NMF vs ICA for Audio Source Separation

Rahul Mahendru
u7213186
The Australian National University
Canberra, ACT - 2601
u7213186@anu.edu.au

Abstract

Blind Source Separation (BSS) is a problem aiming to separate a mixture signal into its components and extract the source signals. This paper compares and studies the independent component analysis (ICA) and non-negative matrix factorization (NMF) techniques on mixed audio data. Earlier works have compared their performance on other forms of data. The first stage of this project aims to separate an instrumental audio mix using both techniques. The second stage is a testing phase for mixed speech audio signals and also reports the accuracy results of the two algorithms using the Mean Squared Error (MSE) statistic. The FastICA algorithm results in a good model for separating the audio sources with accurate MSE scores. The NMF algorithm struggles with configuration, and the results for the extracted sound and MSE values do not align. The code for the implementation is available at https://github.com/rahulmahendru/ICASoundSeparation

1 Introduction

Blind Source Separation (BSS) refers to working with an unknown mixed signal to separate the individual sources without knowing the mixing methodology or the individual source data [1]. The problem uses the unsupervised form of learning to process the signals and find individual sources. We can represent the mixed signals as a linear combination of the individual sources and the mixing weights.

This paper analyses and compares the performance of two notable methods used for BSS: the Independent Component Analysis (ICA) method and the Non-Negative Matrix Factorization (NMF). We use audio data for our analysis. The primary motivation for this work comes from the cocktail party problem [2], which demonstrates the need for an efficient audio separation algorithm. This paper analyzes the algorithms mentioned above by solving an instance of the cocktail party problem.

Even though the algorithms mentioned above for BSS are well studied, we could not find many works that directly compare these algorithms' performance on audio data. In [3], Andri undertakes a similar effort to compare the performance of the two methods on image datasets. The work in [3] is large scale and compares many different algorithms for both ICA and NMF. This paper works on a smaller scale and aims to extend the comparison of the two techniques using audio data. More specifically, this paper aims to show the performance and analysis of the FastICA algorithm and the Kullback-Leibler NMF (Kl-NMF) algorithm using graphs and statistics.

The FastICA algorithm works by finding statistically independent components using a fixed-point iteration method. It maximizes the non-gaussianity by maximizing the negentropy of the extracted signals [4]. On the other hand, the KL-NMF algorithm aims to decompose the input matrix into the basis weight vectors and the gain function [5]. After separating the instrumental mix, we test the algorithms on mixed speech signals and report the Mean Squared Error (MSE) scores for the algorithms' accuracy.

In a nutshell, this paper aims to make the following main contributions:

1. Solve an instance of the cocktail party problem on instrumental and speech audio mixes using ICA and NMF.
2. Compare and analyze the performance of the FastICA and the KL-NMF algorithms in separating audio source signals using the given audio mixture signals.

The following section describes the definition of the problem in terms of BSS. In section 3, we describe the methods used throughout this project. Section 4 introduces the data and presents the results using graphs and statistics. In section 5, we present the conclusion and inferences for this work.

2 Problem Definition

The Blind Source Separation (BSS) problem separates the sources from mixture signals given little or no information about the mixture process [1]. This paper aims to apply the concept of BSS to a mixture of audio signals to retrieve their original music signals. We take three audio mixture signals as input and denote them as x_1, x_2 and x_3. Each of these signals is a weighted sum of three instrumental signals s_1, s_2 and s_3. We may denote the mixture signals as linear equations below:

$$\begin{aligned} x_1 &= a_{11}s_1 + a_{12}s_2 + a_{13}s_3 \\ x_2 &= a_{21}s_1 + a_{22}s_2 + a_{23}s_3 \\ x_3 &= a_{31}s_1 + a_{32}s_2 + a_{33}s_3 \end{aligned}$$

where the a_{ij} coefficients are weight parameters. We may simplify these equations in vector-matrix notation as:

$$x = As \tag{1}$$

where A denotes the vector of weights and s denotes the vector of individual sound sources. In this paper, we aim to solve the above linear equation using the two methods of ICA and NMF to retrieve the values of s. This problem is similar to solving an instance of the cocktail party problem [2].

In the project's second phase, we test the algorithms using speech data comprising two individual source signals, a male speech and a female speech. We may denote the problem equation for the speech mixture similarly as in equation 1.

3 Method

We propose using and comparing two BSS methods to separate the original audio sources from the given audio mixtures: ICA and NMF. To test the accuracy scores for these algorithms, we use a separate mix of speech audio data. We execute the procedure using the Python language and only basic libraries such as NumPy, Matplotlib and sklearn.preprocessing. These methods are described in detail below:

3.1 ICA for BSS

The ICA model is generative and aims to describe the mixing process of the constituents to generate the data. We make two assumptions while using ICA: First, the components s_i are statistically independent and second, the independent components must have non-gaussian distributions [4]. Using ICA, we aim to find the unknown matrix A and compute the independent components simply by calculating $s = Wx$ where W is the inverse of A.

3.1.1 Preprocessing

Data preprocessing is required for the FastICA algorithm to work efficiently [6]. This phase includes two steps:

1. **Centering:** To make the signals zero-mean, we center the data by subtracting the mean from all signals, $D = x - \mu(x)$. This step is taken to simplify the ICA algorithm.

2. **Whitening:** In this step, we transform the observed vector x linearly to obtain a new vector $\tilde{x}$ with uncorrelated components whose variances equal one and $\tilde{x}$ is orthogonal. We eliminate the correlation of each signal with the other using the eigenvalue decomposition of the covariance matrix $E[xx^T] = EDE^T$, where E is the orthogonal matrix of the eigenvectors of the covariance matrix and D is the diagonal of eigenvalues. We may now compute $\tilde{x} = ED^{-1/2}E^T x$. The orthogonality may be verified using $E[\tilde{x}\tilde{x}^T] = I$

3.1.2 FastICA

In this paper, we will use the FastICA algorithm for BSS. The FastICA algorithm uses a fixed-point iteration scheme. It extracts the independent audio source signals by maximizing the non-gaussianity by maximizing the negentropy of the extracted signals $J(w^T x)$. A basic version of the algorithm is as follows [7]:

Algorithm 1 FastICA

Require: $g(u) = tanh(a_1 u), g'(u) = uexp(-u^2/2)$

$w \leftarrow randomize$
while $w \neq converges$ **do**
  $w^+ \leftarrow E[xg(w^T x)] - E[g'(w^T x)]$ ▷ Optimize w by maximizing non-gaussianity
  $w+ \leftarrow w^+ - \Sigma(w + w^t)w$
  $w = w^+ / ||w^+||$
end while
return $w^T x$

This algorithm returns the optimal weight value for one unit/component. We repeat this process for the number of components in the mixed audio signal. Finally, we compute the individual components s using the dot product of w^T and x.

3.2 NMF for BSS

Non-negative matrix factorization aims to decompose a matrix into a product of two matrices. We decompose the matrix containing the mixed signals x into matrix H, which denotes the weights or activation values and W, which contains the basis weight vectors and may be used to retrieve the separated audio sources. We may directly use the spectrogram data of the audio signals. However, the NMF method expects the values of the audio signals to be non-negative. Thus the data is normalized in order to remove any negative values. We make use of the KL-NMF (Kullback-Leibler NMF) in this project. A basic version of the algorithm is as follows [5]:

Algorithm 2 KL-NMF

$W \leftarrow randomize$
$H \leftarrow randomize$
while $converges$ **do**
  $H \leftarrow H * \frac{W^T * \frac{X}{W*H}}{W^T 1}$ ▷ Update Activation
  $W \leftarrow W * \frac{\frac{X}{W*H} * H^T}{H^T 1}$ ▷ Update Basis Weights
end while
Normalize W
return W, H

3.3 Testing

We use two different mixes of speech audio files of varying lengths to calculate the accuracy scores for our algorithms. We measure mean squared error (MSE) to calculate the accuracy by observing the squared distance between the predicted and actual sound values [8]. A value closer to zero signifies a perfect model while it increases with more error. We undertake the following procedure:

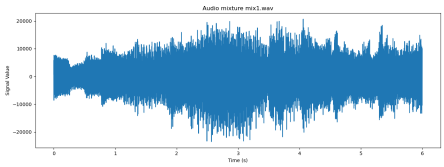
1. Mix the audio files to get matrix x using a mixing weight matrix A and the given individual speech data s.
2. Run the algorithm on the mixed data to retrieve the original signal data.
3. Calculate the MSE using the formula $\frac{\sum_i^n (y_i - \hat{y}_i)}{n}$

4 Experimental Results

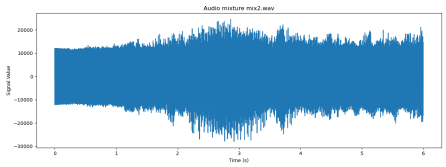
The following sections describe the data used, the results gathered using the two algorithms, and the performance metrics gathered to compare and evaluate the performance of FastICA and KL-NMF for audio source separation.

4.1 Data

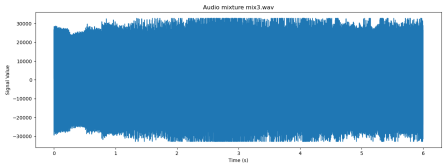
During the algorithm development stage, the data that we use includes three mixed audio files in the wav format. Each audio contains a mix of three instruments. Figure 1 shows the source signal data in a graphical plot against time.



(a) Original Mix 1



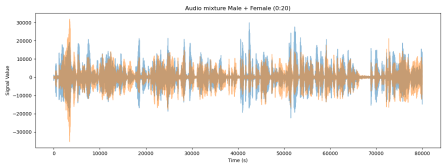
(b) Original Mix 2



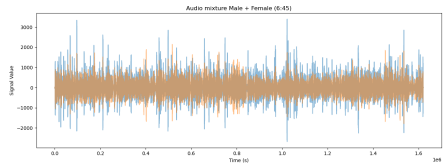
(c) Original Mix 3

Figure 1: Original instrumental mixture signals

During the testing phase, we used four different audio files. These audio files consist of two male speech files and two female speech files. These occur in pairs of two different durations, with one of the male and female files lasting for 20 seconds while the other pair lasts 6 minutes and 45 seconds. Each pair is mixed, and Figure 2 shows the source signals graphically.



(a) Male + Female Mix (0:20)



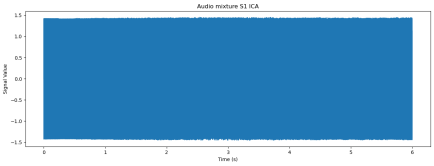
(b) Male + Female Mix (6:45)

Figure 2: Speech mixture signals

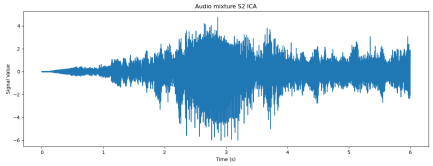
4.2 Experimental Results for Algorithms

The FastICA algorithm offers reasonably good separation of the audio signals into separate individual sources. From the output sources, we may infer that the three instruments in the audio mix are a

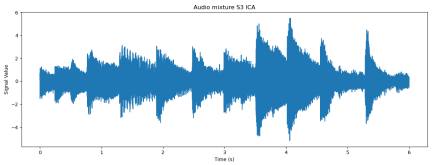
cello, a piano and a violin. Figure 3 shows the plot of the individual audio sources constructed using the FastICA algorithm.



(a) Individual Sound 1



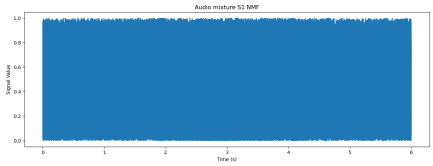
(b) Individual Sound 2



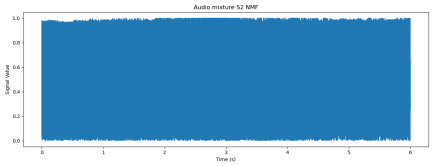
(c) Individual Sound 3

Figure 3: Individual instrument source signals using ICA

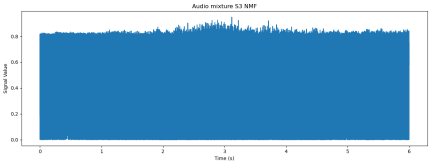
On the other hand, it is interesting to question the NMF algorithm's capacity to separate the audio mixes separately. As shown in Figure 4, while the NMF algorithm separates the individual audio sources, it does so based on the different frequency bins. The separated audio sources seem to have different pitches rather than different instrumental sounds on hearing the output sound sources.



(a) Individual Sound 1



(b) Individual Sound 2



(c) Individual Sound 3

Figure 4: Individual instrument source signals using NMF

4.3 Performance Metrics

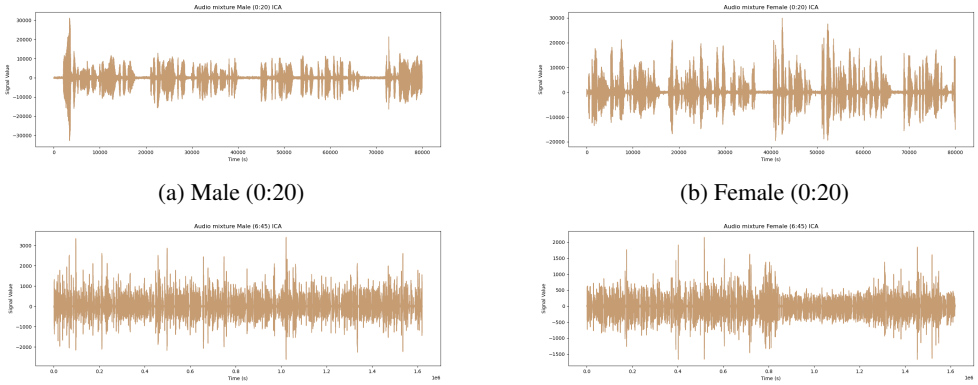
To further compare the two algorithms statistically and get sound results, we apply them to the speech data and measure the MSE values to record the accuracy of the predicted signals. Table 1 shows the MSE values for the two duration varying male and female speech data, and Figures 5, 6 shows the overlay for the predicted and actual values.

We expect the FastICA algorithm to give us accurate results after hearing the separated audio files. From Table 1, we can see that the FastICA algorithm results in small MSE values, indicating a model close to perfect. We may verify this claim further in Figure 5.

On the other hand, it is interesting to note that while using the NMF algorithm, the output audio only describes different pitches while listening to them. However, the MSE and graphical results seem accurate, with small MSE values in Table 1 and a perfectly overlapping plot in Figure 6. More

Table 1: MSE values recorded for speech data using ICA and NMF

Algorithm	Source Signal	MSE
ICA	Male (0:20)	19.3
	Female (0:20)	31.9
	Male (6:45)	1.08
	Female (6:45)	1.08
NMF	Male (0:20)	13.18
	Female (0:20)	24.19
	Male (6:45)	0.95
	Female (6:45)	0.93



(a) Male (0:20)

(b) Female (0:20)

(c) Male (6:45)

(d) Female (6:45)

Figure 5: Predicted vs actual signals using ICA

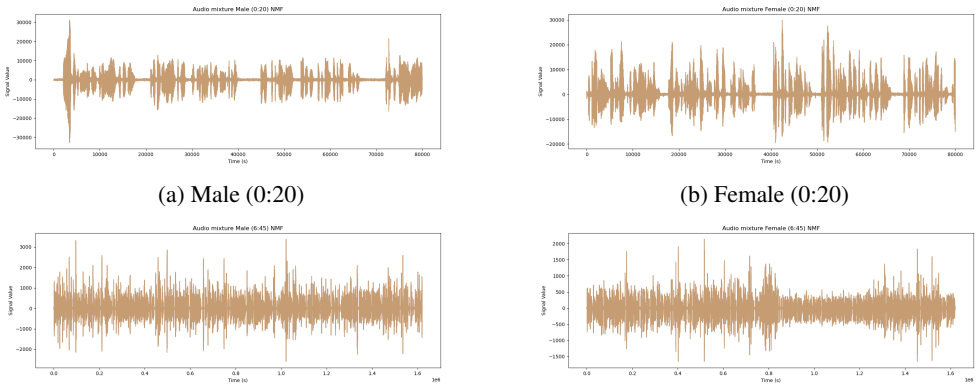
interestingly, the NMF algorithm tends to be working better than the FastICA algorithm for the speech data based on the MSE scores.

5 Conclusion

This paper compares the ICA and NMF methods for blind source separation for audio source separation. From the results, it is evident that for the first part of the project to separate the instrument sources, the ICA algorithm gives a clear sounding output to conclude that the instruments in the sound mix are a piano, a cello and a violin. The NMF separates the sound sources based on the pitch, and thus the instruments cannot be differentiated on hearing the output sources. Thus, the ICA algorithm is successful in solving an instance of the cocktail party problem.

Similarly, while hearing the separated audio signals during the testing phase, it is inferred that the FastICA algorithm separates the individual speeches clearly, and the NMF does so based on pitch levels. However, after calculating the MSE scores, it is interestingly noted that the NMF algorithm is excellently separating the speech signals into their constituent sources. Using the above results as a reference, we may conclude that the ICA algorithm is an excellent method for audio source separation. It is difficult to conclude if NMF is a better method because it appears to be separating the audio mix based on the frequency bins, and additional manual configuration is required to obtain the required output and results. Moreover, the MSE results do not intuitively align with the audio output, indicating a possible error in the NMF execution or the testing phase.

Additionally, as further work, we suggest working solely on the KL-NMF algorithm to explore the different configurations for the best possible results while separating audio sources. The work may



(a) Male (0:20)

(b) Female (0:20)

(c) Male (6:45)

(d) Female (6:45)

Figure 6: Predicted vs actual signals using NMF

include using different kinds of audio sources to observe the algorithm's behaviour with different pitches and noise.

References

[1] Madhab Pal, Rajib Roy, Joyanta Basu, and Milton S. Bepari. Blind source separation: A review and analysis. In *2013 International Conference Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–5, 2013.

[2] Kevin JP Woods and Josh H McDermott. Schema learning for the cocktail party problem. *Proceedings of the National Academy of Sciences*, 115(14):E3313–E3322, 2018.

[3] Andri Mirzal. Nmf versus ica for blind source separation. *Advances in Data Analysis and Classification*, 11(1):25–48, 2017.

[4] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[5] Nicholas Bryan and Dennis Sun. Source separation tutorial mini-series ii: Introduction to non-negative matrix factorization, 2022.

[6] Alaa Tharwat. Independent component analysis: An introduction. *Applied Computing and Informatics*, 2020.

[7] Fastica, 2022.

[8] Jim Frost. Mean squared error (mse), 2022.