

AN AUDIO STEM EXTRACTOR

Sujay Aniruth P V¹, Syleshkumar N J², Jai Krishnan A³, Yamuna S⁴

¹ *Sujay Aniruth P V Student, Information Technology, Bannari Amman Institute of Technology,
Tamil Nadu, India*

² *Syleshkumar N J Student, Information Technology, Bannari Amman Institute of Technology,
Tamil Nadu, India*

³ *Jai Krishnan A Student, Computer Science Engineering, Bannari Amman Institute of Technology,
Tamil Nadu, India*

⁴ *Yamuna S Assistant Professor II, Information Technology, Bannari Amman Institute of Technology,
Tamil Nadu, India*

ABSTRACT

This project introduces Demucs (Deep Extractor for Music Sources), a cutting-edge music source separation model, particularly focusing on drums, bass, and vocals extraction from complex music mixtures. Demucs leverages a U-Net convolutional architecture inspired by Wave-U-Net, incorporating state-of-the-art techniques in deep learning. The latest version, Demucs v4, introduces the Hybrid Transformer Demucs, a novel approach employing a hybrid spectrogram/waveform separation model using Transformers. This model features a dual U-Net structure with a cross-domain Transformer, achieving a Signal-to-Distortion Ratio (SDR) of 9.00 dB on the MUSDB HQ test set. Additionally, by using sparse attention kernels and per-source fine-tuning, a state-of-the-art SDR of 9.20 dB is attained. The paper provides comprehensive insights into the architecture, training methodology, and performance evaluation of Demucs. It discusses model comparisons, system requirements, and practical instructions for using Demucs for music separation tasks. The release notes highlight significant updates and additions, including support for the SDX 2023 Challenge and integration with torchaudio. Moreover, the paper offers guidelines for training Demucs models and reproducing results from the MDX Challenge. To facilitate adoption, Demucs is made available as a Python package, compatible with various operating systems and environments. The paper concludes with citations for proper attribution and licensing information. Demucs represents a significant advancement in music source separation, offering researchers and practitioners a powerful tool for audio processing and analysis tasks.

Keyword: - Demucs, Music source separation, U-Net convolutional architecture, Hybrid Transformer Demucs, Signal-to-Distortion Ratio (SDR)

1. LITRATURE REVIEW

[1]. **Jansson et al.**, "Singing Voice Separation with Deep U-Net Convolutional Networks": This paper introduces a method for separating singing voice from music accompaniment using deep U-Net convolutional networks.

[2]. **Takahashi et al.**, "Multi-scale Multi-band DenseNets for Audio Source Separation": The authors propose a technique for audio source separation using multi-scale and multi-band DenseNets, a type of neural network architecture known for its dense connections between layers.

[3]. **Takahashi et al.**, "MMDENSELSTM: AN EFFICIENT COMBINATION OF CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS FOR AUDIO SOURCE SEPARATION": This paper presents MMDenseLSTM, which combines convolutional and recurrent neural networks for efficient audio source separation.

[4]. **Choi et al.**, "PHASE-AWARE SPEECH ENHANCEMENT WITH DEEP COMPLEX U-NET": The authors introduce a method for speech enhancement using a deep complex U-Net architecture, which considers both magnitude and phase information in the audio signal.

[5]. **Jansson et al.**, "Learned complex masks for multi-instrument source separation": This paper proposes a technique for multi-instrument source separation using learned complex masks, which capture both magnitude and phase information of audio signals.

[6]. **Liutkus et al.**, "The 2016 Signal Separation Evaluation Campaign": This work presents an evaluation campaign focused on signal separation techniques, likely providing benchmarks and datasets for assessing the performance of various separation algorithms.

[7]. **Takahashi et al.**, "Multi-scale Multi-band DenseNets for Audio Source Separation": This appears to be a duplicate reference to item 2.

2. INTRODUCTION

In an era where music production and audio manipulation are central to various creative endeavors, the quest for tools that empower users to extract, manipulate, and reimagine audio content has never been more pronounced. Among these tools, the Audio Stem Extractor, with a primary focus on vocal removal, emerges as a pivotal solution, catering to the evolving needs of musicians, producers, and audio enthusiasts alike.

The Audio Stem Extractor project represents an innovative endeavor aimed at harnessing the power of advanced signal processing techniques, particularly leveraging the capabilities of deep learning, to facilitate the separation of vocals from mixed audio recordings. By delving into the intricate nuances of sound waves and spectral analysis, this project seeks to unlock new possibilities in audio post-production, remixing, and content creation.

At its core, the Audio Stem Extractor is driven by a fundamental aspiration: to provide users with a seamless and intuitive toolset for isolating vocals from music tracks, thereby enabling unprecedented levels of creativity and experimentation. Whether it's the desire to craft personalized karaoke versions, explore new remixing possibilities, or enhance audio clarity in video productions, the capabilities of the Vocal Remover module within the Audio Stem Extractor project stand poised to revolutionize the audio processing landscape.

With a keen emphasis on both technical prowess and user-centric design, the project endeavors to deliver a robust and versatile solution that transcends conventional boundaries. By integrating cutting-edge algorithms, curated datasets, and intuitive user interfaces, the Audio Stem Extractor aims to democratize the art of audio manipulation, making complex tasks accessible to novices and seasoned professionals alike.

Moreover, the Audio Stem Extractor project embodies a commitment to continual innovation and refinement, driven by a relentless pursuit of excellence and a deep-seated passion for the transformative potential of sound. Through ongoing research, collaboration, and community engagement, the project seeks to evolve in tandem with the dynamic needs and aspirations of its user base, forging new frontiers in the realm of audio engineering and creativity.

In essence, the Audio Stem Extractor project represents not only a technical endeavor but a testament to the boundless possibilities inherent in the fusion of technology and artistic expression. As it endeavors to redefine the boundaries of audio manipulation and empower creators worldwide, the project stands as a beacon of innovation, ingenuity, and sonic exploration in an ever-evolving digital landscape.

3. OBJECTIVES AND METHODOLOGY

3.1.OBJECTIVES:

3.1.1. Advance Audio Source Separation Techniques: The primary objective of the project is to contribute to the advancement of audio source separation techniques by exploring deep learning methodologies and related approaches. This involves improving the accuracy, efficiency, and robustness of source separation algorithms.

3.1.2. Develop Novel Architectures: To design and implement novel deep learning architectures tailored specifically for audio source separation tasks. This includes investigating various network architectures such as U-Net, DenseNets, and combinations thereof to achieve optimal separation performance.

3.1.3. Address Real-World Challenges: To address real-world challenges encountered in audio source separation, such as overlapping sources, reverberation, background noise, and variations in audio quality.

3.1.4. Evaluate Performance: To evaluate the performance of developed algorithms using standardized datasets and evaluation metrics, comparing against existing state-of-the-art methods. This involves assessing metrics such as Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), and perceptual evaluation of audio quality.

3.1.5. Contribute to Audio Processing Applications: To contribute insights and methodologies that can be applied in various audio processing applications, including music production, speech enhancement, sound recognition, and related fields.

3.2. METHODOLOGY:

3.2.1. Literature Review: Conduct an extensive review of existing literature, including research papers, conference proceedings, and relevant publications, to understand the current state-of-the-art techniques in audio source separation.

3.2.2. Data Collection and Preprocessing: Gather appropriate audio datasets for training, validation, and testing purposes. Preprocess the audio data by standardizing formats, handling missing values, and augmenting the dataset to improve model generalization.

3.2.3. Model Architecture Design: Design and implement deep learning architectures suitable for audio source separation tasks. Experiment with variations of U-Net, DenseNets, and hybrid architectures to determine the most effective model configurations.

3.2.4. Training and Validation: Train the developed models using the collected dataset, optimizing model parameters using techniques such as stochastic gradient descent, Adam optimization, and learning rate scheduling. Validate the trained models using separate validation datasets to assess generalization performance.

3.2.5. Performance Evaluation: Evaluate the performance of trained models using standardized evaluation metrics such as SDR, SIR, and perceptual evaluation of audio quality. Compare the performance of developed models against baseline methods and state-of-the-art approaches.

3.2.6. Analysis and Interpretation: Analyze the results obtained from performance evaluation experiments, identifying strengths, weaknesses, and areas for improvement in the developed models. Interpret the findings to draw meaningful conclusions about the effectiveness of different methodologies.

3.2.7. Optimization and Fine-Tuning: Conduct optimization and fine-tuning experiments to further improve the performance of developed models. Explore techniques such as transfer learning, hyper parameter tuning, and data augmentation to enhance model robustness and generalization.

3.2.8. Documentation and Reporting: Document the entire research process, including data collection, model development, training procedures, evaluation metrics, and results analysis. Prepare comprehensive reports and presentations summarizing the key findings and contributions of the research.

3.2.9. Dissemination and Collaboration: Share the research findings and methodologies with the academic and professional community through publications, conference presentations, and collaboration with industry partners. Foster collaboration and knowledge exchange to advance the field of audio source separation.

4. SYSTEM DESIGN:

4.1. DATA COLLECTION AND PREPROCESSING:

4.1.1. Data Acquisition: Collect audio datasets from various sources, including public repositories, online databases, and proprietary sources.

4.1.2. Data Preprocessing: Standardize audio formats, handle missing values, and perform data augmentation techniques such as time stretching, pitch shifting, and noise addition to enhance the diversity and quality of the dataset.

4.2. MODEL DEVELOPMENT AND TRAINING:

4.2.1. Architecture Selection: Choose appropriate deep learning architectures for audio source separation tasks, such as U-Net, DenseNets, or hybrid models.

4.2.2. Model Implementation: Implement the selected architectures using deep learning frameworks such as TensorFlow or PyTorch, incorporating necessary layers, activation functions, and optimization algorithms.

4.2.3. Training Procedure: Train the models using the preprocessed audio datasets, optimizing model parameters through techniques like stochastic gradient descent, Adam optimization, and learning rate scheduling.

4.2.4. Validation and Testing: Validate the trained models using separate validation datasets to assess generalization performance. Evaluate the models' performance using metrics like Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), and perceptual evaluation of audio quality.

4.3. SYSTEM INTEGRATION AND DEPLOYMENT:

4.3.1. API Development: Develop APIs or interfaces to interact with the trained models, allowing users to input audio files for source separation and receive separated audio tracks as output.

4.3.2. Web Application Development: Build web applications or user interfaces to provide a user-friendly experience for interacting with the audio source separation system.

4.3.3. Scalability and Performance Optimization: Implement strategies for scaling the system to handle large volumes of audio data and optimize performance for real-time processing and low-latency requirements.

4.3.4. Deployment Environment: Deploy the system on cloud platforms like AWS, Google Cloud, or Azure, leveraging scalable compute resources and managed services for efficient operation.

4.4. MONITORING AND MAINTENANCE:

4.4.1. Performance Monitoring: Implement monitoring solutions to track system performance, resource utilization, and user interactions, enabling proactive identification and resolution of issues.

4.4.2.Model Maintenance: Periodically update and retrain the models using new data and improved techniques to adapt to evolving requirements and address emerging challenges.

4.4.3.Bug Fixing and Optimization: Continuously address bugs, performance bottlenecks, and usability issues through iterative development and optimization cycles.

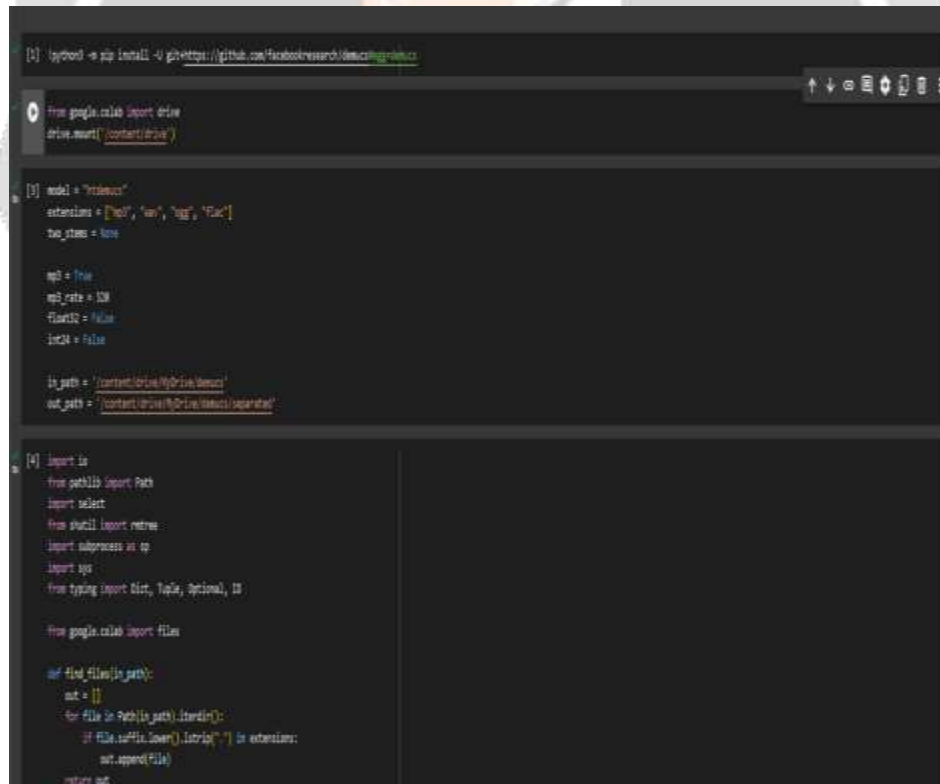
4.4.4.Documentation and Knowledge Sharing: Document the system architecture, design decisions, and implementation details to facilitate knowledge sharing among team members and stakeholders.

4.5.CONTINUOUS IMPROVEMENT:

4.5.1.Feedback Mechanism: Establish mechanisms for collecting user feedback and incorporating user suggestions and requirements into future iterations of the system.

4.5.2.Research and Innovation: Stay abreast of the latest advancements in audio source separation techniques and incorporate new methodologies and technologies to enhance the system's capabilities.

4.5.3.Collaboration and Community Engagement: Foster collaboration with researchers, practitioners, and industry partners to exchange ideas, share best practices, and contribute to the broader audio processing community.



```
[1] !python -m pip install -U git+https://github.com/Hzarkodresser/htdemucs

from google.colab import drive
drive.mount('/content/drive')

[2] model = "htdemucs"
extensions = ["wav", "wav", "ogg", "flac"]
two_stems = True

mp3 = True
mp3_rate = 128
float32 = False
int24 = False

in_path = "/content/drive/MyDrive/htdemucs"
out_path = "/content/drive/MyDrive/htdemucs/separated"

[4] import os
from pathlib import Path
import select
from shutil import rmtree
import subprocess as sp
import sys
from typing import Dict, Tuple, Optional, List

from google.colab import files

def find_files(in_path):
    out = []
    for file in Path(in_path).iterdir():
        if file.is_file() and file.suffix in extensions:
            out.append(file)
    return out
```

Fig.4.1 Installing Required libraries and setting parameters and file paths for separating audio stems using the 'htdemucs' model, specifying input and output directories, and MP3 bitrate.

```

11)
import sys
from pathlib import Path
import shutil
from shutil import move
import argparse
from typing import List, Union, Optional, IO
from argparse import ArgumentParser

def find_files_in_path(path):
    out = []
    for file in Path(path).iterdir():
        if file.is_dir():
            out.append(file)
        else:
            out.append(file)
    return out

def process_streams(processor, options):
    out_stream = StringIO()
    in_stream = StringIO()
    return processor(in_stream, out_stream)

def main():
    parser = ArgumentParser()
    parser.add_argument('input', type=Path, help='Input file path')
    parser.add_argument('-o', '--output', type=Path, help='Output directory')
    parser.add_argument('-m', '--model', type=Path, help='Model path')
    parser.add_argument('-n', '--num_threads', type=int, help='Number of threads')
    parser.add_argument('-v', '--verbose', action='store_true', help='Verbose output')
    args = parser.parse_args()

    input_path = Path(args.input)
    output_path = Path(args.output)
    model_path = Path(args.model)

    files = find_files_in_path(input_path)

    if not files:
        print("No files found in the input path.")
        return

    print(f"Going to separate the files.")
    print(f"Input: {input_path}")
    print(f"Output: {output_path}")
    print(f"Model: {model_path}")
    print(f"Number of threads: {args.num_threads}")
    print(f"Verbose: {args.verbose}")

    processor = create_processor(model_path)

    for file in files:
        print(f"Processing {file}")
        processor(file, output_path)

    print("All files processed successfully.")

def create_processor(model_path):
    # Implementation of the processor function
    # This function would load the model and return a function that takes an input stream and an output stream
    # and returns the processed streams.
    # This is a placeholder for the actual implementation.
    def processor(in_stream, out_stream):
        # Placeholder for the actual processing logic
        pass
    return processor

if __name__ == '__main__':
    main()

```

Fig.4.2 Function to separate audio stems from input files or uploaded files using the specified parameters and model, executing the separation process and handling the output.

```

separate()
print("Audio stems Extracted Successfully.")

Going to separate the files:
/content/Drive/Drive/Venue/Belkhalil - Amir Singh 128 (Opn.mp3)
With command: python3 -m venue.separate -s /content/Drive/Drive/Venue/separated -n 4 -m -k16 -v
Selected model is a bag of 1 models. You will see that many progress bars per track.
Separated tracks will be stored in /content/Drive/Drive/Venue/separated/Venue
Separating track /content/Drive/Drive/Venue/Belkhalil - Amir Singh 128 (Opn.mp3)
100% |#####| 174.4/174.4 [00:01:00.00, 31.45seconds]
Audio stems Extracted Successfully.

```

Fig.4.3 Function call to separate audio stems and separates the audio

5. ADVANTAGES AND DISADVANTAGES:

5.1.ADVANTAGES:

5.1.1.Improved Audio Quality: Audio source separation techniques can enhance the quality of audio recordings by isolating individual sound sources, reducing interference, and improving clarity.

5.1.2.Enhanced Creativity in Music Production: Source separation enables musicians, producers, and sound engineers to manipulate individual components of a music track, facilitating remixing, rearranging, and post-production editing with greater flexibility and creativity.

5.1.3.Better Speech Recognition and Enhancement: Audio source separation can aid in speech recognition systems by isolating speech signals from background noise and other audio sources, improving the accuracy and reliability of speech recognition algorithms.

5.1.4.Effective Noise Reduction: Source separation techniques can help reduce background noise and interference in audio recordings, resulting in cleaner and more intelligible audio output, particularly in applications like audio conferencing, speech recording, and surveillance systems.

5.1.5.Facilitates Audio Analysis and Understanding: Separating audio sources can aid in the analysis and understanding of complex audio signals, enabling researchers, analysts, and machine learning algorithms to extract meaningful information and insights from audio data.

5.2.DISADVANTAGES:

5.2.1.Loss of Signal Integrity: Source separation algorithms may introduce artifacts, distortions, or loss of fidelity in the separated audio signals, especially in complex mixtures with overlapping sources or in the presence of reverberation and ambient noise.

5.2.2.Computational Complexity: Source separation algorithms can be computationally intensive, requiring substantial processing power and memory resources, particularly for real-time or large-scale applications.

5.2.3.Challenges in Real-world Environments: Source separation techniques may struggle to perform effectively in real-world environments with varying acoustic conditions, background noise, and overlapping sources, limiting their applicability in certain scenarios.

5.2.4.Difficulty in Handling Overlapping Sources: Separating overlapping sources, such as multiple instruments playing simultaneously in a music recording, poses significant challenges for source separation algorithms, often leading to imperfect results and degraded audio quality.

5.2.5.Limited Generalization: Source separation models trained on specific datasets or music genres may exhibit limited generalization capabilities when applied to unseen data or different musical styles, necessitating continuous adaptation and fine-tuning to new environments and contexts.

6.CONCLUSION:

In conclusion, audio source separation techniques represent a powerful tool in the field of audio processing and machine learning, offering numerous advantages such as improved audio quality, enhanced creativity in music production, and better speech recognition and enhancement. By isolating individual sound sources from complex audio mixtures, source separation enables a wide range of applications including music remixing, speech processing, noise reduction, and audio analysis. However, it is important to acknowledge the challenges and limitations associated with source separation algorithms, including potential loss of signal integrity, computational complexity, and difficulties in handling real-world environments and overlapping sources. Despite these challenges, ongoing research and development efforts are focused on addressing these issues and advancing the state-of-the-art in source separation technology.

In the future, continued innovation and collaboration among researchers, practitioners, and industry experts will be crucial for driving further advancements in audio source separation techniques. By leveraging emerging technologies, exploring new methodologies, and fostering interdisciplinary collaboration, we can unlock new possibilities in audio processing, improve user experiences, and enable new applications across a variety of domains.

In summary, while audio source separation presents both opportunities and challenges, its potential to revolutionize audio processing and analysis remains undeniable. With dedication, creativity, and a commitment to excellence, we can harness the power of source separation to unlock new insights, enhance audio experiences, and shape the future of sound.

7. REFERENCES:

- [1]. D. Petermann, G. Wichern, Z. -Q. Wang and J. L. Roux, "The Cocktail Fork Problem: Three-Stem Audio Separation for Real-World Soundtracks," ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, Singapore, 2022, pp. 526-530, doi: 10.1109/ICASSP43922.2022.9746005.
- [2]. R. Das, D. Deshwal, P. Sangwan and N. Nehra, "Music Source Separation: A Guide," 2021 International Conference on Industrial Electronics Research and Applications (ICIARA), New Delhi, India, 2021, pp. 1-4, doi: 10.1109/ICIARA53202.2021.9726721.
- [3]. T. Bhagwat, S. Deolalkar, J. Lokhande and L. Ragha, "Enhanced Audio Source Separation and Musical Component Analysis," 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC), Gunupur Odisha, India, 2020, pp. 1-6, doi: 10.1109/iSSSC50941.2020.9358850.
- [4]. Takahashi et al., "Multi-scale Multi-band DenseNets for Audio Source Separation", <https://arxiv.org/pdf/1706.09588.pdf>
- [5]. Jansson et al., "Singing Voice Separation with Deep U-Net Convolutional Networks", <https://ejhumphrey.com/assets/pdf/jansson2017singing.pdf>
- [6]. Takahashi et al., "MMDENSELSTM: AN EFFICIENT COMBINATION OF CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS FOR AUDIO SOURCE SEPARATION", <https://arxiv.org/pdf/1805.02410.pdf>
- [7]. Choi et al., "PHASE-AWARE SPEECH ENHANCEMENT WITH DEEP COMPLEX U-NET", <https://openreview.net/pdf?id=SkeRTsAcYm>
- [8]. Jansson et al., "Learned complex masks for multi-instrument source separation", <https://arxiv.org/pdf/2103.12864.pdf>
- [9]. R. Cosbey, A. Wusterbarth and B. Hutchinson, "Deep Learning for Classroom Activity Detection from Audio," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 2019, pp. 3727-3731, doi: 10.1109/ICASSP.2019.8683365.
- [10]. M. A. Martínez Ramírez and J. D. Reiss, "Stem audio mixing as a content-based transformation of audio features," 2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP), Luton, UK, 2017, pp. 1-6, doi: 10.1109/MMSP.2017.8122275.
- [11]. D. Stoller, S. Ewert and S. Dixon, "Adversarial Semi-Supervised Audio Source Separation Applied to Singing Voice Extraction," 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 2018, pp. 2391-2395, doi: 10.1109/ICASSP.2018.8461722.
- [12]. Mierswa, I., Morik, K. Automatic Feature Extraction for Classifying Audio Data. Mach Learn 58, 127–149 (2005). <https://doi.org/10.1007/s10994-005-5824-7>

[13]. R. P. Alarcón, L. M. P. Alvaro, C. Rodriguez, F. G. Puente, I. Petrlik and Y. Pomachagua, "Comparison of the Use of the DEMUCS Neural Network On Different Platforms for the Separation of Sources Of Musical Origin," 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN), Al-Khobar, Saudi Arabia, 2022, pp. 185-188, doi: 10.1109/CICN56167.2022.10008289.

[14]. Kim, Hyun-Tae, "Vocal Separation in Music Using SVM and Selective Frequency Subtraction," The Journal of the Korea institute of electronic communication sciences, vol. 10, no. 1, pp. 1–6, Jan. 2015.

[15]. H.-T. Kim and J.-S. Park, "A Study on Vocal Separation from Mixture Music," Journal of information and communication convergence engineering, vol. 9, no. 2. The Korean Institute of Information and Communication Sciences, pp. 161–165, 30-Apr-2011.

