International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 03, March-2021

Audio Stems Separation using Deep Learning

Deepak Kharah, Dhruv Parekh, Kirtesh Suthar
UG Student,
Dept. of Information Technology
Vidyavardhini's College of Engineering and Technology,
Vasai, India

Prof. Vaishali Shirsath
Assistant Professor,
Dept. of Information Technology
Vidyavardhini's College of Engineering and Technology,
Vasai, India

***Abstract-* Music recordings are usually a mix of several individual instrument tracks. Retrieving Stems from a piece of music is a common need in the professional music industry and music signal researchers. The task of music source separation is: given a mix can we recover these separate tracks. This stem separation has many potential applications like remixes, active listening, educational purposes, and also pre-processing for other tasks such as transcription. The manual process of stem separation is both expensive and time-consuming. There are attempts to automate the Stem separation process to reduce the hassle, but the results were not very promising. The pace in the recent development of machine learning and deep learning brings new approaches to the table. This helps us make some giant leaps in the stem separation approaches that will help us construct more cleaner, distortion-free stems out of audio signals.**

Keywords— S*tem separation, audio separation, music information retrieval, deep learning*

I. INTRODUCTION

Stems are unit of an audio signal. When we mix these stems, it produces complex and harmonized sounds. Humans are excellent in isolating these audio signals and process only the sounds we need. With the advancement of technology, it is possible to record or create sounds that we can hear later. Now, various sectors deal with manipulation and study of the sound signals that require stem. Hence, we need algorithms that can separate audio signal effectively. Since we can pick up little inconsistencies in the sound, the stem separation must be clear and crisp as possible. Hence, we would harness the capabilities of deep learning, masking and regeneration to recreate individual stems.

II. RELATED WORK

In the first paper, they have attempted the problem of identifying the instrumentation of a music signal at any given time using several machine learning techniques (logistic regression, K-NN, SVM).They approached the problem as a series of separate binary classifications (as opposed to a multivariate problem) so that we could mix and match the best algorithm for each instrument to create the best overall classifier. They used examples from multiple recordings in order to create a more robust system. An instrument will have many unique features on a given recording. Every individual instrument has a unique character (due to materials. construction process, body shape, age, etc.), and every individual performer creates a different sound with the instrument. There are also many different playing techniques, and then microphone choice and positioning as well as digital effects and equalization will all create significant variations between recordings. So, in order to effectively analyze an arbitrary recording, they needed to train their classifiers with multiple examples. The data was first segregated into frames of 1024 samples each (23 ms). This frame size was selected for its wide use in speech processing applications. Once divided, we chose to describe each frame with three types of features, which were decided based on acoustic knowledge of the instruments:

- Magnitudes of the Discrete Fourier Transform (DFT)
- Mel Frequency Cepstral Coefficients (MFCCs)
- Change in energy from frame to frame

Algorithms used:

- Logistic Regression
- K-Nearest Neighbors (K-NN)
- Support Vector Machine (SVM) with Linear Kernel
- SVM with Gaussian Kernel

Their resultant accuracy was just under 80% for 2-instrument case while for 3-instrument case it dropped to 52% and it dropped further for 5-instrument case to around 42%. But this result was not satisfactory and a better improved algorithm was required for better accuracy. Also, rich set of feature set needs to be setup for better accuracy and also more powerful computer [1].

We managed to overview the latest datasets that we could use in our implementation that are rich in sample collections of different musical instruments. Our approach is a first step in determining the effectiveness of NSynth with the end goal of live instrument detection for an entire piece of music or for a point in time. There is potential to detect new instruments in music as well as non-traditional instruments based on sound combinations and we might even be able to identify and predict notes in music from a recording [2].

We learned how to use Fourier transform. An audio signal is a complex signal composed of multiple 'single-frequency sound waves' which travel together as a disturbance (pressure-change) in the medium. When sound is recorded, we only capture the resultant amplitudes of those multiple waves. Fourier Transform is a mathematical concept that can decompose a signal into its constituent frequencies. Performing deep learning on image samples is easier and hence we used Fourier transform to portray important and characteristic information in image form. Using Fast Fourier transform we can separate our different frequency signals i.e., different instrument/vocal and then regenerate separate audio files using inverse-Fourier transform. They have proposed a new multi-channel audio source separation method based on separating the waveform directly in the time-domain without extracting any hand-crafted features. We introduced a novel multi-resolution convolutional auto-encoder neural network to separate the stereo waveforms of the target sources from the input stereo mixed signals. Their experimental results show that the proposed approach is very promising. In future work we will investigate combining the multi-resolution concept with generative adversarial neural networks (GANs) for waveform audio source separation [3].

III. METHODOLOGY

Sound as a data is data in time series. The time-domain representation of a signal is a visual representation that demonstrates how the loudness (amplitude) of a sound wave changes over time.

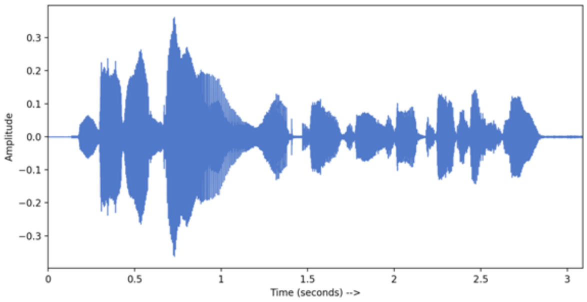


Fig. 1. Sound amplitude representation

These amplitudes aren't really useful since they just refer to the volume of an audio recording. It is important to convert the audio signal into the frequency domain in order to better understand it. The frequency-domain representation of an audio signal reveals the various frequencies present in the signal. So, in comes Fourier Transform [4].

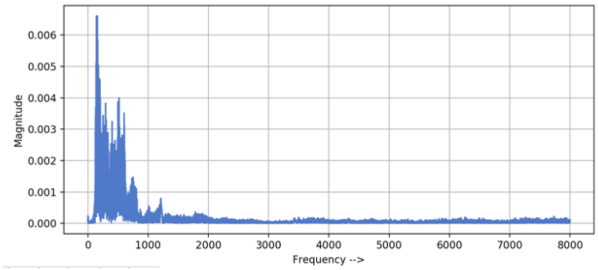


Fig. 2. Sound frequency domain representation

The Fourier Transform is a mathematical phenomenon that allows you to break down a signal into its individual frequencies. The Fourier transform determines the frequencies present in the audio signal as well as the magnitude of each frequency [5].

Assume we're developing a speech recognition system. We have a narration audio file (for example: How are you). These three terms should be predicted in the same order by our recognition scheme. We've decomposed the audio signal into its frequency values. We'll use this frequency value as a function in our recognition scheme. However, converting the audio signal to frequency domain removes the previously present time information. As a result, our machine is unable to determine the order in which the audio signals arrived. As a result, we must devise a new method of calculating characteristic for our system so that it has frequency values that correspond to the time observed. And this is where spectrograms come into play [6]. Our plan is to divide the audio signal into smaller frames (windows) and measure the FT for each one. By monitoring the order of the windows, we now have frequency characteristics as well as time information. Window one appears first, followed by window two, and so on.

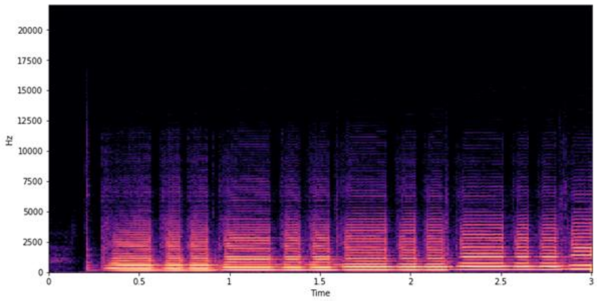


Fig. 3. Phase spectrogram

We discovered that almost all of the critical frequency data is below 12,500 Hz in this spectrogram. As a result, this does not reflect how humans interpret frequencies. We sense frequencies on a logarithmic scale in addition to loudness. We'd listen to the same frequency range between 50 and 100 Hz as we would between 400 and 800 Hz. The difference between 500 and 1000 Hz is obvious, while that between 7500 and 8000 Hz is barely discernible [7].

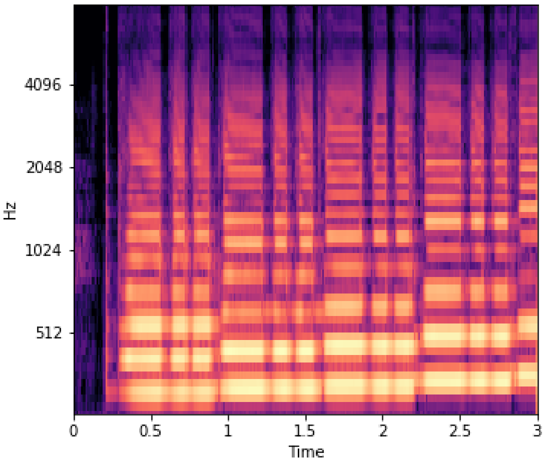


Fig. 4. Mel Spectrogram

In mathematical terms, the Mel Scale is the result of a non-linear transformation of the frequency scale. In comparison to the Hz scale, the Mel Scale is composed in such a way that the scale of loudness is close to how humans interpret it. We now know what a Spectrogram and Mel Scale are. Hence, by that knowledge, a Mel Spectrogram is a spectrogram with Mel scale in the vertical axis. Consider this spectrogram to be an image. We've turned our audio file into an image. As a result, we've narrowed it down to an image classification problem. Although it is not yet possible to classify raw audio waveform data, image classifiers are commonly used against Mel spectrograms, and they work very well. To use this method, we must first convert our entire dataset to image files.

Artificial neural networks are the first thing that comes to mind when we talk about deep learning. Let's pretend we're using an Artificial Neural Network. As a result, this approach simulates timbre features over several time frames. That is, they do not take advantage of local time-frequency functionality. Rather, they depend on global characteristics that span the entire frequency range. CNNs use less memory and resources than fully linked neural networks, making the model faster and more efficient [8].

In the field of image processing, CNNs accept a two-dimensional vector of pixel intensities across the spatial dimension and learn localised features by exploiting local spatial correlation among input neurons. In our model for the audio signal, we use a two-dimensional representation called the Short-Time Fourier Transform (STFT), which has frequency and time dimensions. As a result, the filters will adapt to the FT representation of audio using CNNs [9].

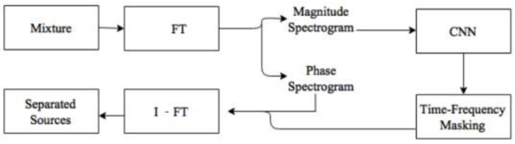


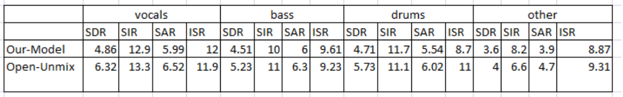
Fig. 5. Block diagram

To sum up the procedure, STFT was used to create a phase spectrogram and a Mel spectrogram from input audio. After that, an optimal filter is used to mask the Mel spectrogram. A CNN was used to build this filter. Now, we will generate a new stem boosted spectrogram image.

When it came to training the model, we first considered separating the four stems (vocals, bass, drums, and other). We used U-net, which is a skip-connected encoder/decoder Convolutional Neural Network architecture. We used U-nets with 12 layers (6 layers for the encoder and 6 for the decoder). For each source, a U-net was used to estimate a soft mask (stem). Between masked input mix spectrograms and source-target spectrograms, training loss is an L1-norm. The model was trained using the Musdb18 dataset. The musdb18 dataset contains 150 full-length music tracks (10 hours in length) from various genres, along with isolated drums, bass, vocals, and other stems. On a single GPU, training took approximately a week. Soft masking Wiener filtering is then used to separate the approximate source spectrograms. Tensorflow was used to perform the training and inference. It enables the code to run on a Central Processing Unit (CPU) or a Graphics Processing Unit (GPU).

IV. RESULTS AND DISCUSSION

The model inference is fast since the entire separation pipeline can run on a GPU and the model is based on a CNN (which allows for very efficient computation parallelization). For example, our model can divide the entire musdb18 test dataset (approximately 3 hours and 27 minutes of audio) into four stems in under 6 minutes, including model loading time (approximately 55 seconds) and audio.wav file export. Following are the comparisons of our model with some other models:



	vocals				bass				drums				other			
	SDR	SIR	SAR	ISR	SDR	SIR	SAR	ISR	SDR	SIR	SAR	ISR	SDR	SIR	SAR	ISR
Our-Model	4.86	12.9	5.99	12	4.51	10	6	9.61	4.71	11.7	5.54	8.7	3.6	8.2	3.9	8.87
Open-Unmix	6.32	13.3	6.52	11.9	5.23	11	6.3	9.23	5.73	11.1	6.02	11	4	6.6	4.7	9.31

Fig. 6. Results Comparison

We present our findings using standard source separation metrics. Signal to Distortion Ratio (SDR), Signal to Artifacts Ratio (SAR), Signal to Interference Ratio (SIR), and source Image to Spatial Distortion Ratio were the parameters used (ISR). We compared the findings to Open-Unmix, which is the only publicly available system that, to the authors' knowledge, performs well. Soft masking and multichannel Wiener filtering results are presented (applied using Norbert). As can be seen, our model is competitive with Open-Unmix for the majority of the metrics, and particularly on SDR for all instruments.

V. CONCLUSION

So, after conducting literature surveys and gathering the required details, we were able to design the process workflow. Our model also appears to have shown competitive results. A prototype implementation of this also shows good results when trying to perform stem separation. We are also considering exploring more applications of this technology as a future goal.

REFERENCES

[1] Keunwoo Choi, György Fazekas, Kyunghyun Cho, Mark Sandler, a Tutorial on Deep Learning for Music Information Retrieval, arXiv preprint arXiv:1709.04396v2, 2018.

[2] Greg Sell Gautham J. Mysore Song Hui Chon, Musical Instrument Detection, arXiv preprint arXiv:6323.08463, 2006.

[3] Brandi Frisbie, Music instrument detection using LSTMs and the Nsynth Dataset, arXiv preprint arXix:8446.63362, 2017.

[4] Emad M. Grais, Dominic Ward, and Mark D. Plumbley, Raw Multi-Channel Audio Source Separation using Multi-Resolution Convolutional Auto-Encoders, arXiv:1803.00702v1, 2018.

[5] Juan J Bosch, Jordi Janer, Ferdinand Fuhrmann, and Perfecto Herrera. A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals. In *ISMIR*, pages 559–564, 2012.

[6] Ferdinand Fuhrmann et al. Automatic musical instrument recognition from polyphonic music audio signals. PhD thesis, *Universitat Pompeu Fabra*, 2012.

[7] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[8] Has¸im Sak, Andrew Senior, and Franc¸oise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv preprint arXiv:1402.1128*, 2014.

[9] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural audio synthesis of musical notes with wavenet autoencoders. arXiv preprint arXiv:1704.01279, 2017.