.725	-11.672	-21.459	-22.929	-5.003
Reggae	-22.038	-20.295	-6.516	-8.930	-11.365	-19.437	-4.861
Electronic	-22.464	-18.708	-4.970	-6.539	-17.083	-17.383	-1.692
Heavy Metal	**-19.997**	**-18.704**	-3.449	**-5.051**	-15.950	**-16.113**	**-1.260**
Pop	-22.344	-20.705	-4.942	-5.616	-15.524	-16.865	-1.966
Jazz	-25.633	-25.371	-6.413	-9.122	**-6.270**	-20.925	-3.626

Table 10. Mixture component SDR (dB) scores by genre.

	Methods						
Genre	FastICA	NMF	DUET	Hybrid Demucs	Open Unmix	Spleeter	Wave-U-Net
Singer/Songwriter	-15.131	-13.712	-3.008	-8.389	-2.529	-2.525	0.972
Pop/Rock	-14.430	-13.329	-2.232	-7.544	-1.885	-1.877	0.813
Country	-13.824	-16.960	**-1.071**	-6.811	-1.159	-1.168	**1.607**
Rock	**-13.133**	-15.032	-1.142	**-6.610**	-1.244	-1.236	0.982
Rap	-14.774	-14.377	-3.831	-7.967	-2.177	-2.223	0.903
Reggae	-13.203	-11.779	-1.265	-6.746	-1.311	-1.348	1.042
Electronic	-13.601	**-11.407**	-1.424	-6.708	**-0.825**	**-0.820**	1.375
Heavy Metal	-13.823	-13.364	-1.591	-6.985	-1.350	-1.346	0.498
Pop	-14.720	-14.803	-2.478	-8.043	-2.193	-2.190	1.092
Jazz	-17.955	-15.141	-4.474	-10.373	-4.117	-4.148	0.243