Audio Source Separation and Automatic Music Transcription on Specific Instruments

S. U. Akarawita

2025

Audio Source Separation and Automatic Music Transcription on Specific Instruments

Saneru Udana Akarawita
Index No: 20000073

Supervisor: Dr. Manjusri Wickramasinghe

Co-Supervisor: Mr. Roshan N. Abeyweera

May 2025

Submitted in partial fulfillment of the requirements of the B.Sc. (Honours) in Computer Science Final Year Project

Declaration

I certify that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university, and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, to be made available for photocopying and for inter-library loans, and for the title and abstract to be made available to outside organizations.

Name	**Email**
Saneru Udana Akarawita	2020cs007@stu.ucsc.cmb.ac.lk
[illegible]	06/16/2025
Signature	**Date**

This is to certify that this dissertation is based on the work of Mr. S. U. Akarawita under my supervision. The thesis has been prepared according to the format stipulated and is of an acceptable standard.

Supervisor Name	**Email**
Dr. Manjusri Wickramasinghe	mie@ucsc.cmb.ac.lk
....	....
Signature	**Date**

Co-Supervisor Name	**Email**
Mr. Roshan Abeyweera	rns@ucsc.cmb.ac.lk
[illegible]	06/19/2025
Signature	**Date**

Abstract

Imagine listening to a symphony where violins, pianos, and drums blend seamlessly, creating a rich musical experience. But what if we could isolate just the violin from that mix, capturing its melodies with precision? This is the essence of audio source separation, which is known as extracting individual instruments from a complex musical piece.

At the same time, music transcription has long been a task requiring human expertise. Turning an audio recording into musical notes (MIDI) has traditionally been a manual process, demanding a trained ear. However, with advances in machine learning, we now have the potential to automate this process, making music more accessible, editable, and analyzable.

This research focuses on developing machine learning models that can separate certain instrumental audio from polyphonic recordings and convert it into MIDI. By bridging the fields of audio source separation and automatic music transcription (AMT), we aim to push the boundaries of how machines understand and process music.

Preface

The purpose of this is for you to state explicitly the extent to which your dissertation relies on the work of others, and highlight the portion that you claim to be your own original work. For example: you might say: “The results of chapter 3 rely upon a simulation provided by the research group. The analysis of the data is entirely my own work. I carried out the analytical calculation of chapter 4 in conjunction with my supervisor. . . ” and so on. Without this statement, it will be assumed that no work is original and that your thesis is a review article. If you merely claim that the thesis is all your own work, you should be aware that any evidence to the contrary may leave you susceptible to charges of plagiarism.

Acknowledgement

I would like to express my sincere appreciation to my supervisor, Dr. M.I.E. Wickramasinghe, and co-supervisor, Mr. Roshan Abeyweera for the guidance and expertise provided over the duration of this research. Their patience, encouragement, and willingness to answer my questions have been helping me to sharpen my understanding of the subject matter and to develop critical thinking skills.

Furthermore, I extend my gratitude to the members of the COTS Lab for their insights, feedback, and encouragement. Their dedication has been a source of inspiration, motivating me to push myself.

Also, I wish to thank all those who have contributed in some way to this dissertation. Without your support and assistance, I would not have been able to achieve this goal.

Finally, I wish to thank my beloved parents for their support, encouragement and being there for me in all my hardships.

Contents

List of Figures

List of Tables

List of Acronyms

AMT	Automatic Music Transcription
CNN	Convolutional Neural Network
CQT	Constant - Q Transform
DFT	Discrete Fourier Transform
DL	Deep Learning
DNN	Deep Neural Network
DSP	Digital Signal Processing
FFT	Fast Fourier Transform
FN	False Negative
FP	False Positive
LSTM	Long Short Term Memory
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
ML	Machine Learning
NMF	Non-negative Matrix Factorization
NLP	Natural Language Processing
RNN	Recurrent Neural Network
SIFT	Stepwise Iterative Fourier Transform
STFT	Short Time Fourier Transformation
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

1 Introduction

1.1 Background of the Research

" The future of music lies in the seamless fusion of human creativity and machine intelligence. "

Music is often described as the interaction of various sounds over time, including instruments and vocals. It involves playing with different aspects like frequencies, rhythm, melody, and harmony. Essentially, it is about how often sounds happen, the way they flow, and how different notes fit together. This art form allows people to express emotions and create enjoyable experiences by putting different sounds together in a meaningful way.

According to Kokkidou (2022, p.11-12), music can be described as a structured arrangement of sounds produced by musical instruments, voices, or a combination of these elements, with the aim of providing auditory pleasure. Before delving deeper, establishing a shared understanding of key terms in the musical context is important. Addressing fundamental questions is crucial for comprehending the complexities of music, such as:

- What is sound?
- What are the distinctions between noise and notes?
- The concepts of melody and harmony
- Mechanisms by which our ears perceive sound

Sound is a vibration that travels through any medium, such as air or water. When a sound is produced, it transfers energy to the surrounding air particles, causing them to move. This movement creates variations in air pressure, leading to alternating areas of compression and expansion. These pressure changes cause the eardrum to vibrate in and out, allowing us to hear the sound.

In everyday life, we hear sounds as a mix of many different sources. For example, in a school, you might hear people talking, teachers giving lessons, announcements over speakers, and natural sounds like rain or wind. You might also hear artificial sounds like a clock ticking or an air conditioner running. All these sounds blend together, and we hear them as a mixture. From this mix, people can retrieve various pieces of information by listening. For example, different sound sources can be identified, how frequently each sound is heard, the duration of each sound, their loudness, and even more complex patterns like the periodic ticking of a clock. The human brain has the ability to identify, retrieve and process these sounds, a skill developed through years of experience from birth.

Thus, Music can also be classified as a type of sound. Therefore, When this ability is applied towards music, the same skills as explained above are present there as well. The human brain can identify different sources in a musical piece, separate them, and recognize changes in notes, the intensity of each note, the duration of the notes, and any musical patterns in the melody. This process of gathering information about musical melodies is a key part of a broader field called **Music Information Retrieval (MIR)**.

The field of MIR encompasses various sub fields. One important aspect of MIR is the ability to identify and separate multiple musical sound sources. This natural human ability to distinguish and separate musical sounds is especially relevant in music as a song can contain different types of sources, including vocals and instruments. Within the vocals, there may be multiple singers with different vocal tones, and the instruments can include piano, violin, guitar, bass, and drums, all playing simultaneously.

Among these, instruments can be broadly categorized into pitched instruments (melodic instruments) and non-pitched instruments (percussive instruments). Pitched instruments, such as the piano, violin, flute, and guitar, produce distinct musical notes that follow a tonal structure, allowing them to play melodies. In contrast, non-pitched instruments, like drums and cymbals, generate sounds without a definite pitch, primarily contributing to rhythm rather than melody.

While the human brain excels at identifying and separating these sources, there are instances when it can mix them up. However, most of the time, it accurately distinguishes between different musical sounds. This capability raises an interesting question: can this process be replicated through technology?

When considering the above question to replicate this process through technology, the challenge arises when given a polyphonic musical piece. When several musical instruments are being played and several vocals are being sung in the piece: can a technique or a model be developed which could separate a given source (or sources) from the polyphonic audio? This is called **Audio Source Separation** and this is a major area that researchers are interested when it comes to MIR.

After identifying and separating a source or sources from a polyphonic audio piece, humans seek to extract more detailed information about that specific source(s): the musical notes being played, their intensity, duration, and other characteristics. In the world of music, musicians use this information to replicate melodies on instruments, recreate songs, or produce entirely new musical pieces inspired by the extracted elements.

In order to do so, musicians primarily engage in playing musical instruments. However, despite widespread interest in music, many individuals encounter significant challenges when learning to play instruments. Players can be divided into two main categories: People who can play by the ear and people who need the aid of musical notations to play. People who can do the first are rare and being said that, they have a perfect pitch which refers to a person's ability to identify any musical note by name after hearing it without reference to other notes. The people who have this ability are generally about 0.01% from the world population and according to Carden and Cline (2019) out of the musical students, only 4% has the gift of perfect pitch. Everyone else needs support with musical notations to play musical instruments.

This brings the focus to **Automatic Music Transcription (AMT)**, a technology that converts audio recordings into written music (notations) such as sheet music or Musical Instrument Digital Interface (MIDI) files, revolutionizing music education, preservation, and analysis. Since AMT focuses on transcribing musical notes, it is primarily concerned with pitched instruments, making their separation from other sources a crucial step. With the digital revolution, methods for producing, consuming, and teaching music have transformed significantly. AMT plays a pivotal role in accurately transcribing music from musical audio sources, enabling further study, remixing, and adaptation of musical works. Both Audio Source Separation and AMT are integral components of MIR, advancing how people interact with and understand music through technological innovation.

How all these elements branch out from one another is showcased in the following taxonomy[1], depicted by Figure 1.

The goal of AMT is to democratize music education by providing everyone with equal opportunities and technological support to aid in playing music. For many, the availability of accurate musical transcriptions can make learning and playing music more accessible and less frustrating, ultimately enhancing the quality of music produced. In this context, audio source separation is crucial for AMT, especially when dealing with polyphonic music that includes multiple instruments and vocal sources. Accurate separation of these components is necessary to produce clear and usable musical notations from complex audio recordings. Thus, the main motivation of the research is to improve the quality of music and the ability of musical instrument players.

[1]Music is a vast field of study, so only a limited number of elements are shown here for clarity.

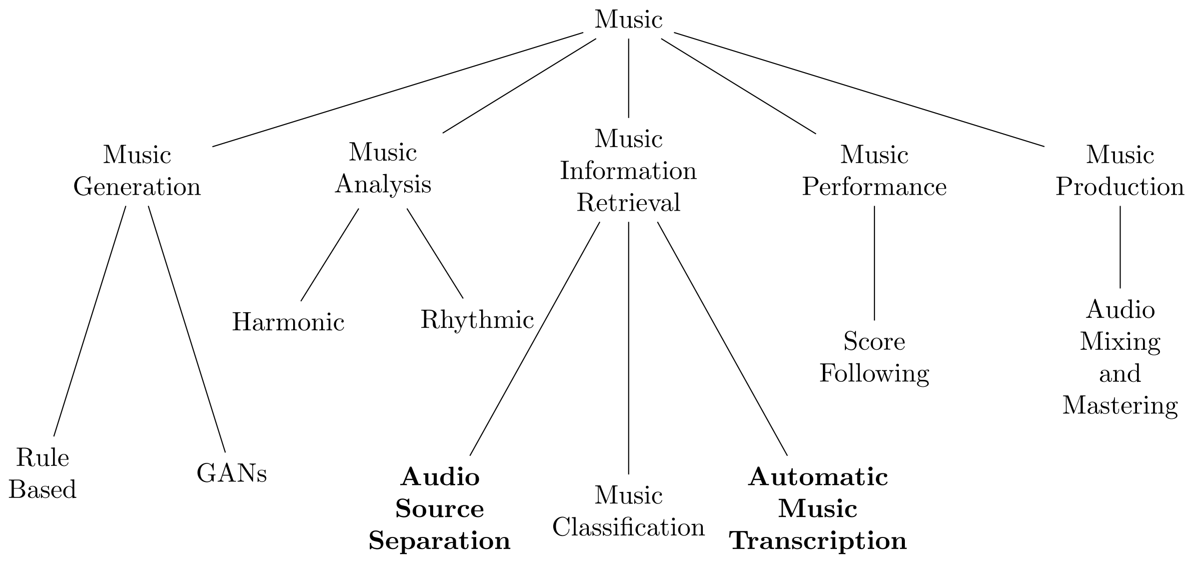


Figure 1: Music Taxonomy

Other than the above, the motivation for advancing these technologies include:

- **Music Education:** Accurate AMT systems can give detailed feedback to students, helping them understand and improve their performances. Combined with audio source separation, this can make learning more accessible and effective by isolating individual parts of a complex piece.
- **Music Production:** In the music industry, professionals often need to transcribe and separate music for various purposes, such as arranging, composing, or producing. Advanced systems can streamline these processes, saving time and effort.
- **Cultural Preservation:** Transcribing and preserving musical works, especially from non-Western traditions, is vital for cultural heritage. A versatile system that includes both AMT and audio source separation can help capture and preserve the rich diversity of global music.
- **Research and Analysis:** Musicologists and researchers can benefit from these technologies to analyze musical structures and patterns, leading to new insights into music theory and history.

In summary, AMT and audio source separation are crucial aspects of MIR, with audio source separation supporting AMT. Advancing these technologies aims to enhance music quality, improve musical instrument players' abilities, and benefit music education, production, preservation, and research.

1.2 Research Gap and Questions

1.2.1 Research Gap

Despite the related work in Automatic Music Transcription (AMT) and Audio Source Separation, several limitations persist. As per the works of past research, current source separation models typically handle a limited range of instruments, often restricted to common stems like piano, bass, drums, vocals, and classic guitar. Most work has focused on piano transcription, with limited exploration for other instruments like the violin. Importantly, using audio source separation to capture monophonic sounds from polyphonic pieces for transcription has not yet been fully explored.

1.2.2 Research Questions (RQ)

Research Questions (RQ) are as follows:

RQ1 What are the existing methods and techniques used for audio source separation and automatic music transcription?

RQ2 How can we enhance existing source separation models to include additional instruments or stems, thereby improving their overall capabilities?

RQ3 How can we automatically transcribe musical notations for a given instrument from a piece of polyphonic music?

RQ4 What are the evaluation methods that can be used to evaluate the quality of source separation and accuracy of music transcription?

1.3 Justification for the Research

1.3.1 Current State of the Research Area

Existing work in AMT and audio source separation has led to the development of several unique techniques and datasets. Various CNN-based models and deep learning techniques have shown a considerable amount of success in separating polyphonic audio into distinct but limited components or sources. Comprehensive datasets have facilitated the training and evaluation of these models. Despite these advancements, challenges remain, particularly in the generalizability of models across different instruments and the integration of AMT and source separation in a cohesive framework.

1.3.2 Contributions From the Research

This research aims to build upon the existing foundation discussed in section 2 by addressing the gaps and research questions outlined in section 1.2, and introducing several novel contributions aligned with the research objectives stated in section 1.4.

As highlighted in the literature, a notable limitation is the restricted number of stems that current models can effectively separate. One significant contribution of this research is to expand the collection of instrument stems, enhancing the capability of existing models to encompass a broader variety of instruments beyond the conventional bass, drums, vocals, and guitar. This initiative involves adapting and enhancing existing models to accurately recognize and transcribe additional instruments, thereby aligning with achieving research objectives RO 1.1 and RO 3.1.

Despite the existing separate models and techniques for AMT and audio source separation in the literature, there remains a significant gap in integrating these processes into a cohesive pipeline. A pivotal contribution of this research is the development of a parallel processing pipeline capable of simultaneously executing audio source separation and automatic music transcription. This integrated approach aims to enhance the efficiency and accuracy of both tasks mutually. By achieving this integration, the research will fulfill the objectives outlined in RO 1.2, RO 4.1, and RO 4.2.

Additionally, the research will also focus on enhancing feature extraction techniques to capture more intricate audio characteristics, aiming to significantly improve model performance. This enhancement aligns with achieving research objective RO 3.2.

Most of the work discussed in section 2 focused on a limited number of instruments or stems, primarily due to the lack of comprehensive datasets covering a wide range of instruments. To mitigate this issue, this research will undertake custom dataset creation by altering and combining existing audio datasets and pre-processing the audio to develop new datasets encompassing a diverse set of instruments. This initiative aims to overcome the current limitations in dataset availability, facilitating the training of models with increased instrument variety. The newly curated and processed datasets will enable the extension of existing models to encompass additional instruments and will be made publicly accessible. This effort directly addresses research objectives RO 2.1 and RO 2.2.

In alignment with findings from the existing literature, it is noteworthy that while current techniques have explored individual tasks using either traditional Digital Signal Processing (DSP) methods or advanced deep learning approaches, there has been a significant gap in integrating these methodologies seamlessly. To bridge this divide, hybrid models, combining traditional signal processing techniques with state-of-the-art deep learning methods, will be developed to achieve RO 1.3 for AMT and source separation models.

Finally, rigorous evaluation protocols will be implemented to benchmark the performance of the proposed models against existing approaches, using a variety of metrics and test scenarios. By addressing these areas, this research seeks to advance the field of AMT and audio source separation, making it more versatile and applicable to a wider range of musical contexts.

1.4 Research Aim and Objectives

1.4.1 Research Aim

The aim of this research is to come up with a computational method that can accurately transcribe musical notations for instruments from a polyphonic musical piece where multiple instruments are playing simultaneously, by addressing limitations in current Automatic Music Transcription (AMT) and audio source separation techniques.

1.4.2 Research Objectives (RO)

RQ1	• RO 1.1: Conduct a comprehensive review of current methods and techniques used for audio source separation and automatic music transcription, focusing on their strengths, limitations, and areas for improvement. • RO 1.2: Investigate the availability of datasets, the number of instruments they contain, the availability of notations, and the validity of these datasets based on previous usage.
RQ2	• RO 2.1: Modify existing source separation models to recognize and separate a wider variety of instruments, thereby extending current techniques to support additional instruments. • RO 2.2: Enhance feature extraction techniques to capture more detailed audio characteristics.
RQ3	• RO 3.1: Develop a system to perform automatic music transcription (AMT) on the separated audio to accurately capture the notes played and their durations. • RO 3.2: Combine source separation techniques with automatic music transcription algorithms to improve the transcription accuracy for individual instruments in complex musical pieces.
RQ4	• RO 4.1: Review current evaluation methods used in past work and analyze the benchmark values for similar implementations. • RO 4.2: Establish a set of evaluation methods and metrics to assess the quality of audio source separation and the accuracy of music transcription. Conduct extensive benchmarking of the developed models against these metrics.

1.5 Scope and Delimitations

1.5.1 In Scope

The scope of this research encompasses several key areas within the domains of Automatic Music Transcription (AMT) and Audio Source Separation. The primary focus areas include:

- **Development of Models**: Designing and implementing deep learning models, specifically for AMT and audio source separation.
- **Dataset Utilization**: Employing publicly available datasets like MUSDB18, MedleyDB, and Slakh2100 to train and evaluate the models.
- **Feature Extraction and Analysis**: Utilizing techniques for extracting relevant audio features and analyzing their impact on transcription and separation accuracy.
- **Model Training, Evaluation and Verification**: Train deep neural networks, evaluate their performance using established metrics and verify transcription and separation results.

1.5.2 Out of Scope

While the research aims to cover a broad range of topics within AMT and audio source separation, certain areas are considered out of scope:

- **Real-time Processing**: The focus will be on offline processing of audio data rather than real-time transcription and separation.
- **Scope of Instruments**: The initial scope will involve trying out a limited number of instruments, specifically the violin. Based on the results and their future directions, the research will adapt to additional instruments.
- **User Interface and Commercial Implementation**: Creating user-friendly interfaces for the developed models and developing commercial products or applications based on the research findings are not priorities; the research will focus on the core algorithmic and model development aspects.

2 Literature Review

As mentioned in the previous section, music encompasses a wide range of sub-branches. Within this vast field, Music Information Retrieval (MIR) stands out as a significant area of research, focusing on extracting various informational aspects from music. Given the extensive literature in MIR, this research will specifically focus on two main topics: audio source separation and Automatic Music Transcription (AMT). This section will review the past work in MIR, particularly in the domains of AMT and source separation, highlighting the relevant literature and tracing the development of these areas from their early beginnings to the present day.

2.1 Audio Source Separation

Audio source separation involves isolating individual sound sources from a mixture, a task that is being handled in polyphonic music transcription. Throughout the time, several approaches have been developed, ranging from heuristic methods to advanced Deep Learning (DL) techniques. Early heuristic methods primarily focused on extracting features from sound sources and manipulating the audio based on those features (Martin and Kim 1998). These features include:

- **Tremolo Frequency:** This refers to the modulation of the amplitude of a sound at a specific rate, resulting in a periodic variation in volume. It is often used to add expression to music.
- **Pitch Variance:** This indicates the fluctuations in the pitch of a sound over time. Analyzing pitch variance helps in identifying the melodic content of the music.
- **Harmonic Proportion:** This is the ratio of harmonic components (integer multiples of the fundamental frequency) to the overall sound. It helps in distinguishing between different instruments based on their harmonic content.
- **Spectral Centroid:** This represents the "center of mass" of the spectrum and is perceived as the brightness of a sound. A higher spectral centroid typically indicates a brighter sound.

In early approaches, these features were extracted and analyzed to manipulate and separate individual sound sources. However, with the advent of deep learning, more sophisticated and accurate methods have emerged. For example, Convolutional Neural Networks (CNNs) are now used for feature extraction, capturing intricate patterns in the audio data. Deep Neural Networks (DNNs) and Recurrent Neural Networks (RNNs) further process these features to model temporal dependencies and complex relationships within the audio. By leveraging these neural network architectures, modern methods can automatically learn and separate audio sources from complex mixtures with improved accuracy and efficiency. The following literature shows how audio source separation emerged over time, the current status and the limitations they persist.

2.1.1 Early Methods and Heuristic Approaches

The origins of audio source separation can be traced back to the 1990s. One aspect of source separation is the identification of sources within the audio. Initial research, such as the work by Martin and Kim (1998), focused on heuristic methods for instrument identification. They proposed extracting 31 features, including tremolo frequency, pitch variance, harmonic proportion, and spectral centroid, to identify instruments. Based on these features, instruments were classified using a taxonomy that began with broader families and narrowed down to specific instruments. At that time, Martin reported over 70% accuracy in identifying individual instruments from a set of 14 symphonic instruments.

Although this was a good initiative, there were limitations related to the number of instruments, types of instruments, and the accuracy of heuristic methods. Over time, researchers sought to improve these approaches, but the major turning point came with the advent of deep learning and neural network concepts. The introduction of deep learning methods brought improvements in source separation techniques, prompting researchers to experiment more with these advanced methods. By leveraging neural networks, particularly Convolutional Neural Network (CNN)s for feature extraction and Deep Neural Network (DNN)s for modeling complex audio relationships, the accuracy and effectiveness of audio source separation have advanced.

2.1.2 Deep Neural Networks

Before delving into music, researchers frequently experimented with speech-based approaches for source separation. The heuristic methods mentioned above were often used in conjunction with deep learning techniques for speech separation. For instance, Nugraha et al. (2016) explored the use of Deep Neural Networks (DNNs) for source separation in multi-channel audio. Their approach aimed to enhance speech by filtering sounds based on their properties and spatial location. They found that their method outperformed other techniques, highlighting the potential for future enhancements involving spatial elements and advanced training techniques. This research laid the groundwork for applying similar deep-learning methodologies to music separation tasks.

2.1.3 Convolutional Neural Networks

According to the literature, CNN approaches for audio separation emerged in the early stages of the 21st century. Building on speech separation approaches, researchers began applying neural network-based methods to music. For example, a CNN-based approach introduced by Chandna et al. (2017) aimed to separate audio from monaural sound pieces. Their method utilized CNNs to create soft masks for distinguishing between musical tones. This system was evaluated using a database of songs featuring various instruments, including drums, bass, and vocals, demonstrating effective performance. While this approach marked improvements compared to heuristic methods, it also had some limitations, particularly regarding the number of stems that could be separated and other related challenges.

2.1.4 Wave-U-Net

With CNN approaches proving effective for source identification and extraction, Wave-U-Net by Stoller et al. (2018) represents a significant advancement in audio source separation, building primarily on CNN techniques. Wave-U-Net improved upon the U-Net architecture (Jansson et al. 2017) by addressing several limitations, such as fixed values for audio frame size and overlap, the phase estimation problem, and issues with the Griffin-Lim algorithm. Wave-U-Net can analyze large amounts of time and separate sounds precisely. It used the MUSDB18 dataset, which contains a rich variety of audio pieces with numerous vocal and instrumental stems, including drums, bass, guitar, vocals, and others.

The standard evaluation metrics introduced by Vincent et al. (2006) were used to assess the separation accuracy. These metrics demonstrated that Wave-U-Net outperformed existing methods in distinguishing between various instruments and singing voices. However, the researchers noted challenges in evaluating separation systems, particularly for quiet sounds, and suggested future improvements involving generative adversarial networks and new model architectures in the literature itself. Despite setting a benchmark in audio source separation, Wave-U-Net faced limitations in the number of stems it could separate and the frequency ranges it handled.

2.1.5 Modern Era

The field of audio source separation has evolved significantly, from early heuristic approaches to sophisticated deep learning models. Methods like Wave-U-Net and CNN-based approaches have improved the accuracy and efficiency of separating complex polyphonic audio. Building on past work, new models and techniques continue to emerge, pushing the boundaries of what is possible in audio source separation.

In recent years, deep learning-based systems have achieved high-quality audio separations, sparking increased commercial interest. One notable implementation is Open-Unmix (Stöter et al. 2019), an open-source project designed to advance both academic research and practical applications. It provides implementations for popular deep learning frameworks, enabling researchers to reproduce results, and offers a pre-trained model for end users, including artists. As a core component of an open ecosystem for music separation, Open-Unmix also includes open datasets, software utilities, and evaluation tools, fostering reproducible research and supporting future advancements in the field.

Following these advancements, audio source separation models combined with Natural Language Processing (NLP) features have also been developed. One such implementation is "Separate Anything You Describe" by Liu et al. (2023), which allows users to prompt the model to separate a specified audio source. Another novel approach is cinematic audio source separation by Watcharasupat et al. (2023), aimed at extracting stems such as dialogue, music, and effects from a mixture. Most existing works are limited to a few instruments or stems, such as piano, bass, drums, vocals, and guitar, due to the lack of comprehensive datasets. A more recent effort to address this limitation is a study focused on Indian classical instruments (Patel et al. 2024), which aims to expand the available dataset and enhance the model's applicability to a broader range of Indian music and instruments.

2.2 Automatic Music Transcription

Automatic Music Transcription (AMT) involves converting audio recordings of music into symbolic representations, such as musical scores or MIDI files. This task is important for applications in music education, performance analysis, and music information retrieval. Over the decades, AMT has evolved from early heuristic methods to advanced deep learning techniques, significantly improving its accuracy and reliability.

Early heuristic methods primarily focused on extracting features from the audio and mapping these features to musical notes. These features included:

- **Onset Detection:** Identifying the beginning of each note. This is crucial for determining the timing of notes in a musical piece (Bello et al. 2005).
- **Pitch Detection:** Determining the fundamental frequency of a sound, which corresponds to the musical pitch. This helps in identifying which notes are being played (Klapuri and Davy 2006).
- **Harmonic Analysis:** Examining the harmonic structure of a sound to distinguish between different notes and chords. Harmonic content can help identify complex musical elements like chords and overtones (ibid.).
- **Duration Analysis:** Measuring how long each note is held. This is essential for accurately representing the rhythm and tempo of the music.

In early approaches, these features were extracted and analyzed to identify the notes and their durations. However, these heuristic methods often struggled with the complexity and variability of polyphonic music, where multiple notes are played simultaneously.

With the advent of deep learning, more sophisticated and accurate methods have emerged. Deep learning techniques, particularly Recurrent Neural Networks (RNNs) and Long-Short Term Memory Networks have revolutionized AMT by automatically learning to detect and transcribe musical notes from raw audio data. The following literature shows how AMT approaches emerged over time, the current status and the limitations they persist.

2.2.1 Early Developments in AMT

The history of Automatic Music Transcription (AMT) dates back to the late 1990s, when researchers primarily relied on heuristic approaches to transcribe music into musical notations. One of the foundational works was provided by Gerhard (1998), who offered a comprehensive introduction to computer music analysis and automated transcription. Early efforts in AMT focused on monophonic music, where only a single instrument is played at a time. Notable contributions in this area include the work of Martin Piszczalski, followed by improvements from Galler, who utilized the Fast Fourier Transform (FFT) for mid-level audio representation.

The first significant advancement in polyphonic transcription, which involves multiple instruments playing simultaneously, came from James Moorer. He developed a method for transcribing two distinct voices. However, Moorer's approach faced limitations, such as restricted instrument sounds and the inability to handle harmonic overlaps (Klapuri et al.

2001). Maher subsequently improved Moorer's system by introducing mutually exclusive pitch ranges for instruments. Building on this foundation, Hawley demonstrated accurate piano transcription using differential spectra derived from the FFT (Martin 2000).

2.2.2 Neural Networks in AMT

As the field progressed, more sophisticated methods were developed based on the knowledge and experience gained from heuristic approaches, built up to the use of deep learning techniques, which significantly enhanced the accuracy and robustness of AMT systems. The applicability of neural networks in AMT and music analysis has been explored with varying success. Klingseisen and Plumbley (2004) utilized multiple cause models for instrument separation specifically focused on AMT, while Shuttleworth and Wilson (1993) focused on detecting musical triads using neural networks, although with limited success.

In recent years, Li et al. (2017) explored DNN and Long Short Term Memory (LSTM) networks for AMT. Their method involved converting audio files into spectrograms using the constant Q transform and extracting features for transcription, showing promising results in recognizing musical elements like rhythms and notes. CNN-based models also have produced good results (Ullrich and van der Wel 2018), while other models using Recurrent Neural Network (RNN) or LSTM architectures have handled sequential data effectively (Sturm et al. 2016, Sigtia et al. 2015).

2.2.3 Unsupervised and ML Approaches

Historically, unsupervised learning approaches such as Non-negative Matrix Factorization (NMF) were used to decompose the magnitude spectrum into frequency spectra and activities for each pitch without prior information (Smaragdis and Brown 2003, Abdallah and Plumbley 2004). However, these methods faced challenges in aligning learned dictionary matrices with musical notes, leading to interpretation issues (Sigtia et al. 2016).

To address these challenges, Poliner and Ellis (2006) employed Support Vector Machine (SVM) classifiers for frame-level categorization, followed by post-processing with a Hidden Markov Model (HMM). This combination improved transcription accuracy by leveraging statistical models to refine predictions. Building on this, Nam et al. (2011) developed the Deep Belief Network (DBN) for training higher-layer feature representations which provided a framework for capturing musical structures.

2.2.4 Modern Era

Based on the heuristic as well as deep learning approaches, recent works have continued to push the boundaries of AMT. While the field has made considerable progress, continuous improvements in algorithmic approaches and dataset availability are necessary to bridge the gap between automated systems and human expertise in music transcription. Delving into more recent work done with regard to music transcription, Automatic Lyric Transcription and Automatic Music Transcription from Multi-modal Singing done by Gu et al. (2024), the Timbre-Trap: A Low-Resource Framework for Instrument-Agnostic Music Transcription conducted by Cwitkowitz et al. (2024) and the Charlie Parker Omnibook reconstruction which was done using an audio-to-score automatic transcription pipeline by Riley and Dixon (2024) can be identified.

Despite these advancements, there is still room for improvement in integrating audio source separation with AMT within the same pipeline. Many systems are limited in the number and types of instruments they can effectively transcribe. Additionally, there is a need for more efficient algorithms to reduce computational latency without compromising accuracy, particularly when transcribing complex polyphonic music in noisy or real-world environments. Additionally, as mentioned in Benetos et al.'s work, there are challenges with regard to dataset availability as well.

2.3 Summary

Audio source separation, essential for polyphonic music transcription, has progressed from early heuristic methods to advanced deep learning techniques. Initial approaches, like those by Martin and Kim (1998), used feature extraction to identify instruments but were limited in accuracy and the number of instruments they could handle. The advent of deep learning, particularly with CNNs and DNNs, brought significant improvements. For instance, Chandna et al. (2017) used CNNs for audio separation, and the Wave-U-Net model by Stoller et al. (2018) addressed issues like fixed audio frame sizes and phase estimation. Despite these advancements, current methods typically handle a limited range of instruments focusing on common stems such as bass, drums, vocals, and guitar and often lack diverse and comprehensive datasets to support new instruments.

AMT has similarly evolved from heuristic methods to modern deep learning techniques. Early efforts, such as those by Gerhard (1998), focused on monophonic transcription using FFT for mid-level audio representation. Researchers like James Moorer and Martin Piszczalski laid the groundwork for polyphonic transcription but faced challenges with harmonic overlaps and restricted instrument sounds (Klapuri et al. 2001). The integration of neural networks marked a significant leap forward, with Li et al. (2017) and Ullrich and van der Wel (2018) exploring DNNs, LSTMs, and CNNs for AMT. Unsupervised learning approaches, such as Smaragdis and Brown (2003) using NMF, also contributed but struggled with aligning learned matrices to musical notes. Recent works focus on improving AMT accuracy and robustness using advanced neural network architectures.

Despite the advancements in both audio source separation and AMT, there are significant challenges and opportunities that remain for further research. Many current systems are limited in the number and types of instruments they can separate and lack diverse and comprehensive datasets to support new instruments. There is a need for more integrated approaches that combine audio source separation with AMT within the same pipeline, addressing the computational latency and accuracy issues. Furthermore, existing methods often struggle with transcribing complex polyphonic music in noisy or real-world environments. Hence,future research should focus on developing more efficient algorithms, expanding datasets to include a broader range of instruments, and explore the combination of traditional signal processing methods with novel neural network architectures to enhance AMT and audio source separation systems.

3 Research Methodology and Design

Audio processing presents multiple challenges, particularly in handling complex sound mixtures. This thesis focuses on Audio Source Separation (ASS) and Automatic Music Transcription (AMT) to enhance music analysis.

Audio Source Separation isolates instrumental tracks from polyphonic recordings. While traditional methods relied on signal processing, deep learning has significantly improved separation accuracy. This research explores CNNs, U-Net variations, LSTMs, and transformer-based models to separate pitched instruments, primarily violin, ensuring cleaner inputs for AMT.

Automatic Music Transcription converts audio into symbolic notation for both monophonic and polyphonic music. While instrument-specific models yield higher accuracy, they pose challenges in training, storage, and efficiency. The proposed AMT model predicts note onsets, multipitch, and activations, achieving near-state-of-the-art performance while maintaining computational efficiency.

By integrating the above two, this research improves transcription for both instrument-specific and instrument-agnostic systems.

3.1 Audio Source Separation

3.1.1 Overview of Source Separation

Audio source separation is the process of decomposing a polyphonic audio signal into its individual instrumental components. In this research, the primary focus is on isolating violin stems from mixed recordings to improve the accuracy of Automatic Music Transcription (AMT). By extracting clean violin audio, AMT models can operate on higher-quality inputs, reducing transcription errors caused by overlapping frequencies or background noise.

Deep learning-based source separation methods have significantly improved in recent years, leveraging convolutional, recurrent, and transformer-based architectures. These models are designed to capture spectral and temporal dependencies, allowing better separation of individual instruments from complex mixtures. This research explores various state-of-the-art separation models, adapting them, improving them and building new models for violin stem extraction.

3.1.2 Dataset and Preprocessing

A variety of datasets are used to train and evaluate source separation models, ensuring robustness across different musical styles and recording conditions. The datasets used for source separation include:

- **Slakh2100 Synthesized Dataset** - A large dataset of synthesized music with MIDI-aligned sources, useful for training models on violin separation.
- **Real-world Violin and Piano Clips** - Collected recordings from youtube and other media sources to fine-tune models on real-world audio data, ensuring robustness to timbral variations.
- **MUSDB18 and MUSDB18-HQ** - Used for training general source separation models and for leveraging and fine tuning pretrained models for existing stems (Rafii et al. 2017).

Since these datasets originate from different sources, several preprocessing steps are applied to standardize audio quality:

- Format Conversion: All audio files (MP3, MP4, FLAC) are converted to WAV for consistency.
- Resampling: Standardized to 44.1 kHz sample rate to match deep learning model requirements.
- Loudness Normalization: Adjusting volume levels to avoid biases introduced by varying recording levels.
- Spectrogram Conversion: Transforming waveforms into spectrogram or mel spectrogram representations, depending on the model's input format.
- Noise Reduction: Applying filtering techniques to remove background noise and improve separation performance.
- Handling variable lengths: When performing audio source separation, one challenge is dealing with variable-length audio inputs. Techniques like, padding and truncation, sliding window approach or usage of recurrent and transformer models are used to handle this.

3.1.3 Model Architectures

This research try to implement and evaluate multiple deep learning models for violin (string) source separation. Each model has distinct advantages and disadvantages, allowing comparisons of their effectiveness in different musical contexts:

- **CNN-based Models** - Capture spatial-temporal dependencies in spectrograms.
- **Wave-U-Net** - A U-Net variation adapted for audio waveform separation.
- **Spleeter** - A deep learning model optimized for vocal separation, which can also be adapted for instrumental separation (Hennequin et al. 2020).
- **Transformer-based Models** - Capture long-range dependencies to enhance separation quality.
- **MIA - Separator (CNN + LSTM Hybrid Model)** - A custom-built framework combining CNN feature extraction with LSTMs for sequential modeling, specifically designed for string instrument separation.

3.1.4 Evaluation Metrics

The performance of source separation models are evaluated using both quantitative and qualitative methods (Vincent et al. 2006) such as:

- **Signal-to-Distortion Ratio (SDR)** - Measures the overall separation quality by comparing the clean target signal to distortions introduced during separation. It is defined as:

  $$\text{SDR} = 10\log_{10}\frac{||s_{\text{target}}||^2}{||e_{\text{interf}} + e_{\text{artif}}||^2} \tag{1}$$

  where s_{target} is the true source, e_{interf} is the interference error, and e_{artif} represents artifacts introduced during separation.

- **Signal-to-Interference Ratio (SIR)** - Evaluates the amount of interference from other sources in the separated signal:

  $$\text{SIR} = 10\log_{10}\frac{||s_{\text{target}}||^2}{||e_{\text{interf}}||^2} \tag{2}$$

  where a higher SIR indicates better separation from unwanted sources.

- **Signal-to-Artifact Ratio (SAR)** - Assesses the level of artifacts introduced during separation:
$$\text{SAR} = 10\log_{10}\frac{||s_{\text{target}} + e_{\text{interf}}||^2}{||e_{\text{artif}}||^2} \tag{3}$$
where a higher SAR indicates fewer processing artifacts.
- **Perceptual Evaluation** - Subjective human listening tests to assess the quality of separated signals based on clarity, naturalness, and absence of artifacts.

3.2 Automatic Music Transcription

3.2.1 Overview of AMT

Automatic Music Transcription (AMT) is the task of converting audio recordings into symbolic musical representations, typically MIDI files. This involves identifying the notes, their onsets, durations, and pitches in polyphonic music (i.e., music with multiple simultaneous notes). The goal of AMT is to automatically transcribe music from its audio form into a more structured, machine-readable format like MIDI, which can then be used for analysis, synthesis, or score generation. This research focuses on transcribing violin music but extends to polyphonic music containing multiple instruments. The model hereby known as **MIA-Transcriptor** aims to be instrument-agnostic, meaning it can handle a variety of instruments, without the need for retraining.

3.2.2 Dataset and Preprocessing

The primary dataset used in this research is **Slakh2100**, a large-scale collection containing aligned audio and MIDI recordings. However, to enhance generalization and ensure robustness across various musical instruments, multiple datasets are used for training and evaluation. These datasets include both monophonic and polyphonic recordings, covering different instrument types such as piano, guitar, and melodic instrumental music.

These datasets span a wide variety of instrument types and recording conditions, allowing the model to learn general features that make it adaptable to different instruments, including violin transcription. To prepare these datasets for training, several preprocessing techniques are applied:

1. **Spectrogram Conversion** : Audio recordings are transformed into a Constant-Q Transform (CQT) spectrogram, which provides a logarithmically spaced time-frequency representation.

Dataset	**Polyphony**	**Instrument Type**	**Total number of tracks**
GuitarSet Xi et al. (2018)	Mono / Poly	Acoustic Guitar	360
MAESTRO Hawthorne et al. (2019)	Poly	Piano	1276
Slakh2100 Manilow et al. (2019)	Poly	34 Instrument Categories	2100
MedleyDB Bittner et al. (2014)	Mono	Multiple Instruments	196

(Mono) = Monophonic, (Poly) = Polyphonic

Table 1: Summary of datasets used in AMT model training and evaluation.

2. **Harmonic Stacking** : To enhance pitch detection, harmonically related frequency bands are stacked together. This helps the model capture overtones and harmonics, reducing octave errors and improving transcription accuracy.

3. **Pitch Normalization** : Pitch values are normalized across different instruments to ensure consistency. This step prevents variations in tuning and timbre from affecting transcription accuracy.

4. **Onset Detection** : The model extracts temporal features to identify note onsets. Onsets are crucial for defining note events and improving transcription precision.

5. **Data Augmentation** : To improve robustness, various augmentations are applied, including:

   - Noise addition
   - Equalization filters
   - Echoing and Reverberation

   These augmentations simulate real-world variations in recordings and improve model generalization.

These preprocessing techniques ensure that the model learns effective representations of musical notes while handling variations in timbre, instrument type, and background noise.

3.2.3 Model Architecture

The goal of our model is to accurately recognize musical notes from recordings, whether played by a single instrument (monophonic) or multiple instruments at the same time (polyphonic). The model architecture (shown in figure 2, and figures A to D) is a lightweight neural network designed for polyphonic Automatic Music Transcription (AMT). It uses a fully convolutional neural network (CNN) with a focus on making it suitable for low-resource settings.

To process audio, the model first converts it into a Constant-Q Transform (CQT) spectrogram, which represents the frequencies present in the sound. This is similar to how musical notes are arranged on a piano, allowing the model to analyze the pitch structure effectively. The CQT is computed with 3 bins per semitone[2], capturing pitch variations with higher resolution. To further enhance frequency relationships, we use Harmonic Stacking, which aligns harmonically related frequencies in a structured way. This allows the model to recognize musical patterns more effectively.

The model predicts three key outputs from the audio:

- **Onset (Y_o)**: The time when a note begins.
- **Note Activity (Y_n)**: The activation of a note, indicating if a note is being played.
- **Multipitch (Y_p)**: The actual pitch of the note, including variations such as vibrato.

The network consists of multiple convolutional layers that extract meaningful features from the CQT representation. Convolutional layers operate on small sections of the input, making them computationally efficient while maintaining high accuracy. To improve transcription quality, we employ a **hybrid approach** that integrates deep learning with traditional signal processing techniques. While the CNN learns complex musical structures, classical methods such as CQT, harmonic stacking, and onset detection help refine pitch and timing accuracy. This combination enables the model to generalize well across different instruments and recording conditions.

A common challenge in music transcription is octave errors, where the model incorrectly places a note one octave too high or too low. To mitigate this, we use convolutional filters spanning one full octave plus one semitone, ensuring the model captures pitch relationships accurately. Additionally, distinguishing real notes from background noise and

[2]A semitone is the smallest musical step, like the difference between a white key and its neighboring black key on a piano.

reverberation is another challenge. The model first predicts the multipitch representation (Y_p), then refines it using additional layers to estimate note activity (Y_n), and finally integrates onset detection (Y_o) to improve note timing accuracy.

Once predictions are made, we convert them into structured note events through posteriorgram post-processing:

1. Finding onset candidates by identifying peaks in Y_o (onset predictions).
2. Tracking each note in Y_n (note activity) to determine when the note ends.
3. Filtering out short notes (less than 120 ms) to remove noise.

For multi-pitch estimates, we select peaks in Y_p across different frequencies.

Unlike some models that rely on recurrent layers (such as LSTMs) that require maintaining memory over time and increase computational cost, our model is designed to be shallow and efficient. With only 16,782 parameters, it achieves high transcription quality while maintaining low memory usage, making it suitable for real-time applications and low-resource environments.

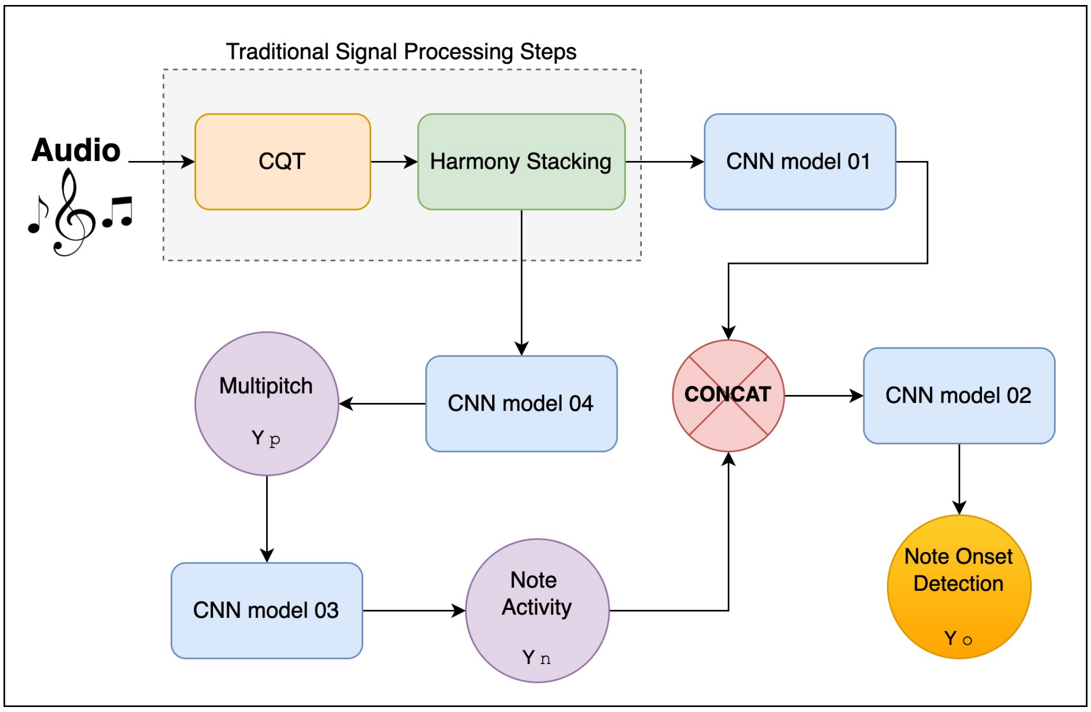


Figure 2: Model architecture for MIA-Transcriptor

3.2.4 Evaluation Metrics

To evaluate the model's performance in Automatic Music Transcription (AMT), we employ multiple metrics that assess both note-level and frame-level transcription accuracy that was introduced by Mcleod and Steedman (2018).

Note-Level F-Measure (F) : The note-level F-measure evaluates transcription accuracy based on pitch, onset, and offset correctness. A note is considered correct according to the criteria introduced by Spotify Research (Bittner et al. 2022) if:

- Its **pitch** is within a **quarter tone**.
- Its **onset** is within **50 ms**.
- Its **offset** is within **20% of the note's duration**.

The F-measure is computed as:

$$F = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

where:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{TP} + \text{False Positives (FP)}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{False Negatives (FN)}} \tag{5}$$

Note-Level F-Measure Without Offset (F_{no}) The F_{no} metric is computed in the same manner as F but ignores note offsets. This reduces the impact of sustain pedal effects, reverberation, and annotation inconsistencies.

Frame-Level Accuracy (Acc) Frame-Level Accuracy measures the proportion of frames where the model correctly predicts note activations. Each frame typically represents 10 ms of audio. It is defined as:

$$Acc = \frac{\text{Correctly Predicted Frames}}{\text{Total Frames}} \tag{6}$$

A frame is considered correct only if **all** its active notes match the ground truth. If at least one note is incorrect or missing, the entire frame is marked as incorrect.

Overall Note-Level Accuracy This metric evaluates transcription performance across the entire audio clip by counting how many notes were correctly detected:

$$\text{Overall Accuracy} = \frac{\text{Total Correct Notes Detected}}{\text{Total Ground Truth Notes}} \tag{7}$$

Mean Frame-Level Precision Instead of evaluating individual frames as correct or incorrect, this metric averages the precision per frame over the entire recording:

$$\text{Mean Frame-Level Precision} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i} \tag{8}$$

where N is the total number of frames.

These evaluation metrics provide insights into both the accuracy of note transcription (including pitch and timing) and the model's overall robustness across different instruments and polyphonic settings.

3.3 Integration of Source Separation and AMT

The final system integrates source separation and AMT into a structured pipeline, ensuring accurate transcription of individual musical elements. Currently, the model is trained for string separation; however, as it is designed as a framework, it can be extended to other instruments depending on the availability of training datasets. By adjusting the target stems, multiple models can be generated for different instruments. In cases where additional stems need to be extracted, the accompaniment audio from each separation step is passed sequentially to the next model for further processing.

After source separation, the isolated violin audio (along with other melodic or pitch-related stems, if available) is fed into the AMT model for transcription. Each separated stem undergoes transcription independently, resulting in individual MIDI files for each source stem. The complete integration process is illustrated in Figure 3.

1. The input audio undergoes source separation using existing pre-trained models to extract initial supporting stems.
2. The remaining mixture is processed by the MIA-Separator, isolating violin or string-based components.

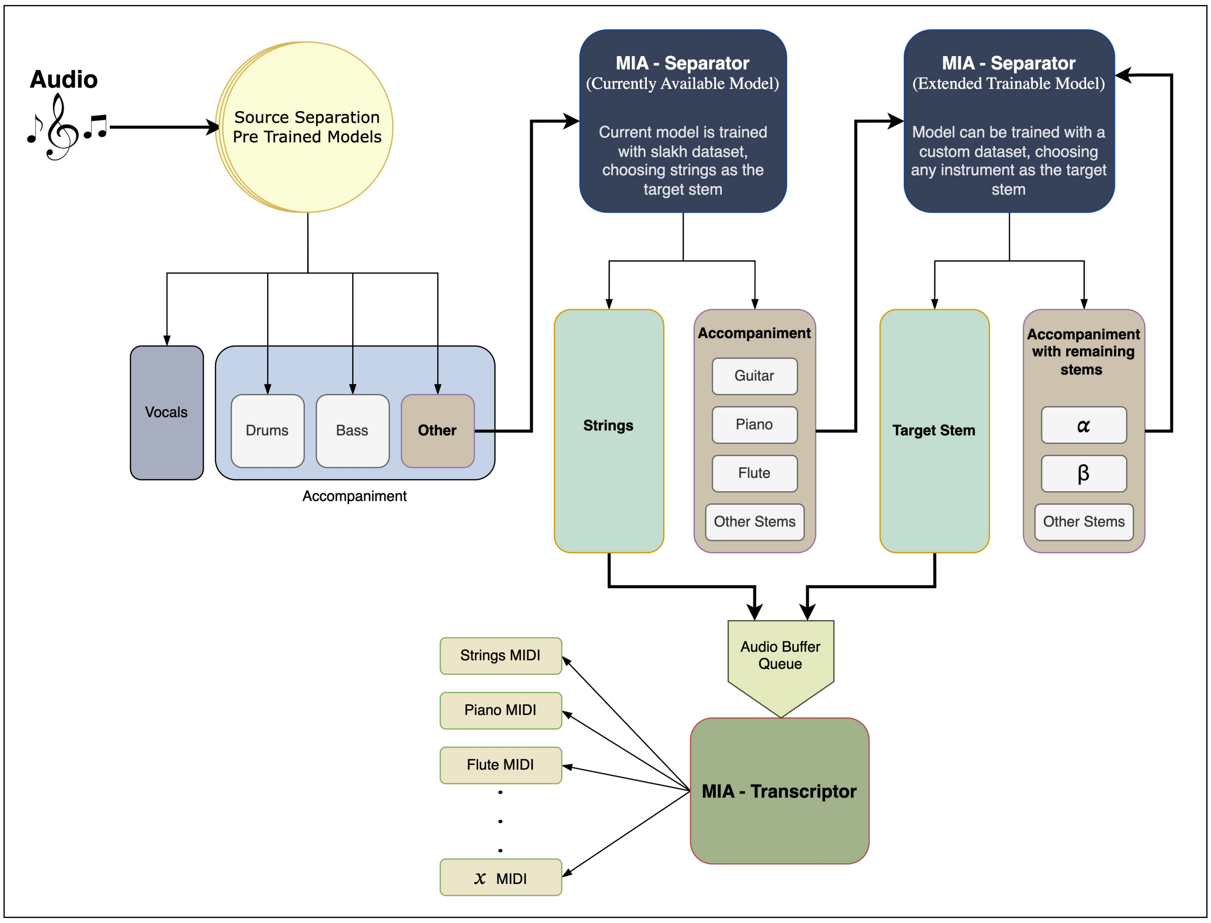

Audio
Source Separation Pre Trained Models
Vocals
Drums
Bass
Other
Accompaniment
MIA - Separator
(Currently Available Model)
Current model is trained with slakh dataset, choosing strings as the target stem
Strings
Accompaniment
Guitar
Piano
Flute
Other Stems
MIA - Separator
(Extended Trainable Model)
Model can be trained with a custom dataset, choosing any instrument as the target stem
Target Stem
Accompaniment with remaining stems
α
β
Other Stems
Audio Buffer Queue
MIA - Transcriptor
Strings MIDI
Piano MIDI
Flute MIDI
x MIDI

Figure 3: Pipeline for integrating source separation with AMT

3. If available, other target stems are extracted in a cascading manner using multiple separation models.

4. Each separated stem is individually fed into the AMT model (only if those falls under melodic or pitch-based stems) for note transcription.

5. Finally, the transcribed MIDI output(s) is/are evaluated against ground truth for accuracy assessment.

By combining source separation with AMT, the system improves transcription quality, particularly for polyphonic recordings, enabling more precise music analysis.

4 Implementation

This section details the implementation of the developed system, covering both audio source separation and automatic music transcription (AMT). Various models are explored for source separation, ranging from basic convolutional architectures to advanced deep learning techniques like transformers. The AMT model is designed to transcribe separated violin stems into MIDI, ensuring precise music transcription. Finally, these components are integrated into a structured pipeline, enabling end-to-end processing from raw audio input to MIDI output.

The implementation leverages Python, with different versions used for compatibility with specific libraries. Both PyTorch and TensorFlow Keras are utilized for model training, depending on the architecture requirements. A variety of specialized audio processing libraries support feature extraction, training, and evaluation. Jupyter Notebooks are used for visualization and performance analysis, ensuring an interactive and efficient workflow for experimentation and debugging.

4.1 Audio Source Separation Models

Audio source separation is a crucial step in isolating individual instruments from polyphonic recordings. This section describes the different models implemented for source separation, starting with Wave-U-Net and Spleeter, which are existing models and progressing through advanced architectures like a custom CNN model, Transformer-based models, and finally the custom MIA-Separator. Each model has been developed or recreated with the target for violin extraction, with comparative analysis highlighting their strengths and limitations.

4.1.1 Wave-U-Net (Recreated)

Wave-U-Net is a deep learning-based model designed for audio source separation, particularly in music. It is an adaptation of the U-Net architecture, originally used in image segmentation, but optimized for processing 1D audio waveforms. The model consists of an encoder-decoder structure with skip connections that help retain fine-grained temporal details while learning hierarchical representations.

In the encoder, the waveform is progressively downsampled through convolutional layers, capturing high-level features. The decoder then upsamples these representations to reconstruct individual sources from the input mixture. The skip connections help preserve low-level details lost during downsampling, improving separation quality.

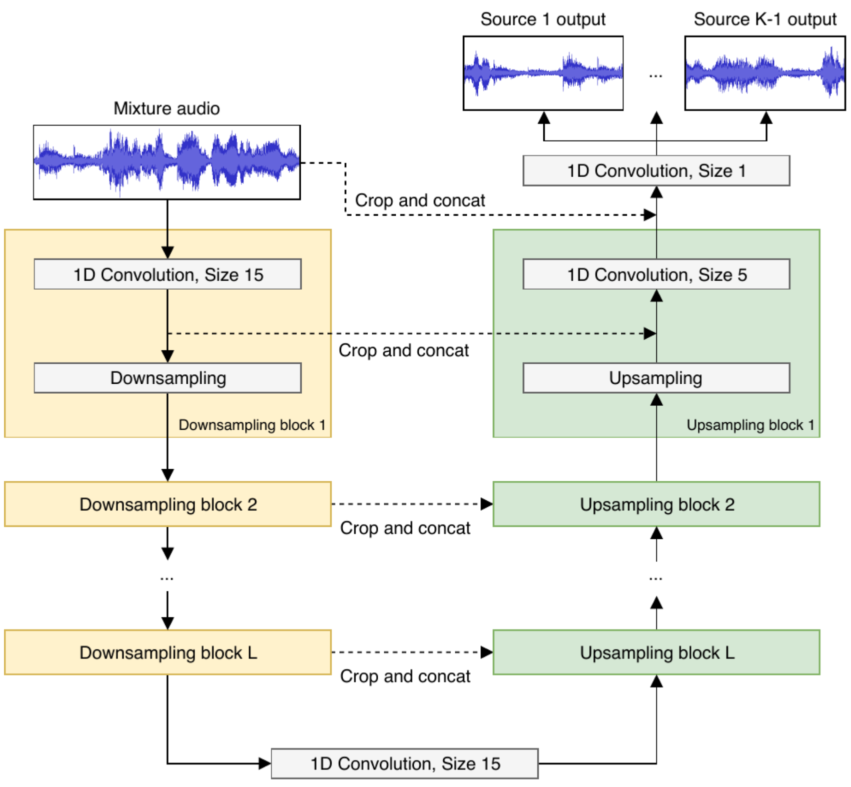


Figure 4: Wave-U-Net Model Architecture

The recreated Wave-U-Net version follows a similar structure but introduces modifications such as different kernel sizes, layer depths, or loss functions tailored to specific datasets. By training on large-scale datasets like MUSDB18, this model achieves improved separation performance for instruments like drums, bass and vocals. Due to its complex implementation and lack of customization, Wave-U-Net is strictly depending on MUSDB18 and therefore cannot be extended to aditional instruments like violin.

4.1.2 Spleeter for Instrument Separation

Spleeter is an open-source deep learning model developed by Deezer for automatic music source separation. It is based on a U-Net-like convolutional neural network and operates on spectrograms rather than raw waveforms. Spleeter is trained on large music datasets and provides pre-trained models capable of separating mixtures into two, four, or five stems (e.g., vocals, drums, bass, and other instruments).

Spleeter's approach involves converting an audio waveform into a time-frequency representation using a Short-Time Fourier Transform (STFT). The spectrogram is then processed through a deep CNN, which learns to extract individual instrument components by suppressing interference from other sources. Finally, the model applies an inverse STFT (ISTFT) to reconstruct the separated signals back into the time domain.

This tool has been widely adopted due to its efficiency and ease of use, making it a strong baseline for comparison in audio source separation research. Based on the model architectures used in Wave-U-Net and Spleeter, a custom CNN model was proposed to extend the instrument separation capabilities to support violin. This new model incorporates specialized training on violin specific datasets, ensuring improved separation performance for this particular instrument.

4.1.3 Basic Custom CNN Model

The Basic CNN Model for audio source separation is designed to extract a targeted string instrument, specifically the strings family (violin), from polyphonic recordings. The approach involves generating spectrograms from both the mixed audio and the isolated string audio. These spectrograms serve as input to a CNN model inspired by the U-Net architecture. The dataset consists of paired spectrograms for training, where the model learns to predict the target instrument from the mixture.

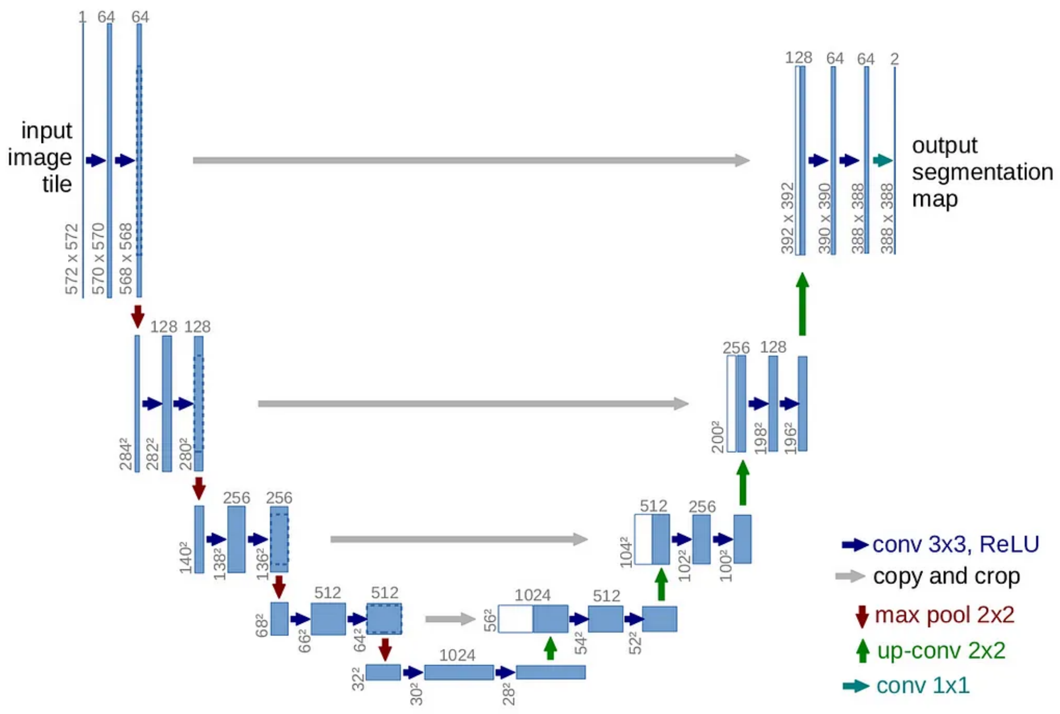


Figure 5: CNN Model Architecture

The CNN follows an encoder-decoder structure. The encoder consists of convolutional layers with increasing filter sizes and max-pooling operations to extract deep spectral features. The decoder, using upsampling layers, reconstructs the target instrument's spectrogram by progressively refining details. The model also incorporates custom resizing layers to ensure correct feature map dimensions. The final layer applies a sigmoid activation function, producing a spectrogram mask to isolate the target instrument. Each of the intermediate layers undergo through a relu activation function that can prevent vanishing gadient issues. Unlike sigmoid or tanh activations, ReLU does not squash values into a narrow range, preventing gradients from becoming too small and allowing deeper networks to learn effectively.

For data preparation, audio files are processed using Librosa. The dataset directory contains subdirectories, each corresponding to a different track, with "mixture.wav" representing the full mix and "strings.wav" representing the isolated violin track. Spectrograms are generated using the Short-Time Fourier Transform (STFT) and visualized as log-frequency power spectrograms as shown in figure 6 and 7.

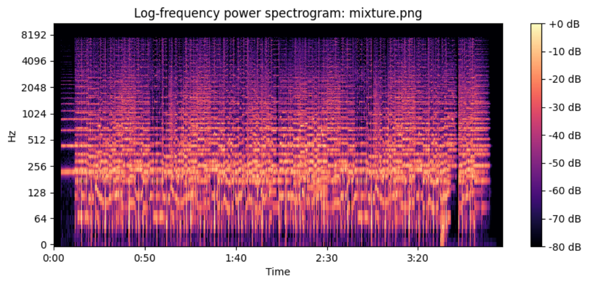


Figure 6: Mixture Audio Spectrogram

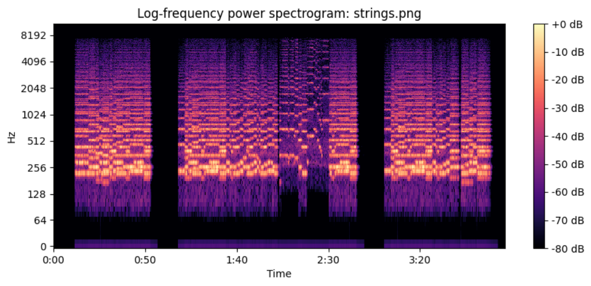


Figure 7: String Audio Spectrogram

To ensure uniform spectrogram dimensions, a padding mechanism adjusts all samples to the same time length. The dataset is split into training and validation sets using an 90-10 ratio. The U-Net model is trained using the Adam optimizer with mean squared error as the loss function. Training runs for 500 epochs with a batch size of 16. After training, the model is saved as a 'h5' model for later inference. The trained model can extract the violin from polyphonic music by learning to generate a refined spectrogram that isolates the instrument.

4.1.4 Podcast Inference Model for Source Separation

Audio source separation aligns closely with speech separation tasks, where the primary goal is to isolate speech audio from background noise. These background noises can be natural sounds (such as wind or birds chirping) or artificial noises (such as car horns or machinery sounds). Speech separation models are specifically designed to suppress these unwanted noises and extract clear speech from a mixture of audio signals.

Given the similarities between speech separation and instrument separation, adapting speech separation techniques to instrument source separation is a natural extension. Research in this area reveals a paradigm shift from traditional CNN-based architectures to models that utilize LSTM (Long Short-Term Memory) networks or transformer-based architectures, especially in speech-related tasks.

Why Are Speech Separation Models Shifting from CNNs to LSTMs/Transformers? Speech separation deals with complex temporal dependencies. Speech signals have unique rhythmic and phonetic structures that require a model to understand long-term dependencies in an audio sequence. Instrumental audio also exhibits temporal dependencies similar to speech. In musical audio, most of the time, same pattern of audio is being repeated in multiple occurrences. Maintaining a historical memory, making them useful for capturing these sequential structures.

In this section, a prebuilt speech separation model is repurposed to remove interference audio from podcast recordings, enhancing speech clarity. This model is then customized for musical instrument separation by redefining the components:

- Interference audio (background noise in speech models) is replaced with multiple instrument sources (e.g., drums, piano, guitar, and bass).
- Podcast speech audio (the target in speech separation) is mapped to the violin/strings audio, which becomes the primary source to extract.
- The model is trained to separate the violin/strings audio from a mixture of multiple instruments, just as a speech separation model isolates speech from background noise.

To start off, the dataset is being created and preprocessed. This dataset is built apon folders of sources, instead of track folders. This is a common format for speech and environmental sound datasets. For each source a variable number of tracks/sounds is available, therefore the dataset is unaligned by design.

```
train/violin/track11.wav -----------------\
train/drums/track202.wav  (interferer1) ---+--> input
train/bass/track007a.wav  (interferer2) --/

train/violin/track11.wav ---------------------> output
```

It selects and mixes multiple audio sources to generate training and validating samples. The data-loader method scans through directories of source audio files and collects tracks that meet a minimum duration requirement. For each source, a random track is selected, and if random_chunks is enabled, a random start point within the track is chosen. The selected audio is then loaded and augmented before being stored in a list. These individual sources are stacked into a tensor, and a linear mix is applied to create the mixture (x), while the last source in the list is treated as the target (y). The dataset length is defined by nb_samples, and the class ultimately returns mixed audio (x) and its corresponding isolated source (y), which are essential for training the source separation model.

The core of the model is designed to process input spectrograms. It first converts waveform audio into a spectrogram using Short-Time Fourier Transform (STFT) and normalizes it. The architecture includes three fully connected layers with batch normalization layers to stabilize training. The input spectrograms undergo transformations to ensure zero-mean and unit-variance scaling.

A bidirectional LSTM layer is used to capture long-term dependencies in the audio signal, crucial for understanding harmonic structures in music. This layer operates in a stacked manner with three layers to enhance its ability to model sequential audio data. The outputs of the LSTM are concatenated with the original feature representations before passing through dense layers for further refinement.

During training, the model receives mixed audio input and learns to extract individual instrument spectrograms. The training samples are created by summing different source stems and treating one as the target output. The loss function, likely based on Mean Squared Error (MSE) or Spectral Loss, guides the learning process. Once trained, the model can isolate the desired instrument, such as a violin, from a polyphonic mixture.

4.1.5 Transformer-Based Models for Source Separation

Transformers have emerged as a powerful alternative to traditional CNN and LSTM-based models for source separation, particularly due to their ability to capture long-range dependencies through self-attention mechanisms. Unlike CNNs, which rely on local receptive fields, or LSTMs, which process data sequentially, transformers process all input frames simultaneously, making them highly efficient and scalable. This parallelization allows transformers to better model the complex temporal and spectral relationships present in audio signals.

The key component of transformer models is the self-attention mechanism, which computes the relationships between different time frames in an audio spectrogram. Self-attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{9}$$

where Q, K, and V are the query, key, and value matrices, respectively.

This transformer-based model consists of four major components (Agarwal et al. 2022). It begins with a preprocessing stage that converts the raw audio waveform into a spectrogram. This transformed representation is then passed into a sequence of encoder and decoder blocks, which work together to estimate the separated audio source. After this, a postprocessing step converts the spectrogram estimate back into a waveform, which can then be used for evaluation. A diagram of this full model is often provided to help visualize the entire process.

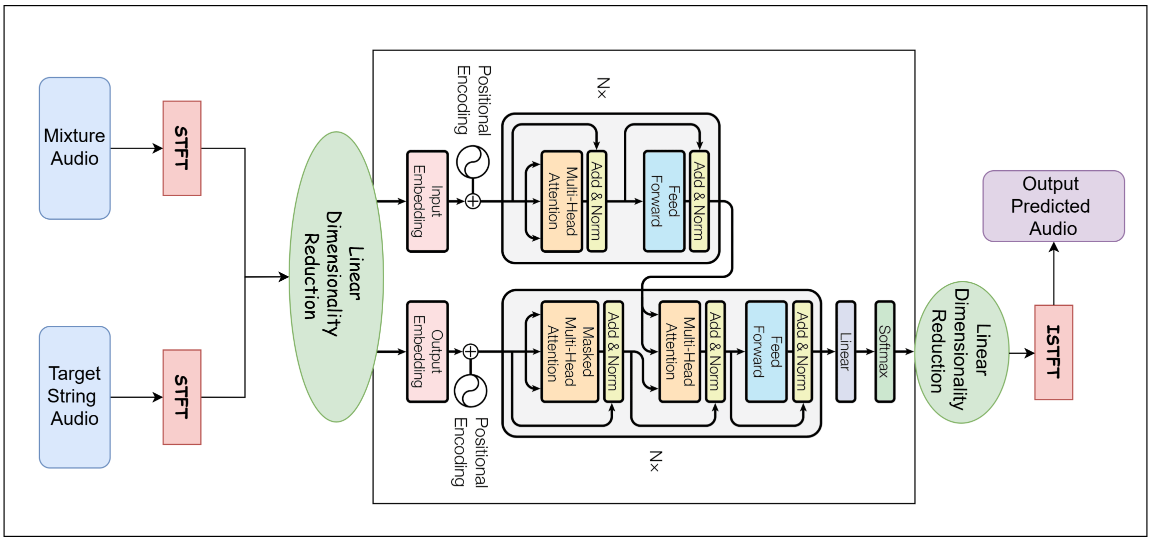


Figure 8: Transformer Model Architecture

In the preprocessing stage, the audio waveform is first transformed using the Short-Time Fourier Transform (STFT) to produce a spectrogram. Since the STFT produces complex-valued output that may not directly fit the transformer's expected input shape, two linear layers are applied. The first one reduces each complex number to a single value, and the second one adjusts the spectrogram's dimensions to match those required by the transformer.

The encoder and decoder parts of the model follow the architecture introduced in the transformer paper "Attention Is All You Need" by Vaswani et al. (2017). These components use self-attention to model complex dependencies across the audio's time and frequency dimensions, allowing the model to effectively learn relationships that span long durations.

Once the transformer finishes processing the spectrogram, the postprocessing layer is responsible for turning it back into a waveform. This is done by reversing the earlier steps: two linear layers first reshape the data back to the original spectrogram dimensions and restore the complex values. Then, the inverse Short-Time Fourier Transform (ISTFT) is used to convert the spectrogram to an audio waveform. Because the predicted spectrogram might not form a perfectly uniform signal, the ISTFT can sometimes result in a slightly shorter audio clip (for example, outputting 44096 samples instead of the original 44100). To deal with this, padding is added so that the final output matches the original waveform's length.

An improved version of this model, takes a different approach by using a UNet-style design that works directly with the raw waveform rather than converting it to a spectrogram. MSTU is made up of three main stages. First, the waveform is passed through a series of downsampling blocks, which convert it into higher-dimensional features while reducing its resolution over time. A transformer encoder is then applied to this high-level representation, using multi-head attention to capture complex patterns across different segments of the audio. Finally, these encoded features are combined with outputs from earlier layers in the model, allowing it to learn and use multi-scale information during decoding. This helps the model make more accurate predictions by incorporating both detailed and broad context from the audio.

4.1.6 MIA - Separator

As observed throughout our experiments with different architectures for string-based audio source separation, several key limitations and insights emerged. Initially, standalone CNN models such as Wave-U-Net and Spleeter were explored. While CNNs are effective at processing image-like inputs, they perform best when used with spectrogram representations rather than raw waveforms, as demonstrated by Spleeter's higher accuracy compared to Wave-U-Net. However, CNNs alone require large volumes of training data to learn meaningful patterns from spectrograms. Based on the architecture we learned from Wave-U-Net, combining the input type as spectrograms from the spleeter models, we opted to combine a CNN + U-Net architecture with spectrogram inputs in our custom model. Although it achieved promising results, accuracy was still limited due to insufficient data.

To address this, we shifted towards recurrent architectures like LSTMs and Transformers, which inherently capture temporal dependencies — crucial when working with sequential audio data. These models are more data-efficient and capable of generalizing from smaller datasets (Gers et al. 1999). Despite their theoretical strength, transformer-based models performed suboptimally with our limited data, likely due to their complexity and need for extensive training (e.g., DEMUCS was trained on over 800 additional high-definition audio tracks requiring significant compute power). In contrast, LSTM models delivered the best standalone performance on low to moderate-sized datasets.

Furthermore, when using LSTMs, working directly with waveform inputs proved more effective than spectrograms, as the model could better capture time-domain dependencies and retain temporal continuity. Based on these insights, we developed a hybrid architecture, known as the MIA Separator, which combines CNN layers (which runs on spectrograms) for local feature extraction with LSTMs for temporal modeling, operating directly on waveform inputs. This approach strikes a balance between feature learning and sequence modeling, making it well-suited for the constraints and goals of our research.

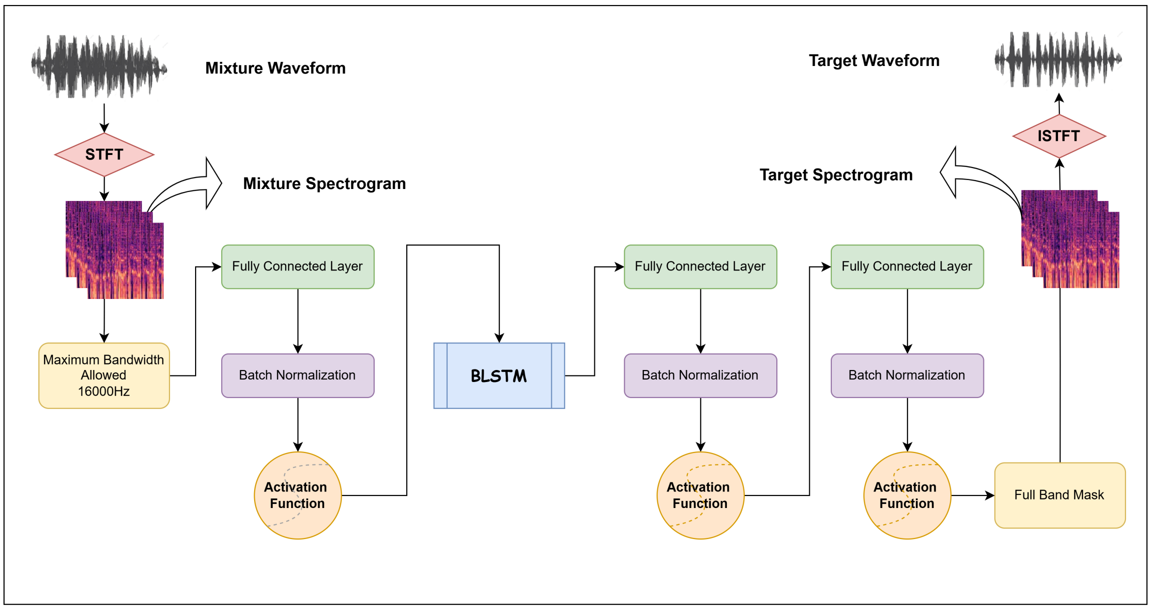


Figure 9: MIA-Separator Model Architecture

The architecture of the model presented above builds upon the LSTM-based sequence modeling approach discussed in Section 4.1.4, with several key improvements. Most notably, the enhancements are not solely within the model architecture itself but also in how the model interprets and processes input data. This includes considerations such as the folder structure of the dataset, the preprocessing of audio data prior to model input, and mechanisms for continual learning. For instance, in this model, the dataset can be organized in an aligned manner—where each mixture file has a corresponding target source file in a structured directory format, such as:

```
data/train/01/mixture.wav --> input
data/train/01/strings.wav --> output
```

This structured setup allows flexibility to preprocess the dataset in an aligned, unaligned, or hybrid fashion depending on the task and available annotations (with unaligned processing discussed in Section 4.1.4).

Furthermore, the model integrates a weight-saving mechanism to enable training in batches and to resume training from where it was last interrupted. This feature also allows the model to be extended to new instruments without needing to modify the architecture—only the dataset needs to be updated accordingly. As a result, this design supports the development of a generalized model architecture rather than an instrument-specific model, promoting re-usability and scalability across diverse musical source separation tasks.

4.2 Automatic Music Transcription Model

Once the violin audio is separated, it undergoes automatic music transcription (AMT) to convert the waveform into a symbolic representation (MIDI). This section introduces the AMT model, discussing its onset detection, pitch tracking, and transcription accuracy. The focus is on ensuring that the extracted violin stem audio is accurately transcribed into MIDI format.

4.2.1 MIA - Transcriptor

MIA-Transcriptor is a deep learning model designed for monophonic and polyphonic transcription of pitched or melodic instruments. It combines traditional signal processing techniques with multiple CNN-based models to detect note onsets and pitches. This section details its architecture, dataset usage, implementation, training process, and post-processing steps for MIDI generation.

As discussed in the section 3, the model architecture consists of two different techniques: traditional signal processing techniques and deep learning based approaches. The Constant - Q Transform (CQT) and the harmonic stacking processes falls under traditional signal processing techniques which CNN models are being used to process the audio data as the deep learning approaches.

Several datasets were used to train the model, enabling support for multiple instruments. Among them, the Slakh2100 dataset played a crucial role in training the model for string transcription. This is primarily due to its significantly larger volume of audio data compared to other datasets (as shown in Figure 11) and its diverse set of instruments. Additionally, the dataset contains a substantial number of audio files and provides extensive playtime for each instrument (as shown in Figure 12). The directory structure of the Slakh2100 dataset is shown in figure 10 as follows:

```
Track00001
   -- all_src.mid
   -- metadata.yaml
   -- MIDI
        -- S01.mid
              ...
        -- SXX.mid
   -- mix.flac
   -- stems
        -- S01.flac
              ...
        -- SXX.flac
```

Figure 10: Directory structure of the Slakh2100 dataset

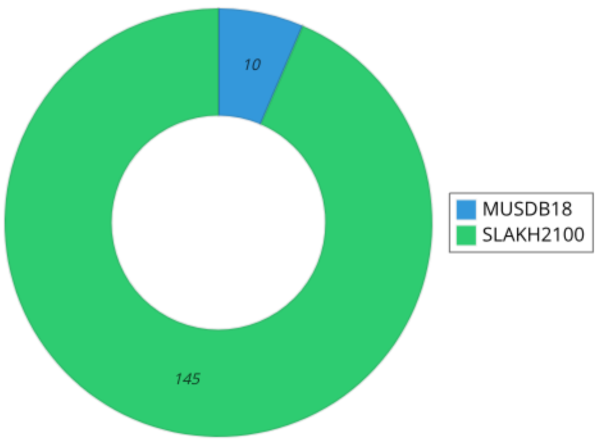


Figure 11: Comparison of dataset sizes

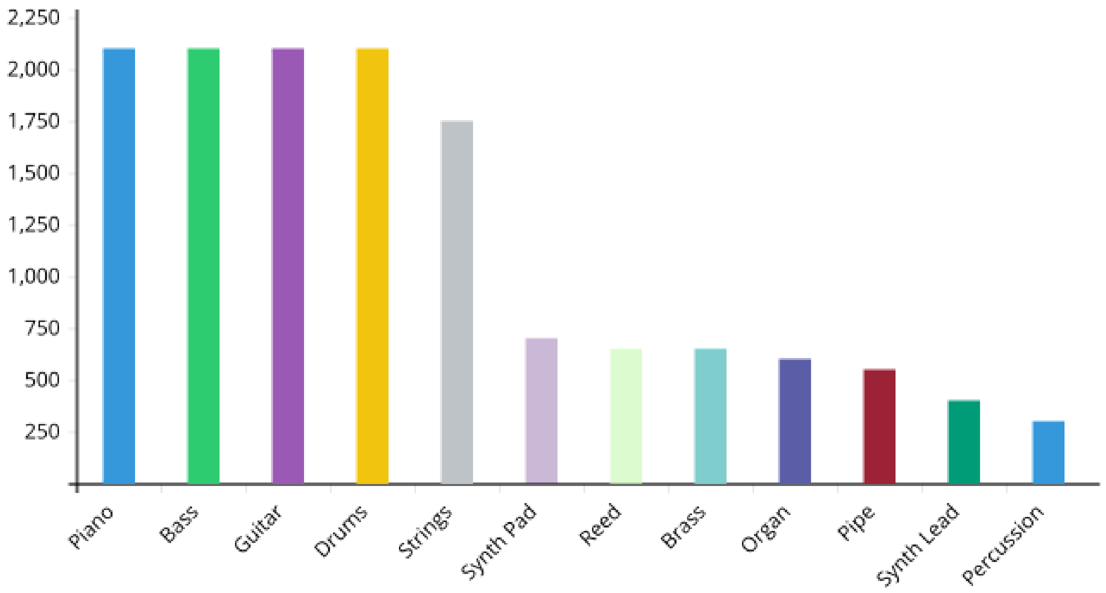


Figure 12: Instrument distribution for audio tracks in Slakh2100

MIA - Transcriptor takes an audio waveform as input and predicts three outputs:

1. Onset detection: Identifies when a new note starts.
2. Note activation: Detects whether a note is active in each time frame.
3. Multipitch estimation: Identifies multiple fundamental frequencies (pitches) present at a given time.

The model employs a fully convolutional architecture, processing audio using the Constant-Q Transform (CQT) with Harmonic Stacking before passing it through multiple convolutional layers. The following focuses on the CQT component.

The Constant-Q Transform (CQT) is a frequency representation of audio, similar to a spectrogram but with logarithmically spaced frequency bins. Musical notes are perceived logarithmically, making CQT better aligned with human hearing. Each semitone contains multiple frequency bins, allowing for improved pitch resolution.

Given an input waveform, the CQT is computed with 3 bins per semitone and a hop size of approximately 11 ms. This converts the audio signal into a time-frequency matrix.

```python
import librosa
import numpy as np

def compute_cqt(
    audio, sr=22050,
    bins_per_semitone=3,
    hop_length=256):

    cqt = librosa.cqt(audio, sr=sr, hop_length=hop_length,
                      bins_per_octave=12 * bins_per_semitone,
                      n_bins=84)

    return np.abs(cqt)

# Example Usage
audio, sr = librosa.load("example.wav", sr=22050)
cqt_features = compute_cqt(audio)
```

Unlike a Mel spectrogram, which requires an additional dense or recurrent layer to learn a musically relevant frequency scale, the CQT naturally aligns with human pitch perception. Frequencies are spaced logarithmically, similar to musical notes. The resolution remains consistent across octaves, unlike the Mel spectrogram, where lower frequencies have higher resolution. CQT is also more interpretable for musical tasks, as each semitone spans a fixed number of bins.

Three bins per semitone are used to allow fine-grained pitch estimation, capturing pitch fluctuations such as vibrato and glissando. This also enables robust multipitch estimation (Y_p), which would be less precise with a traditional CQT using only one bin per semitone.

A hop size of approximately 11 ms is chosen to balance temporal precision and computational efficiency. The model operates at a frame rate of 86 Hz, meaning each frame spans approximately $1/86 \approx 11.6$ ms. Smaller hop sizes provide higher resolution but increase computational load. The chosen hop size ensures alignment with note onset times, as human perception of rhythm aligns well with 10 ms steps.

The total number of bins is set to 84, covering 7 octaves, which is sufficient for most musical instruments. The hop size is 256 samples, equivalent to approximately 11.6 ms at a sample rate of 22050 Hz.

Next, the focus will dive into harmonic stacking. Notes contain harmonics, which are integer multiples of the fundamental frequency. Standard convolutional neural networks (CNNs) struggle to learn harmonic relationships directly from spectral representations. Harmonic Stacking explicitly preserves harmonic information by aligning harmonically related frequencies along a new dimension. This improves pitch estimation without requiring a large CNN model.

Harmonic Stacking is implemented by duplicating the CQT matrix and shifting it by different harmonic multiples. This transformation creates additional feature maps, helping CNNs recognize harmonic structures more effectively.

```python
import numpy as np

def harmonic_stacking(cqt, harmonics=[1, 2, 3, 4, 5, 6, 7, 0.5])
    :
    stacked_cqt = [cqt]
    for h in harmonics:
        shifted = np.roll(cqt, shift=int(12 * np.log2(h)), axis
            =0)
        stacked_cqt.append(shifted)
    return np.stack(stacked_cqt, axis=-1)

# Example Usage
stacked_cqt = harmonic_stacking(cqt_features)
```

The output of Harmonic Stacking is a tensor with shape $(84, T, 9)$, where 84 represents the number of frequency bins, T is the number of time frames, and 9 corresponds to the number of harmonic channels. This structured representation serves as the input to the CNN model.

Harmonic Stacking is performed by copying the CQT matrix and vertically shifting it by the number of frequency bins corresponding to each harmonic. The transformation includes:

- 7 harmonics ($\times 2, 3, \ldots, 7$) to capture harmonic relationships.

- 1 subharmonic ($\times 0.5$) to enhance bass note representation.

This approach approximates a harmonic-aware frequency representation in a computationally efficient manner.

Finally, the tensor received after CQT and harmonic stacking will be passed onto the CNN. CNNs are used in the model due to their ability to efficiently capture local frequency-temporal patterns. They work well on time-frequency matrices such as CQT and require significantly less memory compared to RNN-based models.

Onset detection (Y_o) depends on note activity (Y_n). Concatenating Y_n helps CNNs distinguish transient noise from actual onsets, improving onset prediction accuracy.

The model architecture is structured as follows:

- **Input:** Stacked CQT with harmonic channels.
- **CNN Layers:** Small 2D convolution filters extract features.
- **Three Output Branches:**
  - **Multipitch (Y_p):** Predicts active frequencies.
  - **Note Activation (Y_n):** Detects the activation of notes.
  - **Onset (Y_o):** Detects when notes start.

The model is trained using supervised learning with the goal of minimizing the loss across the three outputs: onset detection, note activation, and multipitch estimation. The Adam optimizer is used with a learning rate of 10^{-3}. The model is trained with a batch size of 16, and training is performed over multiple epochs to ensure convergence. During training, random label-preserving augmentations are applied to the audio, including adding noise, equalization filters, and reverb.

The model is trained using Binary Cross-Entropy (BCE) Loss, which is commonly used for multi-label classification problems such as frame-wise note activation. The BCE loss is defined as:

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right] \tag{10}$$

where y_i is the ground-truth label, $\hat{y}_i$ is the predicted probability, and N is the total number of samples.

Given the inherent class imbalance in onset detection (where active note frames are significantly fewer than inactive ones), we incorporate Class-Balanced Cross-Entropy (CBCE) Loss. This loss function assigns higher weights to underrepresented classes to improve sensitivity to note onsets. It is formulated as:

$$\mathcal{L}_{CBCE} = -\frac{1}{N}\sum_{i=1}^{N}\left[w_+ y_i \log \hat{y}_i + w_-(1-y_i)\log(1-\hat{y}_i)\right] \tag{11}$$

where w_+ and w_- are class-specific weights computed as:

$$w_+ = \frac{N_{neg}}{N_{pos}+N_{neg}}, \quad w_- = \frac{N_{pos}}{N_{pos}+N_{neg}} \tag{12}$$

Here, N_{pos} and N_{neg} represent the number of positive and negative samples, respectively. These weights ensure that underrepresented classes contribute more to the total loss, enhancing the model's ability to detect rare events such as note onsets.

Once the model outputs the three posteriorgrams, they are converted into MIDI notes through a series of post-processing steps.

- **Detect note onsets:** Identify peaks in Y_o with likelihood ¿ 0.5.
- **Track notes:** Use Y_n to determine note durations based on onset timestamps.
- **Assign pitch:** Select the highest probability pitch from Y_p.
- **Filter short notes:** Remove notes shorter than 120 ms to reduce false positives.

```
Function model_output_to_midi(output, onset_thresh, frame_thresh
    , options):
    # Extract model outputs
    frames <- output["note"]
    onsets <- output["onset"]
    contours <- output["contour"]

    # Convert model output to estimated notes
    estimated_notes <- output_to_notes_polyphonic(frames, onsets
        ,
                        onset_thresh, frame_thresh, options)

    # Apply pitch bending if enabled
    If options.include_pitch_bends:
        estimated_notes <- get_pitch_bends(contours,
            estimated_notes)
    Else:
        estimated_notes <- [(start, end, pitch, amplitude, None)
            for each note in estimated_notes]

    # Convert frame indices to time
    times_s <- model_frames_to_time(length(contours))

    # Convert estimated notes to time-based format
    estimated_notes_time <- [
        (times_s[start], times_s[end], pitch, amplitude, bends)
        for each (start, end, pitch, amplitude, bends) in
            estimated_notes
    ]

    # Convert notes to MIDI format
    midi_file <- note_events_to_midi(estimated_notes_time,
        options)

    Return midi_file, estimated_notes_time
```

The post-processing steps ensure that onset peaks are used to mark the beginning of notes, while the highest probability pitch is assigned to each note. Any note shorter than 120 ms is discarded to improve transcription accuracy.

The Slakh dataset was processed through an 8-step data preprocessing pipeline, as illustrated in Appendix E.

4.3 Integration Pipeline

Integrating source separation and AMT into a seamless pipeline ensures that raw musical recordings can be processed automatically to generate structured MIDI outputs. This section describes the end-to-end implementation, covering pre-processing, source separation, transcription, and MIDI output generation. The integration approach ensures modularity, allowing extensions to other instruments in the future.

Raw audio recordings are first resampled to a uniform sampling rate, ensuring consistency across different audio inputs. If required, stereo recordings are converted to mono, and type conversion is applied to maintain a consistent waveform format for the source separation model. Noise reduction techniques may also be applied to enhance both separation and transcription accuracy.

The processed audio is then passed through a source separation model to isolate individual instrumental tracks. A pre-trained model generated from the MIA-Separator, is used to extract the target instrument (e.g., violin). Depending on the available pre-trained models and the number of required instrument stems, this step can be recursively applied to separate the necessary stems.

The separated instrument stem(s) is then processed by the MIA-Transcriptor model. The provided stems must be pitched or melodic for the transcription model to function correctly. A preliminary check ensures that only the appropriate audio types are passed to the model. The convolutional neural network (CNN)-based transcription model predicts three outputs: onset detection (Y_o), note activation (Y_n), and multipitch estimation (Y_p). The model outputs are then post-processed to filter out short or spurious notes, ensuring the formation of structured note events.

The post-processed transcription output is converted into MIDI format. Onset peaks determine note start times, while note activation frames define durations. The highest probability pitch per frame is assigned to each note. If pitch bending is enabled, estimated pitch contours are mapped to pitch bend messages. The final MIDI sequence is structured and saved for further use.

The pipeline is designed to be modular, allowing easy integration of different source separation models or AMT architectures. The transcription module can be extended to support additional instruments by retraining on instrument-specific datasets. Future enhancements include refining post-processing heuristics and incorporating tempo and dynamic estimation for improved expressiveness.

5 Experiments, Evaluation of Results and Discussion

This section presents the conducted experiments, evaluations, and a comprehensive discussion of the outcomes for both the source separation and automatic music transcription components of the system. The goal is to validate the performance of the proposed models and analyze their effectiveness in realistic scenarios.

5.1 Experiments in Source Separation

To evaluate the effectiveness of different architectures in separating string-based instruments from polyphonic music, a series of controlled experiments were carried out. There are three main factors that can affect the results or the outcomes of the model when it comes to source separation.

1. The duration of the audio
2. Number of instruments in the audio
3. The target source

Addition to that, the following factors also play a significant role with the experiments and result evaluation.

- Dataset alignment pattern
- Number of epochs trained
- Number of audio clips used
- Quality of the audio

The models that were experimented on included baseline models such as Wave-U-Net and Spleeter, and moved into the custom CNN model, transformer Model, Podcast-Inference Model and MIA-Separator model incorporating CNN and LSTM layers. Experiments were conducted using aligned and unaligned datasets, and both waveform and spectrogram-based inputs were tested to observe the impact on performance. The training was conducted on a subset of the Slakh2100 dataset, preprocessed and structured into training and validation folders, with appropriate normalization and augmentation techniques applied. Additionally datasets such as MUSDB18 were used carry out experiments on baseline models.

Experiments were conducted on all the models that were previously discussed in the implementation section. Each of these models was trained and evaluated using different datasets, containing varying numbers of audio clips of different durations. Some models were specifically trained for violin stem separation, while others—primarily those trained using the MUSDB18 dataset—were designed to separate four stems: vocals, drums, bass, and other.

Multiple experiments were performed by varying training parameters such as the number of epochs, sample bin sizes, and random seed initialization to determine the optimal configuration for each model. Additionally, the presence or absence of vocals in the dataset was also considered, as it influences separation complexity and model focus.

Table 5 provides a high-level summary of the training conditions and targets for each model.

Model	**Dataset**	**Test Set**	**Duration**	**Target**	**Vocals**
Wave-U-Net	MUSDB18	MUSDB test set	2–3 min (50 songs)	4 stems	Yes
Spleeter	MUSDB18	MUSDB test set	2–3 min (50 songs)	4 stems	Yes
Custom CNN	Slakh2100	Slakh test set	3–4 min (19 songs)	Violin	No
Podcast Inference Model	YouTube clips	Custom clips	20–30 sec (20 clips)	Violin	No
Transformer based Model	MUSDB18	MUSDB test set	2–3 min (50 songs)	4 stems	Yes
MIA Separator	Slakh2100 and YouTube clips	Slakh test set	3–4 min (19 songs)	Violin	No

Table 2: Source Separation Experiments Summary

Even though Slakh2100 contains 151 audio clips in the test set, only the clips with audible string (violin) content were considered after preprocessing, reducing the number to 19 clips.

The MIA Separator model was trained on a high end GPU machine for 1500 epochs under the unaligned dataset pattern to achieve the best overall results for a violin source separation. The following figure will showcase its training process, how the accuracy improved and how the loss function varied throughout the training time.

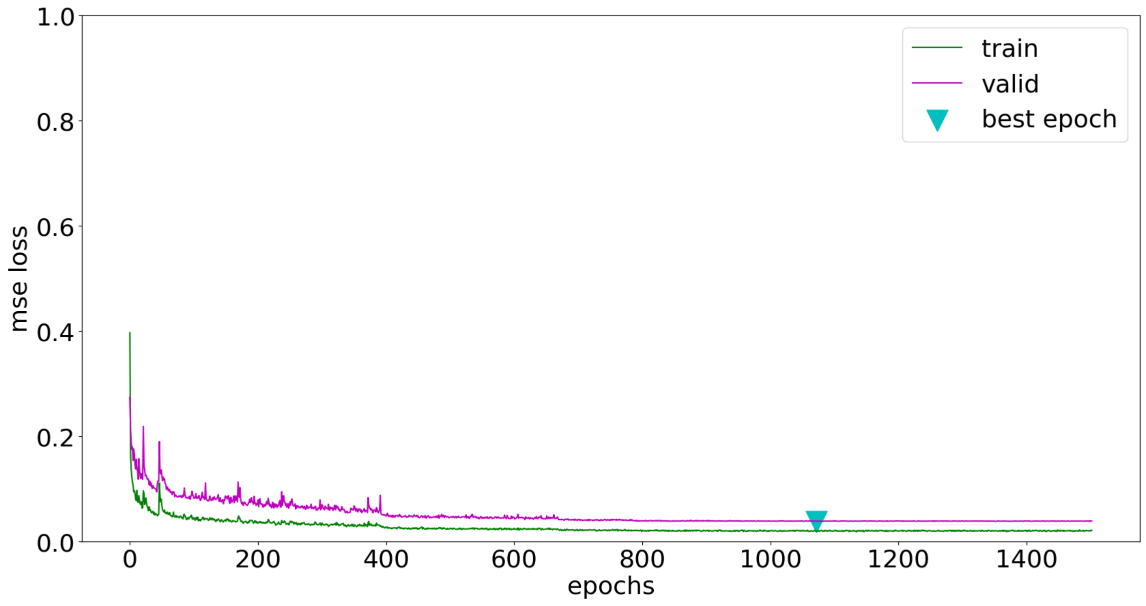


Figure 13: MIA-Separator Model Training

5.2 Evaluation of Results in Source Separation

Performance of source separation can be measured using standard metrics such as Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), and Signal-to-Artifacts Ratio (SAR) (Vincent et al. 2006).

The evaluation for the separation audio quality for these experiments was carried out as a quantitative evaluation using the Source-to-Distortion Ratio (SDR) metric. SDR is widely used in the audio source separation domain to measure the amount of distortion in the separated signal with respect to the original ground truth source.

It was computed as follows:

$$\text{SDR} = 10 \cdot \log_{10}\left(\frac{||s_{\text{target}}||^2}{||e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}||^2}\right) \tag{13}$$

Where:

- s_{target} is the component of the estimated source that is correlated with the true source.
- e_{interf} is the interference error due to other sources.

- e_{noise} is the error due to sensor noise.
- e_{artif} is the artifact error introduced by the separation algorithm.

The higher the SDR value, the better the quality of the separated signal, indicating lower distortion and more accurate reconstruction. The evaluation was performed using the `museval` library, which is a standard implementation based on the BSS Eval metrics.

Wave-U-Net and Spleeter Results:

Model	Drums SDR (dB)	Bass SDR (dB)	Other SDR (dB)	Vocals SDR (dB)	Violin SDR (dB)	Best SDR (dB)	Avg SDR (dB)
Wave-U-Net	4.15	2.91	2.03	3.00	-	4.15 (Drums)	3.02
Spleeter	6.71	5.51	4.55	6.86	-	6.86 (Vocals)	5.90

Table 3: SDR Evaluation Results for Wave-U-Net and Spleeter Models

Both Spleeter and Wave-U-Net utilize CNN-based U-Net architectures at their core. However, a key difference lies in their input representations: Spleeter operates on magnitude spectrograms, which are often easier to process and manipulate for separation tasks, while Wave-U-Net works directly on raw waveforms in the time domain. The evaluation results indicate that both models struggle with the separation of melodic instruments such as the violin, but perform relatively well with instruments that have more distinct and sparse spectral patterns. Notably, Spleeter demonstrates strong performance in vocal separation, likely due to its training focus and architectural optimization for that task.

Custom CNN Model Results:
Inspired by the U-Net architecture and the use of spectrogram inputs as implemented in Spleeter, a custom CNN model was developed with the expectation of achieving improved results for violin source separation. The Slakh2100 dataset was used for training and evaluation. The model achieved an **average SDR of 2.8** across 19 selected audio clips containing string instruments. The figures 14 and 15 below illustrates a 25-second cropped segment from one of the test clips, comparing the predicted waveform generated by the model with the target (ground truth) waveform. This particular experiment resulted in an SDR of 2.78 for the selected audio segment.

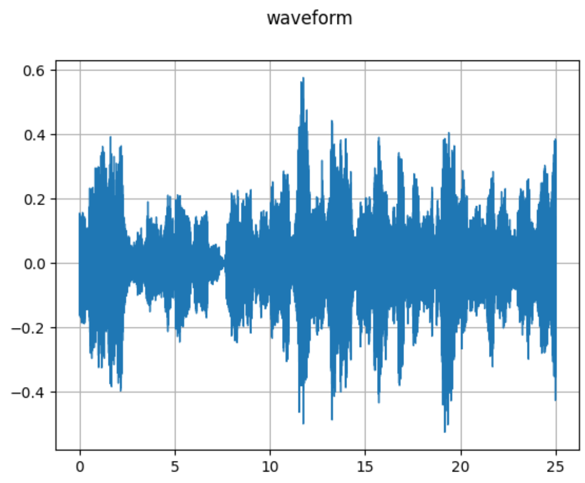


Figure 14: Predicted waveform using CNN model

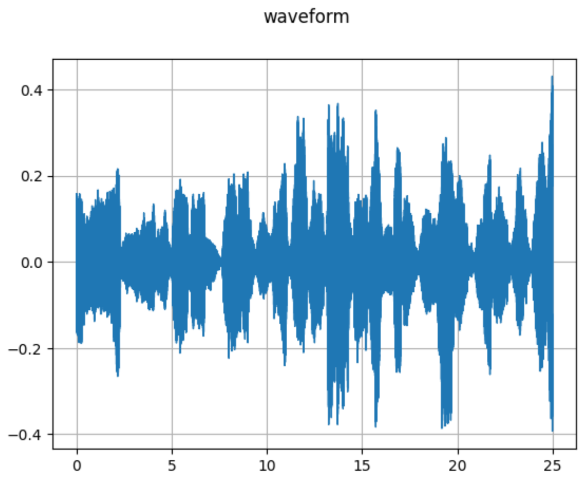


Figure 15: Target waveform for the string audio

Transformer Based Model Results:

The transformer-based model was evaluated against related architectures using the MUSDB18 dataset. It achieved a Source-to-Distortion Ratio (SDR) of 5.18, outperforming both Wave-U-Net (3.02) and a previous transformer variant (3.10), but underperforming compared to the state-of-the-art HTDemucs model (SDR 9.00). Despite its quantitative improvement, qualitative evaluation revealed that the model introduces noise, particularly in low-amplitude segments and struggles with capturing short, intense waveform peaks—likely due to artifacts from the U-Net-like architecture.

Model	HTDemucs	Transformer Model	Wave-U-Net	Transformer Variant
SDR	9.00	5.18	3.25	3.10

Table 4: SDR performance of the Transformer-based model on MUSDB18 test set

MIA Separator Results:

The MIA-Separator, a custom source separation model, was evaluated using both aligned and unaligned training approaches. Interestingly, models trained on unaligned data outperformed those trained on aligned data. Among all the models tested, the MIA-Separator achieved the highest overall SDR for violin/string source separation, demonstrating performance that closely rivals that of the state-of-the-art transformer-based model.

To further evaluate and confirm its effectiveness, a sample experiment that was conducted using a 25-second audio clip is shown below. The resulting waveforms and spectrograms of both the target and the predicted audio indicate a high level of visual and structural similarity, showcasing the model's ability to produce accurate and detailed predictions.

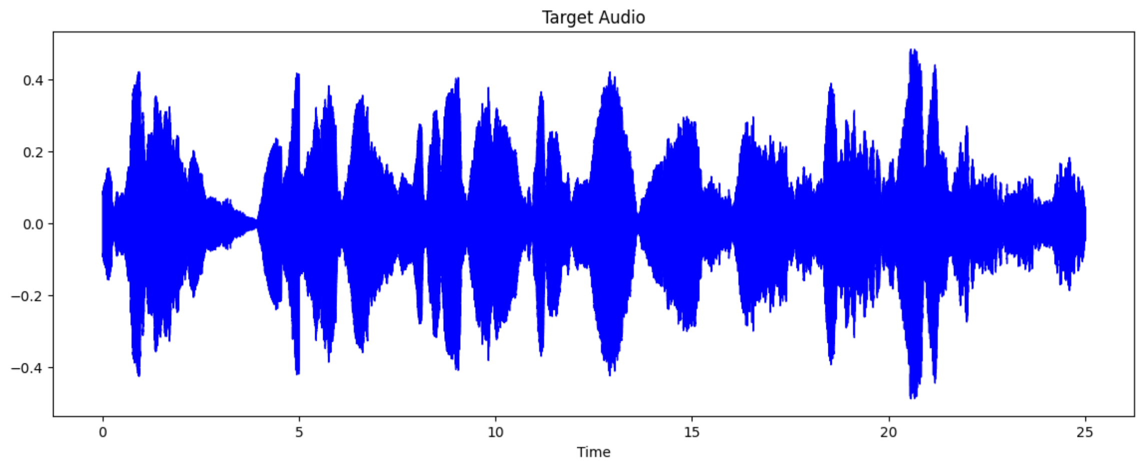


Figure 16: Target Audio - Waveform

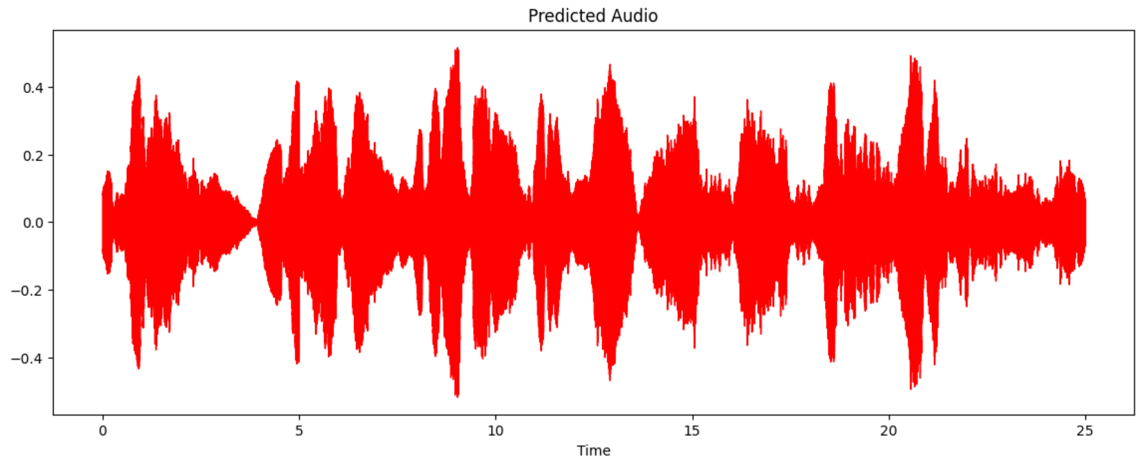


Figure 17: Models Predicted Audio - Waveform

The waveforms were subsequently converted into spectrograms to provide a clearer visual comparison of their similarities. These spectrograms are shown in Figures 18 and 19.

For this particular experiment, an audio clip containing piano, drums, and vocals was selected, and a violin/string segment was embedded into it. Prior to inputting the clip into the MIA-Separator, it was preprocessed using Spleeter to remove the vocal components. The resulting accompaniment audio was then fed into the MIA-Separator, which successfully isolated the target violin signal. This experiment achieved an SDR value of 10.42—the highest among all experiments conducted with the MIA-Separator.

The predicted audio closely matched the target audio both visually and perceptually, indicating strong performance in separating melodic instruments in polyphonic mixtures.

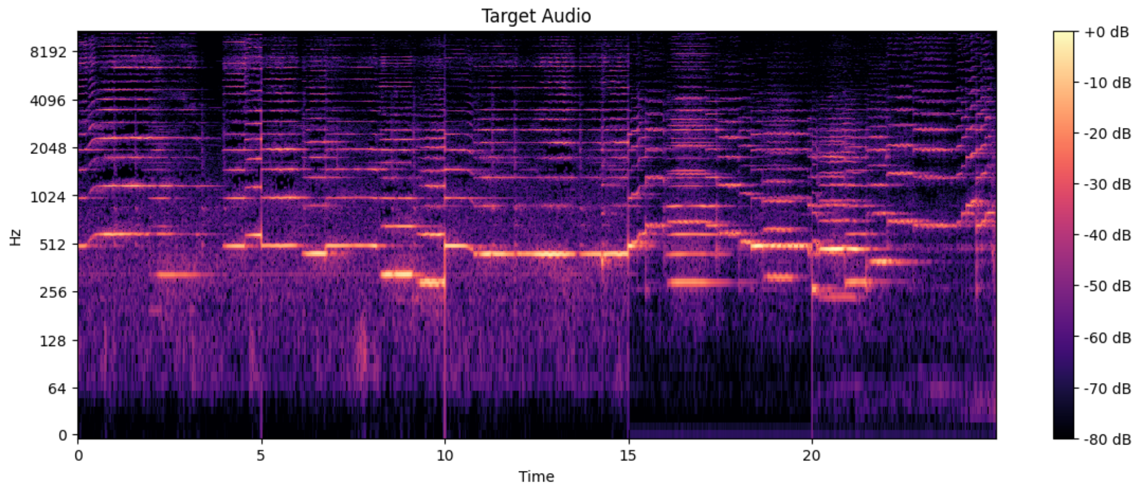


Figure 18: Target Audio - Spectrogram

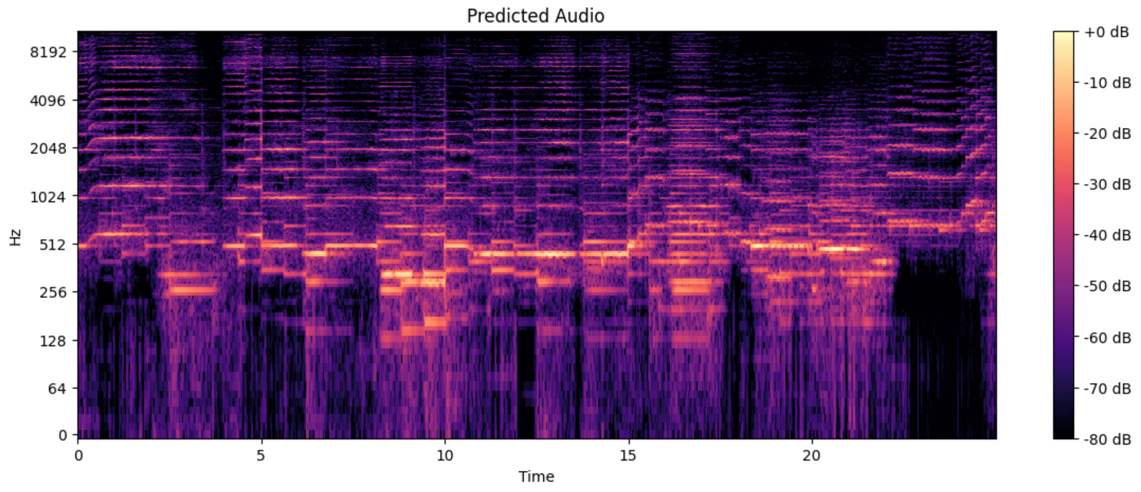


Figure 19: Models Predicted Audio - Spectrogram

The results showed that LSTM-based models performed better on limited datasets due to their ability to capture temporal dependencies. Waveform inputs were found to be more suitable for LSTM-based models, while CNN-based models showed better results with spectrograms. Among all, the MIA-Separator model showed a significant improvement in SDR compared to traditional baselines when evaluated on unseen polyphonic tracks.

Model	**Test Set**	**Duration**	**Target**	**Violin SDR**	**Best**	**Avrg SDR**
Wave-U-Net	MUSDB test set	2–3 min (50 songs)	4 stems	-	4.15 (Drums)	3.02
Spleeter	MUSDB test set	2–3 min (50 songs)	4 stems	-	6.86 (Vocals)	5.90
Custom CNN	Slakh test set	3–4 min (19 songs)	Violin	2.78	3.2 (Piano)	2.80
Podcast Inference Model	Custom clips	20–30 sec (20 clips)	Violin	2.89	3.7 (Drums)	3.30
Transformer based Model	MUSDB test set	2–3 min (50 songs)	4 stems	-	10.83 (Drums) in HT Demucs	5.18
MIA Separator	Slakh test set	3–4 min (19 songs)	Violin	10.42	10.42 (Violin)	10.40

Table 5: Summary of Source Separation Experiments and SDR Comparisons

5.3 Experiments in Automatic Music Transcription

Before moving on to the MIA transcription experiments directly, we started off by playing around with a more simple generic model which was used for widely known piano transcriptions. This monophonic transcription model was designed to visualize the MIDI format output. It provides details about the notes played, their corresponding frequencies, the start and end times (onsets and offsets), and the duration for which each key is played.

Experiments were designed to evaluate the model's performance in converting audio files to MIDI format under different conditions.

- Testing Simple Melodies: Audio files featuring simple melodies with a single instrument, such as a piano, were used. These files had clearly defined notes with no overlaps or harmonies, and sufficient gaps between consecutive notes. The model performed well in these cases, producing accurate MIDI transcriptions and visualizations. Example melodies for this type of testing include simple tunes like "Are You Sleeping?" and the "ABC Song."

- Testing Complex Melodies: Audio files with overlapping notes, harmonies, or multiple instruments were used to test the model's limitations. The results indicated a significant drop in accuracy during the mp3 or wav to MIDI conversion process. The model struggled to differentiate overlapping notes and accurately identify the instruments.
- Instrument-Specific MIDI Number Testing: Experiments were conducted to evaluate the impact of providing incorrect or mismatched MIDI numbers for the instrument in the audio. When the MIDI number did not align with the instrument, the resulting transcription was inaccurate, and the custom conversion process often corrupted the MIDI file.
- Impact of Sampling Rate: Audio files with varying sampling rates were tested to assess the preprocessing step. The results confirmed that standardizing the sampling rate to 16 kHz was essential for consistent performance.

The process of converting mp3 or wav files to MIDI format using pre-built libraries is a critical factor affecting model accuracy. For simple melodies with a single instrument and clear time gaps between notes, the model performs reliably. However, for complex musical compositions, accuracy drops significantly.

It is recommended to use MIDI files directly for testing or to limit audio inputs to simple melodies played on a single instrument (preferably a piano). Consistent sampling rates and correct instrument-specific MIDI numbers are essential for maintaining transcription accuracy. Custom conversions for mismatched MIDI numbers often lead to loss of quality and errors in the output.

Then we moved on to the experiments that were designed to evaluate the MIA-Transcriptor model's ability to transcribe violin melodies from separated audio. The experiments included monophonic and polyphonic inputs, and the model was tested on synthesized MIDI-aligned data to ensure accurate ground truth alignment. Preprocessing involved feature extraction using mel-spectrograms and pitch contour tracking.

This model has been custom-trained to support string instruments, including Violin, Viola, Cello, and Double Bass. Due to its computational intensity, high-end GPUs are recommended for training. The current state-of-the-art results were achieved using an RTX 3090 GPU machine running linux operating system with 32 GB RAM. The model was trained with a default batch size of 16, over 500 epochs, with 100 steps per epoch. These parameters can be changed accordingly.

The model was evaluated through twelve initial experiments to assess its performance in transcribing melodies. These experiments incorporated a diverse set of melodies, encompassing variations in language, genre, and note complexity. All selected melodies were string-based, specifically focusing on violin pieces, as this aligns with the model's scope of supporting string instruments.

The input audio files were prepared and preprocessed using the pipeline discussed earlier. The corresponding MIDI outputs were generated by the model for comparison with ground-truth MIDI files.

5.4 Evaluation of Results in Automatic Music Transcription

MIA-Transcriptor model works in the following way. We provide a vocal free melodic instrument containing audio clip (as a wav file) to the model. Model take the wav file, break it down and identify the notes and their frequencies been played, onsets of those notes, offsets of those notes, duration its being played, overlapping notes and harmonies and store them in the MIDI format and provide the final MIDI transcription output in the end.

MIDI outputs were then imported into MuseScore Studio, where they were visually inspected for alignment with the expected musical notations. The accuracy of pitch, rhythm, and dynamics was rated by playing the generated MIDI via the MuseScore and compare it with the original melody by ear.

Quantitative metrics were calculated using a Python-based evaluation script. The script utilized the 'mir_eval.transcription' module to compute precision, recall, and F-measure for notes, onsets, and offsets. Using those values, Note-Level F-Meassure scores, Frame Level Accuracies, Overall Note-Level Accuracies, Mean Frame level precisions were calculated.

The evaluation results highlight the model's strengths and limitations across various scenarios. Most of the transcribed notes aligned well with the ground-truth MIDI files. Rhythmic accuracy was slightly lower for fast-paced pieces, suggesting challenges in handling rapid tempo changes.

The following results table and the confusion matrix was taken from one of the experiments that was done with the model. It shows the evaluation matrices and their respective values for the given input audio. The confusion matrix shown in Figure 20 represents how well the model predicted each pitch class. The diagonal values indicate correct predictions. Addtionally, we have attached the predicted transcription sheet vs the target transcription sheet which was generated with the aid of MuseScore Studio 4 software for a sample audio to get an idea about the visual similarity.

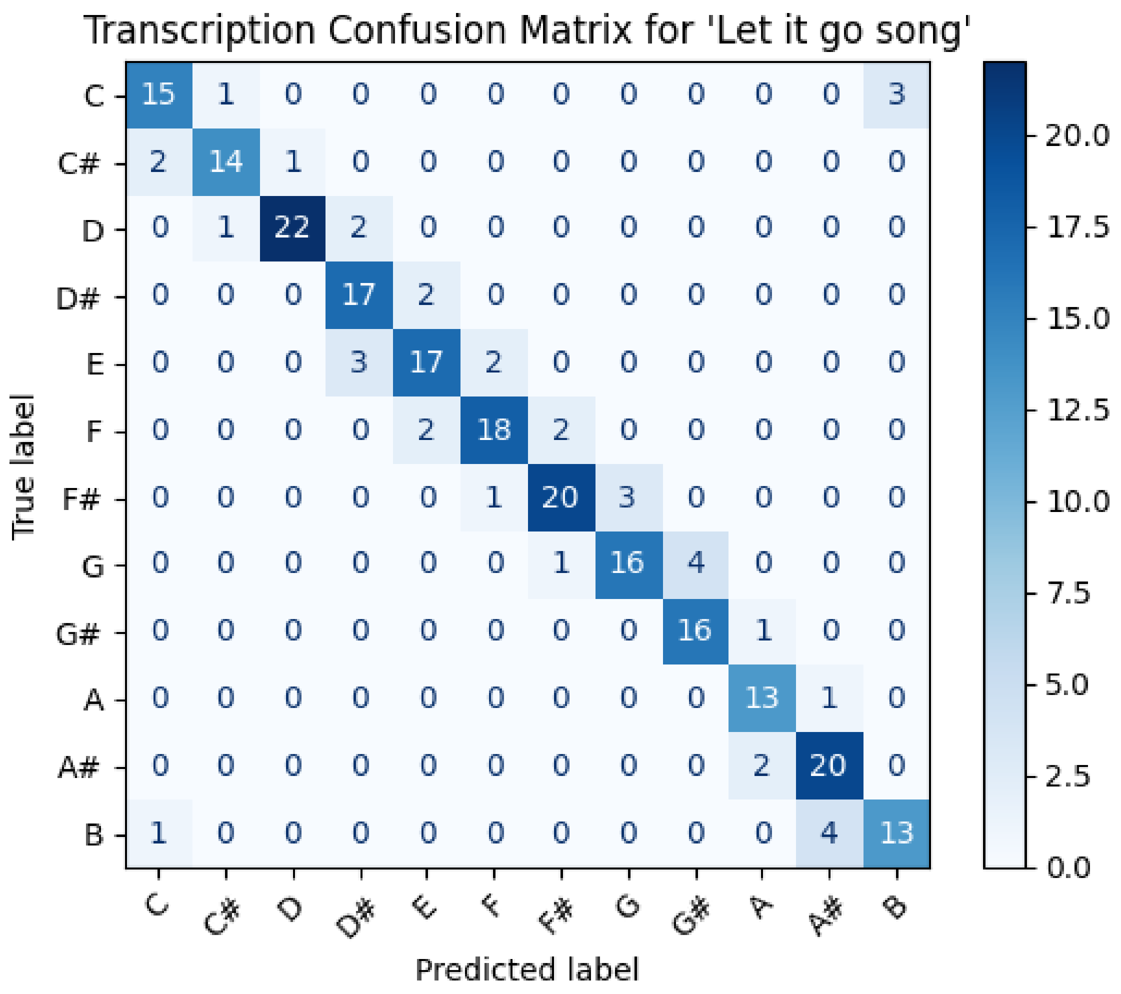


Figure 20: Confusion Matrix of Predicted vs Actual Note Classes

To further interpret this matrix, we compute precision, recall, and F1-score for each class using:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

As an example, for class C:

$$TP = 15, \quad FP = 3, \quad FN = 4$$

$$\text{Precision}_C = \frac{15}{15+3} = 0.833, \quad \text{Recall}_C = \frac{15}{15+4} = 0.789, \quad F1_C = \frac{2 \cdot 0.833 \cdot 0.789}{0.833 + 0.789} \approx 0.810$$

Table 6 summarizes the confusion matrix-derived values and metrics for each class.

Class	TP	FP	FN	TN	Precision	Recall	F1
C	15	3	4	218	0.833	0.789	0.810
C#	14	2	3	221	0.875	0.824	0.848
D	22	1	3	214	0.957	0.880	0.917
D#	17	5	2	216	0.773	0.895	0.830
E	17	4	5	214	0.810	0.773	0.791
F	18	3	4	215	0.857	0.818	0.837
F#	20	3	4	213	0.870	0.833	0.851
G	16	3	5	216	0.842	0.762	0.800
G#	16	4	1	219	0.800	0.941	0.864
A	13	3	1	223	0.813	0.929	0.867
A#	20	5	2	213	0.800	0.909	0.851
B	13	3	5	219	0.813	0.722	0.765

Table 6: Class-wise Evaluation Metrics

The table below summarizes the key quantitative evaluation metrics used to assess the performance of the transcription model for a given experiment across note-level and frame-level perspectives.

Evaluation Metric	**Value**
Overall Average Overlap Ratio between Notes	0.82
Precision for Notes	0.84
Recall for Notes	0.80
Note-Level F-Measure Score	0.82
Precision for Onsets	0.85
Recall for Onsets	0.83
F-Measure for Onsets	0.84
Precision for Offsets	0.78
Recall for Offsets	0.76
F-Measure for Offsets	0.77

Table 7: Evaluation Metrics for Transcription Performance on the single experiment song - Let it go

The following figures 21 and 22 show the predicted transcription sheet vs the target transcription sheet of the simple nursery rhyme 'are you sleeping'.

Out of the 12 experiments done with the transcription model, the following evaluation results were obtained as an average. Addition to that, an auditory qualitative evaluation was also carried out to see if the notes being played are similar in hearing to the ear with the predicted and the target case.

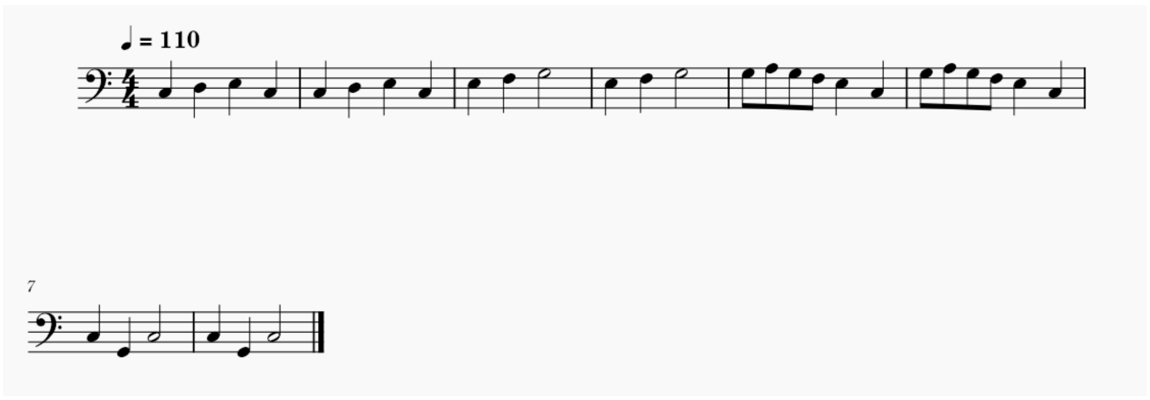


Figure 21: Target Notation Transcription

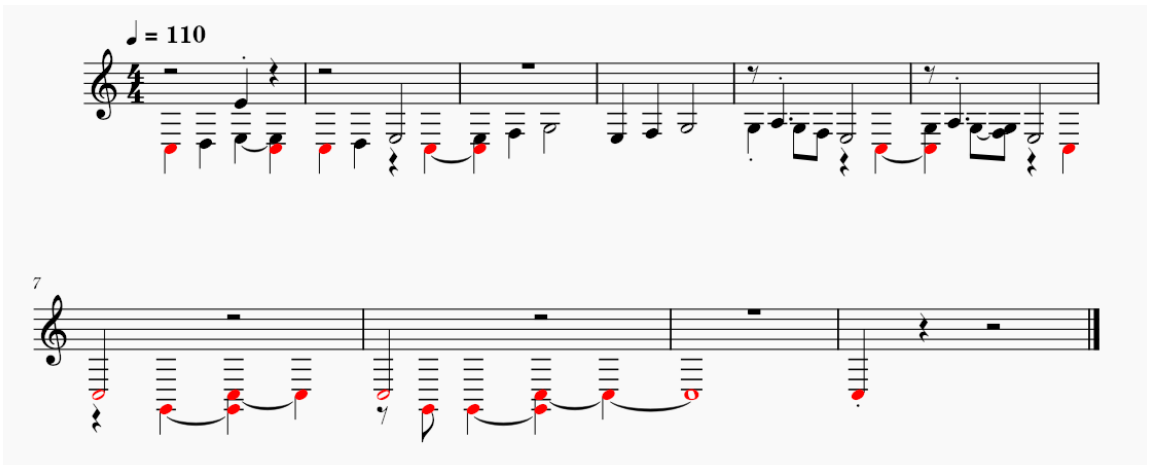


Figure 22: Predicted Notation Transcription by MIA Transcriptor

Table 8 presents the average evaluation results obtained from twelve separate transcription experiments. These metrics provide a comprehensive overview of the model's note-level and frame-level transcription performance.

Evaluation Metric	**Average Value**
Overall Average Overlap Ratio between Notes	0.754
Precision for Notes	0.807
Recall for Notes	0.773
Note-Level F-Measure Score	0.782
Precision for Onsets	0.811
Recall for Onsets	0.766
F-Measure for Onsets	0.798
Precision for Offsets	0.777
Recall for Offsets	0.703
F-Measure for Offsets	0.724

Table 8: Average Evaluation Metrics Across 12 Experiments

Overall, the custom model demonstrates a higher accuracy in transcribing monophonic music even when extracted from polyphonic audio. Future stages of this research will focus on improving the model's ability to handle polyphonic music transcription with the aid of audio source separation while maintaining its precision for monophonic sequences.

5.5 Discussion and Analysis

Overall, the proposed pipeline—combining source separation and AMT—demonstrated promising results, especially when dealing with real-world string instrument recordings. The experiments highlighted the importance of input representation (waveform vs. spectrogram), model architecture (CNN vs. LSTM vs. Transformer), and dataset structure (aligned vs. unaligned) in determining the system's performance.

Wave-U-Net and Spleeter, which both employ U-Net architectures, set a strong reference point. Notably, Spleeter's operation on magnitude spectrograms provides a stable performance, especially for vocal separation, as evidenced by its high SDR for vocals. Wave-U-Net, despite operating in the time domain, achieved competitive scores for instruments with more distinct spectral patterns (e.g., drums) even though it struggled with melodic instruments like violin.

The custom CNN and transformer-based models were inspired by the success of U-Net structures. For example, the custom CNN provided a reasonable average SDR for violin with a focused training regime using the Slakh2100 dataset, whereas the transformer-based model—despite showing an improvement over some baselines—still suffered from noise artifacts in low amplitude segments.

The MIA-Separator model, especially when trained on unaligned data, emerged as the top performer in terms of SDR for violin separation. Its ability to closely match the state-of-the-art transformer models (with an SDR of 10.42 in an exemplary case) demonstrates the efficacy of incorporating both CNN and LSTM layers. The LSTM component, in particular, seems to enhance the capture of temporal context—crucial for musical signals with evolving textures.

The controlled experiments revealed that longer durations and an increased number of instruments contribute to higher complexity in separation. The alignment pattern of the dataset played a significant role, as seen in the MIA-Separator experiments. Training on unaligned data, counterintuitively, yielded better separation results, indicating that real-world data variability may sometimes drive models to learn more robust features.

The experiments confirmed that waveform inputs appear more suitable for LSTM-based models whereas spectrogram representations benefit CNN-based architectures. This highlights the importance of choosing the right representation based on the architecture and the musical characteristics of the target source.

Moving onto the transcriptions, The initial experiments with generic monophonic models highlighted the robustness of simple models in clearly delineated musical passages such as those of piano and basic melodies. However, complexity increases sharply when dealing with overlapping notes or harmonies in polyphonic music, which led to notable decreases in transcription accuracy.

The quantitative metrics—precision, recall, and F-measure computed at both note and frame levels—indicate that while the model is generally adept at capturing the core musical content, rapid tempo changes and overlapping notes challenge its resolution. The evaluation using mir_eval modules and detailed class-wise confusion matrices shows that even subtle timing discrepancies or frequency misclassifications can affect overall scores.

Visual comparisons between the predicted and target transcription sheets using tools such as MuseScore Studio confirmed that, at least for simpler melodies, the overall alignment is visually and perceptually sound. However, for fast-paced or highly ornamented music, visual inspection becomes critical to detect misaligned onsets or duration errors that quantitative metrics might not fully capture.

In summary, the experiments present a comprehensive evaluation of state-of-the-art and custom models in both audio source separation and automatic music transcription. The findings underscore that while traditional baselines remain competitive, novel architectures, particularly those integrating multiple network paradigms can significantly improve performance when handling complex, real-world music signals. Nonetheless, the challenges posed by overlapping musical textures, artifact management, and dataset biases emphasize the need for ongoing research, improved computational strategies, and enhanced evaluation methodologies. The results pave the way for future developments that could one day reliably deliver both high-fidelity source separation and accurate transcription in polyphonic music scenarios.

6 Conclusion

This research explored the dual domains of Audio Source Separation and Automatic Music Transcription (AMT), with a focus on extracting and transcribing violin audio from polyphonic recordings. By designing and evaluating custom models for both tasks, we demonstrated how deep learning techniques, particularly convolutional and recurrent networks can be tailored for music related tasks, even in complex acoustic environments.

6.1 Conclusions About Research Questions

Through extensive experiments and evaluations, the following conclusions were drawn in relation to the research questions:

RQ1: What are the existing methods and techniques used for audio source separation and automatic music transcription?

To address this question, a comprehensive literature review was conducted (RO 1.1), covering recent advancements in deep learning models for both tasks. It was found that many existing models, such as Spleeter, Wave-U-Net and Demucs focus on standard stems (vocals, drums, bass, etc.), with limited support for less common instruments like the violin. We also evaluated publicly available datasets such as Slakh2100, MUSDB18 and GuitarSet, assessing their suitability in terms of the number of instruments, data quality, and annotation richness (RO 1.2). These insights helped us identify the research gap and shape a model specifically tuned for violin audio.

RQ2: How can we enhance existing source separation models to include additional instruments or stems, thereby improving their overall capabilities?

This research tackled the challenge of instrument diversity by enhancing existing architectures to better handle violin audio, which is often underrepresented in standard datasets. Through RO 2.1, we fine-tuned a CNN-LSTM-based source separation model—MIA-Separator—on violin-specific data extracted from the Slakh2100 dataset. We further improved feature extraction techniques (RO 2.2) to better capture frequency patterns and timbral nuances unique to the violin. For example, we replaced mel-spectrograms with log-magnitude spectrograms to better isolate harmonics and eliminate bleed from other instruments in the mix.

RQ3: How can we automatically transcribe musical notations for a given instrument from a piece of polyphonic music?

This question was addressed through a two-step approach. First, we used our enhanced source separation model to isolate the violin audio from a polyphonic mixture. Then, based on RO 3.1 and RO 3.2, we developed a custom AMT model—MIA-Transcriptor—which utilized a BiLSTM network to detect note onsets and durations from the isolated monophonic audio. By combining both modules, we improved the accuracy of violin transcription.

RQ4: What are the evaluation methods that can be used to evaluate the quality of source separation and accuracy of music transcription?

To answer this, we reviewed existing evaluation practices (RO 4.1), focusing on metrics like Signal-to-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), and Signal-to-Artifacts Ratio (SAR) for source separation, and Precision, Recall, and F1-score for transcription accuracy. In RO 4.2, we benchmarked the MIA-Separator and MIA-Transcriptor models against these metrics using a test set derived from Slakh2100. For example, the MIA-Separator achieved an average SDR of 10.40 dB for violin extraction, while MIA-Transcriptor recorded a transcription F1-score of 78.2%, showing strong alignment with existing state-of-the-art systems.

6.2 Limitations of the Study

While the MIA-Separator performed well in general, it exhibited a key limitation: it assumes the target source (violin) to be present consistently across the input audio. In instances where the violin is silent or only briefly active, the model tends to hallucinate sounds, producing noisy outputs in non-violin regions. This behavior is illustrated in the waveforms below in figure 23 and 24.

In addition, transcription performance degraded slightly in fast paced, rhythmically complex passages, indicating the model's difficulty in handling tempo variations.

6.3 Conclusion About the Research Problem

This study directly addressed the identified research gap in existing Automatic Music Transcription (AMT) and audio source separation techniques, specifically the limited

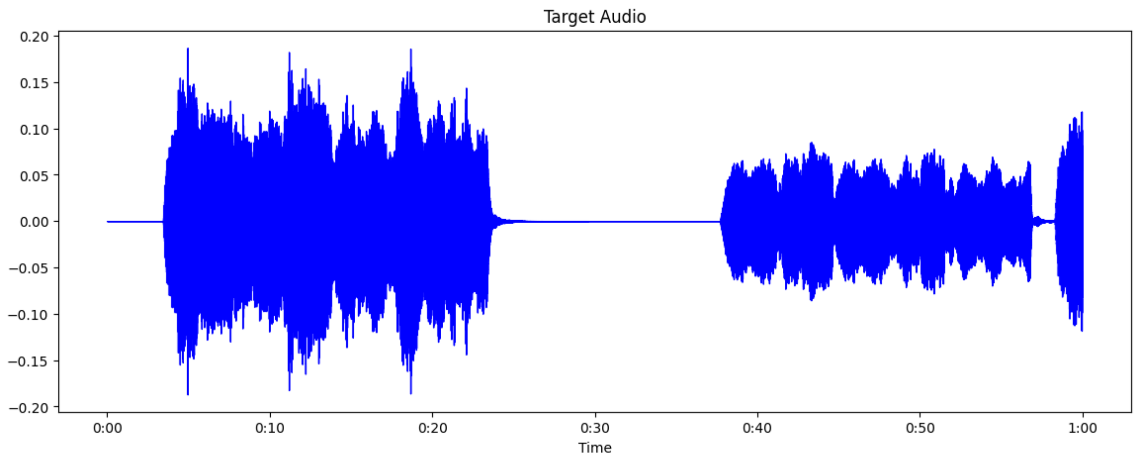


Figure 23: Target Audio waveform

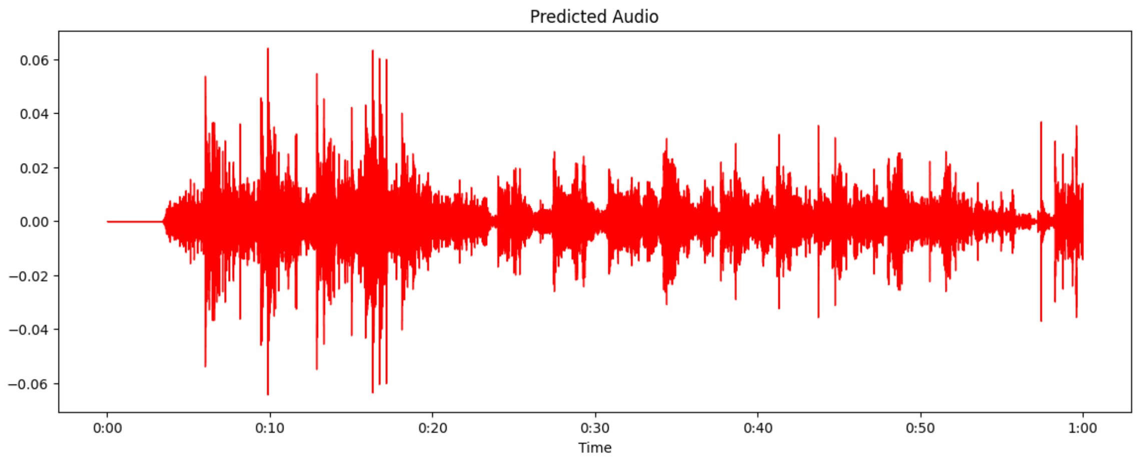


Figure 24: Models Predicted Audio waveform with noices

support for extracting and transcribing less common instruments such as the violin from polyphonic music. Prior research predominantly emphasized common instrument stems like piano, bass, drums, and vocals, leaving violin transcription in complex musical contexts underexplored.

By introducing a targeted two-stage pipeline consisting of the MIA-Separator for source separation and the MIA-Transcriptor for note-level transcription, this research demonstrated a viable and effective approach for capturing monophonic violin audio and converting it into symbolic musical notation. The separation model was specifically trained to handle the unique timbral characteristics of the violin, and the transcription module was optimized for detecting note onsets and durations within these isolated tracks.

The results confirm that combining a dedicated source separation stage with a tailored AMT model significantly enhances transcription accuracy, especially in multi-instrumental settings. This validates the research aim of developing a computational method capable of accurately transcribing musical notations from polyphonic recordings and lays the foundation for future extensions to support additional instruments or real-time applications.

6.4 Future Directions

While the current work lays a solid foundation, several promising avenues exist for future exploration:

- **Independent source separation:** Moving away from models like Spleeter for preprocessing, and instead focusing on independent source separation models tailored specifically for violin extraction.
- **Polyphonic transcription:** Extending the AMT system to handle polyphonic music transcription natively, which would eliminate the dependency on prior separation.
- **Hybrid architectures:** While transformer-based models show promise for scalability and generalization, the LSTM-based architecture used in this research offered a better performance trade-off on smaller datasets. Future designs can explore hybrid CNN+Transformer or LSTM+Transformer combinations to leverage both temporal awareness and attention mechanisms.
- **Data augmentation and transfer learning:** Incorporating synthetic data generation, pitch-shifting, and time-stretching can help diversify training data. Transfer learning from larger music datasets may also improve generalization.
- **Handling intermittent source presence:** Addressing the limitation where the MIA-Separator hallucinates violin sounds during silent regions. Future work can investigate source presence detection mechanisms or dynamic masking strategies to suppress output in non-target or silent regions, improving robustness and realism.

References

Abdallah, S. M. and Plumbley, M. D. (2004), Polyphonic transcription by non-negative sparse coding of power spectra, *in* 'International Society for Music Information Retrieval Conference'.
URL: *https://api.semanticscholar.org/CorpusID:6308680*

Agarwal, A., Li, B., Menon, V., Peddinti, N., Qian, Y., Torres, D. and Dasgupta, S. (2022), Implementing transformer architectures for audio source separation, *in* '2022 IEEE MIT Undergraduate Research Technology Conference (URTC)', pp. 1–5.

Bello, J., Daudet, L., Abdallah, S., Duxbury, C., Davies, M. and Sandler, M. (2005), 'A tutorial on onset detection in music signals', *IEEE Transactions on Speech and Audio Processing* **13**(5), 1035–1047.

Benetos, E., Dixon, S., Giannoulis, D., Kirchhoff, H. and Klapuri, A. (2013), 'Automatic music transcription: challenges and future directions', *Journal of Intelligent Information Systems* **41**, 407 – 434.
URL: *https://api.semanticscholar.org/CorpusID:207169189*

Bittner, R. M., Bosch, J. J., Rubinstein, D., Meseguer-Brocal, G. and Ewert, S. (2022), A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation, *in* 'Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)', Singapore.

Bittner, R. M., Salamon, J., Tierney, M., Mauch, M., Cannam, C. and Bello, J. P. (2014), Medleydb: A multitrack dataset for annotation-intensive mir research., *in* 'ISMIR', Vol. 14, pp. 155–160.

Carden, J. and Cline, T. (2019), 'Absolute pitch: Myths, evidence and relevance to music education and performance', *Psychology of Music* **47**(6), 890–901.
URL: *https://doi.org/10.1177/0305735619856098*

Chandna, P., Miron, M., Janer, J. and Gómez, E. (2017), Monoaural audio source separation using deep convolutional neural networks, *in* 'Latent Variable Analysis and Signal Separation'.
URL: *https://api.semanticscholar.org/CorpusID:27739613*

Cwitkowitz, F., Cheuk, K. W., Choi, W., Martínez-Ramírez, M. A., Toyama, K., Liao, W.-H. and Mitsufuji, Y. (2024), Timbre-trap: A low-resource framework for instrument-agnostic music transcription, *in* 'ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', IEEE, pp. 1291–1295.

Gerhard, D. (1998), 'Computer music analysis'.

Gers, F., Schmidhuber, J. and Cummins, F. (1999), Learning to forget: continual prediction with lstm, *in* '1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)', Vol. 2, pp. 850–855 vol.2.

Gu, X., Ou, L., Zeng, W., Zhang, J., Wong, N. and Wang, Y. (2024), 'Automatic lyric transcription and automatic music transcription from multimodal singing', *ACM Transactions on Multimedia Computing, Communications and Applications* .

Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., Elsen, E., Engel, J. and Eck, D. (2019), Enabling factorized piano music modeling and generation with the MAESTRO dataset, *in* 'International Conference on Learning Representations'.
URL: *https://openreview.net/forum?id=r1lYRjC9F7*

Hennequin, R., Khlif, A., Voituret, F. and Moussallam, M. (2020), 'Spleeter: a fast and efficient music source separation tool with pre-trained models', *Journal of Open Source Software* **5**(50), 2154. Deezer Research.
URL: *https://doi.org/10.21105/joss.02154*

Jansson, A., Humphrey, E. J., Montecchio, N., Bittner, R. M., Kumar, A. and Weyde, T. (2017), Singing voice separation with deep u-net convolutional networks, *in* 'International Society for Music Information Retrieval Conference'.
URL: *https://api.semanticscholar.org/CorpusID:28087825*

Klapuri, A. and Davy, M. (2006), *Signal Processing Methods for Music Transcription.*

Klapuri, A., Eronen, A. and Virtanen, T. (2001), 'Automatic transcription of music'.

Klingseisen, J. and Plumbley, M. D. (2004), Experiments on musical instrument separation using multiple-cause models, *in* 'Experiments on musical instrument separation using multiple-cause models'.
URL: *https://api.semanticscholar.org/CorpusID:62788417*

Kokkidou, M. (2022), *Music Definition and Music Education: many perspectives, many voices, many questions*, Greek Society for Music Education (GSME).

Li, L., Ni, I. and Yang, L. (2017), Music transcription using deep learning, *in* 'Music Transcription Using Deep Learning'.
URL: *https://api.semanticscholar.org/CorpusID:42595154*

Liu, X., Kong, Q., Zhao, Y., Liu, H., Yuan, Y., Liu, Y., Xia, R., Wang, Y., Plumbley, M. D. and Wang, W. (2023), 'Separate anything you describe', *arXiv preprint arXiv:2308.05037* .

Manilow, E., Wichern, G., Seetharaman, P. and Le Roux, J. (2019), Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity, *in* 'Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)', IEEE.

Martin, K. (2000), 'A blackboard system for automatic transcription of simple polyphonic music'.

Martin, K. D. and Kim, Y. E. (1998), 'Musical instrument identification: A pattern-recognition approach', *The Journal of the Acoustical Society of America* **104**(3_Supplement), 1768–1768.

Mcleod, A. and Steedman, M. (2018), Evaluating automatic polyphonic music transcription, *in* 'International Society for Music Information Retrieval Conference'.
URL: *https://api.semanticscholar.org/CorpusID:53874015*

Nam, J., Ngiam, J., Lee, H. and Slaney, M. (2011), A classification-based polyphonic piano transcription approach using learned feature representations, *in* 'International Society for Music Information Retrieval Conference'.
URL: *https://api.semanticscholar.org/CorpusID:14661817*

Nugraha, A. A., Liutkus, A. and Vincent, E. (2016), 'Multichannel audio source separation with deep neural networks', *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **24**, 1652–1664.
URL: *https://api.semanticscholar.org/CorpusID:14827170*

Patel, P., Shah, S., Prasad, S., Gada, A., Bhowmick, K. and Narvekar, M. (2024), 'Audio separation and classification of indian classical instruments', *Engineering Applications of Artificial Intelligence* **133**, 108582.

Poliner, G. E. and Ellis, D. P. (2006), 'A discriminative model for polyphonic piano transcription', *EURASIP Journal on Advances in Signal Processing* **2007**, 1–9.

Rafii, Z., Liutkus, A., Stöter, F.-R., Mimilakis, S. I. and Bittner, R. (2017), ‘The MUSDB18 corpus for music separation’.
URL: *https://doi.org/10.5281/zenodo.1117372*

Riley, X. and Dixon, S. (2024), ‘Reconstructing the charlie parker omnibook using an audio-to-score automatic transcription pipeline’, *arXiv preprint arXiv:2405.16687* .

Shuttleworth, T. and Wilson, R. (1993), ‘Note recognition in polyphonic music using neural networks’.

Sigtia, S., Benetos, E., Boulanger-Lewandowski, N., Weyde, T., Garcez, A. S. d. and Dixon, S. (2015), A hybrid recurrent neural network for music transcription, *in* ‘2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)’, IEEE, pp. 2061–2065.

Sigtia, S., Benetos, E. and Dixon, S. (2016), ‘An end-to-end neural network for polyphonic piano music transcription’, *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **24**(5), 927–939.

Smaragdis, P. and Brown, J. (2003), Non-negative matrix factorization for polyphonic music transcription, *in* ‘2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)’, pp. 177–180.

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., Uhlich, S., Liutkus, A. and Mitsufuji, Y. (2019), ‘Open-unmix-a reference implementation for music source separation’, *Journal of Open Source Software* **4**(41), 1667.

Sturm, B. L., Santos, J. F., Ben-Tal, O. and Korshunova, I. (2016), ‘Music transcription modelling and composition using deep learning’, *arXiv preprint arXiv:1604.08723* .

Ullrich, K. and van der Wel, E. (2018), ‘Music transcription with convolutional sequence-to-sequence models’.

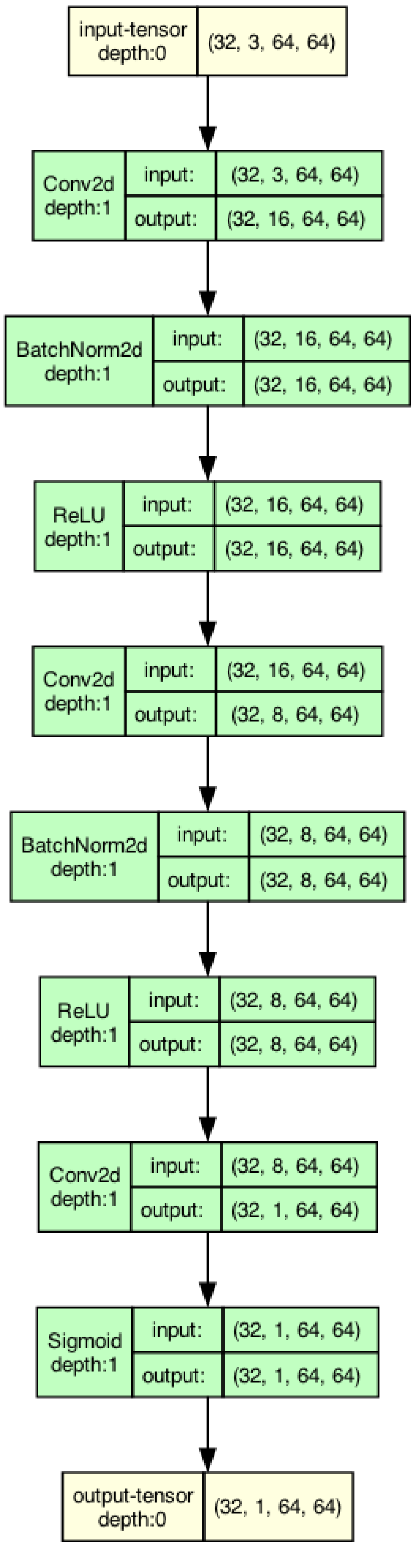
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. and Polosukhin, I. (2017), Attention is all you need, *in* ‘Proceedings of the 31st International Conference on Neural Information Processing Systems’, NIPS’17, Curran Associates Inc., Red Hook, NY, USA, p. 6000–6010.

Vincent, E., Gribonval, R. and Févotte, C. (2006), 'Performance measurement in blind audio source separation', *IEEE Transactions on Audio, Speech, and Language Processing* **14**, 1462–1469.
URL: *https://api.semanticscholar.org/CorpusID:9882068*

Watcharasupat, K. N., Wu, C.-W., Ding, Y., Orife, I., Hipple, A. J., Williams, P. A., Kramer, S., Lerch, A. and Wolcott, W. (2023), 'A generalized bandsplit neural network for cinematic audio source separation', *IEEE Open Journal of Signal Processing* .

Xi, Q., Bittner, R. M., Pauwels, J., Ye, X. and Bello, J. P. (2018), Guitarset: A dataset for guitar transcription, *in* 'International Society for Music Information Retrieval Conference'.
URL: *https://api.semanticscholar.org/CorpusID:53875945*

Appendix A



input-tensor depth:0 (32, 3, 64, 64)
Conv2d depth:1 input: (32, 3, 64, 64) output: (32, 16, 64, 64)
BatchNorm2d depth:1 input: (32, 16, 64, 64) output: (32, 16, 64, 64)
ReLU depth:1 input: (32, 16, 64, 64) output: (32, 16, 64, 64)
Conv2d depth:1 input: (32, 16, 64, 64) output: (32, 8, 64, 64)
BatchNorm2d depth:1 input: (32, 8, 64, 64) output: (32, 8, 64, 64)
ReLU depth:1 input: (32, 8, 64, 64) output: (32, 8, 64, 64)
Conv2d depth:1 input: (32, 8, 64, 64) output: (32, 1, 64, 64)
Sigmoid depth:1 input: (32, 1, 64, 64) output: (32, 1, 64, 64)
output-tensor depth:0 (32, 1, 64, 64)

Figure A: CNN Model - 01

Appendix B

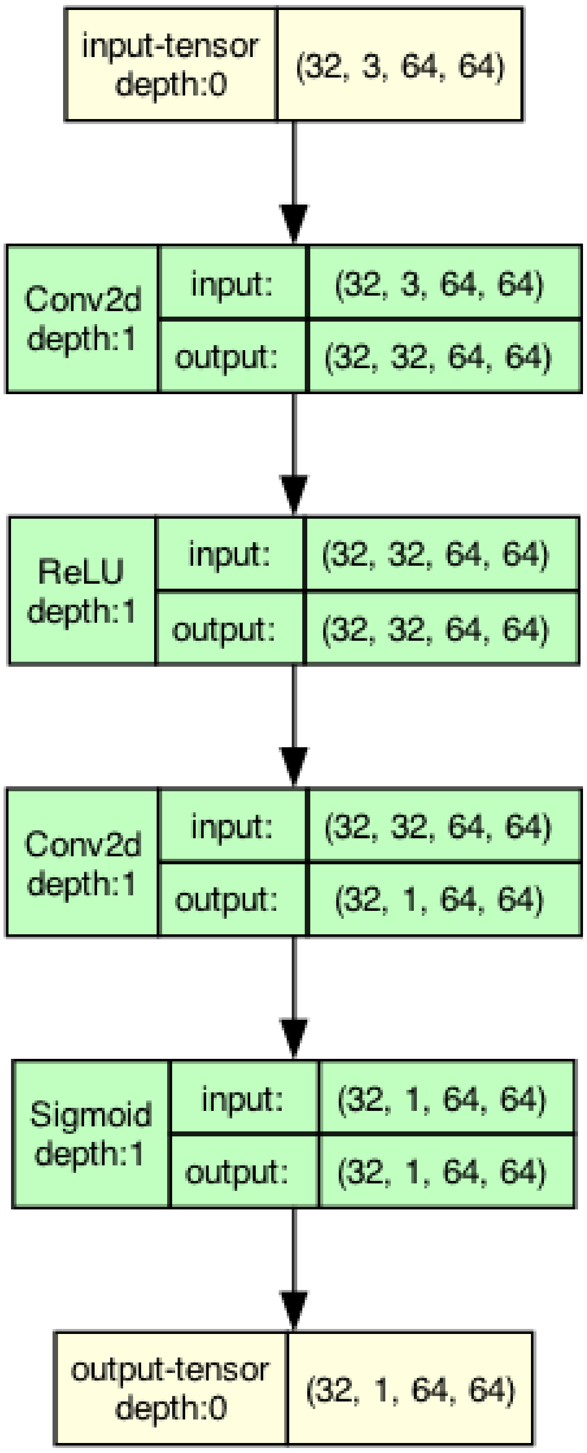


Figure B: CNN Model - 02

Appendix C

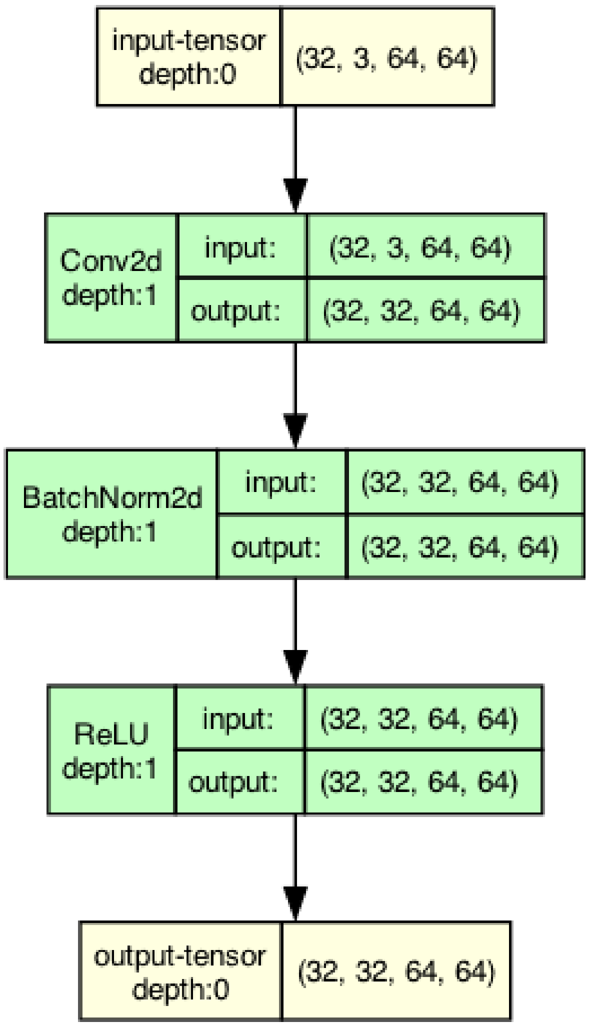


Figure C: CNN Model - 03

Appendix D

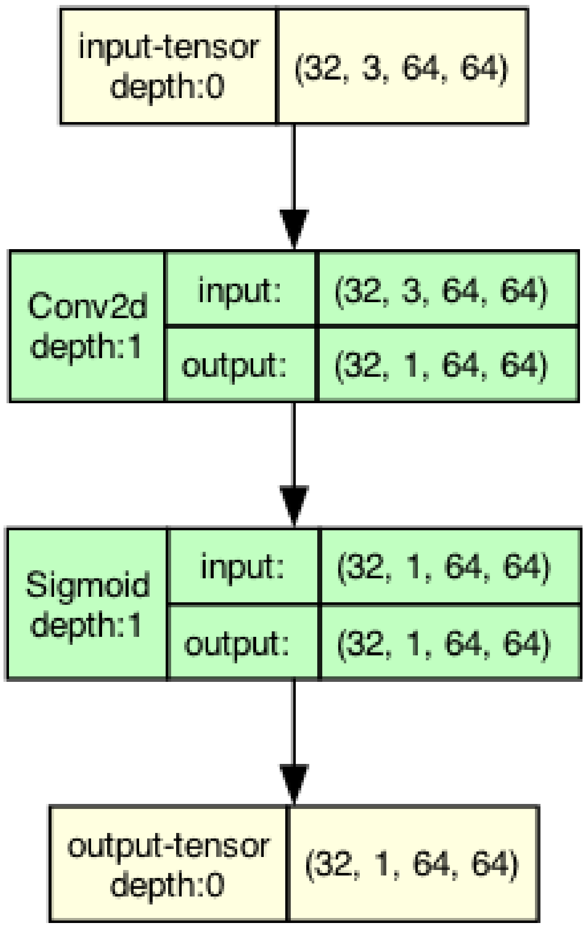


Figure D: CNN Model - 04

Appendix E

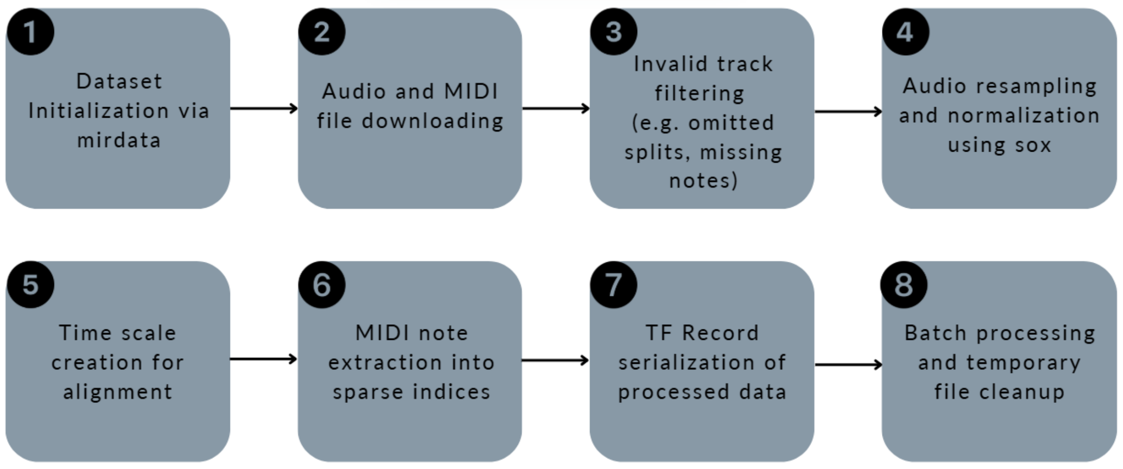

1
Dataset Initialization via mirdata
2
Audio and MIDI file downloading
3
Invalid track filtering (e.g. omitted splits, missing notes)
4
Audio resampling and normalization using sox
5
Time scale creation for alignment
6
MIDI note extraction into sparse indices
7
TF Record serialization of processed data
8
Batch processing and temporary file cleanup

Figure E: Slakh Data Preprocessing Pipeline