Analysis of Demucs for Music Source Separation on waveform domain.

Sourav Panda

Master of Science in Data Science
The University of Bath
(2021-2023)

Analysis of Demucs for Music Source Separation on waveform domain.

Submitted by:
Sourav Panda
For the degree of MSc in Data Science of the
University of Bath

Copyright

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of Master of Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Abstract

⟨ The process of separation of individual sources from a mixture of sounds recorded individually and arranged together to form a melody is known as Music Source Separation. Voice, bass, percussion, and any other accompaniments are examples of such components. In contrast to many audio synthesis jobs where models that directly create the waveform get the greatest results, the state-of-the-art in source separation for music is to calculate masks on the magnitude spectrum. In this dissertation, we use demucs architecture a waveform Music source separation. We compare model generalisation with other waveform models and test model response to noisy data sets of music songs from various genres using Additive Gaussian White noise.⟩

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my supervisors Jordan Taylor and Alessio Guglielmi for their constructive feedback and support during the project. I would also like to thank my parents, wellness team and mentor Bikramaditya Singhal for their moral support. Lastly, I would like to thank my roommate for providing me with meals during this project.

Chapter 1

Introduction

1.1 What is Music Source Separation?

The process of separating individual sound sources from a mixture of sounds from multiple isolated sources s known as Music Source Separation as shown in Figure 1.1. Bass, drums, guitar, or other musical instruments, are examples of isolated sources. To express these words in terms of an equation, we can write the mathematical Equation 1.1

$$y(t) = \sum_{i=1}^{N} T_i(x_i(t)) \tag{1.1}$$

In Equation 1.1, the mixture signal is represented by $y(t)$, N is the number of isolated sources the mixture signal $y(t)$ is composed of,$x_i(t)$ is the individual audio signal from i^{th} isolated source, T_i is a transformation function used on the individual source before the mixing of sources which is unknown information to the source separation model and can be different for each source. For a given $y(t)$, the main objective of Music Source Separation is to recover individual sources $x_i(t)$. (Manilow, Seetharman and Salamon, 2020)

1.2 Motivation of Music Source Separation

The motivation for the project is, that there is a significant growth in content creation due to the boom in social media and easy internet access (Perrin, 2015). Music is an important part of the content to improve the user experience (Chaney, 2012). This has resulted in the importance of copyright-free music for background music in videos or the making of instrumental covers.To produce an instrumental rendition of a song, the background track must be separated from the vocals and finding a copyright-free background track might be difficult. (Gateau, 2014). This is faced by budding content creators and musicians. Using Music Source Separation techniques, we can obtain the individual audio signal from individual instruments used in making the mixture. The individual audio signal also known as stem is then used by mix-matching with our own recording to create new content.

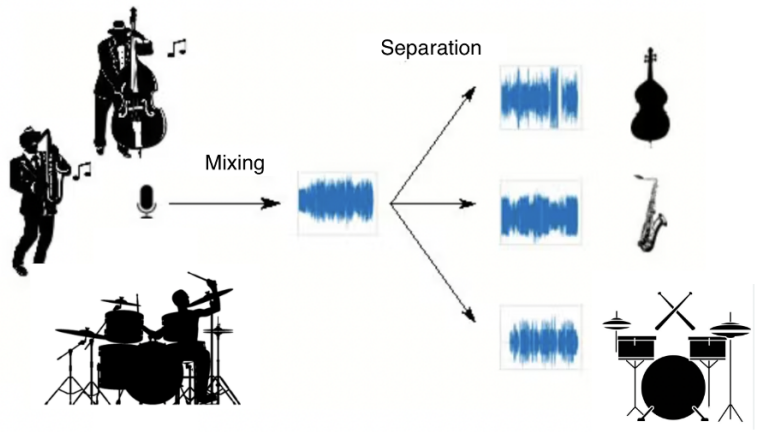

Figure 1.1: Music Source Separation.

1.2.1 Applications for Music Source Separation

In recent years, there is a lot of advancement in Music Source Separation techniques due to its increasing demand in the field of Music Source Separation(Casey et al., 2008). Few applications are as follows:

- It is often the case that it is easier to process audio from isolated sources over a mixture of sources. Also, it adds a range of flexibility to be able to treat independently using different processing techniques like reverb, pitch correction, etc to the audio form individual sources.(Stöter et al., 2019)
- The idea of automatic music transcription is a great tool for musicians. Due to the complexity of different scales in music, it is a very difficult and important task to transcribe music by listening to it, specifically for budding musicians. This example is shown in Figure 1.2.(Manilow, Seetharaman and Pardo, 2020)
- Automatic music instrument and singer identification to help the search engine for music apps.
- As we know that during the production of any music these days, there is a lot of mixing. Different instruments are recorded in isolation and it consumes a lot of time to achieve lyric music alignment manually. We can automate the process using Music Source Separation Techniques (Mesaros and Virtanen, 2010).

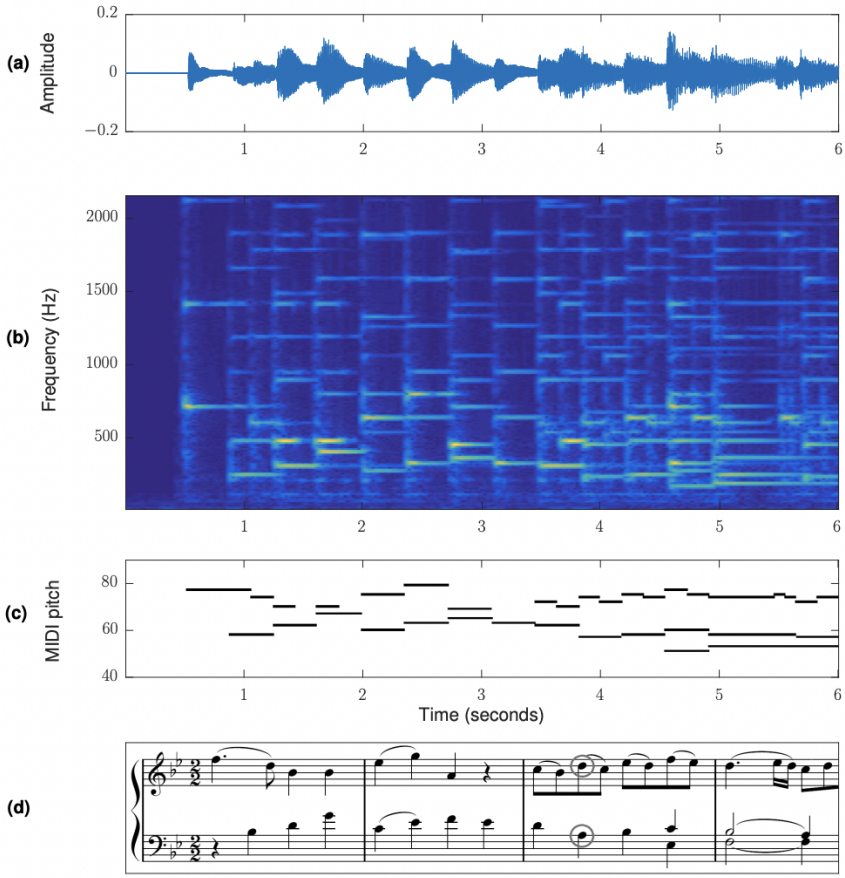

Figure 1.2: (a) input waveform, (b) internal time-frequency representation, (c) output piano-roll representation, and (d) output music score, with notes A and D marked in Gray circles. Image courtesy of Benetos et al. (2018)

1.2.2 Challenges in Music Source Separation

There are many challenges which make Music Source Separation unique and distinct from other source separation techniques. The following characteristics of the musical audio wave make it different from other source separation techniques:

1. Any song or a piece of music is melodious and sounds good to our ears because of the harmony in the stems. Stems change in sync at the same time, which results in a strong correlation among the sources.

2. In the modern world, musical output in most cases is not a linear addition because they are mixed and processed by the composers and music producer in a non-linear manner. The use of reverb and other non-linear signal processing techniques on stems makes the separation more complex. Also, we do not have the information about the transformation to the individual stems or the whole mixture that adds to the complexity (Cano et al., 2018).

3. The end product of Music Source Separation needs to be highly accurate. To be used further by the end user, it is very important that the output is close to the original.

This raises the bar for quality much higher in comparison with other source separation techniques.

1.3 Objectives

The objective of this project is to test the performance of the Music Source Separation model on the waveform domain. Across many music synthesis applications, models that directly create the waveform outperform, whereas, in the case of Music Source Separation, the state-of-the-art is to estimate mask on the spectrogram. (Défossez et al., 2019). We will test with an unseen genre of music with different musical instruments to know how well the model generalises the music space and analyse the benefits of data augmentation techniques in the temporal domains. Additionally, we would examine the effects of noise in the input mixture and its impact on the performance of the source separation model by implementing a model for Music Source Separation in the waveform domain to achieve equal or better performance on the existing state-of-the-art models.

Chapter 2

Background

Signal Source separation originated from the cocktail party effect in 1953. Cherry observed that the brain can segregate the audio waves from the person a human is conversing with, from all other audio waves like background music, surrounding noise from other people chatting in the room, sound from cutlery, etc. Music Source Separation is a part of signal source separation that deals with extracting individual sounds or stems from a mixture of sounds from multiple isolated sources such as bass, drums, guitar, or other musical instruments. In the last 50 years, due to improvement in computing capacity and availability of various data sources along with improvement in the field of deep networks and various competitions like the Music Demixing (MDX) Challenge, Music Source Separation has invited a lot of interest from researchers. (Lee, Choi and Lee, 2019)

2.1 Audio as input

Audio waves are the vibrations in air molecules. Audio inputs are represented in the form of waveform or time-frequency in Music Source Separation models.

1. **Waveform** is the most fundamental representation of any sound wave. It is the digital representation of the sound signal and it is closest to sound in its physical state. Waveforms are generated by converting the changing air pressure to voltage through the microphone, which is then sampled in regular time intervals and a quantized form is stored in the form of an array. This digital array is known as a waveform. An important point to note is that the signal is discretised in the space of both time and magnitude. Audio signals with one audio channel are called mono or monophonic, and the array of these signals is of shape $(t \times 1)$ where t is the number of time steps. Similarly, signals with two audio channels are called stereo or stereophonic, and the array of these signals is of shape $(t \times 2)$. The sample rate is a very crucial parameter for the waveform. The sample rate represented by sr is the number of samples per second and measured in hertz. While working with deep learning models, we upsample and downsample to increase and decrease the computational load respectively. An important point to be aware of is downsampling removes the information at higher frequencies. Figure 2.1 shows the representation of waveform in different time scales. (Manilow, Seetharman and Salamon, 2020)

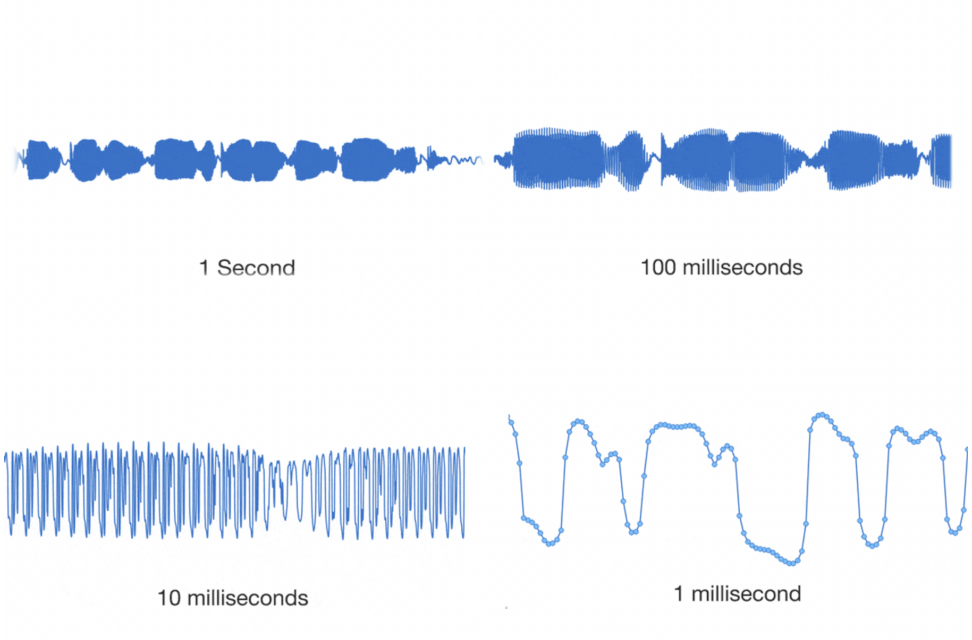

Figure 2.1: Variation in waveform for different time scale. Image courtesy of Balen (2020)

2. **Phase** indicates the timing and the location of a waveform. It is measured in radian or degree. Most of the machine learning models have focused on finding the best mask to filter out the frequencies. Phase is one of the key aspects of sound. Short Time Fourier Transform (STFT)on phase outputs noise makes modelling of a phase difficult as shown in Figure 2.2.

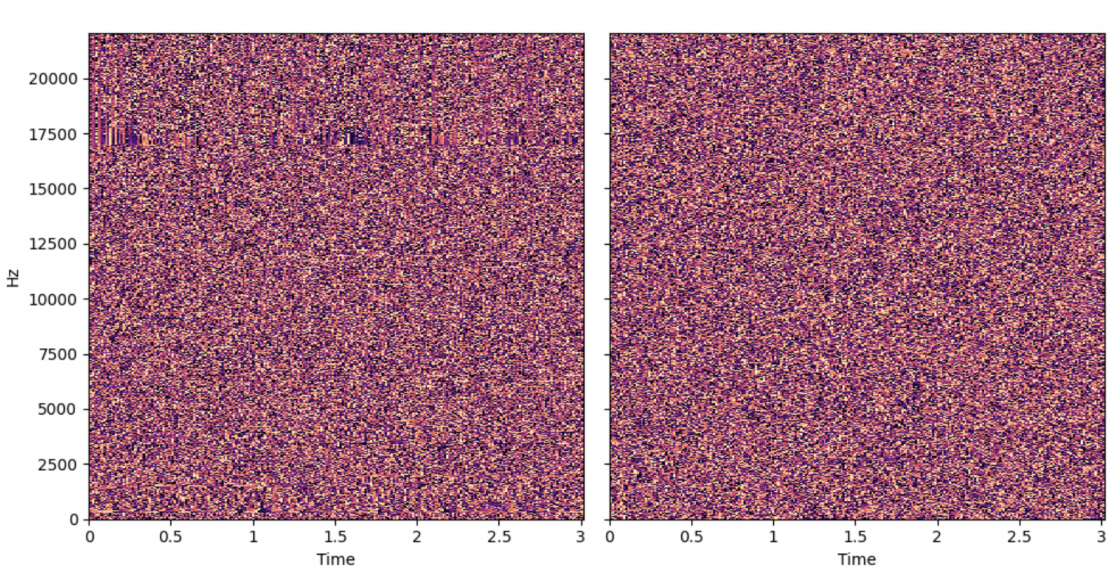

Figure 2.2: The left image in figure is the structure of with STFT and the right image is random noise. Image courtesy of Manilow, Seetharman and Salamon (2020).

3. **Time-Frequency** representation depicts the frequency components of an audio signal over time in a two-dimensional matrix. The visual representation of Time-Frequency

representation is achieved by plotting a heat map where time is represented on the x-axis, frequency is represented on the y-axis the intensity of light represents the amplitude. This representation is also known as a spectrogram.

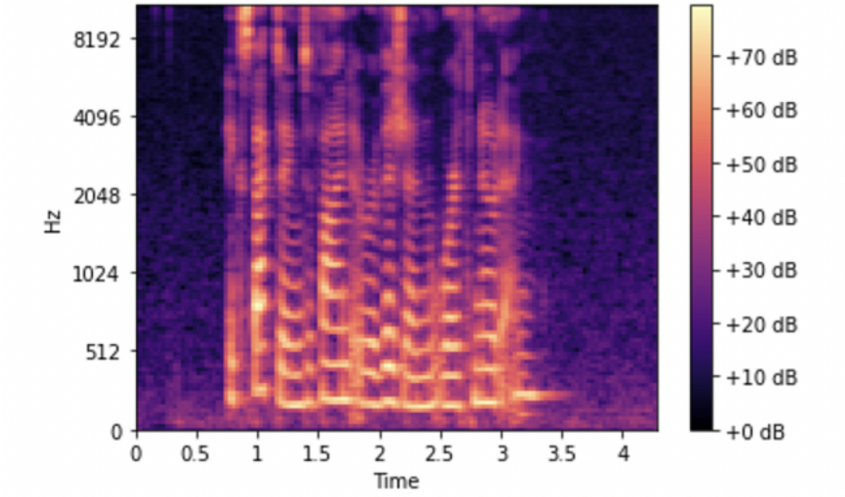

Figure 2.3: Spectrogram.Image courtesy of (Manilow, Seetharman and Salamon, 2020)

2.2 Deep Learning

Neural Network methods are the state-of-the-art methods for Music Source Separation. Neural Network based methods are also known as deep nets or deep learning methods. The first model of Deep Learning was built by Pitts and Warren McCulloch in 1943. This model was based on the neurons of the human brain. The development of polynomial and non-linear activation functions by Alexey Grigoryevich Ivakhnenko and Valentin Grigorevich Lapa proved the early success of Deep learning models (Tappert, 2019). Neural Networks work on large amounts of training data. In Music Source Separation, the model is trained on large sets of mixed and isolated audio waves. The output of this deep net model is compared with the original isolated source and with the help of a loss function, the model gives quantified feedback. This feedback is used to update the weight and parameters of the model. This process is called backpropagation and the model comes under the umbrella of supervised machine learning models. In source separation techniques, deep net models are complex due to the presence of a large number of weights also known as training parameters. The setup choice of these parameters is known as hyperparameter.

2.2.1 Neural Network Components

1. **Layers:** The neural network is built of layers and each layer has a set of weights. These weights are an important feature which affects the accuracy of the model. These weights are updated using gradient descent and the model with the least loss or most accuracy is saved. Types of layers in a neural network are as follows:

   (a) **Fully Connected Layers:** In the Fully Connected (FC) layers, as the name suggests, each neuron is connected to every other neuron from its next layers as shown in Figure 2.4. One primary application of the FC layers is to expand or

compress the dimensionality from the input or the output from the previous layers to be fed as input to the next layers. In the context of source separation, FC layers with an activation function are used in the process of changing the dimensionality as required to match the dimension of the input to the next layers or output.

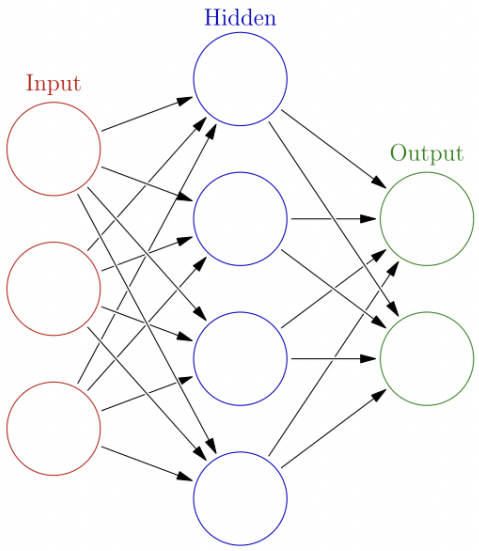

Figure 2.4: Fully connected layers. Input layers is in the red, hidden layer is in blue and the output layer in green. Image courtesy of Manilow, Seetharman and Salamon (2020)

(b) **Convolution Layers** Convolutional layers are like FC layers as discussed above, except in the case of the convolution layers, every node is not connected, rather a small set of nodes are only connected. This helps the model prevent overfitting the data set. An important characteristic of convolution layers is that they are linear and translationally invariant which means they produce the same output even if there is a shift in the input (Manilow, Seetharman and Salamon, 2020). As the name suggests, convolution layers are associated with the signal processing concept of convolution, as shown in Figure 2.5. Stride is the number of units the kernel moves and kernel is a weighted matrix that is multiplied with the input for relevant feature extraction.

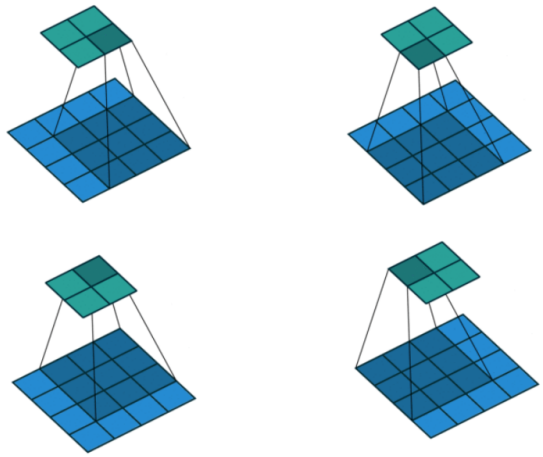

Figure 2.5: Step-by-step convolution operation for a 2-dimensional input with stride 1 and kernel size 3. Image courtesy of Dumoulin and Visin (2016)

(c) **Recurrent Layers:** In recurrent layers, part of the output is taken as input for the same node, which forms a loop and thus information can persist. Due to the formation of these loops, recurrent layers are specifically best suited for audio signals because they can process audio as it varies over time. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are two types of common recurrent layers used in signal source separation space because these types of recurrent layers are able to retain information in a more stable way. Recurrent Layers can see the information over the time axis. Recurrent layers with the ability to see the information in both directions, that is, forward in time and backward in time, contain two sets of units and are known as Bidirectional Recurrent Layers. Similarly, Unidirectional Recurrent Layers can see information in only one direction. As bidirectional Recurrent Layers can see the future, they can not be used in real-time processing. (Manilow, Seetharman and Salamon, 2020)

2. **Activation Functions** Activation functions helps the model to learn the non-linearity of the data. It sets the degree to which one layer can affect the next layer. All activation functions have the property of being differentiable and are non-linear in nature. Activation Functions are also used to decide when to switch the node on and off. Switching the node off means that there is no effect on the next layer and switching it on would result in certain effects based on the type of activation function. (Manilow, Seetharman and Salamon, 2020)

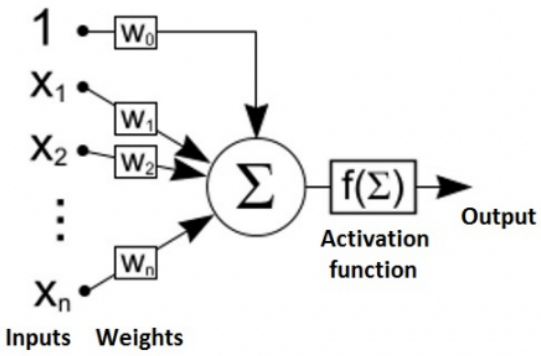

Figure 2.6: Application of Activation Function to neural networks. the combined output of the node is passed through the Activation function. Image courtesy of Sibi, Jones and Siddarth (2013)

In most Neural Network Models, there will be at least one layer with an activation function to learn non-linearity in the model. There are many types of activation functions. Few relevant activation functions in the field of Music Source Separation are as follows:

(a) **Sigmoid** As shown in Figure 2.7, the out put of the sigmoid function bound between 0 and 1. This makes the sigmoid function a good choice to be used as an activation function for Masks. The sigmoid function is denoted by the symbol σ.

$$s(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$

In Equation 2.1 σ(x)is the sigmoid function for an input x.

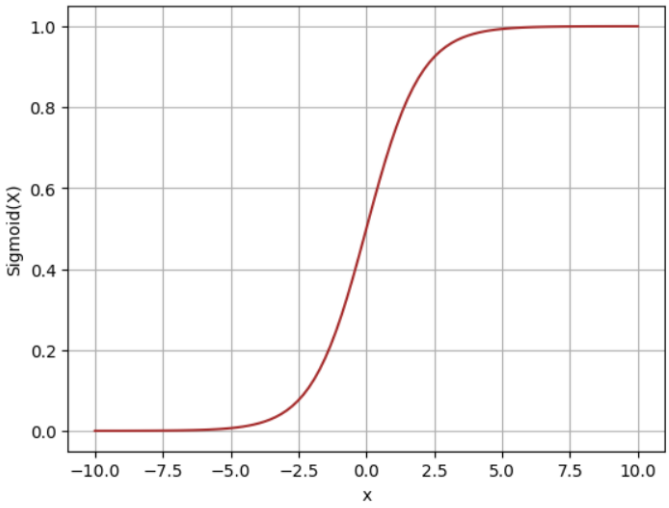

Figure 2.7: A Sigmoid activation function plot.

(b) **Rectified Linear Units** The shape of Rectified Linear Units (ReLU) is shown in Figure 2.8. ReLU outputs the input directly if the input is greater than 0 and

returns 0 if the input is less than or equal to 0. It is one of the most common activation functions because it has induced linear functions but has non-linear mathematical characteristics, which allows the model to learn complex patterns in the data set. In certain models, ReLU is used as an activation function to create a mask and also in the output layer which generates waveform.

$$f(x) = max(0, x) \tag{2.2}$$

In Equation 2.2, x is the input and $f(x)$ is the output of the ReLU activation function.

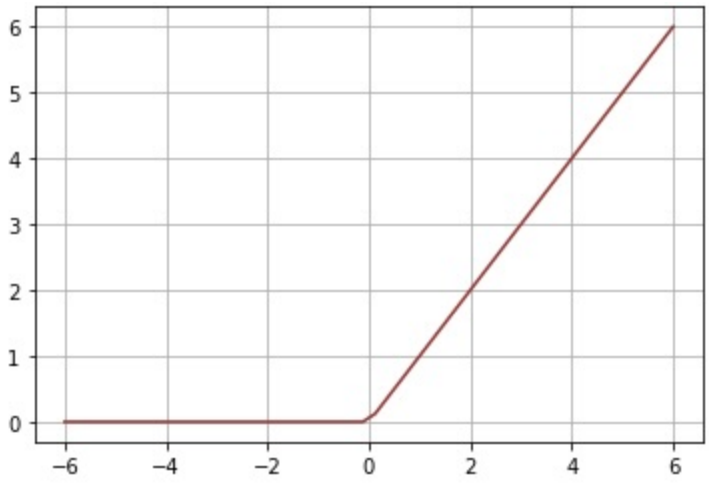

Figure 2.8: A ReLU activation function plot.

There are two activation functions, Leaky ReLU and PReLU which share the same family. Similar to ReLU, Leaky ReLU returns the same input for input greater or equal to 0 and returns a weighted output for input less than 0. The weight is kept close to 0 so that it can make the smallest updates. Similarly, PReLu returns the same output except the weight is replaced by a learnable network parameter α.

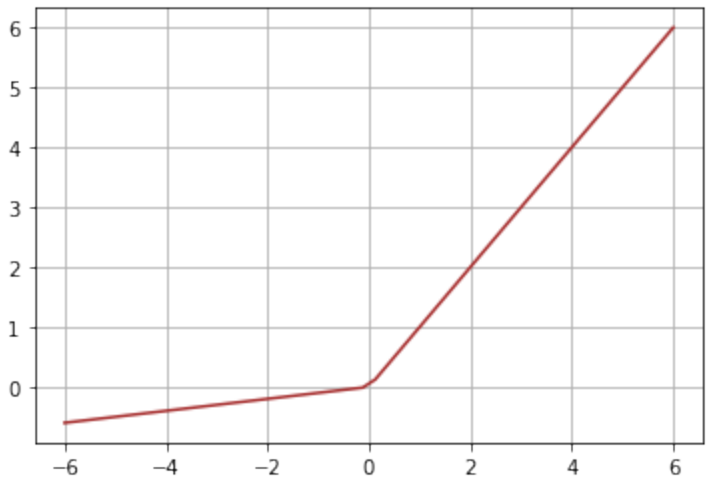

Figure 2.9: A Leaky activation function plot.

Chapter 3

Literature and Technology Survey

In this Section, we will have an overview of existing work in the field of source separation. Find out the scope of Music Source Separation in the waveform domain and analyse different research work to find a common metric of evaluation and choose a state-of-the-art end-to-end waveform model for Music Source Separation. Broadly, Music Source Separation Models are classified into two categories based on the form of audio input they use. Models estimating through masking the spectrogram fall under the frequency domain or spectrogram-based method. Model estimating from the audio as the waveform is known as temporal domain or waveform-based methods. The first category of Music Source Separation techniques worked on the time-frequency domain. In these models, the goal is to predict the mask which is then applied to the spectrogram, and the phase information from the input is used to generate the waveform. We also identify other models in the time-frequency or spectrogram domain in case the waveform end-to-end model does not yield desired results.

We reviewed different models to compare different approaches and know more about their characteristics and performance in source separation. Models in time-frequency space work on mask estimation for the estimation of each isolated audio source also known as stems. As the computing power increased and deep learning methods developed, large public data sets like MUSDB18, and Slakh2100 were available, and fully supervised learning gained traction. The first few models on source separation were focused on speech source separation.

Grais, Sen and Erdogan (2014) proposed a Deep Neural Network (DNN) for single channel source separation. In this paper in contrast to classification-based speech separation, where there are classifiers to divide time-frequency bins into sources, masks are estimated using the DNN approach.As shown in Figure 3, the DNN neural network architecture has two outputs, one for each source. $h_1, ..., h_4$ are the hidden layer and $w_1, ..., w_4$ are weight of the respective hidden layers. DNN model performed well for speach estimation with a SDR value of 8.96. However, it was not good for music source separation and achieve a SDR of 1.97.

U-Net is a convolutional neural network that was first developed at the University of Freiburg's Computer Science Department for biomedical picture segmentation. The network uses a fully convolutional network as its foundation and the network's architecture was updated and extended to work with fewer training photos and thereby provide better segmentation (Ronneberger, Fischer and Brox, 2015). Jansson et al. (2017) investigated the U-Net architecture in the context of singing voice separation and discovered that it provides significant gains over current approaches. Uhlich, Giron and Mitsufuji (2015) worked on Music Source

Separation with fully connected layers over limited spectrogram frames. The introduction of recurrent networks and Long Short-Term Memory (LSTMs) in Liu and Yang (2018) and Uhlich et al. (2017) proved to be ground-breaking. This also established that winner filtering is the best post-processing technique for time-frequency based models. Winner filtering is still used by best performing time-frequency or spectrogram based models. These models were the best performing in SiSec 2018 evaluation Stöter, Liutkus and Ito (2018a) for source separation on the MUSDB data set. Stöter et al. (2019) released a reproducible baseline called Open Unmix after the evaluation that matches the top submissions trained only on MusDB. Takahashi, Goswami and Mitsufuji (2018)'s MMDenseLSTM model trained on 807 unreleased songs currently holds the absolute record of SDR in the SiSec campaign. Jansson et al. used skip connection with U-Net and compared it with ordinary convolutional encoder-decoders, to demonstrate the advantages of low-level skip connections. In U-Net layers are coupled as encoder and decoder, also they have the same number of filters, size, strides and output dimensions making the architecture symmetrical. They are connected using skip connections as the U-Net is designed for a symmetric architecture. The goal behind these connections is to improve the retrieval of the separated signal by supplying finer details in decoding (from the encoder) which will be not utilised by the decoder without this connection and information will be lost. Takahashi and Mitsufuji (2020) on D3Net, which performed convolution on a wider kernel by inserting space between the kernel elements with dense connection, surpassed the state-of-the-art model in the time-frequency or spectrogram domain.

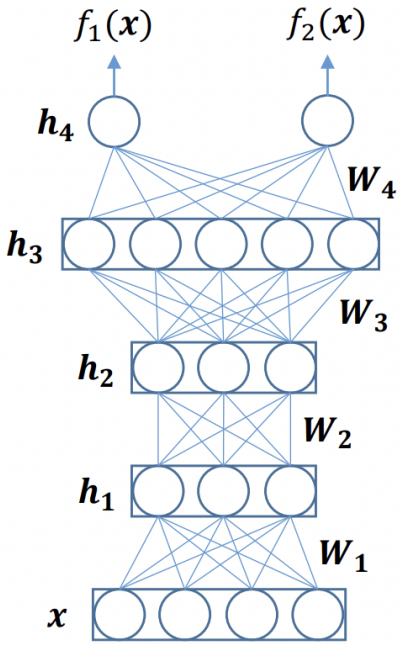

Figure 3.1: DNN architecture. Image courtesy (Grais, Sen and Erdogan, 2014)

In recent years, there have been few developments in the waveform domain, also known as the temporal domain. However, these models outperformed spectrogram based models by a wide margin. Stoller, Ewert and Dixon (2018) first adapted the U-Net structure to a wave-form domain, which was first used by Jansson et al. (2017) on the spectrogram domain. As shown

in figure 3 the U-Net architecture by Stoller, Ewert and Dixon is for K sources and L layer. Concat concatenates the current, high-level features with more local features, size is the kernel size of the convolution layer.

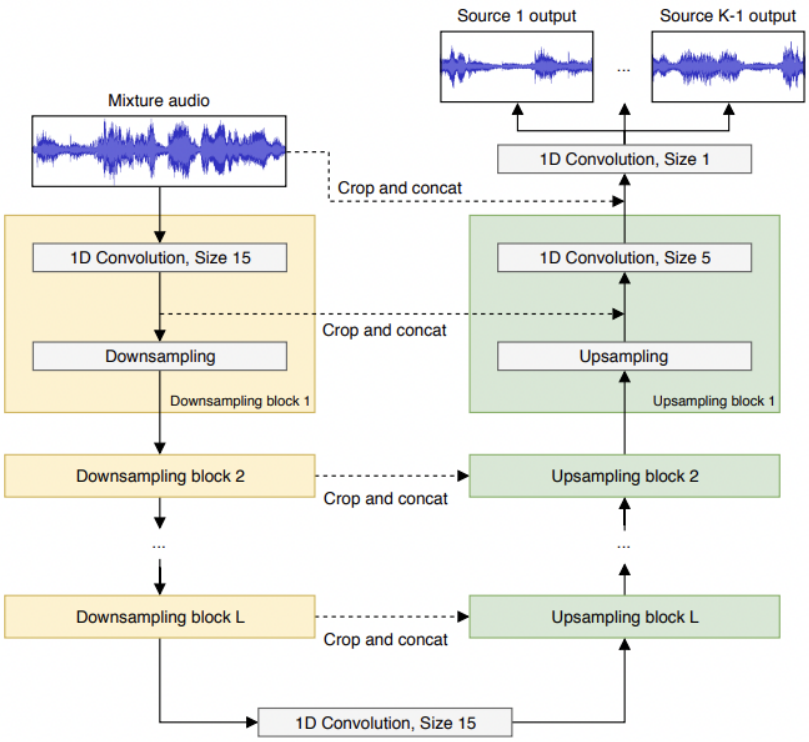

Figure 3.2: Unet architecture for music source separation on waveform domain. Image courtesy (Stoller, Ewert and Dixon, 2018)

Lluís, Pons and Serra (2018) gives us detailed information on the feasibility of waveform-based source separation models and the advantages of waveform-based end-to-end models over signal filtering or time-frequency transformation-based models. However, it did not offer enough detailed information on current cutting-edge waveform models to allow experimentation. More details on the characteristics and performance of a convolutional encoder in the waveform domain and how Conv-TasNet could outperform the current Time Frequency technique for speech separation can be found in Luo and Mesgarani (2018a). Tansnet model by Luo and Mesgarani (2018b) on the waveform domain matched the good performance of models like (Kolbæk et al., 2017) and (Isik et al., 2016) in speech source separation in the waveform domain. This model by Luo and Mesgarani (2018b) applies masking over a learnable front-end obtained from LSTM. The original tansnet model was trained by Lancaster and Souviraà-Labastie (2020) with 800 extra songs, about 30 times larger than MUSDB. This model performs good but fails in comparison to Conv-tansnet, Demucs, and Hybrid Demucs, even after being trained on a large data set. Using masking over a learnable front-end derived from an LSTM, Luo and Mesgarani (2018b) created Tasnet, a waveform domain method that achieved the same accuracy as cutting-edge frequency domain methods. Luo and Mesgarani improved their method by replacing the LSTM in favour of a superposition of dilated convolutions, which increased the SDR and outperformed spectrogram-based approaches like the Oracle ideal ratio mask (IRM).

Défossez et al. adapted the Conv-TasNet architecture from (Luo and Mesgarani, 2018a) originally designed for speech separation and proposed a model for Music Source Separation. Conv-Tasnet model by Luo and Mesgarani (2018a) has a receptive field of 1.5 seconds for audio sampled at 8kHZ. Conv-TasNet had a kernel size of 3, the stride of 1, and a dilation factor of 2^{nmodN} (Dilated Convolution is a method of expanding the kernel by inserting gaps between its consecutive parts). Here n is the 0-based index of residual blocks and N is a hyperparameter. However, to adapt to the MUSDB data set, Défossez et al. set N at 10 and the kernel size of the encoder-decoder is increased from 16 to 20 Luo and Mesgarani (2018c). With these modifications, Conv-Tasnet achieved cutting-edge performance on the MUSDB data set, outperforming all existing spectrogram-based algorithms with an SDR of 5.7. (Défossez et al., 2019) The original Conv-Tasnet model was designed for speech separation and short sentences. When using whole songs as input, however, there will certainly be both quiet and loud periods. Normalisation will not help as it will normalise the quiet and loud parts, resulting in a training-evaluation gap. To overcome this, Défossez et al. tested it on a whole track, breaking the input track into 8-second pieces for best results(Défossez et al., 2019). Défossez et al. did not see any adverse effects while transitioning from one chunk to the next, therefore defossez2019music did not investigate overlap-add approaches. While Conv-TasNet accurately isolated the sources, the output of the model was affected by artefacts like persistent broadband noise, hollow instrument attacks, or even missing pieces, specifically from drums and bass. Défossez et al. shows through experiments that Conv-Tasnet surpasses many existing spectrogram methods. In his experiment Défossez et al. (2019), Conv-Tasnet could achieve an SDR of 5.7. However, it does not benefit from data augmentation techniques like pitch/tempo shift. In addition, severe audio artefacts are present in its audio samples as determined by human analyses.

We found that better outcomes could still be attained using a different waveform technique, by Défossez et al. (2019) called Demucs. I wanted to research more on Demucs architecture. Défossez et al. (2019) first presented the Demucs architecture. Demucs is a U-net architecture that consists of a convolutional encoder and a decoder based on wide transposed convolutions with large strides. The network is a combination of encoder and decoder, and this is created using sets of convolutional layers. The encoder is used to reduce the dimension of the input while keeping valuable information for the job at hand. Transposed convolution layers make up the decoder. Typically, The encoder's compressed form is used by the decoder to reassemble the input. This is known as Auto-Encoder. In contrast to Conv-Tasnet, Demucs is less contaminated by outside sources.

3.1 Evaluation Metrics

It is critical to understand the model's performance to compare it to other models and to improve model performance. Subjective measures and objective measures are the two main approaches for evaluating a source separation model. In subjective measures, humans examine the model's output and judge its correctness whereas an objective measure is a mathematical approach that rates the separation quality using various mathematical computations.

(Manilow, Seetharman and Salamon, 2020) is very educational about the evaluation process of Music Source Separation. It gives a clear idea about subjective and objective measures like Source-to-Distortion Ratio (SDR), Source-to-Interference Ratio (SIR), Source-to-Artefact Ratio (SAR), and Signal-to-Noise Ratio (SNR). Evaluation criteria is discussed in detail in

section 3.1.1 and 3.1.2. These are the measures we will use to evaluate and test the models, as they are used by a large percentage of the literature found during this analysis. Both of these measures have got advantages and disadvantages of their own. The most credible way to assess the model's performance might be through the subjective metric. However, comparing a model's output against that of other models while analysing a model output over a vast corpus of music would simply result in opinions and take a great deal of time. (Raffel et al., 2014)

Based on the particular concepts discussed at the Ninth International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA 2010), Vincent et al. (2012) proposed standard evaluation criteria which are found in other papers to evaluate Music Source Separation models. This paper explained in detail the evaluation criteria and provided a comparison of existing models in these evaluation criteria.

3.1.1 Objective Measures

Vincent, Gribonval and Févotte created objective measurements for the performance of source separation models for blind source separation and were later used in (Stöter, Liutkus and Ito, 2018a). Objective measures are created to provide a well-defined method of calculating a score that shows the correctness of each system's output. These metrics usually entail a heuristically driven comparison of the model's output to ground truth. The evaluation of algorithms used to extract information from music data is at the heart of Music information retrieval (MIR) research. Museval is an open-source python library that implements these measures for evaluating MIR algorithms in a clear and user-friendly manner. These measures give a standardised approach toward source separation among the existing work and compared to subjective measures, they are much faster and cheaper to obtain. However, they struggle with aspects of human perception like the presence of hollowness in the sound, which is specifically important for Music Source Separation. Using (Vincent, Gribonval and Févotte, 2006) Vincent's notations consider a source, $j \in 1, 2, 3, 4$ and orthogonal projections $\prod\{y_1, y_2, ...y_n\}$ on subspace spanned by vectors $y_1, y_2, ..., y_k$ Now,

$$P_{s_j}(\hat{s}_j) = \prod\{s_j\} \tag{3.1}$$

$$P_{s_j}(\hat{s}_j) = \prod\{(s_{j'})_{1 \leq j' \leq n}\} \tag{3.2}$$

$$P_{s,n}(\hat{s}_j) = \prod\{(s_{j'})_{1 \leq j' \leq n}, (n_i)_{1 \leq j' \leq m}\} \tag{3.3}$$

In the above equations 3.1, 3.2, 3.3 s_j is the source we are trying to estimate, $(S_{j'})_{j'/nej}$ is the component from unwanted sources, n is the number of sources, n_i is the noise from the m sources like sensors and/or caused by forbidden distortions of the sources and/or artefacts.

An estimated source from the model $\hat{s}_j$ can be decomposed into four parts as in the equation 3.4(Vincent, Gribonval and Févotte, 2006).

$$\hat{s}_j = s_{target} + e_{interference} + e_{noise} + e_{artefacts} \tag{3.4}$$

where,

In equation 3.4 s_{target} is the true source, $e_{interference}$ is the error term for interference, e_{noise} is the error term for noise, $e_{artefacts}$ is the error term for induced artefacts. These four terms can be now used to define the following energy ratio as objective measures.

- Source-to-Distortion Ratio (SDR) is measured in decibels and gives an overall measure of the goodness of the sound. It is one of the most common measures reported in the literature in the field of source separation. The higher the value of SDR the model estimates better. The mathematical formula for SIR is in the equation 3.5

$$\text{SDR} = 10\log_{10}\left(\frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2}\right) \tag{3.5}$$

- Source-to-Interference Ratio (SIR) is also measured in decibels and it gives a measure of the presence of other sources in a source estimate. The higher the value of SIR implies less presence of other sources. In music terminology higher value of sir will have less interference from other sources. It is one of the key measures when it comes to Music Source Separation. The mathematical formula for SIR is in the equation 3.6

$$\text{SIR} = 10\log_{10}\left(\frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}}\|^2}\right) \tag{3.6}$$

- Source-to-Artefact Ratio (SAR) is also measured in decibels and it gives a measure of the presence of unwanted artefacts in the source estimate. The higher the value of SAR the better it is. The mathematical formula for SIR is in the equation 3.6

$$\text{SAR} := 10\log_{10}\left(\frac{\|s_{\text{target}} + e_{\text{interf}} + e_{\text{noise}}\|^2}{\|e_{\text{artif}}\|^2}\right) \tag{3.7}$$

3.1.2 Subjective Measures

Subjective measures involve human rating the estimates of the models. Due to the involvement of humans, ratings are subject to variability also this is a more expensive and time-consuming process with no scope of standardisation to compare the models among the literature. These measures can be more accurate to measure the quality of the models for certain aspects of the separation like hollowness in the sound as it involves a human ear. However, these measures are not so popular in the literature because of the reliability of the evaluation Manilow, Seetharman and Salamon (2020). In this project given all of the advantages and downsides of the evaluation measure, we have decided to proceed with the objective measure and compare it to baseline models in the literature.

3.2 Dataset

In recent years, Music Source Separation goal is focused mainly on separating four defined sources namely, Drums, Bass, Vocal, and others in a supervised manner. The MUSDB18 dataset consists of 150 stereophonic music tracks all encoded at 44.1kHz from diversified genres. Each track is in multi-track format, a collection of five stereo streams namely, mixture, drums, bass, vocals, and the rest of the accompaniments. MUSDB18 is an open-source dataset that can be downloaded from Rafii et al. (2019). MUSDB18 has two directories: test and train, which contains 50 and 100 music tracks respectively. As the name suggests, for training in supervised learning approaches we can use the training set and the testing set for testing the model.

3.2.1 MUSDB18

MUSDB18 is available in un-compressed and compressed formats. The uncompressed format data set is of size 22.7 gigabytes. This data set is known as MUSDB18-Hq. MUSDB18-Hq has four tracks for respective stems, bass, drums, vocals, others, and one track for the mixture. In the MUSDB18-Hq data set all of the tracks are in Waveform Audio File Format (.wav), dual channel in nature and have a bandwidth up to 22 kHz. Theoretically, the higher bandwidth helps in better prediction. However, the processing is computationally demanding while loading the data during training and testing the model. The compressed data set is of the size of 4.4 gigabytes. This data set is known as MUSDB18. It has only one compressed track per song which is multi-track in nature and composed of 5 dual audio or stereophonic streams corresponding to stems, bass, drums, vocals, others, and mixture, each one is Advanced Audio Coding (AAC) encoded at a bit rate of 256 kbps. Due to AAC encoding of mixture separately there is a small difference between the mixture and the sum of individual stems. Tracks in MUSDB18 are of Native instrument stem format(.mp4) and have bandwidth limited to 16 kHz due to compression. The comparison of tracks from both the data set is in Figure 3.3. In Figure 3.3 the Y axis represents the frequency and we can observe that in the left image MUSDB18 spectrogram bandwidth is limited to 16 kHz and it is completely dark above the 16 kHz mark. Each of the data points in the data set MUSDB18 and MUSDB18-Hq are full-length tracks which help the models learn long-term structures of music, and silence in the sources. From early experiments and literature, it can be inferred that MUSDB18 is computationally less intensive and yields similar results. The difference in bandwidth does not affect the evaluation. To compare the performance of the model with existing Music Source Separation literature we should use the MUSDB18 data set(Rafii et al., 2017). From experiments, Défossez et al. observed that the loading of compressed audio on the flow is leading to unreliable speed. During experiments in this process loading compressed audio reduced the speed by a factor of 2. To avoid this compressed audio can be converted to raw audio format and stored. The MUSDB18 data set is extracted from the music of various genres like jazz, metal, electro, etc. However, it is focused on the western musical style as all the songs are from western music and have western musical instruments.

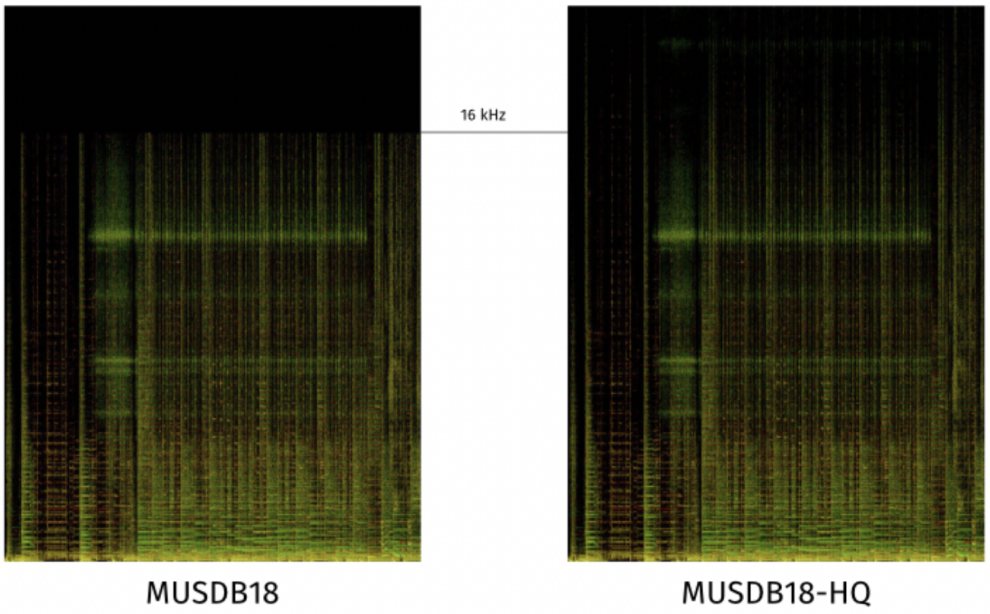

Figure 3.3: Comparison of Compressed and uncompressed. Image courtesy of Rafii et al. (2019)

3.3 Data Augmentation

Data augmentation is the technique of increasing the amount of data artificially by slightly changing current data and adding it as additional data points, hence expanding the training data. We increased the coverage of the real-world signal space by employing appropriate data augmentation techniques. When it comes to Music Source Separation, it is critical to augment the data by keeping the ground truth or changing it in a predictable fashion (Cohen-Hadria, Roebel and Peeters, 2019). When compared to traditional networks, recurrent neural networks with Bidirectional Long Short-Term Memory (BLSTM) layers have the advantage of not experiencing vanishing or exploding gradient difficulties (Uhlich et al., 2017). Demucs has more configurable weights than other models, so it always improves with new data and data augmentation by preventing overfitting and helps in the generalisation of the model Défossez et al. (2019). We have implemented the following data augmentation techniques.

- Time Shift Randomly shifting the audio signal to by the desired number of samples to the left or right as in Figure 3.4 is known as time shifting. We can augment the raw data by shifting the audio to the left or right by a specified number of samples, In this project, the default value of shift is set at 10. In spectrogram-based models, even a small shift will reflect on the phase of the spectrogram and these models estimate the mask only from the magnitude and reuse the phase directly from the input. However, in waveform-based models like Demucs, Défossez et al. noticed that it is not true. Hence we sample S random shifts of an input mixture x and average our model's predictions for each after applying the opposite shift (Défossez et al., 2019).

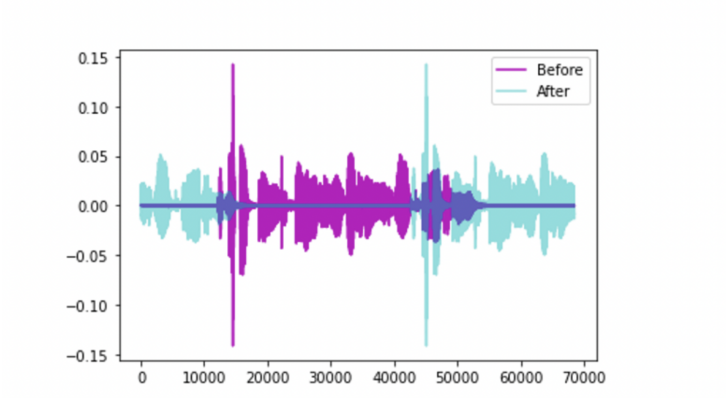

Figure 3.4: Comparison of audio waves before and after the time shift. Image courtesy of (Doshi, 2021)

- Flip Channels In this method, the channels are switched at random. In the case of a two-channel audio wave, channel 1 is substituted with channel 2 and vice versa.
- Scale As the name implies, in this method we randomly scale the amplitude of the audio sample by a factor uniformly distributed between 0.25 and 1.25. When audio is scaled by a factor less than one, the amplitude or loudness of the audio wave is reduced, and vice versa.
- FlipSign This is also referred to as polarity inversion. In this approach, the signal is turned upside down along the horizontal axis. Mathematically it is a simple operation of multiplying the signal by a negative one.
- Remix Remix: In this method, we shuffle the sources coherently within the batch to generate a new mix. Each batch is separated into groups and shuffling is performed individually inside each group to maintain the same probability distribution. Without this grouping, utilising parallelisation increases the likelihood of retaining two sources from the same track together, which might influence performance. (Défossez et al., 2019)
- Repitch and time scaling: We change the pitch of an audio input by a couple of semitones up or down in the range of -2 to +2 and change the tempo to increase or decrease the speed of the audio. The negative shift in pitch causes the audio to be pitched to a lower scale, while the positive shift causes the audio to be tuned to a higher scale.
- Shift Tricks An ideal source separation model must have symmetry over time, which means that changing the input mixture by x sample changes the output Y by the same amount. In Conv-Tans dilated convolutions with stride 1 make it symmetry over time(Luo and Mesgarani, 2018b). In the case of spectrogram models, the small shift will only impact the phase of the spectrogram. As the mask is estimated with only the magnitude of the spectrogram and the original phase is reused the output will be shifted automatically. However, Demucs naturally do not satisfy the property of time equivariant. We take S random shifts from an input mixture x and average our model's predictions for each, following the application of the opposite shift. Taking 10 random shifts gives a 0.3 gain in SDR. However, it makes the evaluation slow by S times. (Défossez et al., 2019)

- Re-sampling from (Defossez, Synnaeve and Adi, 2020) had a positive effect to deal with speech de-noising. (Défossez et al., 2019) found enhanced performance while upsampling the input by a factor of two and downsampling the output to recover the right sample rate. We use a sinc re-sampling filter Smith and Gossett (1984)to accomplish this operation as part of the end-to-end training loss.

From the study, the frequency model tends to outperform Demucs based on SDR. However, human evaluation proves that Demucs have fewer artefacts with a close SDR. Additionally, by experimentation and human evaluation, Défossez et al. (2019) proved the superior performance of Demucs architecture with respect to artefacts. In this project, we will use the SDR as the standard metric to compare it with other models as SDR is the common metric in the literature. As most of the model in the literature is trained and tested on MUSDB data set we would use the same.

Chapter 4

Model Description

4.1 Demucs Architecture

This section will discuss the architecture of Demucs. Demucs is an abbreviation meaning Deep Extractor for Music Sources. Stereo, two channels audio wave is fed as input to the Demucs model. It is composed of the convolutional encoder, a convolutional decoder following an encoder-decoder architecture along with a bidirectional LSTM in between the encoder, decoder linked with the skip U-Net connection as shown in figure 4.1. The skip U-Net is motivated from (Défossez et al., 2018). Batch normalisation is not used because early experiments in (Défossez et al., 2019) suggest that it degrades the performance of Demucs architecture. In figure 4.1 the black wavy line at the bottom represents the two-channel audio wave of the mixture as input to the model, and the magenta, yellow, red, and blue are the output audio waves of individual sources from the model. The arrows represent the U-Net skip connections.

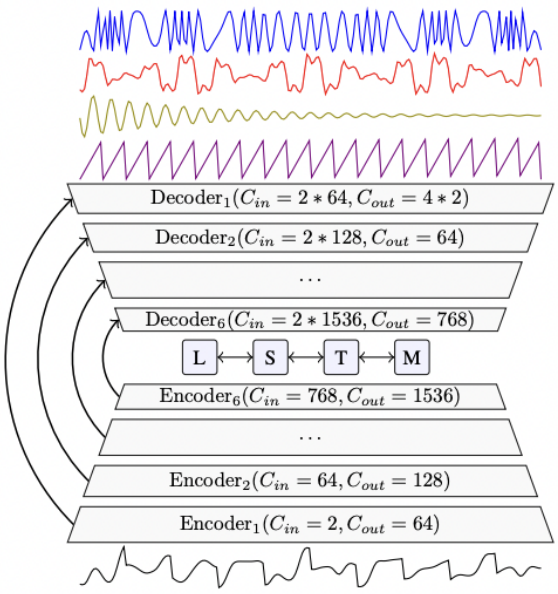

Figure 4.1: Architecture of Demucs. Image courtesy of (Défossez et al., 2019)

4.1.1 Convolutional auto-encoder

Encoder As in figure 4.1, there are 6 encoder blocks, named *Encoder*$_1$ to *Encoder*$_6$. Each of these blocks is composed of convolution with kernel (k) size 8 stride (s) 4, C_{i-1} input channels and C_i output channels and a ReLU followed by another convolution layer of the kernel (k) 1 stride (s) 1 and gated linear unit as activation function (Dauphin et al., 2017). The convolution with k 1 and s 1 makes the model more expressive and increases depth at a low computational cost(Défossez et al., 2019). Note as the gated linear unit (GLU) halves the number of channels C_i number of output channels is half of the output of 2nd convolution layer in the encoder. Figure 4.2 shows the architecture of an encoder-decoder block. The number of channels fed to the model (C_0) is 2, while C_1 is 64 and subsequently, each encoder layer doubles the number of Chanel, hence C_6 is 2048. The detail of each layer is shown in figure 4.2. The output of the 6^{th} encoder is passed through a Bidirectional LSTM with 2 layers to the decoder. As the LSTM double the channels per time position we use a linear layer to set the channel of the output of LSTM to the same as the output of C_6 and feed this into the decoder.

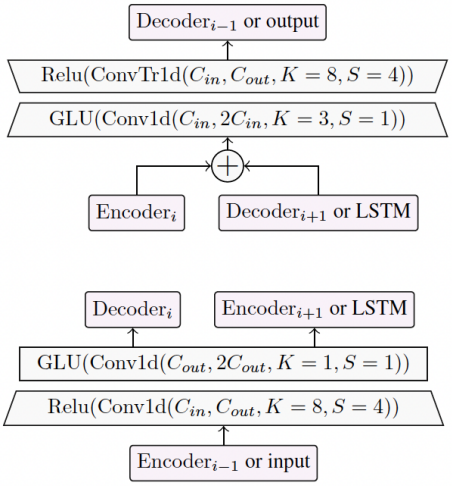

Figure 4.2: Layout of encoder and decoder. Image courtesy of (Défossez et al., 2019)

The **decoder** is functionally very similar to the inverse of the encoder and there is an equal number (6) of the decoder and encoder blocks, decoder is named *Decoder*$_6$ to *Decoder*$_1$ as in Figure 4.1. Each block begins with a convolutional layer with stride 1 and kernel size 3 as shown in 4.2. This helps to contextualise neighbouring time steps, input/output channels Ci, and a ReLU activation followed by transposed convolution layer with kernel size 8 and stride 4 except for the final layer. The final layer is a linear layer which synthesises the source without any activation function, 4 dual channels audio waveform in our case of Music Source Separation in the MUSDB data set, after all, decoder blocks. The setting of a large stride with a large number of channels is inspired from (Défossez et al., 2018).

4.2 Loss function

We train the model g which is parameterised by θ such that $g(x) = g_s(x;\theta)_{s=1}^{S}$, $g_s(x;\theta)$ is the predicted waveform for the source s for an input x an audio mixture. We aim to minimise reconstruction error. S is number of source in our case for MUSDB it is 4 Mathematically,

$$\min_{\theta} \sum_{x \in D} \sum_{s=1}^{S} L(g_s(x;\theta), x_s) \tag{4.1}$$

In equation 4.1 D is the data set MUSDB training data and L is the reconstruction error. The reconstruction loss $L(g_s(x;\theta), x_s)$ in equation 4.1 is the Mean Absolute Error (MAE). For a waveform x_s from source s and x

$_s$ an estimated source with T samples, t denoting the t^{th} sample of the source and estimate, we use MAE loss.

$$L(\hat{x}_s, x_s) = \frac{1}{T} \sum_{t=1}^{T} |\hat{x}_{s,t} - x_{s,t}| \tag{4.2}$$

In the generation task for audio on the waveform domain, direct reconstruction losses on the waveform do not reflect correctly as they are highly sensitive to the initial phase. However, in the case of source separation, the problem of a phase shift is not severe, as it is an unconditional generation and the model has access to the phase of the original signal through skip connections. From the experiments of Défossez et al. (2019). It was evident that there was no difference in performance by using MAE Mean Absolute Error(L) or Mean Square Error (MSE) hence I choose to go with MAE as they are computationally less expensive.

Chapter 5

Experiment and Results

In this section, we discuss the experimental results on the MUSDB data set and Bollywood data set described in section 5.2.1 for Demucs and Conv-tasnet. We also check the performance of these models on noisy signals and how well they generalise the model. We will perform two experiments, one with the noisy data set and the other with the Bollywood data set.

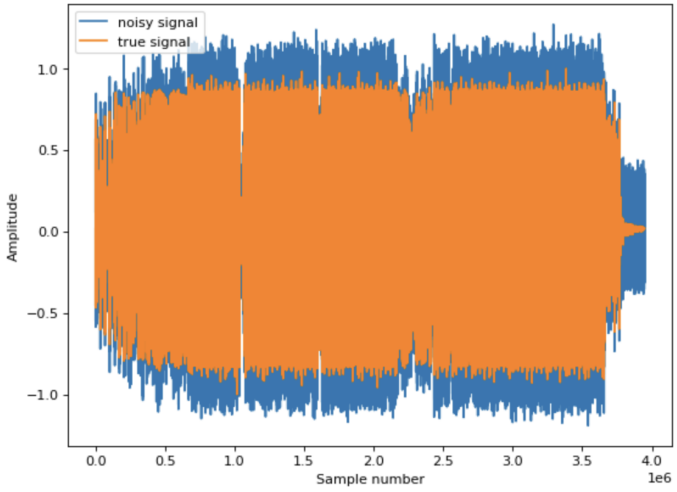

Figure 5.1: Noisy and clean mixture wave

5.1 Noisy Testing

In the field of music separation adaptation to noise is important. To understand the effects of noisy input on the model, we need a data set with a noisy mixture and use this noisy mixture as input to test the model. To generate a noisy mixture from the existing MUSDB data set we are using the python package librosa. Librosa is a Python tool that analyses music and audio. It provides various functions for making changes to the audio wave (McFee et al., 2022). For this new data set, we have used Additive White Gaussian Noise (AWGN) as our noise signal. AWGN mimics the randomness in the real world which makes it a good noise to test Music Source Separation (Wijayasingha, 2018). This noise can be introduced to the signal directly by arithmetic element-wise addition. The mean value of the noise is zero as it is sampled from a Gaussian distribution and includes all ranges of frequency in equal proportion hence a type

of white noise. In Figure 5.1 the noisy audio mixture with Signal to Noise Ration (SNR) value of 10 dB in blue and the clean mixture is in orange. However, in practical scenarios, the noise will be less. We have used the pretrained Demucs and Conv-Tasnet model by Défossez et al.. A high adverse effect of noise can be seen in sections of the audio tracks with low amplitude as evident in the spectrogram in Figure 5.1. In both Demucs and Conv-Tasnet bass has the lowest SDR whereas the highest SDR in Demucs and Conv-Tasnet are attained by others and drums respectively. Figure 5.2 and 5.3 show the comparison between the ground truth stems and output stems separated by the model from the noisy mixture.

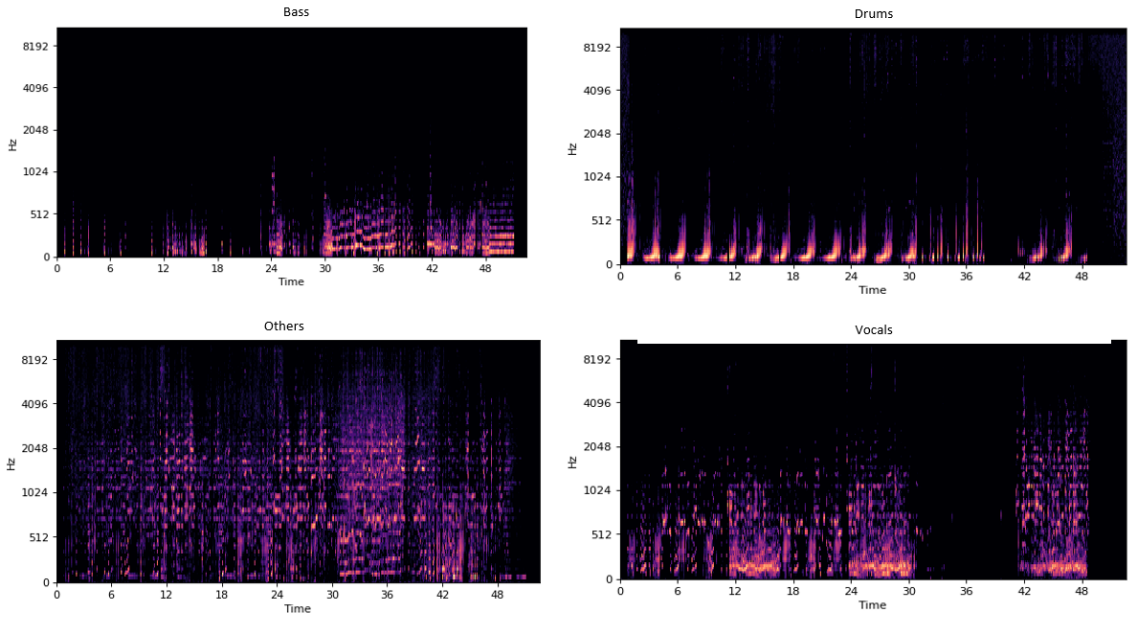

Figure 5.2: Output stems from the noisy mixture

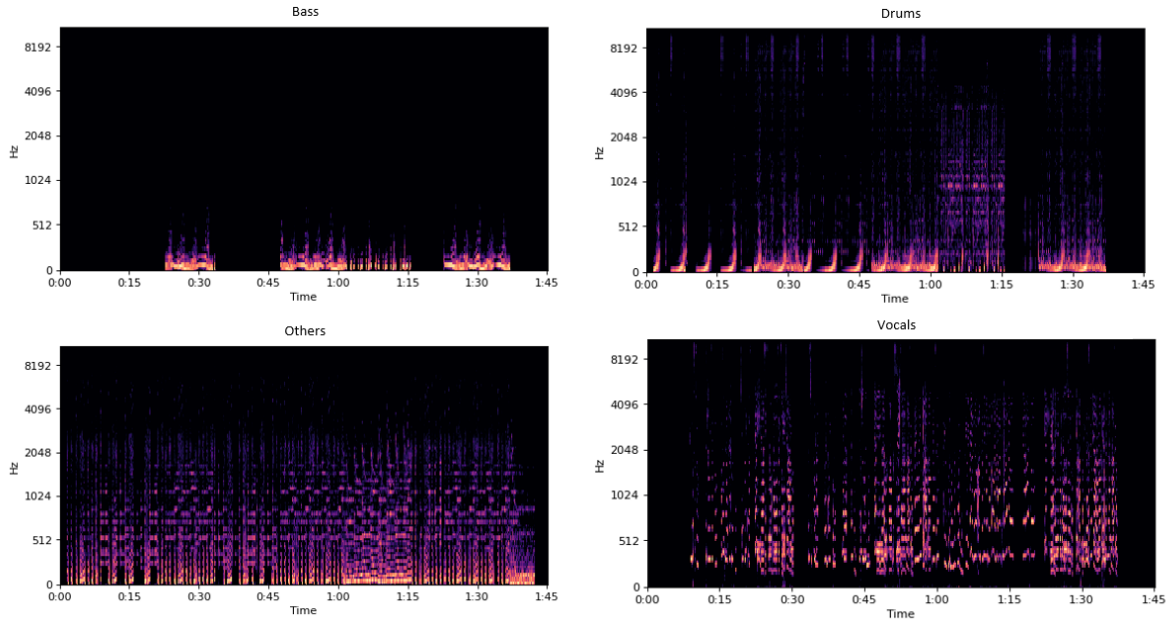

Figure 5.3: Clean ground truth stems

Table 5.1 shows the overall performance of the models when tested on noisy data set. A negative SDR value indicates that the distortion is greater than the signal. Both models have performed poor to the noisy data set. The average SDR of Demucs and Conv-Tasnet on clean data set is 6.28 and 5.73 respectively Défossez et al. (2019). From Table 5.1 we can see that Conv-Tasnet has tackled noise better than Demucs and performs better by a good margin. However, for the stem other and bass Demucs has a slight edge. As discussed in Chapter 3 Conv-Tasnet splits the track into parts of 8 seconds and during training, the quiet parts is be scaled to a mean volume. As the noise effect prediction predominantly for the quiet part Conv-Tasnet can cope better to noise.

Table 5.1: SDR for individual sources from noisy data

	SDR	
STEMS	DEMUCS	CONV-TASNET
VOCAL	-4.91	-0.34
BASS	-6.15	-7.32
OTHER	-4.24	-5.75
DRUMS	-2.69	-1.39
OVERALL	-4.50	-3.70

5.2 Generalisation Test

Music is very diverse, and the nature of composition changes from region to region. There are countless instruments in every genre of music. In the real world, Music Source Separation models should be transferable and agile to diverse sets of music. To understand the generalisation capability of the model we have we will use the Bollywood data set as described in section 5.2.1. We have used the same pretrained Demucs model on the Bollywood data set to understand the generalisation capability of Demucs architecture.

5.2.1 Bollywood dataset

To evaluate the model performance in a different genre we need a data set with tracks which have components of different musical instruments. The musical instruments used in Indian music is different from the western musical instrument in terms of frequency. (Agarwal, Karnick and Raj, 2013). Testing the model on a data set with Indian music will explain if the diversity is covered by the model. We have collected a data set from various sources. There are many genes in Indian music based on region, language, culture, etc. However, this data set is called Bollywood data set as it originated from Bollywood songs which are a mixed genre but do not cover all the genres. To retain compatibility with the model's training and testing codes, the format of the Bollywood data set has been set to the same format as the MUSDB18-Hq data set. There are twenty-five songs in the entire Bollywood data collection. Twenty of these songs are used for the training, and five for testing. The goal of the training data is to fine-tune the model by using the pooled training data from MUSDB18-Hq and Bollywood data set to train the model and the testing folder to gauge how well the models perform.

5.2.2 Test on the pretrained Demucs

In this experiment, we have again used the pretrained Demucs model by Défossez et al.. We have tested the pretrained Demucs and Conv -Tasnet model on the Bollywood data set without any fine-tuning. Conv-Tasnet was chosen for comparison because, in waveform domain, Conv-Tasnet and and Demucs have the closest overall SDR of 5.7 and 6.3, respectively. Table 5.2 it can be seen that the model has performed significantly well with an SDR score of 6.35 and hence we can conclude that model has good generalisation characteristics. The pretrained Conv-Tasnet model has a decent performance However, in comparison to Demucs, it has poor performance. The comparison of both models is shown in Table 5.2. To know more about the transfer learning capacity of the Demucs model we have fine-tuned the model on the Bollywood data set. From the Table 5.2, we can also infer that Demucs have performed better overall. However Conv-Tasnet has a better SDR score for individual stem vocals.

5.2.3 Test on pretrained Demucs with fine tuning

To further extend our experiment we fine-tuned the model with 20 new training tracks from the Bollywood data set. The performance for the model increased as in Table 5.2. Due to restrictions in computation resources, we could not fine-tune the Conv-Tasnet model for comparison. However, we can see that Demucs has improved its performance.

Table 5.2: SDR for individual sources from Unseen data

	SDR		
STEMS	DEMUCS	CONV-TASNET	DEMUCS (fine tuned)
DRUMS	7.41	7.13	8.44
BASS	6.51	3.44	7.38
OTHER	1.89	1.96	2.07
VOCALS	9.59	10.83	11.28
OVERALL	6.35	5.84	7.29

Chapter 6

Conclusion and future work

Overall we have experimented with Demucs and Conv-Tasnet, two state-of-the-art Music Source Separation models in the waveform domain. From our experiment on noisy data set, Demucs is more prone to be affected by noise whereas Conv-Tasnet handles noise better than Demucs. Form the generalisation test Demucs can generalise the unseen data and performs the same for a different genre of music on the contrary Conv-Tasnet deteriorates in performance by a small amount. Additionally, Demucs improves it performance when fine-tuned by the new data set Bollywood and responds well to data augmentation.

There are numerous potential directions for future works to be explored. To begin with, to create a GUI or a user-friendly application which can be used by musicians and content creators for Music Source Separation. Demucs architecture needs a lot of memory or computation power. It can be optimised to be less compute-intensive specifically for the evaluation metrics. Museval necessitates a substantial quantity of memory for evaluation. Evaluation of all sources can take up to 4 GB of ram and it is too slow as there is no parallelisation. For multiprocessing, the ram requirement goes up to 16 GB of RAM The current implementation of MusEval needs a lot of computing power and can be improved Stöter, Liutkus and Ito (2018b). While there are promising aspects of using waveform-based models as it responds to data augmentation better than other state-of-the-art models in the spectral domain still the performance in terms of SDR needs to improve. In this project due to limitations in computing resources, we have trained our model to 30 epochs only as it needs about 4 hrs to train for 1 epoch. The natural extension would be to train the model for a higher number of epochs and save the model at the convergence point to explore its full potential of the model. In our experiments, we have used Additive White Gaussian Noise and the input signal has an SNR of 10, further extension to this can be experimenting with different types of noise at different levels. Additionally, (Défossez, 2021) discovered improvements to the source separation model by integrating a hybrid temporal and frequency domain model in the U-Net architecture. This modification resulted in better quality results, with a large reduction in artefacts, as well as significant increases in overall quality and the absence of interference between sources (Défossez, 2021). Finally, deep learning techniques such as Generative Adversarial Networks (GANs) a generative modelling methodology that employs deep learning approaches such as cnn should be tried. Subakan and Smaragdis has demonstrated by experiment that Wasserstein GANs can perform well in generative source separation. GANs has research opportunity in music source separation because they assess the likelihood of the sample from the real dataset, which aids in source separation.

Bibliography

Agarwal, P., Karnick, H. and Raj, B., 2013. A comparative study of indian and western music forms. *Ismir*. pp.29–34.

Balen, J.V., 2020. What a wav file looks like [Online]. [Online]. Available from: `https://jvbalen.github.io/notes/waveform.html`.

Benetos, E., Dixon, S., Duan, Z. and Ewert, S., 2018. Automatic music transcription: An overview. *Ieee signal processing magazine*, 36(1), pp.20–30.

Cano, E., FitzGerald, D., Liutkus, A., Plumbley, M.D. and Stöter, F.R., 2018. Musical source separation: An introduction. *Ieee signal processing magazine*, 36(1), pp.31–40.

Casey, M.A., Veltkamp, R., Goto, M., Leman, M., Rhodes, C. and Slaney, M., 2008. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the ieee* [Online], 96(4), pp.668–696. Available from: `https://doi.org/10.1109/JPROC.2008.916370`.

Chaney, D., 2012. The music industry in the digital age: Consumer participation in value creation. *International journal of arts management*, 15(1).

Cherry, E.C., 1953. Some experiments on the recognition of speech, with one and with two ears. *The journal of the acoustical society of america*, 25(5), pp.975–979.

Cohen-Hadria, A., Roebel, A. and Peeters, G., 2019. Improving singing voice separation using deep u-net and wave-u-net with data augmentation. *2019 27th european signal processing conference (eusipco)*. IEEE, pp.1–5.

Dauphin, Y.N., Fan, A., Auli, M. and Grangier, D., 2017. Language modeling with gated convolutional networks [Online]. In: D. Precup and Y.W. Teh, eds. *Proceedings of the 34th international conference on machine learning*. PMLR, *Proceedings of Machine Learning Research*, vol. 70, pp.933–941. Available from: `https://proceedings.mlr.press/v70/dauphin17a.html`.

Défossez, A., 2021. Hybrid spectrogram and waveform source separation. *Proceedings of the ismir 2021 workshop on music source separation*.

Defossez, A., Synnaeve, G. and Adi, Y., 2020. Real time speech enhancement in the waveform domain. *arxiv preprint arxiv:2006.12847*.

Défossez, A., Usunier, N., Bottou, L. and Bach, F., 2019. Music source separation in the waveform domain. *arxiv preprint arxiv:1911.13254* [Online]. Available from: `https://arxiv.org/pdf/1911.13254.pdf`.

Défossez, A., Zeghidour, N., Usunier, N., Bottou, L. and Bach, F., 2018. Sing: Symbol-to-instrument neural generator. *Advances in neural information processing systems*, 31.

Doshi, K., 2021. Audio deep learning made simple (part 3): Data preparation and augmentation [Online]. Available from: `https://towardsdatascience.com/audio-deep-learning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b5`

Dumoulin, V. and Visin, F., 2016. A guide to convolution arithmetic for deep learning. *Arxiv e-prints*. 1603.07285.

Gateau, T., 2014. The role of open licences and free music in value co-creation: the case of misteur valaire. *International journal of arts management*, 16(3), p.49.

Grais, E.M., Sen, M.U. and Erdogan, H., 2014. Deep neural networks for single channel source separation [Online]. *2014 ieee international conference on acoustics, speech and signal processing (icassp)*. pp.3734–3738. Available from: `https://doi.org/10.1109/ICASSP.2014.6854299`.

Isik, Y., Roux, J.L., Chen, Z., Watanabe, S. and Hershey, J.R., 2016. Single-channel multi-speaker separation using deep clustering. *arxiv preprint arxiv:1607.02173*.

Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A. and Weyde, T., 2017. Singing voice separation with deep u-net convolutional networks.

Kolbæk, M., Yu, D., Tan, Z.H. and Jensen, J., 2017. Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks. *Ieee/acm transactions on audio, speech, and language processing* [Online], 25(10), pp.1901–1913. Available from: `https://doi.org/10.1109/TASLP.2017.2726762`.

Lancaster, E.P. and Souviraà-Labastie, N., 2020. A frugal approach to music source separation.

Lee, J.H., Choi, H.S. and Lee, K., 2019. Audio query-based music source separation. *arxiv preprint arxiv:1908.06593*.

Liu, J.Y. and Yang, Y.H., 2018. Denoising auto-encoder with recurrent skip connections and residual regression for music source separation. *2018 17th ieee international conference on machine learning and applications (icmla)*. IEEE, pp.773–778.

Lluís, F., Pons, J. and Serra, X., 2018. End-to-end music source separation: is it possible in the waveform domain? *arxiv preprint arxiv:1810.12187*.

Luo, Y. and Mesgarani, N., 2018a. Tasnet: Time-domain audio separation network for real-time, single-channel speech separation [Online]. *2018 ieee international conference on acoustics, speech and signal processing (icassp)*. pp.696–700. Available from: `https://doi.org/10.1109/ICASSP.2018.8462116`.

Luo, Y. and Mesgarani, N., 2018b. Tasnet: time-domain audio separation network for real-time, single-channel speech separation. *2018 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, pp.696–700.

Luo, Y. and Mesgarani, N., 2018c. Tasnet: Time-domain audio separation network for real-time, single-channel speech separation [Online]. *2018 ieee international conference on acoustics, speech and signal processing (icassp)*. pp.696–700. Available from: `https://doi.org/10.1109/ICASSP.2018.8462116`.

Luo, Y. and Mesgarani, N., 2019. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *Ieee/acm transactions on audio, speech, and language processing*, 27(8), pp.1256–1266.

Manilow, E., Seetharaman, P. and Pardo, B., 2020. Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments. *Icassp 2020-2020 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, pp.771–775.

Manilow, E., Seetharman, P. and Salamon, J., 2020. *Open source tools & data for music source separation* [Online]. https://source-separation.github.io/tutorial. Available from: `https://source-separation.github.io/tutorial`.

McFee, B., Metsai, A., McVicar, M., Balke, S., Thomé, C., Raffel, C., Zalkow, F., Malek, A., Dana, Lee, K., Nieto, O., Ellis, D., Mason, J., Battenberg, E., Seyfarth, S., Yamamoto, R., viktorandreevichmorozov, Choi, K., Moore, J., Bittner, R., Hidaka, S., Wei, Z., nullmightybofo, Weiss, A., Hereñú, D., Stöter, F.R., Nickel, L., Friesch, P., Vollrath, M. and Kim, T., 2022. *librosa/librosa: 0.9.2* (v.0.9.2). Zenodo. Available from: `https://doi.org/10.5281/zenodo.6759664`.

Mesaros, A. and Virtanen, T., 2010. Automatic recognition of lyrics in singing. *Eurasip journal on audio, speech, and music processing*, 2010, pp.1–11.

Perrin, A., 2015. Social media usage. *Pew research center*, 125, pp.52–68.

Raffel, C., McFee, B., Humphrey, E.J., Salamon, J., Nieto, O., Liang, D., Ellis, D.P. and Raffel, C.C., 2014. mir_eval: A transparent implementation of common mir metrics [Online]. *In proceedings of the 15th international society for music information retrieval conference, ismir*. Citeseer. Available from: `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.722.1267`.

Rafii, Z., Liutkus, A., Stöter, F.R., Mimilakis, S.I. and Bittner, R., 2017. The MUSDB18 corpus for music separation [Online]. Available from: `https://doi.org/10.5281/zenodo.1117372`.

Rafii, Z., Liutkus, A., Stöter, F.R., Mimilakis, S.I. and Bittner, R., 2019. Musdb18-hq - an uncompressed version of musdb18 [Online]. Available from: `https://doi.org/10.5281/zenodo.3338373`.

Ronneberger, O., Fischer, P. and Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: N. Navab, J. Hornegger, W.M. Wells and A.F. Frangi, eds. *Medical image computing and computer-assisted intervention – miccai 2015*. Cham: Springer International Publishing, pp.234–241.

Sibi, P., Jones, S.A. and Siddarth, P., 2013. Analysis of different activation functions using back propagation neural networks. *Journal of theoretical and applied information technology*, 47(3), pp.1264–1268.

Smith, J. and Gossett, P., 1984. A flexible sampling-rate conversion method [Online]. *Icassp '84. ieee international conference on acoustics, speech, and signal processing*. vol. 9, pp.112–115. Available from: `https://doi.org/10.1109/ICASSP.1984.1172555`.

Stoller, D., Ewert, S. and Dixon, S., 2018. Wave-u-net: A multi-scale neural network for end-to-end audio source separation. *arxiv preprint arxiv:1806.03185*.

Stöter, F.R., Liutkus, A. and Ito, N., 2018a. The 2018 signal separation evaluation campaign [Online]. *International conference on latent variable analysis and signal separation*. Springer, pp.293–305. Available from: `https://arxiv.org/abs/1804.06267`.

Stöter, F.R., Liutkus, A. and Ito, N., 2018b. The 2018 signal separation evaluation campaign. *Latent variable analysis and signal separation: 14th international conference, lva/ica 2018, surrey, uk*. pp.293–305.

Stöter, F.R., Uhlich, S., Liutkus, A. and Mitsufuji, Y., 2019. Open-unmix-a reference implementation for music source separation. *Journal of open source software*, 4(41), p.1667.

Subakan, Y.C. and Smaragdis, P., 2018. Generative adversarial source separation [Online]. *2018 ieee international conference on acoustics, speech and signal processing (icassp)*. pp.26–30. Available from: `https://doi.org/10.1109/ICASSP.2018.8461671`.

Takahashi, N., Goswami, N. and Mitsufuji, Y., 2018. Mmdenselstm: An efficient combination of convolutional and recurrent neural networks for audio source separation. *2018 16th international workshop on acoustic signal enhancement (iwaenc)*. IEEE, pp.106–110.

Takahashi, N. and Mitsufuji, Y., 2020. D3net: Densely connected multidilated densenet for music source separation. *arxiv preprint arxiv:2010.01733*.

Tappert, C.C., 2019. Who is the father of deep learning? [Online]. *2019 international conference on computational science and computational intelligence (csci)*. pp.343–348. Available from: `https://doi.org/10.1109/CSCI49370.2019.00067`.

Uhlich, S., Giron, F. and Mitsufuji, Y., 2015. Deep neural network based instrument extraction from music. *2015 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, pp.2135–2139.

Uhlich, S., Porcu, M., Giron, F., Enenkl, M., Kemp, T., Takahashi, N. and Mitsufuji, Y., 2017. Improving music source separation based on deep neural networks through data augmentation and network blending. *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, pp.261–265.

Vincent, E., Araki, S., Theis, F., Nolte, G., Bofill, P., Sawada, H., Ozerov, A., Gowreesunker, V., Lutter, D. and Duong, N.Q., 2012. The signal separation evaluation campaign (2007–2010): Achievements and remaining challenges. *Signal processing*, 92(8), pp.1928–1936.

Vincent, E., Gribonval, R. and Févotte, C., 2006. Performance measurement in blind audio source separation. *Ieee transactions on audio, speech, and language processing*, 14(4), pp.1462–1469.

Wijayasingha, L.N., 2018. Adding noise to audio clips.

Appendix A

Code for Spectrogram

```
def spectogram(path):
    #path to the wav file
    audio_data_array, sample_rate = librosa.load(path)
    signal = nussl.AudioSignal(audio_data_array=audio_data_array, sample_rate=sample_rate)
    plt.figure(figsize=(8, 4), dpi=80)
    nussl.utils.visualize_spectrogram(signal, y_axis='mel')
    plt.show()
```