TESIS DE LA UNIVERSIDAD DE ZARAGOZA

2025 172

Carlos Hernández Oliván

Advances in Music Technology with Deep Learning: from Music Analysis to Production

Director/es
Beltrán Blázquez, José Ramón

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

Universidad de Zaragoza
Servicio de Publicaciones

ISSN 2254-7606

Universidad Zaragoza
1542

Tesis Doctoral

ADVANCES IN MUSIC TECHNOLOGY WITH DEEP LEARNING: FROM MUSIC ANALYSIS TO PRODUCTION

Autor

Carlos Hernández Oliván

Director/es

Beltrán Blázquez, José Ramón

UNIVERSIDAD DE ZARAGOZA
Escuela de Doctorado

Programa de Doctorado en Ingeniería Electrónica

2025

Universidad Zaragoza
1542

Tesis Doctoral

Advances in Music Technology with Deep Learning: from Music Analysis to Production

Autor

Carlos Hernández Oliván

Director

José Ramón Beltrán Blázquez

Escuela de Ingeniería y Arquitectura

2024

Advances in Music Technology with Deep Learning: from Music Analysis to Production

Author

Carlos Hernandez Olivan

Supervisor

Dr. Jose Ramon Beltran Blazquez

Department of Electronic Engineering and Communications

University of Zaragoza

A thesis submitted for the degree of

Doctor of Philosophy (PhD)

October 2024

Acknowledgments

First, to my family, especially to my mother, father, brother, and grandparents, for supporting me since the very beginning. There are no words to express my gratitude for having them in my life. To my grandfather who awakened in me the passion for technology and engineering.

To my supervisor José Ramón Beltrán, who has been my first and only support at the University of Zaragoza. For supporting me academically when anyone else did and awakened my interest in music research.

To my viola teacher Juan Luis Arcos, whose patience, understanding, and encouragement allowed me to finish my studies at the conservatory. Without him, it would have not been possible. For being one of the best professors I had ever and will have.

To the AIBeatz team, especially to Daniel and Derek Tingle for teaching me and for being so patient with me.

To the Sony R&D Music Technology team in Tokyo for being the first international company that believed in me, and for their hospitality, availability and help during the internship. Especially to Koichi Saito, Kinwai Cheuk, Marco A. Martinez Ramirez, Weihsiang Liao, Naoki Murata, Chieh-Hsin Lai and Yuki Mitsufuji.

To the NTT Communication Sciences Laboratory in Kyoto for accepting me as a Vulcanus participant. To Tsubasa Ochiai, Naohiro Tawara, Tomohiro Nakatani and Shoko Araki for helping and teaching me during my stay there. To Marc Delcroix, for being the best supervisor I could have had inside and out the company, and for teaching, encouraging and helping me with everything I needed.

皆様に心より感謝申し上げます。

To my friends, Rafa y Bruno who pursued together with me the same studies and were always there supporting me, and to everyone including friends, supervised students, interns, researchers and administration staff who I met during my interships and stays; Nai, Astrid, Dani, Dan, Pedro Ramoneda, Rose Battle, Yutong (Kelly) He, Dongjun Kim, Frank Cwitkowitz, Junyi Peng and Alexis Plaquet and many more.

Resumen

La creación de música mediante tecnologías de Aprendizaje Profundo comprende una serie de técnicas y aplicaciones que van desde el análisis musical hasta la producción. En esta tesis doctoral se abordan distintas aplicaciones que, en su conjunto, forman parte y tienen como objetivos la mejora del flujo de trabajo de profesionales de la música como compositores o productores musicales y la democratización de la música para hacerla más accesible a todos los públicos. Es relevante recalcar que, a pesar de la automatización que puedan significar los modelos utilizados en esta tesis y en otros trabajos de referencia, es siempre necesaria la supervisión de humanos, dado que el propósito de estas técnicas es el de potenciar la creatividad de las personas.

En esta tesis doctoral se presentan ditintos temas relacionados con el proceso de creación musical, desde el análisis, que es el primer paso, pasando por la composición hasta la producción. Cada uno de estos grupos contiene distintos temas de investigación que tratan de resolver varios problemas como por ejemplo el análisis de la estructura formal de piezas musicales como técnica dentro del análisis musical, o la restauración de audio y extración de sonidos relacionadas con la producción de audio.

Primero, se abordan técnicas relativas al análisis musical. Dentro de estas técnicas se encuentra, entre otros, el análisis de la estructura de piezas musicales o análisis formal. Este paso es de vital importancia para entender la música, desde el estilo de un compositor determinado hasta las estructuras que conforman un tipo de pieza como puede ser la conocida forma sonata. Bajo este marco, abordaremos el análisis formal de canciones de pop desde la señal de audio teniendo como objetivo principal la segmentación de dichas canciones en base a su estructura.

En segundo lugar, se presenta un software de código abierto que contiene la implementación de diversas técnicas del estado del arte relativas al análisis musical incluyendo algoritmos de estimación de tonalidad, o a la composición musical simbólica mediante la implementación de varios *tokenizers* que convierten la entrada de música en ficheros MIDI a *tokens* que alimentan modelos de composición. Además, incluye diversos tipos de ficheros de salida como protobuf y JSON para poder trabajar con dicho software en proyectos de investigación o comerciales.

Para finalizar, se abordan dos técnicas relativas a la producción musical. La primera de ellas está relacionada a la restauración de audio, en concreto, de pistas de voz cantada. Mediante novedosas técnicas de modelos de difusión, podemos restaurar las pistas de voz que estén deterioradas por algún proceso como el *clipping* o saturación, o incrementar su ancho de banda. Para evitar la necesidad de reentrenar el modelo para cada degradación, se presentarán técnicas de muestreo posterior o *posterior sampling* y se optimizará el número de etapas durante el proceso de muestreo. Estas técnicas son de vital importancia en estudios de grabación donde la calidad musical debe de cumplir unos estándares para poder ser comercialiaza. Por otro lado, se aborda la extracción de sonidos, tanto de un canal como de dos canales, emulando el sistema auditivo humano. Utilizando una función de pérdida específica, podemos reducir el tiempo estimado de llegada de la señal. Para señales de un solo canal, utilizaremos un modelo fundacional del audio o *audio foundation model* para mejorar la calidad de la extracción cuando utilizamos un *enrollment clue*, o pista de referencia del sonido que se quiere extraer. Esta tecnología es importante en aparatos auditivos para personas con déficits auditivos, aplicaciones de realidad virtual y para productores musicales a la hora de hacer remezclas de canciones.

Como conclusión, en esta tesis se presentan diversos temas dentro del marco de las tecnologías musicales utilizando varias técnicas de aprendizaje profundo. Dado que el proceso de creacion musical precisa de un estudio del lenguaje y análisis musical exhaustivo, en esta tesis se presentan técnicas que van en la línea de la democratización de la música para hacerla más accesible a cualquier público. Pese a que cada tecnología es estudiada de forma independiente, en el futuro, se pueden consolidar como un solo modelo *end-to-end* que ayude a las personas a crear música de una forma más creativa y accesible para todos los públicos.

Preface

The creation of music using Deep Learning technologies encompasses a range of techniques and applications, from musical analysis to production. This doctoral thesis addresses various applications that collectively aim to enhance the workflow of music professionals such as composers or music producers, and to democratize music to make it more accessible to all audiences. It is important to emphasize that, despite the automation that the models used in this thesis and other reference works may imply, human supervision is always necessary, as the purpose of these techniques is to enhance people's creativity.

This doctoral thesis presents different topics related to the musical creation process, starting from analysis, which is the first step, through composition to production. Each of these groups contains various research topics that seek to solve several problems, such as analyzing the formal structure of musical pieces as a technique within musical analysis, or audio restoration and sound extraction related to audio production.

First, techniques for musical analysis are addressed, including the analysis of the structure of musical pieces, also known as formal analysis. This step is crucial for understanding music, from the style of a particular composer to the structures that comprise a type of piece, such as the well-known sonata form. Within this framework, we will address the formal analysis of pop songs from the audio signal, with the main objective of segmenting these songs based on their structure.

Secondly, an open-source software is presented that contains the implementation of various state-of-the-art techniques related to musical analysis, including tonality estimation algorithms, or symbolic musical composition through the implementation of various tokenizers that convert music input in MIDI files to tokens that feed composition models. Additionally, it includes various types of output files such as protobuf and JSON to allow working with this software in research or commercial projects.

Finally, two techniques related to musical production are addressed. The first one is related to audio restoration, specifically for sung voice tracks. Using novel diffusion model techniques, we can restore voice tracks that are degraded by processes such as clipping or saturation, or increase

their bandwidth. To avoid the need to retrain the model for each degradation, posterior sampling techniques will be presented, and the number of steps during the sampling process will be optimized. These techniques are vital in recording studies where musical quality must meet certain standards to be marketable. On the other hand, sound extraction is addressed, both from a single channel and from two channels, emulating the human auditory system. By using a specific loss function, we can reduce the estimated arrival time of the signal. For single-channel signals, we will use an audio foundation model to improve the quality of extraction when using an enrollment clue, or reference track of the sound to be extracted. This technology is important in hearing aids for people with hearing deficits, virtual reality applications, and for music producers when remixing songs.

In conclusion, this thesis presents various topics within the framework of music technologies using several deep learning techniques. Since the process of musical creation requires a thorough study of language and musical analysis, this thesis presents techniques aimed at democratizing music to make it more accessible to any audience. Although each technology is studied independently, in the future, they can be consolidated into a single end-to-end model that helps people create music in a more creative and accessible way for all audiences.

Contents

List of Tables

List of Figures

Part I

INTRODUCTION

Chapter 1

INTRODUCTION, OVERVIEW AND MOTIVATION

Music comprehends a wide variety of intrinsic elements which make it complex to understand and analyze. It is so wide, that it can be studied from several perspectives which include art and science among others. Exploring music from a scientific perspective reveals the influential role of Artificial Intelligence in understanding, composing, and transforming music. The tools that can be developed from this perspective can offer benefits not only to music composers, producers or practitioners, but it can also be used for educational purposes [46, 136].

According to our understanding about how humans learn to compose music, we characterize the process of music creation as a tree [86], wherein building blocks rely on one another. First, the music analysis block comprehends the tasks that make us understand how music is composed in terms of its principles: harmony, *form* or structure, rhythm and texture [113]. After, we can address music composition, where this analysis plays an important role in understanding how music should be composed or, in the case of AI systems, evaluated to determine the similarity between the generated music and human-composed music [142]. Music composition, as well as analysis and production requires creativity [14]. To conclude, after music has been composed, we need to produce it to ensure it aligns with certain standards before distributing it. This phase also demands creativity, as it encompasses music editing or mastering which can fundamentally alter how the music is perceived by the listener [142].

In this thesis, as it is highlighted in Fig 1.1, we will present a scientific approach to music structure analysis II and production III from different domains: audio and symbolic. For simplicity, Fig. 1.1 does not contain all the topics belonging to Music Information Retrieval (MIR).

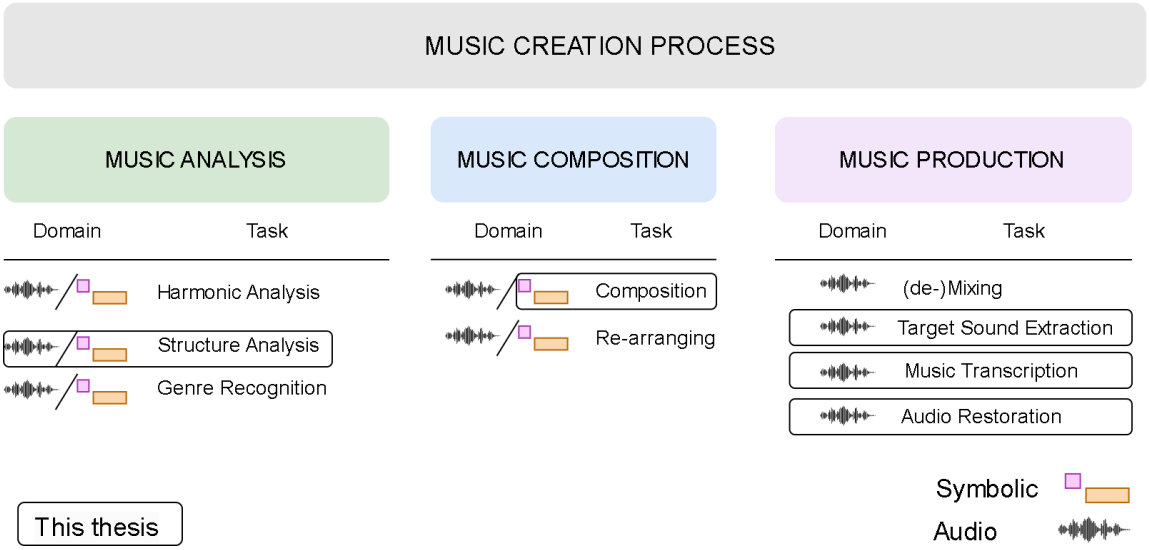

Figure 1.1: General topics of this thesis: music structure analysis, symbolic music analysis and composition, audio restoration and target sound extraction.

1.1 Music Analysis

The first part of this thesis focuses on music analysis, in particular, in Music Structure Analysis (MSA). MSA is an open problem in MIR that has been studied in recent years. MSA involves music principles like harmony or rhythm [113].

Music analysis is necessary to have a better understanding about music and how it has been composed in different periods or eras, and it can be applied to other tasks such as music composition. As happens with other tasks, MSA can be addressed from audio or symbolic representations [113]. We address audio MSA in Chapter 3.

In Western classical music, *form* refers to the structure of a musical piece, which consists on dividing music pieces into smaller sections or parts [86]. Music structure is hierarchical, and it can be segmented by different levels according to the semantic content of each segment: low, middle and high levels. The low level is related to the *motif detection*. Motifs are the shortest music parts in Western classical music pieces. They are developed into longer phrases, which correspond to the middle level. Phrases along with cadences, bridges, etc, compose a section. Sections are considered the highest level of structure and thus, they contain lots of information. Due to the challenges that MSA presents, it is still far from human-level performance. The main challenges that MSA presents are (i) each music genre or style has a different form or structure. Even for pieces from the same period or composer, the structure might change. (ii) Each piece is unique in terms of its

sections. This means that section lengths are different across pieces composed by one composer. The number of motifs, themes and/or musical phrases vary in each piece. (iii) The musical content of the sections that have the same labels changes in each piece too. This means that a section A in one piece can be completely different in terms of rhythm and harmony to other section A from other pieces. (iv) Section boundaries represent a small percentage of the data if we compare them to the number of notes in a piece, which makes difficult to train deep neural networks for section segmentation and/or labeling. (v) Music presents different structure levels across periods and styles, e.g., in Western classical music form we can find motifs, cadences or bridges, but in non-Western classical music this hierarchy is different. (vi) There are only a few datasets in MIR of symbolic music that contain structure annotations. Even if we can train supervised DL models with them, sometimes there are discrepancies between annotators which make this task more difficult to tackle and measure [36].

In particular, this thesis makes contributions to the *boundary detection* task (i) in the audio domain by extending a comparative analysis of the input combinations that are needed to train a Convolutional Neural Network (CNN), and (ii) extending the input context where the network can look at.

1.2 Music composition

The second part of this thesis focuses on Music Composition with deep learning. Music composition has been addressed from the early *algorithmic music* methods [109] to the latest approaches with Language Models (LMs) and Generative Models [46]. Music composition aims to generate new music. This music can be single or multi-instrument, only melodic, accompaniment or both. As for music analysis, we can also address this task from audio or symbolic domains. We can also find here the instrumentation and orchestration tasks.

Developing DL models for music composition has the potential to enhance the capabilities of music creators at all levels, from novices to seasoned professionals. This technology can offer valuable assistance throughout their creative processes and help combat the phenomenon known as *creative block* [29]. For this purpose, it is crucial to take into account the user's proficiency level, who will be engaging with the model, along with designing a user-friendly interface that maximizes every aspect and feature of the model.

In the recent years, music composition or generation has been addressed with Language Models such the Transformer [166]. These models were built in the context of the Natural Language Processing (NLP) domain. Considering the mapping between text and symbolic music were words can be understood as music events, Transformer-based models emerged as a possible solution in the

context of music composition [60, 173]. However, different processing representations or encodings have been proposed for this purpose [58] and there is still a room of improvement in what concerns to music encodings and models.

In spite that our contribution in this topic is targeted towards music composition, we include this part in the music analysis part (Part II), since we focus in music processing and analysis rather than the composition itself.

Our research in this area is targeted towards the development of a framework that includes state-of-the-art (i) tokenizers for pre-processing symbolic music, (ii) evaluation method for symbolic music composition, (iii) algorithms and methods for symbolic music analysis, (iv) Transformer-like architecture which can be trained to generate symbolic music.

1.3 Music Production

Music production comprehends the tasks related to modify an manipulate the music to let it ready for distributing it. The tasks that belong to this block can help music producers and also music practitioners [95].

The goal of the tasks that are related to music production is to disentangle different parts of the audio to be able to manipulate and re-use them. Some music production tasks are: Automatic Music Transcription, Source Separation or Target Sound Extraction [13], Audio Restoration [52, 104], Automatic Mixing and de-mixing [95], and effects modeling and removal. In this thesis, we focus on audio restoration and target sound extraction.

1.3.1 Audio Restoration

Audio restoration consists of restoring signals that are or have been degraded. This task is an inverse problem that can be addressed by many techniques, from supervised to unsupervised diffusion models. Common restoration tasks include inpainting, declipping and bandwidth extension [104].

In the past years, audio restoration tasks were addressed by methods which study only one task [66, 81, 105, 182]. Recent works on audio restoration use diffusion models that are task-agnostic. This means that we can address different tasks with the same pre-trained model. In addition, there have been many diffusion guidance methods proposed in the computer vision field that have not been explored for audio yet [91, 157], and might be applicable to audio.

Our research on this topic addresses singing voice restoration with diffusion models and multiple guidance methods. Our contribution is the following: (i) explore computer vision diffusion guidance and posterior sampling techniques for audio restoration and (ii) the development of a diffusion guidance algorithm which outperform previous works that use the same diffusion model.

1.3.2 Target Sound Extraction

Target sound extraction (TSE) consists of extracting a desired signal from a mixture of sounds and noise [13, 180]. This task is related to music production since with this technology, we can extract desired stems from a mixture or song and re-mix them to create new music.

We explored two different approaches in this topic: binaural target sound extraction with the development of a new loss function and single-channel sound extraction with an audio foundation model.

Binaural Target Sound Extraction

In the past years, different works proposed binaural processing by extending monoaural target speaker extraction systems in speech enhancement [41]. Recent works also addressed binaural TSE [169]. However, not using explicit losses for binaural cues can degrade the spatial metrics such as the *interaural phase difference* (IPD) or the *interaural level difference* (ILD). Some prior works explored the addition of IPD and ILD losses in speech enhancement [160], yielding a degradation of the signal-level metrics, while preserving spatial cues.

Our contribution to this field is the development of a loss based on the interaural time difference (ITD) when doing TSE in binaural mixtures. This loss reduces the ITD while not degrading the signal-level metrics like the *signal-to-noise ratio* (SNR) and the *scale invariant signal-to-noise ratio* (SI-SNR).

Target Sound Extraction with Audio Foundation Model

Previous works in single-channel target sound extraction used enrollment [13] and one-hot clues [13, 168] to extract the desired sound from a mixture. Due to the wide variety of sounds, training a model from scratch can be challenging. However, audio foundation models trained with general audio can provide rich feature representations of sounds within a TSE system. In the speech domain, self-supervised models (SSL) have proven to improve the TSE performance [124]. We draw inspiration from that and explore how Masked-Modeling Duo (M2D), a foundation model which appears especially suited for the TSE task since it is trained using a dual objective consisting of sound-label predictions and improved masked prediction, is suitable for general sounds TSE.

Our contributions to this field are: (i) the development of a new TSE system that integrates the feature representation from M2D into SoundBeam [13], which is a strong TSE system that can exploit both target sound class labels and pre-recorded enrollments (or audio queries) as clues, and (ii) the integration of such system in an online TSE model, Waveformer [168] to improve the TSE performance in real-time systems.

We will discuss how these tasks could be merged together in a single model in Chapter 8 as a conclusion of this thesis.

1.4 Goal and overview

This thesis is divided in two main parts, the first one for music analysis and the second one for music production as follows:

- Part II is devoted to music analysis. In Chapter 3 we investigate music boundary detection, proposing a multi-input model to segment songs by their structure. In Chapter 4, we present an open source software that allows to pre-process, analyze, compose and evaluate symbolic music.
- Part III is devoted to music production. In Chapter 5 investigate signing voice restoration with novel diffusion inference techniques. In Chapter 6 we explore binaural target sound extraction, proposing a new loss function based on the interaural-time difference. In Chapter 7, we improve the overall performance of target sound extraction by using an audio foundation model along with two state-of-the-art TSE systems, one offline and another online models.

1.5 Contributions and measurable results

1.5.1 Publications

- A Review of Music Composition with Deep Learning (Chapter 2, Part I)
  - This work was published as a book chapter in the Springer book series *Signal and Communication Technology*, 2022 [46].
- Music Structure Detection (Chapter 3, Part II)
  - This work was published in the *International Journal of Interactive Multimedia and Artificial Intelligence*, 2021 [45]. This journal has an impact factor of 4.936, and its position in the JCR index is 48th out of 144 (Q2) in the category Computer Science, Artificial Intelligence (data from 2021).
- Symbolic Music Processing (Chapter 4, Part II)
  - This work was published in *SoftwareX*, 2023 [47]. This journal has an impact factor of 2.4, and its position in the JCR index is 54th out of 131 (Q2) in the category Computer Science, Information Systems (data from 2023).

  - Previous results were published in the *Workshop on Generative AI and HCI (GenAICHI)*, 2022 [51].

- Automatic Music Transcription (it was done during the PhD, however, it is not included in the thesis)
  - This work was published in *Electronics*, 2021. This journal has an impact factor of 2.690, and its position in the JCR index is 100th out of 164 (Q3) in the category Computer Science, Information Systems (data from 2021).

- Vocal Restoration with Diffusion Models (Chapter 5, Part III)
  - This work was published in the *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024 [52].

- Target Sound Extraction (Chapters 6, 7, Part III)
  - Binaural Target Sound Extraction work (Chapter 6) was published in the *IEEE 18th International Workshop on Acoustic Enhancement (IWAENC)*, 2024 [49].
  - Target Sound Extraction with an Audio Foundation Model work (Chapter 7) is submitted and currenlty under review in the *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025 [48].

1.5.2 Research stays and intersnhips

- April 2023 - July 2023 (3.5 months): Research Intern at the Music Foundation Team, Creative AI Laboratory, Sony R&D (Tokyo, Japan). Supervisor: Dr. Weihsiang Liao.
- September 2023 - August 2024 (8 months): Research Intern at NTT CSL (Kyoto, Japan). Supervisor: Dr. Marc Delcroix.
- January 2025 - February 2025 (2 months): Visiting Researcher at Queen Mary University of London. Supervisor: Dr. Iran R. Roman.

1.5.3 Supervised students

I directed the following bachelor's theses at the University of Zaragoza:

- 2024-02: María Iriarte Bernal. *Automatic Music Composition in Symbolic Format with Deep Neural Networks.* Grade 8.0/10.0

- 2023-06: Sergio Ferraz Laplana. *Drums Generation with Transformer-based Models.* Grade 9.0/10.0
- 2023-02: Carlota Laborda. *Detection of Boiling in Liquids on an Induction Hob using Acoustic Analysis and Artificial Intelligence.* Grade 8.0/10.0
- 2022-10: Sonia Rubio Llamas. *Music composition with transformer-based models.* Grade 9.0/10.0
- 2022-10: Pablo Castillón Ruiz. *Source separation and music transcription with deep learning.* Grade: 8.5/10.0
- 2022-09: Santiago Pallarés Arnal. *Instruments classification with deep learning.* Grade 8.75/10.0
- 2022-09: Jorge Bajén Gonzalo. *Harmonic Analysis of MIDI files.* Grade 8.5/10.0
- 2022-07: Jorge Abadías Puyuelo. *Study of deep learning-based techniques for music composition.* Grade: 9.0/10.0

I co-directed the following master's thesis at the Pompeu Fabra University of Barcelona:

- 2023: Quoc. *Automatic Score-to-Score Music Generation.* Grade: 8.0/10.0

I also participated in the supervision of two master theses:

- 2021: Carlos Hernández López. *Multitimbral transcription system based on deep neural networks.* Grade: 9.2/10.0
- 2021: Ignacio Zay Pinilla. *Comparative study of automatic score transcription systems using audio separation.* Grade: 8.2/10.0

1.5.4 Research projects

I participate in the following projects:

- Spanish Science, Innovation and University Ministry: RTI2018-096986-B-C31.
- Aragonese Government: AffectiveLab-T60-20R project.

1.5.5 Community Service

I was a volunteer in ISMIR 2021 conference.

I have been a reviewer for the following journals and conferences:

- Transactions on Audio, Speech and Language Processing (TASLP).
- Transactions of the International Society for Music Information Retrieval (TISMIR).
- IEEE Transactions on Neural Networks and Learning Systems (TNNLS).
- International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI).
- Applied Artificial Intelligence (AAI).
- Computer Music Journal (CMJ).
- IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

1.5.6 Others

During two years of my thesis, from 2021 to 2023, I worked full-time as a machine learning engineer at the AIBeatz start-up, which I co-founded. In that period, I developed the package presented in Chapter 4.

Chapter 2

STATE OF THE ART OF DEEP LEARNING FOR AUDIO APPLICATIONS

ABOUT THIS CHAPTER

The work presented in this chapter, although extended here for general audio applications, is inspired in the review published as a book chapter in the Springer book series *Advances in speech and music technology: computational aspects and applications* in 2022. This work was conducted entirely at the University of Zaragoza. Jose Ramon Beltran supervised the work.

C. Hernandez-Olivan and J. R. Beltran.
MUSIC COMPOSITION WITH DEEP LEARNING: A REVIEW
Advances in speech and music technology: computational aspects and applications, pages 25-50, 2022.

2.1 Introduction

Machine learning has become an important part in our daily life, being present in a wide range of tasks in our society. Its applications cover from refining search engine algorithms and filtering content on social media platforms to generating personalized recommendations. Modern consumer devices, including cameras and smartphones, increasingly leverage machine learning to enhance functionality. This technology plays a crucial role in tasks such as object detection in images, converting spoken language into text or aligning

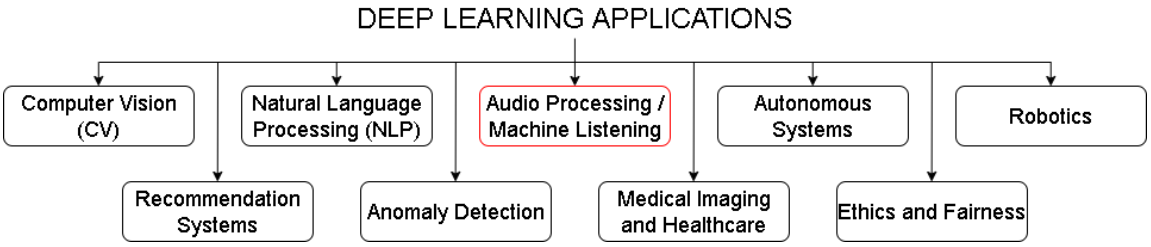

Figure 2.1: Deep learning fields.

news articles and posts with user preferences. A prominent subset of machine learning known as deep learning is at the cutting edge of these advancements, driving many of these applications [33, 85] (see Fig. 2.1).

Learning machines can automatically learn complex representations from raw data. This is called representation learning. Deep learning is a singular case of representation-learning which contains linear and non-linear modules that transform the input into more and more abstract representations to learn multiple levels of representation [85].

Deep learning advances the automatization of many tasks that in previous decades depended on careful engineering design and domain expertise. In the case of audio, the input can be a waveform [165]. In the first layer of representation, the model typically learns to detect basic features like changes in frequency or intensity, which might correspond to simple sound patterns such as tones. The second layer might identify more complex structures, such as harmonics or recurring sound patterns, regardless of small variations in timing or pitch. Subsequent layers could then recognize larger structures, like words, sentences, or entire musical phrases. A key feature of deep learning is that these hierarchical representations are learned directly from the audio data through a general-purpose learning procedure [85].

Many major advances have been done thanks to deep learning. In particular, it revolutionized the field of machine listening in audio generation [165], separation [92], classification [126], sound event detection [78] or automatic speech recognition [42]. Deep learning also helped in aiming for tasks that seemed unreachable by handcraft engineering, such as controllable music generation from audio [17] or real-time target sound extraction [168].

In this chapter, we describe the deep learning methods and their applications in the machine listening field. In particular, we will focus in the neural network architectures and applications that are used in this thesis.

2.2 Deep Learning Methods

In this section, we provide a classification of deep learning methods. We briefly describe the main methods and more in depth, the ones that are used in this thesis.

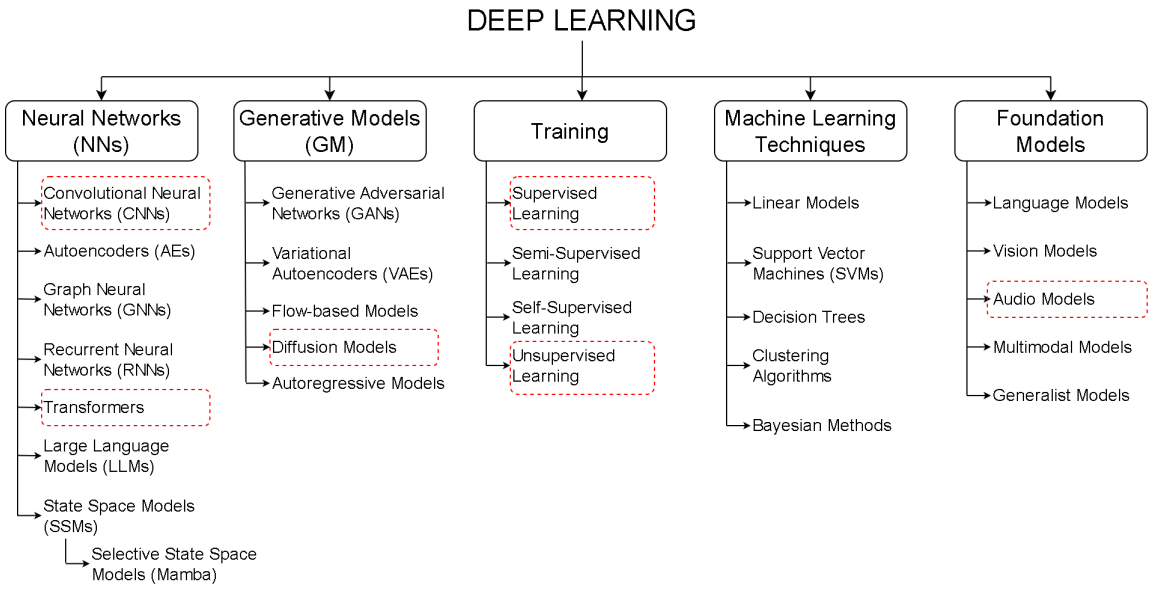

Figure 2.2: Key components of deep learning.

2.2.1 Deep Neural Networks

Deep neural networks (DNNs) are a class of deep learning models which consist of multiple layers [33]. Each layer transforms its input through learned weights and activation functions to produce increasingly abstract representations of data. DNNs can outperform traditional methods in many domains since they capture both low-level and high-level features. DNNs, as shown in Fig. 2.2, contain Convolutional Neural Networks (CNNs) [26, 85], Autoencoders (AEs), Graph Neural Networks (GNNs), Recurrent Neural Networks (RNNs) [3, 57], attention-based models such as Transformers [166], Large Language Models (LLMs) and State Space Models (SSMs).

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) were initially inspired by Neocognitron developed by Kunihiko Fukushima in the 1980s [26]. The idea was to use a hierarchical structure to process visual information in a similar way as humans do. After that, LeCun et al. in 1993 [85] developed LeNet, the first successful application of CNNs for handwritten digit recognition trained using backpropagation. LeNet demonstrated the effectiveness of convolutional layers for detecting local patterns (such as edges) while significantly reducing the number of parameters compared to fully connected networks. In machine listening, CNNs are the most-common neural network architecture due to its nature and ability to learn powerful representations. Common audio feaure representations like spectrograms or mel-frequency cepstral coefficients (MFCCs) are commonly used as input representations [45] (see Chapter 3). Also, CNNs are powerful enough to learn complex representations from directly audio waveforms, as Wavenet proved [165]. We use CNNs in the

encoder and decoder for target sound extraction [48, 49] (see Chapters 6, 7) CNNs can also be combined with other architectures such as Graph Neural Networks (GNNs) or attention-based models.

Autoencoders

Autoencoders [53] are a type of neural network commonly used for unsupervised learning. They excel in tasks such as reducing the dimensionality of data, learning efficient features, and reconstructing original inputs from compressed representations. An autoencoder consists of two parts: the encoder, which maps the input data into a lower-dimensional latent space, and the decoder, which attempts to rebuild the original data from this compressed form. By learning to encode and decode effectively, autoencoders capture essential features while filtering out noise. These networks find applications in areas like image denoising, anomaly detection, and generating new data. Autoencoder architectures, such as UNet [141], are also employed in various machine listening tasks. For example, we use these models in [52] (Chapter 5) and in [48, 49] (Chapters 6 and 7) for extracting target sounds from mixed audio, where the encoder handles the input mixture and the decoder reconstructs the isolated target sound after applying a mask to the audio's feature representation.

Graph Neural Networks

Graphs model data that forms interconnected relationships, such as social networks, molecular structures, or knowledge graphs. Unlike traditional data structures like sequences, graphs allow us to model complex, non-Euclidean data where nodes represent entities, and edges capture relationships or interactions between them. This flexibility makes graph-based models ideal for a wide range of applications, from social network analysis and recommendation systems to drug discovery and traffic prediction. Graph Neural Networks (GNNs) were first introduced by Scarselli et al. in 2009 [149]. They enabled machine learning models to effectively learn from graph-structured data. GNNs aggregate and transform information across nodes and their neighbors, capturing intricate dependencies. Later works made these methods more scalable and applicable to tasks like node classification, link prediction, and graph generation. Graph neural networks have been used in machine listening tasks such as music generation or symbolic music structure analysis [50, 73].

Recurrent Neural Networks

Recurrent Neural Networks (RNNs), until the Transformer was proposed, were the most used networks for sequence processing. Hochreiter and Schmidhuber introduced the long short-term memory (LSTM) in 1997 [57]. LSTMs can handle long-range dependencies in sequences, which allows to model sequential data across different domains including speech recognition, music generation, and audio classification. More recently, xLSTM [3] has been proposed to enhance the capabilities of traditional LSTMs by improving their performance on complex sequence tasks, including those found in audio and music processing.

Transformers

Transformers were introduced in 2017 by Google [166]. This model, first developed for natural language processing tasks, was one of the breakthrough novelties in deep learning. Transformers beat RNNs by using self-attention mechanisms [2] to handle long-range dependencies. After Transformer became increasingly popular across various DL fields, such as for the Vision Transformer (ViT) for images [19], a new family of models were proposed under the name of large language models (LLMs). Their ability of LLMs to process large amounts of data in parallel and model contextual relationships at scale makes them ideal for powering LLMs like GPT [130]. These models leverage the transformer's architecture to generate coherent and contextually rich text across various applications. Transformers have been widely used in audio applications such as music generation [60].

We use Transformer and attention layers in 4 (Chapter 4), where we implement a state-of-the-art symbolic music generation model based on a transformer decoder architecture, and in [48, 49] (Chapters 6 and 7) for real-time target sound extraction with a stat-of-the-art model called Waveformer [168], respectively.

Large Language Models

As we mentioned before, Large Language Models (LLMs) were developed thanks to the success of Transformers in various domains. The applications of LLMs go beyond language tasks, since researchers have found applications in many domains such as audio and music, were LLMs can be leveraged to condition music generation or separation with text or generate lyrics.

State Space Models

State space models provide a powerful framework for representing dynamic systems, allowing for the modeling of time-varying relationships in data. Mamba, introduced in 2021 [39], was the initial framework for integrating these models with reinforcement learning. In machine listening, state space models have been applied to tasks such as speech enhancement and music generation, showcasing their effectiveness in capturing temporal dynamics. The fact that these models are often lightweight, makes them powerful in real-time applications.

2.2.2 Generative Models

Generative models are a relatively new paradigm of learning which allows to generate new samples by learning a data distribution or predicting the next element in a sequence. Among the most well known generative models, we can find Variational Autoencoders (VAEs), Generative Adversarial Networks (GANS), Flow-based models, Diffusion Models and Autoregressive Models. It is important to highlight the fact that the previously describe neural network architectures are normally combined with these models, since generative models focus on how data is learned and neural networks can be understood as the processing blocks that allows that operation.

Variational Autoencoders

Variational Autoencoders (VAEs) [77] combine neural networks with probabilistic inference. They provided the a framework for learning latent representations while generating new data samples. In machine listening, VAEs have been used for tasks such as music generation [139] and audio signal enhancement, showcasing their ability to create diverse and high-quality outputs.

Generative Adversarial Networks

Generative Adversarial Networks (GANs) are models that can generate synthetic data by training two neural networks in a competitive setting. After VAEs were proposed in 2013, Goodfellow et al. proposed GANs in 2014 [34]. GANs were the first models that allowed creating high-quality, realistic data samples across various domains. In machine listening, GANs have been applied to tasks such as music generation [17] and audio synthesis, demonstrating their effectiveness in producing high-fidelity outputs.

Flow-based Models

Flow-based models are a generative modeling framework that use invertible transformations to map complex data distributions into simpler, tractable ones. Dinh et al. in 2014 [82, 138] introduced this approach, which allows for exact likelihood computation and efficient sampling. In machine listening, flow-based models have been applied to tasks such as waveform synthesis and speech generation. Voicebox [83] by Meta is an example of flow matching fo speech generation.

Diffusion Models

Diffusion models are a class of probabilistic generative models that transform noise into data by gradually reversing a diffusion process. These models were first introduced by Sohl-Dickstein et al. in 2015 [155] and have become increasingly popular due to their ability to generate high-quality samples.

Later improvements by Ho et al. in 2020 introduced Denoising Diffusion Probabilistic Models (DDPMs) [54], which have significantly improved the sampling quality. Song et al. in 2021 proposed score-based generative modeling [159], which allows to generate samples by estimating the score function, or gradient of the data distribution. The gradient of the data distribution is known as the score function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$, where x is the known data. Score-based models differ from other generative models in the way that they do not try to model the data distribution but they learn how the probability density function changes in different regions of the data space. This allows the model to generate data samples by gradually moving through the data space, following the score function in a manner similar to how diffusion models perform denoising. The generative process in score-based models can be seen as solving a stochastic differential equation (SDE) that starts from noise and evolves toward the data distribution. The forward process adds noise to data, as in diffusion models, but the reverse process involves solving a differential equation based on the score function. This approach has shown competitive performance, particularly in high-quality image and audio generation tasks, by leveraging the flexibility of continuous-time diffusion processes.

As we mentioned, diffusion models are based on two key processes:

1. **Forward Process**: This process gradually adds noise to the data $\mathbf{x}_0$ over T timesteps, with each timestep resulting in a slightly noisier version of the data. The process is defined by the following transition probability:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}),$$

where β_t is a variance schedule that determines how much noise is added at each timestep.

The marginal distribution of any noisy sample $\mathbf{x}_t$ given the original data $\mathbf{x}_0$ can be written as:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}),$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$.

2. **Reverse Process**: The goal is to learn a model that can reverse the noisy forward process and recover $\mathbf{x}_0$. The reverse process is modeled as a learned Markov chain parameterized by θ, and it is defined by the following transition probability:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)),$$

where μ_θ and Σ_θ are learned functions of the noisy sample $\mathbf{x}_t$ and the timestep t.

The model is trained by minimizing a variational bound on the negative log-likelihood of the data. A simplified objective is to predict the added noise at each timestep:

$$L(\theta) = \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_1,\ldots,\mathbf{x}_T)}\left[\sum_{t=1}^{T}\frac{1}{2\sigma_t^2}\|\mathbf{x}_0 - \hat{\mathbf{x}}_t^\theta\|^2\right].$$

The reverse SDE in score-based models is detailed in Chapter 5.

Diffusion models require to manually adjust the noise schedule to make them work for certain tasks. There are two noise schedules: variance-preserving (VP) and variance-exploding (VE). In the VP process, noise is gradually added to the data in such a way that the variance remains constant throughout the diffusion process. This makes it suitable for modeling and reconstructing data with controlled variance at each step, as used in Denoising Diffusion Probabilistic Models (DDPM) [54]. On the other hand, the VE process adds noise in a way that the variance increases exponentially with time. This schedule is useful for score-based models [159], as it allows for a more robust estimation of the score function over a wide range of noise levels.

Recently, Karras et al. introduced Elucidated Diffusion Models (EDM) [72], a new approach to improve both training efficiency and sample quality in diffusion models. EDM unifies the VP and VE frameworks and provides a principled method for adjusting noise schedules to maximize the quality of generated samples while minimizing computational costs. The main insight of EDM is the combination of continuous noise control and careful selection of the noise level at each step, improving the stability and performance of diffusion models in high-fidelity applications, including audio generation. Our voice restoration work with

diffusion models [52] (Chapter 5) uses the EDM scheduler.

EDM contains *posterior sampling* and *guidance* techniques. On one hand, *posterior sampling* consists of drawing samples from the posterior distribution of the model parameters θ after observing the data x. In diffusion models, this translates to sampling from the learned distributions that represent the diffusion process. This process follows the Bayes theorem $p(\theta|x) = \frac{p(x|\theta)\cdot p(\theta)}{p(x)}$. After observing data x, the posterior distribution $p(\theta|x)$ can be obtained based on the prior distribution $p(\theta)$ and the likelihood $p(x|\theta)$. The likelihood represents how likely the observed data is given the parameters, in other words, it shows how well the model with its parameters θ fits the observed data. On the other hand, *posterior guidance* uses the information obtained from the posterior distribution to influence the sampling process. Posterior guidance adjusts the diffusion steps based on the insights gained from the learned distribution instead of drawing samples randomly from the posterior. This method enhances the generative process by directing it towards higher-quality samples or specific outcomes of interest, thus improving the overall efficacy of the generative model. In summary, while posterior sampling focuses on obtaining samples from the posterior distribution, posterior guidance uses the knowledge of that distribution to infer the sampling process.

In Chapter 5, we also use different diffusion techniques that allows us to improve a state-of-the-art diffusion model for audio restoration. We explore how Posterior-guided Diffusion Models (ΠGDM) [157], Free-conditional Diffusion Models (FreeDOM) [174] and RePaint [91] can be used for this task. These techniques allow to improve the generation quality by leveraging posterior sampling and guidance techniques.

Posterior-guided Diffusion Models (ΠGDM) [157], focuses on improving the sampling process of diffusion models by incorporating posterior guidance. This technique allows to improve the generation quality without needing to retrain the diffusion model from scratch. By leveraging learned distributions from the diffusion process, ΠGDM can effectively navigate the generative space, allowing for more accurate reconstruction of data from noisy inputs. This technique is particularly useful in scenarios where only partial observations are available, enabling the model to infer missing information more effectively. In audio restoration, it is specially useful in bandwidth extension tasks, where part of the signal is removed above the cutoff frequency.

RePaint [91] proposes a two-stage sampling method that refines the outputs of diffusion models iteratively. This approach utilizes a reinforcement learning framework to guide the generation process, allowing the model to revisit and improve previously generated samples. This is done with a feedback loop. In the first step, the data is generated from the posterior distribution. In the second or *refinement* step, the model samples from the previous step output to correct errors and better align the samples with the target distribution.

The idea behind Free-conditional Diffusion Models (FreeDOM) [174] is to avoid retraining by constructing time-independent energy functions. However, our work in Chapter 5 draws inspiration from the definition of the three steps during the sampling process introduced by FreeDOM: chaotic, critical and refinement steps. This allows us to critically reduce the number of diffusion steps during sampling when using the RePaint technique which, as we mentioned before, doubles the number of sampling steps due to its feedback looping strategy.

Current diffusion models and sampling strategies comprehend techniques that allows the diffusion sampling step to be more efficient by maintain a consistent trajectory through the latent space of the model during the generative process. This family of models are called Consistency Models (CM) [158]. Newer models such as Consistency Trajectory Models [75], build upon CM, allows to dramatically reduce the sampling steps by using a single neural network to quickly generate high-quality outputs. This is a great advantage specially for audio generation, since we can get good quality outputs in just one sampling step [143].

Diffusion models have found significant applications in various tasks in machine listening, particularly where high-fidelity generation of waveforms is required. Their gradual denoising approach allows for the synthesis of complex audio signals such as speech and music. In music generation, diffusion models have been applied to generate high-quality music and sound design, preserving intricate time and frequency dynamics. By modeling the gradual transformation of noise into coherent musical structures, diffusion models offer a powerful solution for music generation tasks. DiffWave [79] is an example of an a non-autoregressive diffusion model for audio generation, capable of producing high-fidelity speech. In addition to music generation, diffusion models have also been used in speech synthesis and enhancement tasks, demonstrating their versatility and good performance in generating high-quality audio signals. In this thesis, we use diffusion model, in particular diffusion posterior sampling (DPS) in [52] (Chapter 5).

Autoregressive Models

Autoregressive modeling [131] is a powerful approach in generative modeling that focuses on sequentially generating data by predicting each element based on previous elements. This technique involves modeling the conditional probability of the next data point given a sequence of past points, which allows it to capture temporal dependencies effectively. Notably, autoregressive models model sequence data and are present in Transformers, like GPT for text [130]. In machine listening, autoregressive models have been used for tasks like music generation such as Multi-track Music Machine [21] and speech synthesis.

2.2.3 Learning Approaches in Deep Learning

In machine learning, or particularly in deep learning, there are various paradigms that guide how models learn from data. We can find supervised, semi-supervised, self-supervised or unsupervised learning strategies depending on the task and output that we want to obtain. In supervised learning, a model is trained on labeled data paired with the inputs. In contrast, unsupervised learning learns from unlabeled data, so no labels are needed. This approach is helpful to discover hidden patterns or structures within the data, such as clustering or dimensionality reduction. In addition, self-supervised learning also handles feature learning without requiring labeled datasets by predicting masked parts of the input based on other parts of the input. This means that the model generates a reconstruction of the input based on it. A state-of-the-art model trained under this paradigm is WaVLM [7] for speech. Opposite to this paradigm, semi-supervised learning combines both labeled and unlabeled data. Other learning strategies such as meta-learning focuses on developing models that can learn to learn; by training algorithms on a variety of tasks to enable quick

adaptation to new tasks with minimal data, and making it particularly useful in scenarios where rapid learning and generalization are essential. Meta-learning approaches are few-shot and zero-shot learning, which are useful to generalize to a wider variety of samples. These approaches are specially useful in general audio applications when we want to model a bast amount of different sounds which is unfeasible for training a deep learning model.

2.2.4 Foundation Models

Recently, large datasets for different domains are being proposed for learning general representations. In the audio domain, an example of such kind of dataset is Audioset [30]. Training deep learning models from scratch with these large datasets can be a challenge because the resources needed to do it are not available for all the community. Also, building generalist models can improve the performance of downstream tasks. To address this issue and leverage existing techniques to handle different tasks with a large model, Foundation Models were proposed. Foundation models are large-scale, pre-trained models that serve as a base for a wide range of tasks withing a domain or even across different domains. These models are trained on vast datasets to capture broad patterns, representations and features from the data. This makes them powerful and adaptable to various downstream tasks. Examples include GPT [130] for text, WavLM [7] for speech and Pre-trained Audio Neural Networks (PANNs) for sound event detection or classification [78], or Masked Modelling Duo [115], an audio foundation model that we use to improve target sound extraction performance [48] (Chapter 7), are examples of foundation models for general audio.

2.3 Machine Listening

We gave a general overview of different deep learning approaches and fields and discussed about the techniques and models that we use in this thesis. Now, we will focus on the machine listening field, which in spite of not having the impact or recognition as computer vision, it is gaining interest in the recent years and it is rapidly growing. In Fig. 2.3 we show the machine listening subfields that we discuss in this section.

2.3.1 Music Processing

Music processing, better known as Music Information Retrieval or Research (MIR) is a field focused on the analysis and extraction of music data. MIR combines techniques from signal processing, musicology and machine learning to analyze music in various formats such as symbolic, e.g. MIDI and scores, or audio recordings. MIR tasks include Music Structure Detection or Analysis (MSA) (Chapter 3), Automatic Music Transcription (AMT) or Music Generation or Composition (Chapter 4).

In this thesis, as we previously introduced in Chapter 2, we will focus on Music Boundary Detection in audio recordings, a subtask of Music Structure Analysis (see Chapter 3) and music processing, evaluation for symbolic music composition (see Chapter 4).

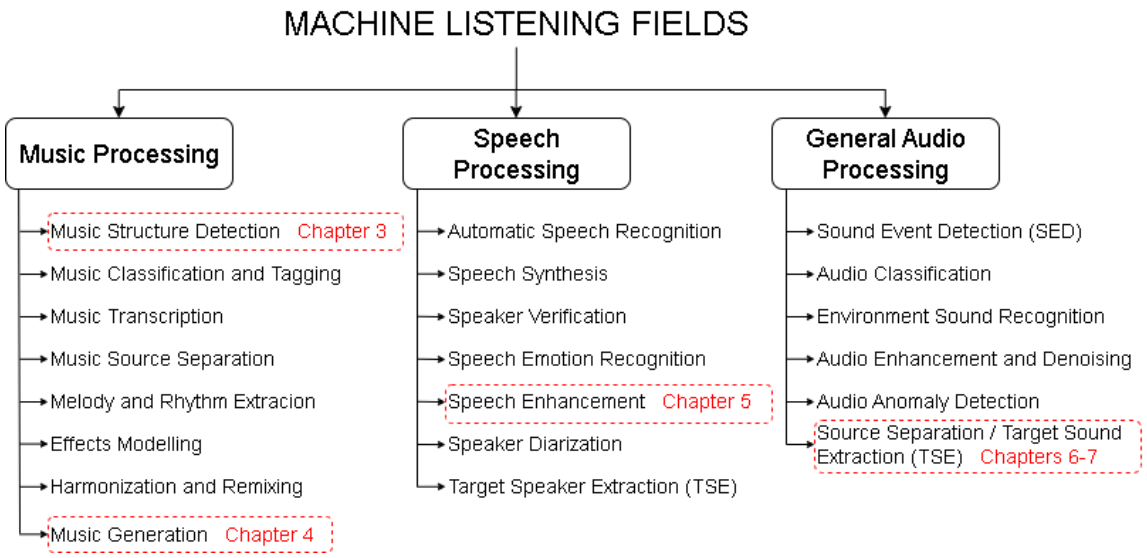

Figure 2.3: Machine listening subfields and tasks.

2.3.2 Speech Processing

Speech processing is the field that focuses on the identification, understanding and generation of human speech. This field combines linguistics, signal processing and machine learning to enable devices interacting like humans do. It covers tasks such as automatic speech recognition, diarization, synthesis, and speaker separation and verification.

This thesis, in spite that it focuses on music processing, it has a component closely related to speech enhancement. As we introduced previously in Chapter 2, we focus on singing voice restoration in Chapter 5, which can be considered as a music restoration or speech enhancement task since it aims to restore singing voice alone, not considering music background or accompaniment in the training nor testing data (see Chapter 5). For this purpose, we use a diffusion model with unsupervised training.

2.3.3 General Machine Listening

Music and speech processing fields can be included in a higher level classifier like general audio. However, we separate the three fields since we understand general audio as sounds that are not music or speech, e.g. environmental sounds. In this regard, we can find several applications and models that have been trained with such data. Sound Event Detection (SED), Audio Anomaly Detection or Target Sound Extraction (TSE) are tasks that are addressed with general sounds.

In this thesis, we focuses on Target Sound Extraction in Chapters 6 and 7 as we train the TSE system with general sounds such as *baby crying* or *car horn*. For this purpose, the models that we use contain convolutional and attention layers. In spite that TSE focuses mainly on general sounds, it is straightforward

to apply it to music data. However, in the music processing field, Source Separation (SS) is more common than TSE. SS focuses on the extraction of all the sources in a mixture, and TSE extracts only the desired sound. Since both approaches are similar, SS is more limited in terms of the sounds it can extract. In music processing, normally SS models can only extract up to 4 or 5 sources including piano, vocals, guitar and *others*. TSE can extract specific sounds but its limitation is the extraction of one source per inference step and also the need of a clue vector or audio recording of the target sound to extract.

2.3.4 Music Applications

In Chapter 1, we presented an overview of the topics that we cover in this thesis. The works presented in this thesis could serve for the following applications: i) Chapter 3 can help musicians to annotate datasets with structure annotations. However, we believe that the metrics should improve to be able to use such results in other useful tasks such as music generation conditioned with structure. ii) Chapter 4 can help processing symbolic music to train sequence models such as Transformer-based models. The package also includes key detection with state-of-the art techniques and it considers time signature changes along the symbolic files, which make it useful for condition the generation of music in different time signatures or keys. Also, it can serve as an evaluation tool for symbolic music composition systems. iii) Chapter 5 can help music producers by cleaning or restoring unclean voice recordings. This is helpful for remixing and can also safe time and resources during recording sessions. iv) Chapter 6 can help in virtual reality (VR) or 3D audio applications, and in hearables or cochlear implants since its lightweight setting. v) Chapter 7 is useful in several applications including music production to remix extracted stems, assistive technologies where isolating sounds in crowded environments is crucial or in surgical healthcare systems for surgical instrument tracking during surgeries.

2.4 Current and Future Research

This thesis makes contributions in different music and audio applications. In spite that music, speech and general sounds are investigated as separated topics as we mentioned before, we believe that future works in these fields could be unified. This can happen in several ways, e.g. building larger foundation models trained on music, speech and general audio data. Audio foundation models to unify speech, music and arbitrary sounds.

Part II

MUSIC ANALYSIS

Chapter 3

MUSIC BOUNDARY DETECTION

ABOUT THIS CHAPTER

The work presented in this chapter has been published in the journal *The International Journal of Interactive Multimedia and Artificial Intelligence* (IJIMAI) in 2021. This work was conducted entirely at the University of Zaragoza. I proposed the idea of the main research topic, implemented the methods and conducted the experiments. David Diaz helped with the implementation and Jose Ramon Beltran supervised the work.

C. Hernandez-Olivan, J. R. Beltran, and D. Diaz-Guerra.
MUSIC BOUNDARY DETECTION USING CONVOLUTIONAL NEURAL NETWORKS: A COMPARATIVE ANALYSIS OF COMBINED INPUT FEATURES.
International Journal of Interactive Multimedia and Artificial Intelligence. Vol. 7 (2), 2021

3.1 Introduction

This is the first chapter in this thesis devoted to music analysis. As we described in Chapter 1, music analysis is the first step in the music creation process (Fig. 1.1). It allows to understand how music is composed which is benefitial for other tasks such as music generation.

Music structure analysis (MSA) consists on extracting all the possible information about a music piece attending to its hierarchical structure. MSA is a complex problem since it involves the analysis of different music principles like harmony and rhythm. Analyzing music helps us understand how composers create

music throughout history and this knowledge can be used to write new music. There are two main approaches to analyzing music: using the written score (symbolic representation) and using the waveform (audio representation).

As discussed in the introduction 1, MSA presents diverse challenges. Music genres and even individual pieces can have unique structures. Sections within a piece might have the same label (like "Section A") but sound completely different across pieces. Additionally, the way music is divided into sections can differ between Western and non-Western styles, adding another layer of complexity. Finally, there is a lack of large, consistent datasets for music structure, making it hard to train computer models effectively.

In this chapter, we will focus on boundary detection. Boundary detection consists of segmenting a music piece by its structure. After this step is completed, we can label the segmented segments according to their similarity. This last step is named *labeling* or *clustering*, however, it is not part of the boundary detection task.

This chapter aims to segment the structure of music pieces. In particular, we study the comparison of different methods of boundary detection between the musical sections with Convolutional Neural Networks.

Our contributions are the following:

- We present a multi-input model for music boundary detection with a long-context network.
- We compare different inputs in the CNN with an ablation study.
- We open source the code and weights for the shake of reproducibility. To our known it is the first open source implementation of music boundary detection with CNNs.

3.2 Related Work

3.2.1 Unsupervised Methods

Several studies have been done in the field of structure recognition in music since self-similarity matrices (SSMs) [23] and self-similarity lag matrices (SSLMs) [35] were introduced. Self-similarity Matrices have been used under the name of Recurrence Plot for the analysis of dynamic systems [181], but their introduction to the music domain was done by Foote [23] in 1999. Since then, there have appeared different techniques for computing these matrices.

Before the introduction of the SSMs and SSLMs, the studies were based on processing spectrograms [176]. In recent years SSMs and SSLMs calculated from audio features in combination with spectrograms have proven better performance [37].

Paulus et al. [121], summarized unsupervised MSA methods in three approaches based on: novelty, homogeneity and repetition. The **novelty-based** approach consists on detecting the transitions between contrasting parts [113]. The techniques to address this approach go from checkerboard-like kernel methods introduced by Foote [24] to new techniques like multiple-temporal-scale kernel [71], which outperformed previous works by proposing a fusion of the novelty and repetition approaches. The **homogeneity-based**

approach is based on the identification of sections that are consistent with respect to their musical properties [113]. These methods use Hidden Markov Models [87, 89, 137] or combinations of SSMs [98, 161]. The **repetition-based** approach refers to finding recurring patterns. These methods apply a clustering algorithm to the SSMs or SSLMs [90, 97, 99, 119, 162]. They are more applicable for labeling the structural parts of music pieces rather than precise segmentation which is required in the boundary detection task.

3.2.2 Supervised Methods

Boundary detection can be accomplished with different techniques. One of the crucial parts in this task is to find an optimal input representation of the audio. Previous works use CNNs trained with mel-scaled log-magnitude spectograms (MLSs) [163], Self-similarity lag matrices (SSLMs) in combination with the MLSs [37], and also combining these matrices with chroma features [36].

In Tables 3.1 and 3.2 we show a recap of the results of previous works that have been done in boundary detection using both unsupervised and supervised neural networks. Results and algorithms nomenclature in Table 3.1 have been extracted from MIREX's campaigns of different years. The results obtained with unsupervised methods on Table 3.1 are not as high as the results obtained with supervised neural networks because the main goal of the unsupervised methods is not the boundary detection (segmentation) but full structure identification (including labeling).

3.3 Music Boundary Detection with CNNs

3.3.1 Self-Similarity Matrices

Supervised neural networks learn from input representations and comparing the model output with the ground truth. The inputs for boundary detection are spectrograms or SSMs which are computed during a pre-processing step (see Fig. 3.1).

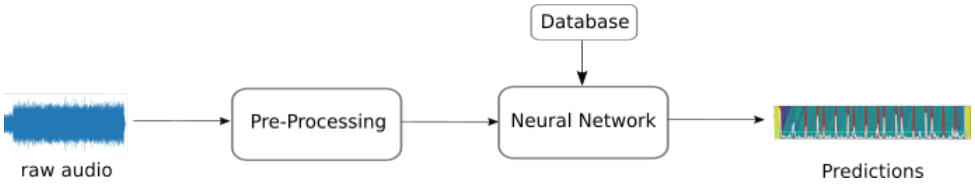

Figure 3.1: General scheme of supervised neural networks.

SSMs are obtained by computing a similarity function that is applied to audio features. The similarity between two feature vectors $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{y} \in \mathbb{R}^N$ is a function that can be expressed as:

$$\mathbf{S} = \delta(u, v), \tag{3.1}$$

[1] https://www.music-ir.org/mirex/wiki/⟨year⟩:MIREX⟨year⟩_Results - headland "Music Structure Segmentation Results".

Table 3.1: Results of boundary detection of previous studies for "Full Structure" and "Segmentation" tasks.

Unsupervised Methods							
Year[1]	**Algorithm**	**Input**	**Method**	**F-measure (F_1) for Testing Databases**			
				MIREX	**RCW-A**	**RCW-B**	**SALAMI**
2009	PK [120]	MFCCs, chromas	*Fitness function*	0.27	-	-	-
2010	MND1 [96]	MFCCs, Discrete Cepstrum	*HMM*	0.325	0.359	-	-
2011	SBVRS1 [147]	Chord estimation	*Viterbi*	0.231	0.324	-	-
2013	MP2 [99]	MLS	*Fisher LD*	0.281	0.355	0.278	0.317
2014	NB1 [111]	MFCCs + chromas	*Check. kernel*	0.289	0.352	0.269	0.299
2015	CC1 [6]	Timbre-type histograms	*HMM*	0.197	0.224	0.203	0.213
2016	ON2 [110]	CQT	*LDA*	0.259	0.381	0.255	0.299
2017	CC1 [6]	Timbre-type histograms	*HMM*	0.201	0.228	0.192	0.212
Supervised Neural Networks							
2014	SUG1 [151]	MLS	CNN	0.434	0.546	0.438	0.529
2015	GS1 [38]	MLS + SSLMs	CNN	0.523	0.697	0.506	0.541

where $\mathbf{S} \in \mathbb{R}^{M \times W}$, M and W are the audio features dimensions and δ is a distance function.

In the literature, this distance is usually calculated as the Euclidean distance:

$$\delta_e(\mathbf{u}, \mathbf{v}) = \|\mathbf{v} - \mathbf{u}\|, \tag{3.2}$$

or the cosine distance δ_c:

$$\delta_c(\mathbf{u}, \mathbf{v}) = 1 - \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \cdot \|\mathbf{v}\|}. \tag{3.3}$$

Once the SSM is computed, we can define a self-similarity lag matrix (SSLM) [35]. The SSLM is a matrix that represents the similarities between low-level features of one time step and previous time steps, up to a certain *lag time*. This allows to compute the relationship between current events and their repetitions in the future. SSLMs can be computed directly from audio features or from SSMs.

Let's define a non-square SSLM as $\mathbf{L} \in \mathbb{R}^{H \times I}$ where I in the lag time factor. We can compute each element of $\mathbf{L}$, $l_{i,j}$ as:

Table 3.2: Results of previous works in boundary detection task for ±0.5s time-window tolerance. It is only showed the best F-measure result of each reference for each database.

Year	Input	Method	Train Set	F-measure (F_1) for different Databases			
				MIREX	RCW-A	RCW-B	SALAMI
Unsupervised Methods							
2007	MFCCs, chromas, spectrogram [162]	-	-	-	-	0.378	-
2011	MFCCs, chromas [146]	-	-	-	-	0.356	-
Supervised Neural Networks							
2014	MLS [150]	CNN	SALAMI	-	-	-	0.465
2015	MLS + SSLMs [37]	CNN	SALAMI	-	-	-	0.523
2015	MLS + PCPs + SSLMs [36]	CNN	SALAMI	-	-	-	0.508
2017	MLS + SSLMs [10]	CNN	SALAMI	-	-	-	0.291

$$l_{i,j} = s_{k+1,j}, \tag{3.4}$$

where $i = 1, ..., H$, $j = 1, ..., I$ and $k = i + j - 2 \bmod H$. This matrix is the input of our model described in Section 3.4.

Once the input matrices of the pre-processing step are obtained, they are padded and normalized to form the input of a CNN. The obtained predictions are post-processed with a peak-picking and threshold algorithm to obtain the final predictions. We define this process in Section 3.4.

3.4 Methods

This work is based on previous works [37, 163] which proposed a pre-proscessing method to obtain the SSLMs from MFCCs features. We extend these works by calculating the SSLMs from chroma features and applying also the Euclidean distance that has not been considered in preliminary studies. The method that we used to generate the SSLMs[2] is based on [36, 37, 152]. The general scheme of the pre-processing block is shown in Fig. 3.2, in which each background color contains the steps that are necessary to compute each of the inputs: MLS (green), SSLM from Chromas (orange) and SSLM from MFCCs (blue). The red background in the max-pooling blocks refers to the 2 variants done in this work: `2pool3` is the one showed

[2] https://github.com/carlosholivan/SelfSimilarityMatrices

in the scheme, while `6pool` is computed by applying the max-pooling of factor 6 in the first red block and removing the second red block of the scheme.

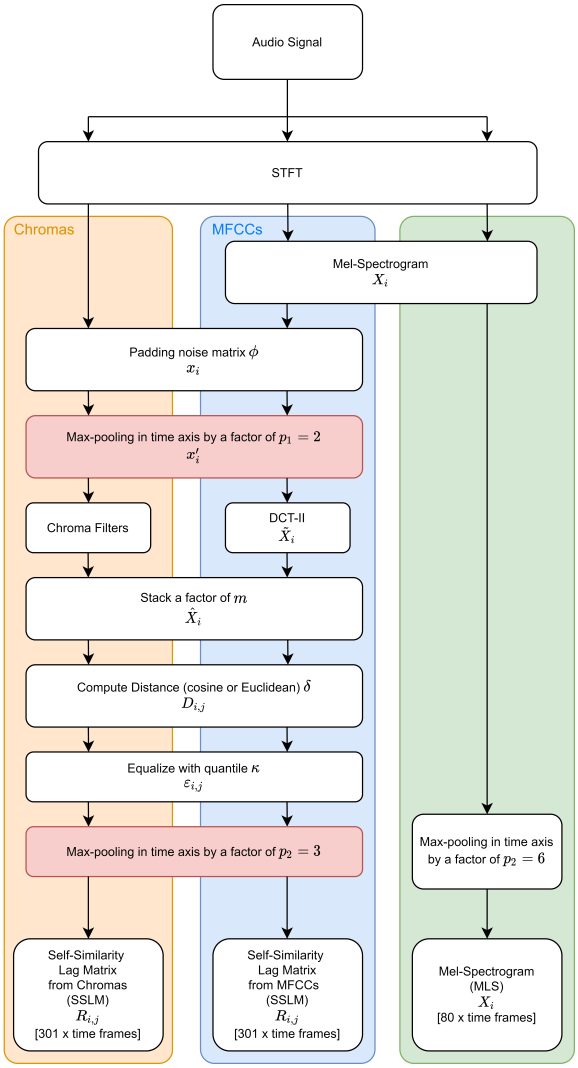

Figure 3.2: General block diagram of the pre-processing block in Fig. 3.1.

3.4.1 Self-Similarity Lag Matrix from MFCCs

The first step of the pre-processing part is to extract the audio features. To do that, we first compute the the Short-Time-Fourier-Transform (STFT). We then compute a mel-scaled filterbank and we scale logarithmically the amplitude magnitudes to obtain the mel-spectrogram (MLS).

Given a signal $\mathbf{r}$, we can compute its MSL as:

$$\mathbf{X} = \text{STFT}(\mathbf{r}) \cdot \mathbf{F}, \tag{3.5}$$

where $\mathbf{X} \in \mathbb{R}^{P \times D}$, P is the number of filter bands, D the frequency bins of the STFT and $\mathbf{F}$ is the Mel filter bank matrix.

After obtaining $\mathbf{X}$, we pad noise of -70dB with a duration of K frames to the pooled matrix before its first time frame. Being $\mathbf{x}_i \in \mathbb{R}^D$ the i-the frame of the MLS, we pad the noise as follows:

$$\check{\mathbf{x}}_i = \mathbf{\Phi}_i \parallel \mathbf{m}_i, \tag{3.6}$$

where $\mathbf{\Phi} \in \mathbb{R}^{K \times P}$. The elements of $\mathbf{\Phi}$ equal to -70dB. $\check{\mathbf{X}} \in \mathbb{R}^{(P+K) \times D}$ is the output matrix. $\mathbf{\Phi}_i \in \mathbb{R}^P$ is derived from the matrix $\mathbf{\Phi} \in \mathbb{R}^{K \times P}$.

Then, we apply a max-pool of a factor of p_1 in the time dimension. Being $\mathbf{m}_i$ the i-th frame of the MLS, we apply the max-pooling as follows:

$$\mathbf{x}'_i = \max_{j=1 \dots p_1} (\check{\mathbf{x}}_{(i-1)p_1+j}), \tag{3.7}$$

where $\mathbf{x}'_i$ is the i-th frame of $\mathbf{X}' \in \mathbb{R}^{((P+K)/p_1) \times D}$ after the max-pooling operation.

We now apply a Discrete Cosine Transform of type II to each frame omitting the first element.

$$\tilde{\mathbf{x}}_i = \text{DCT}^{(\text{II})}(\mathbf{x}'_i)_{[2 \dots P]}, \tag{3.8}$$

where $\tilde{\mathbf{x}}_i$ is the i-th frame of $\tilde{\mathbf{X}}' \in \mathbb{R}^{((P+K)/p_1) \times D}$ after after computing the DCT.

Now we stack the time frames by a factor m so to compute $\hat{\mathbf{X}} \in \mathbb{R}^{[(P-1)m] \times [(P+K+L)/p_1]}$. Note that $P+K$ is the number of frames before the max-pooling operation in Eq. 3.7 and I is the lag factor in frames.

$$\hat{\mathbf{x}}_i = [\tilde{\mathbf{x}}_i^{\text{T}} \parallel \tilde{\mathbf{x}}_{i+m}^{\text{T}}]^{\text{T}}, \tag{3.9}$$

The L matrix is obtained by calculating a distance between the vectors $\hat{\mathbf{x}}_i$. In our work, we use two different distance metrics: the Euclidean distance and the cosine distance.

We compute the distance between $\hat{\mathbf{X}}_i$ and $\hat{\mathbf{X}}_{i-l}$ using the distance metric δ:

$$D_{i,l} = \delta(\hat{\mathbf{X}}_i, \hat{\mathbf{X}}_{i-l}), \quad l = 1 \dots \left\lfloor \frac{L}{p_1} \right\rfloor, \tag{3.10}$$

where δ is the distance metric as defined in Eq. 3.2 and Eq. 3.3.

Then, we calculate an equalization factor $\varepsilon_{i,l}$ with a quantile κ of the distances $\delta(\hat{\mathbf{X}}_i, \hat{\mathbf{X}}_{i-j})$ for $j = 1 \dots \left\lfloor \frac{L}{p} \right\rfloor$:

$$\varepsilon_{i,l} = Q_\kappa \left(D_{i,l}, \cdots, D_{i,\left\lfloor \frac{L}{p} \right\rfloor} \parallel D_{i-l,1}, \cdots, D_{i-l,\left\lfloor \frac{L}{p} \right\rfloor} \right), \tag{3.11}$$

We now remove the first L/p lag bins in the time dimension of the distances matrix D and in the

equalization factor matrix ε, and we apply Eq. 3.7 with max-pooling factor p_2. Finally we obtain the SSLM applying Eq. 3.12.

$$R_{i,l} = \sigma\left(1 - \frac{D_{i,l}}{\varepsilon_{i,l}}\right) \tag{3.12}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$.

We add the noise in the time axis to the begin and end of the SSLM and MLS by padding them with $\gamma = 50$ time frames of pink noise at the beginning and end of the MLS matrix. Then we normalize each frequency band to zero mean and unit variance for MLS and each lag band for the SSLMs. Note also that if there are some time frames that have exactly the same values, the cosine distance would output not-a-number values. We avoid this by converting all this values into zeros as the last step of the SSLM computation.

3.4.2 Self-Similarity Lag Matrix from Chromas

The process of computing the SSLM from chroma features is similar to the method explained previously. The difference lies in padding the STFT instead the mel spectrogram, since we do not need to compute the mel spectrogram to obtain the chroma features. After applying the max-pooling in Eq. 3.7, we compute the chroma filters instead of computing the DCT in Eq. 3.8. The rest of the process is the same as described in section 3.4.1.

All the parameters used to obtain the Self-Similarity Matrices are summarized in Table 3.3. In addition to using the Euclidean and cosine distances, and MFCC and chroma audio features, we compare two pooling strategies. The first one is to make a max-pooling of factor $p_1 = 6$ to the STFT (from MLS calculation), and to the chromas or MFCCs for the SSLMs computation, as it is described in Eq. 3.7. The other pooling strategy is the one showed in Fig. 3.2, where we first do a pooling of $p_1 = 2$ and then a pooling of $p_2 = 3$ once the SSLMs are obtained. We denote these pooling variants as `6pool` and `2pool3` respectively. The total time for processing all the SSLMs (MFCCs and cosine distance) was a factor or 4 faster for `6pool` than `2pool3` because applying a higher padding factor in Eq. 3.7 makes the size of the matrices D and ε lower. Therefore, the calculation of these matrices take more time but it also implies a loss in the resolution that can affect the accuracy of the model as [37] remarks.

3.4.3 Model

The model proposed in this paper (Fig. 3.3) is similar to the model proposed in [37, 163]. It is composed by a CNN which relevant parameters are shown in Table 3.3. The difference between this model and the model proposed in [37, 163] is that our final two layers are not dense layers but convolutional layers in the time dimension. Since we do not crop the inputs and get a single probability value at the output, the input of our model is the matrix of a whole song. The last layer of our implementation outputs one feature map that is passed through a sigmoid function which outputs the boundary probability of each time frame of the

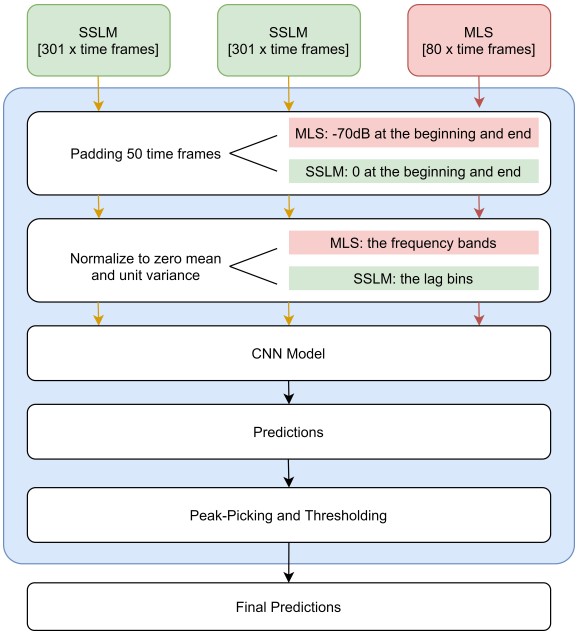

Figure 3.3: General block diagram of our model.

entire music piece, so the output of the model is a vector of length equal to the time frames of the input. This differs from other models where the output is the boundary probability of the segmented part of the input. The general schema of the CNN is shown in Fig. 3.4. Our model is also similar to another work [10] which re-implemented previous works [37, 163]. However, our model presents a larger receptive field as the dilation factor is done in the SSLM matrix branch but we do it also in the MSL branch.

3.4.4 Peak-Picking

Peak-picking consists on selecting the peaks of the output signal of the CNN that will be identified as boundaries of the different parts of the song. Each boundary on the output signal is considered true when no other boundary is detected within a time tolerance. The application of a threshold helps to discriminate boundary values that are not higher than an optimum threshold.

3.5 Experiments and Results

3.5.1 Dataset

We train the model with different input configurations described in Section 3.4 on a subset of the Structural Analysis of Large Amounts of Music Information (SALAMI) dataset [154]. In spite that SALAMI dataset contains 1048 double annotated pieces, we could obtain 1006 pieces since the dataset does not provide the

audio files due to copyright restrictions. For the training, we used the text files of labels from annotator 1. For the songs that were not annotated by annotator 1, we used the annotations from annotator 2.

It is important to highlight that, as described in [10], previous works [36, 37, 163] use a private dataset for training and the results reported in such works could not be reproduced by [10]. Therefore, we re-implemented the work presented in [37] and propose our improvements training our models in publicly available SALAMI pieces and annotations. We split our 1006 SALAMI audio tracks into 65%, 15% and 20%, resulting in 650, 150 and 206 pieces for training, validation and testing respectively.

3.5.2 Labelling Process

Following previous works [163], it is necessary to transform the labels of the SALAMI text files into Gaussian functions so that the Neural Network can better predict the boundary positions. We set the center values of the Gaussian functions by transforming the labels in seconds into time frames as showed in Eq. 3.13. Vector $\mathbf{z} \in \mathbb{R}^Q$ with Q the number of annotations, contains the center of the gaussians. In Eq. 3.13, $\mathbf{w}$ is the vector containing the labels in seconds extracted from SALAMI text files (named "functions") and p_1, p_2, γ are the pooling factors and final padding described in Section 3.4, h is the hop length and sr the sampling rate.

$$\mathbf{z} = \frac{\mathbf{w}}{p_1 \cdot p_2} + \frac{h \cdot sr}{\gamma}. \tag{3.13}$$

Then, we apply a gaussian function with standard deviation $\sigma = 0.1$ and μ_i equal to each label value in Eq.3.13. The i-th element of the gaussian vector containing the labels can be expressed as:

$$\mathbf{g}_i = \mathcal{N}(\mathbf{z}_i, \mu_i, \sigma^2), \tag{3.14}$$

with

$$\mathcal{N}(\mathbf{z}, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{\mathbf{z}-\mu}{\sigma}\right)^2}, \tag{3.15}$$

where μ is a vector of $\frac{\mathbf{z}\cdot\gamma + \frac{w}{2}}{sr}$ frames from $i = 1...\left[\frac{N}{p_1 \cdot p_2}\right]$.

To train the model, we removed the first tag from each text file due to the proximity of the first two tags in almost every file and the uselessness of the Neural Network identifying the beginning of the file. We also want to highlight the fact that we train with full songs and not with short segments that only contain one label as previous work did. This allows the network to learn more context which is crucial in MSA.

3.5.3 Model Configuration

We resampled all the songs in the SALAMI database at a single sampling rate of 44100 Hz. The window size for spectral analysis is set $w = 46$ ms with 50% overlap between consecutive windows. This results in a hop length of $h = 23$ ms. The lag factor is set to $I = 14$s .

For spectral feature downsampling, two separate pooling factors are used. The pooling factor of `6pool` strategy is set to $p_1 = 6$. Additionally, for the strategy named `2pool3` we use $p_1 = 2$ and $p_2 = 3$ along the

time dimension.

The final features are constructed by stacking $m = 2$ consecutive time frames after applying the pooling. A quantile value of $\kappa = 0.1$ is used for potential outlier handling. Finally, we use $\gamma = 50$ frames of zero padding are applied at the end to ensure consistent feature dimensions.

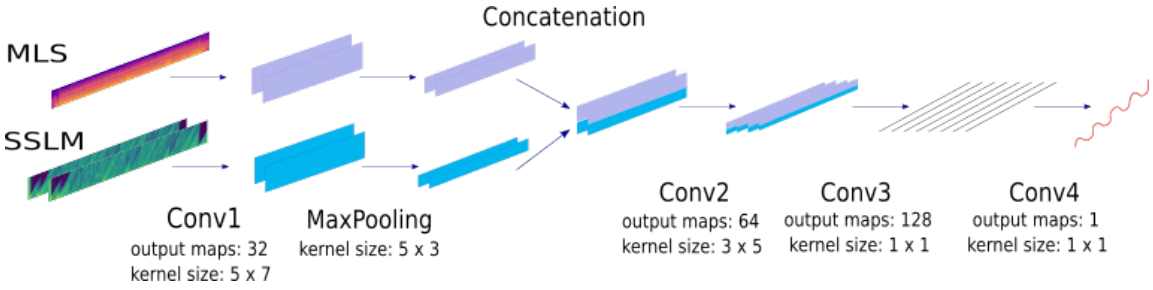

Figure 3.4: Convolutional Neural Network proposed in this Chapter. The main parameters are presented in Table 3.3.

We trained our CNN with *binary cross entropy* or BCEwithLogitsLoss in Pytorch as the loss function which in Pytorch implementation includes a sigmoid activation function in the last layer of the neural network. We use a learning rate of 0.001 and Adam optimizer [76]. We perform *early-stopping* to determine the best-performing model. The SSLMs and MLS have to be passed to the GPU one by one because they have different lengths, which means that 1 song is passed forward and backward through the NN at once. However, to get more robust gradients and a more stable optimization process, the optimizer is executed with the average gradients of batches of 10 songs. We could say that we use a batch size of 1 in terms of GPU calls but a batch size of 10 in terms of the training data. The models were trained on a GTX 980 Ti Nvidia GPU.

3.5.4 Evaluation Metrics

MIREX's campaings use two evaluation measures for boundary detection which are *median deviation* and *hit rate*. The *hit rate* (also called F-score or F-measure) is denoted by F_β, where $\beta = 1$ is the measure most frequently used in previous works. Other works [112] set a value of $\beta = 0.58$, but F_1 continues being the most used metric in MIREX. We evaluate our models for both values of β.

F_1 is normally evaluated for ± 0.5s and ± 3 s time-window tolerances, but in recent works most of the results are given only for ± 0.5s tolerance which is the most restrictive one. We test our model with MIREX algorithm [134] which give us the Precision P, Recall R and F-measure F parameters.

$$\text{Precision} : \text{P} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{3.16}$$

$$\text{Recall} : \text{R} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{3.17}$$

Table 3.3: CNN architecture parameters of the schema presented in Fig. 3.4

Layer	**Parameters**
Convolution 1 + Leaky ReLU	output feature maps: 32 kernel size: 5 x 7 stride: 1 x 1 padding: (5-1)/2 x (7-1)/2
Max-Pooling	kernel size: 5 x 3 stride: 5 x 1 padding: 1 x 1
Convolution 2 + Leaky ReLU	output feature maps: 64 kernel size: 3 x 5 stride: 1 x 1 padding: (3-1)/2 x (5-1)*3/2 dilation: 1 x 3
Convolution 3 + Leaky ReLU	output feature maps: 128 kernel size: 1 x 1 stride: 1 x 1 padding: 0 x 0
Convolution 4 + Sigmoid	output feature maps: 1 kernel size: 1 x 1 stride: 1 x 1 padding: 0 x 0

$$\text{F measure} : \mathrm{F}_\beta = (1+\beta^2)\frac{\mathrm{P}\cdot\mathrm{R}}{\beta^2\cdot\mathrm{P}+\mathrm{R}}, \tag{3.18}$$

where:

- TP: True Positives. Estimated events of a given class that start and end at the same temporal positions as reference events of the same class, taking into account a tolerance time-window.
- FP: False Positives. Estimated events of a given class that start and end at temporal positions where no reference events of the same class does, taking into account a tolerance time-window.
- FN: False Negatives. Reference events of a given class that start and end at temporal positions where no estimated events of the same class does, taking into account a tolerance time-window.

3.5.5 Experimental Results

Before measuring the results, we need to calculate the optimum threshold to perform the peak picking. We calculate the optimum threshold for our experiments by computing the average F_1 in our validation set for all possible threshold values in the range [0, 1] and then selecting the highest value. Therefore, the optimum threshold is the value between [0, 1] for which the average F_1 is higher in our validation set. The optimum threshold value may vary when training our model with the different combination of inputs shown

in Table 3.5. When we train our model with isolated inputs (see Table 3.4), we compute the threshold with the MLS but we do not vary it when testing the models trained with the SSLMs. We vary the threshold value when we train our model with different input combinations in order to optimize each case of study and provide the best performing method (see Table 3.5). In Fig. 3.5, we obtained a threshold of 0.205 for the models using only the MLS as the input and we used the values shown in Table 3.5 for the rest of the models. From the optimum threshold calculation, we can observe that almost all optimum threshold values for each input variant belong to $[2.05, 2.6]$. Fig. 3.5 shows Recall, Precision and F-score values of the testing dataset evaluated for each possible threshold value.

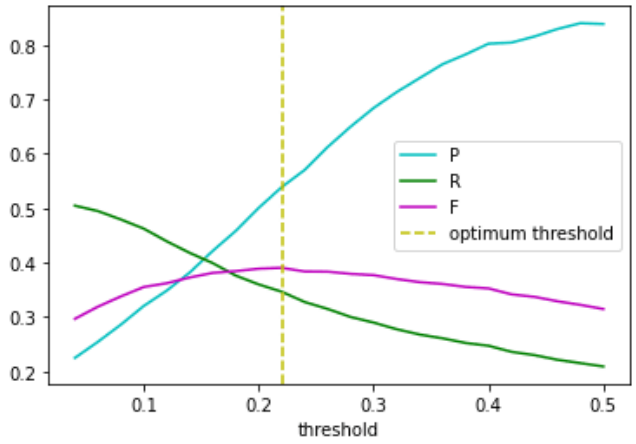

Figure 3.5: Threshold calculation through MLS test after 180 epochs of training with MLS.

Isolated Inputs: Distances, Audio Features and Pooling Strategies

We provide an ablation study of how the audio features and distances to compute the SSLMs impact in the model performance. We trained the model with each input matrix (see Fig. 3.3) separately in order to know what input performs better. We also trained the model using the MLS and SSLMs obtained from MFCC and chroma features, applying Euclidean and cosine distances, and with both pooling strategies mentioned before: `6pool` (lower resolution) and `2pool3` (higher resolution). As mentioned in Section 7.6.1, we removed the first label of the SALAMI text files corresponding to 0.0s label. Results in terms of F-score, Precision and Recall are showed in Table 3.4. Note that the results showed from previous works used a different threshold value.

The best performing model when training our model with isolated inputs is the on e which uses the MLS, with a F_1 value of 0.389 (see Table 3.4). Fo the `6pool` pooling strategy, regarding the SSLMs computed from audio features (MFCCs and chromas), we found that the best performing SSLMs are the ones computed with the MFCCs with more than a 5% difference against the SSLMs computed from chroma features.

Using Euclidean or cosine distances does not have a high impact on the results. The F_1 difference

Table 3.4: Results of boundaries estimation according to different pooling strategies, distances and audio features for ±0.5s and a threshold of 0.205.

Tolerance: ±0.5s and Threshold: 0.205				
	Input	**P**	**R**	**F_1**
6pool	MLS	0.501	0.359	0.389
	$\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$	0.472	0.318	0.361
	$\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$	0.477	0.311	0.355
	$\text{SSLM}^{\text{chromas}}_{\text{euclidean}}$	0.560	0.228	0.297
	$\text{SSLM}^{\text{chromas}}_{\text{cosine}}$	0.508	0.254	0.312
2pool3	$\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$	0.422	0.369	0.375
	$\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$	0.418	0.354	0.366
Previous works				
2pool3	MLS [163]	0.555	0.458	0.465
	$\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$ [37]	-	-	0.430

Table 3.5: Results of boundary estimation with tolerance ±0.5s and optimum threshold in terms of F-score, Precision and Recall. Note that results form previous works did not use the same threshold value.

Tolerance: ±0.5s with 2pool3 matrices					
Input	**Thresh.**	**P**	**R**	**F_1 (std)**	**$F_{0.58}$**
Baseline: MLS + $\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$ [10]		0.279	0.300	0.273 (0.132)	-
MLS + $\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$	0.24	0.441	0.415	0.402 (0.163)	0.414
MLS + $\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$	0.24	0.428	0.407	0.396 (0.158)	0.404
Baseline: MLS + ($\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$ + $\text{SSLM}^{\text{chromas}}_{\text{cosine}}$) [10]		0.470	0.225	0.291 (0.120)	-
MLS + ($\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{chromas}}_{\text{euclidean}}$)	0.24	0.465	0.400	0.407 (0.160)	0.419
MLS + ($\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$ + $\text{SSLM}^{\text{chromas}}_{\text{cosine}}$)	0.24	0.444	0.416	0.404 (0.166)	0.417
MLS + ($\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$)	0.24	0.445	**0.421**	0.409 (0.173)	0.416
MLS + ($\text{SSLM}^{\text{chromas}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{chromas}}_{\text{cosine}}$)	0.24	0.457	0.396	0.400 (0.157)	0.420
MLS + ($\text{SSLM}^{\text{chromas}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{chromas}}_{\text{cosine}}$ + +$\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ + $\text{SSLM}^{\text{MFCCs}}_{\text{cosine}}$)	0.26	**0.526**	0.374	**0.411** (0.169)	**0.451**

between the SSLMs computed with Euclidean or cosine distances is not higher than 1%. Overall, the best performing SSLM for the `6pool` pooling strategy is the $\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ with a F_1 value of 0.361, which is a 2.8% less than the MLS F_1.

In view of the results in Table 3.4, we conclude that the `2pool3` pooling strategy slightly improves the results but it does not make a high impact in the performance. The best performing (`2pool3`) SSLM, the $\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}}$ has a F_1 value of 0.375, which is less than a 2% of the same SSLM but computed with the `6pool` pooling strategy. This procedure not only takes much more time to compute the SSLMs but also the training takes also much more time and it does not lead to better results in terms of F-score.

In Figs. 3.9, 3.8, 3.7, 3.6 we show an example of the boundary detection results for some of our input variants on the MLS and SSLMs. In the figure, the ground truth SALAMI annotations are the gaussians in red, the model predictions is the white curve and the threshold is the horizontal yellow line.

We obtained lower results than [37] but higher results than [10] who tried to re-implement [37]. The reasons for this difference could be that the database used by Grill and Schluter [37] to train their model had 733 non-public pieces. Cohen and Peeters [10], same as we do in our work, trained their model only with pieces from the SALAMI database, so that our results can be compared with theirs, since we trained, validated and tested our models with the same database and songs.

Input Combination

We take the best performing inputs from Table 3.4 to combine them as in [10, 37]. A summary of our results are shown in Table 3.5.

The combination of inputs that perform better in [10] was $\text{MLS} + (\text{SSLM}^{\text{MFCCs}}_{\text{cosine}} + \text{SSLM}^{\text{chromas}}_{\text{cosine}})$ for which $F_1 = 0.291$. We overcome that result for the same combination of inputs obtaining $F_1 = 0.404$. In spite of the previous works [37] claims about using cosine distance leads to better results, we prove that using the Euclidean distance leads to better results. We also found that the best performing input combination is $\text{MLS} + (\text{SSLM}^{\text{chromas}}_{\text{euclidean}} + \text{SSLM}^{\text{chromas}}_{\text{cosine}} + +\text{SSLM}^{\text{MFCCs}}_{\text{euclidean}} + \text{SSLM}^{\text{MFCCs}}_{\text{cosine}})$ with $F_1 = 0.411$. Therefore, we prove that our model, which has a larger receptive field than [10], with the same inputs leads to an improvement of more than 10% in all P, R and F metrics, and the addition of more inputs just improves the performance in 1%.

3.6 Conclusions

3.6.1 Limitations

One of the limitations of the proposed method is that the analysis and results obtained depend largely on the annotations. Inconsistencies between different annotators when analyzing the same piece can lead to ambiguous performance. Therefore, these methods are limited to the quality of the labels given by the annotators and they cannot outperform them.

3.6.2 Benefits

We provided a wide analysis of the input combination for the music boundary detection task. Since state-of-the-art work [36, 37] did not provide the training data, code nor weights, and a posterior work [10] could not reproduce their results, we wanted to democratize the boundary detection task by providing the code and weights. Apart from that, we train our models with the public SALAMI dataset, as [10] and we outperformed their results using the same dataset. We hope that the boundary detection task and, also the labeling task get more attention by the research community in the future since it is crucial to understand music structure, which is one of the music principles [46].

3.6.3 Future Work

There are many ways to improve this work. Since this work was done 4 years ago and since then, many newer deep learning methods emerged, one possible way to improve this work is simply changing the neural network architecture by some of the networks mentioned in Chapter 2 for sequence modelling such as Transformer-based models which have been already used recently for this task [123], or new recurrent neural networks (RNN) like xLSTM [3], or state space models (mamba) [39], which have proven to perform well in temporal sequences. We want to ask the community to help in providing more annotated datasets for this task to be able to build more robust models and improve the reproducibility of future work. In this regard, in a recent work, we propose another work for MSA in the symbolic domain for automatically annotating paired score-audio datasets [50] with graph representations and unsupervised changepoint methods.

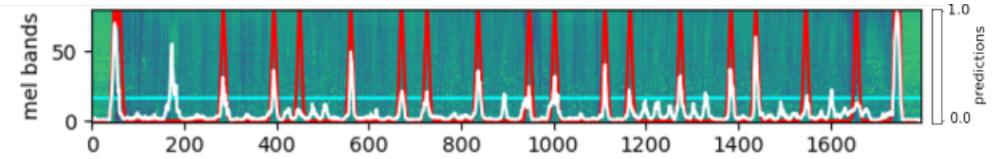

Figure 3.6: CNN predictions on MLS.

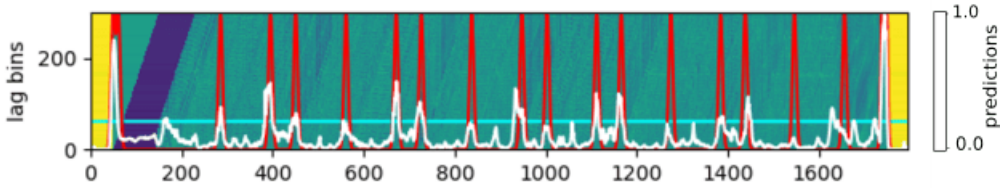

Figure 3.7: CNN predictions on SSLM calculated with MFCCs and Euclidean distance with 2pool3.

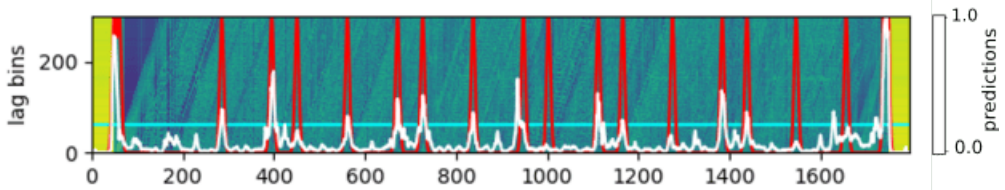

Figure 3.8: CNN predictions on SSLM calculated with MFCCs and cosine distance with 2pool3.

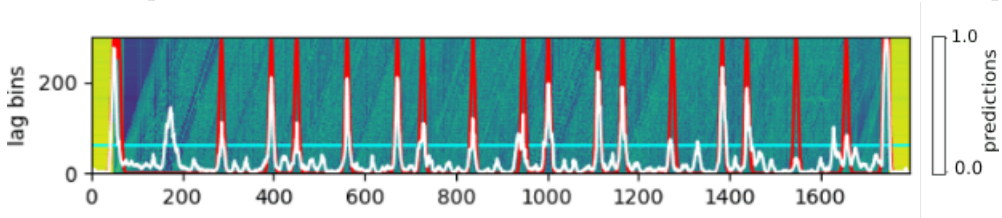

Figure 3.9: CNN predictions on SSLM from MFCCs with cosine distance for the model with MLS and two SSLMs.

Chapter 4

SYMBOLIC MUSIC PROCESSING

ABOUT THIS CHAPTER

The work presented in this chapter has been published in the journal *SoftwareX* in 2023. This work was conducted entirely at the University of Zaragoza. I proposed the idea and implemented the package presented in this chapter. Jose Ramon Beltran supervised the work.

C. Hernandez-Olivan, and J. R. Beltran.
MUSICAIZ: A PYTHON LIBRARY FOR SYMBOLIC MUSIC GENERATION, ANALYSIS AND VISUALIZATION.
SoftwareX 22, 101365, 2023.

4.1 Introduction

In the previous chapters, we presented a method to segment music by its structure in the audio domain. In this chapter, we introduce a package, `musicaiz` which we develop to process, analyze, evaluate and plot symbolic music to help with the symbolic music generation task.

The field of Music Information Retrieval (MIR) is growing significantly in the recent years. With the growth of deep learning, several models for different MIR tasks are being proposed and the need of develop real-world applications is increasing. The automation and scalability of new deep learning (DL) models when it comes to creating pipelines to serve and test such models has become a necessity for music companies such as Sony, Spotify, Apple Music, Pandora or Dolby. Several projects using AI in music applications have

emerged, including Google's Magenta[1] and Microsoft's Muzic[2].

One of the fastest growing tasks of MIR in recent years is music generation. The reason for that is the emergence of new deep learning models such as LLMs that allow to generate music from audio signals and condition such generation with text or tokens [46]. This has increased the interest and need in building open source tools to process music data. On the one hand, some packages work with symbolic music data like `jSymbolic` [101], `music21` [11], `Humdrum` [64], `mido` or `pretty_midi` [133] and, on the other hand, with audio samples such as `librosa` [100]. More specific libraries have been proposed to deal with symbolic music data for music generation purposes such as `muspy` [16] or libraries to help construct encodings for training sequence models such as `miditok` [25]. These softwares is designed to address a particular step of the music generation or analysis tasks. Our proposal, `musicaiz`, aims is to bring together music theory, music processing, models for music generation and evaluation techniques to provide all the steps necessary to build a library that can help future research with reproducibility and scalability. The library is publicly available[3]. We also provide the documentation and examples[4]. The software is tested with `pytest`[5] achieving 80% coverage. The limited coverage is primarily due to incomplete testing of the module that contains the model. Additionally, other algorithms within the package have not been thoroughly tested. These algorithms are outside the scope of this chapter and are therefore not discussed in detail. The methods of each submodule of the package have been tested with different use cases.

One of the fastest-growing subfields in Music Information Retrieval (MIR) in recent years is music generation. The surge in interest can be attributed to advancements in deep learning models, which have outperformed traditional rule-based approaches [46]. This trend has led to an increased focus on developing open-source tools and software to assist researchers in preprocessing stages. These tools cater to both symbolic music data, such as `jSymbolic` [101], `music21` [11], `Humdrum` [64], `mido`, and `pretty_midi` [133], as well as audio samples, for instance, `librosa` [100]. Furthermore, specific libraries have been introduced to handle symbolic music data for music generation purposes, such as `muspy` [16], and to facilitate the construction of encodings for training sequence models, such as `miditok` [25].

In Table 4.1 we compare the features offered by these packages in comparison with the `musicaiz` library.

4.2 Software Description

`Musicaiz` is written in Python 3. We divided the package in different submodules aligned with the music theory principles such as rhythm, harmony and structure. The implementation is done in an object-oriented way. Once we build the music principles, we add a `features` submodule that allows to extract information from the raw symbolic music data. With these features, we can implement evaluation measures in the `eval` submodule. This submodule contains a collection of functions to evaluate music generation inspired by [172].

[1] https://magenta.tensorflow.org/, accessed October 2024
[2] https://www.microsoft.com/en-us/research/project/ai-music/, accessed October 2024
[3] https://github.com/carlosholivan/musicaiz
[4] https://carlosholivan.github.io/musicaiz/index.html
[5] https://github.com/pytest-dev/pytest

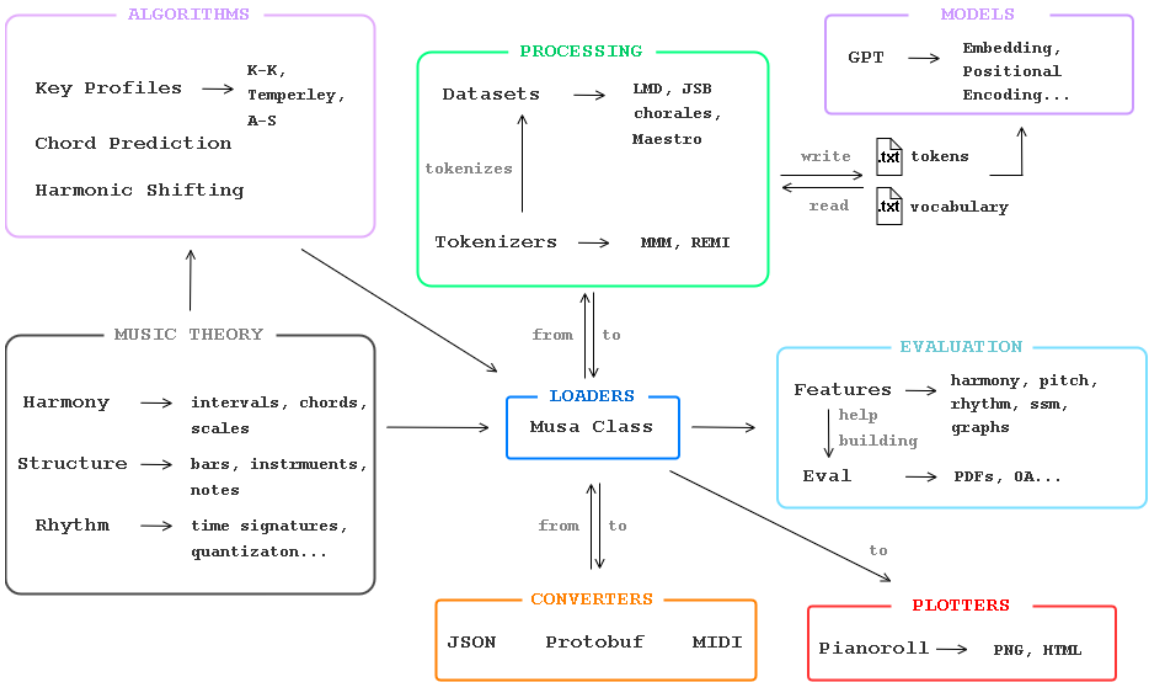

Figure 4.1: Musicaiz utilities and submodules.

Table 4.1: Frameworks for MIDI and symbolic music generation, analysis and representation.

Package	Lang.	Music Theory	Tokenizers	Models	Eval.
jSymbolic	java	✓	✗	✗	✓
music21	python	✓	✗	✗	✗
mido	python	✗	✗	✗	✗
pretty_midi	python	✗	✗	✗	✗
Humdrum	awk	✓	✗	✗	✓
Miditok	python	✗	✓	✗	✗
Muspy	python	✗	✗	✗	✓
Musicaiz	python	✓	✓	✓	✓

Deep learning models often require a preprocessing step since they require identifying a representation that aligns with the objectives of the model's desired outcome. To deal with this, `musicaiz` package also provides a tokenizer module that allows tokenizing symbolic music data in a MIDI-like representation: the Multi-track Music Machine encoding [21], and in a score-like representation: REMI [62]. This submodule can be easily extended to different encodings due to its OOP design. In the `datasets` submodule, we provide classes that allow processing symbolic music datasets used in the field of music generation. Furthermore, symbolic music can be represented as a pianoroll and there are only a few software such as Pypianoroll [18] that allow to plot these representations with a grid of subdivisions as digital audio workstations (DAWs) do. We provide a plotting submodule that helps to plot symbolic musical pianorolls and save them as HTML files with `plotly`. This can not only help to visualize the symbolic musical data, but also could be used as a debugging tool. Finally, the exports submodule converts `musicaiz` objects such as notes or bars (see Section 4.2) to a JSON format so that music information can be sent via REST APIs. This allows building music generation applications with `musicaiz` on the backend that could be used not only for research purposes, but also in industry.

The goal of `musicaiz` is to provide researchers, practitioners and users with a framework for music generation. The software is structured in different submodules which represent a high-level abstraction that caters to the software's functionalities (see Fig. 4.1). These modules contain: Western tonal music theory, processing, generation, evaluation, visualization and export.

Loaders

The module `loaders` contains the main class of the library which is called `Musa`. It allows to import a MIDI file and initialize all its basic attributes. This class uses `pretty_midi` in the backend. The resolution or Ticks per Quarter Note (TPQN) is set to 96. To group the notes in bars, beats and subbeats, we consider both tempo and time signature changes. The Bar, Beat and Subbeat objects store attributes such as the time signature and the tempo to handle their respective changes in the file. This initialization allows to prepare the data for other modules such as the tokenizers. When dealing with MIDI files that capture expressive performances, it is important to remember that the way the notes are grouped into bars or beats

might not be entirely accurate. This is because it is difficult to capture the nuances of a performance using a notation system like a score.

Western Tonal Music Theory

This work utilizes a modular design for symbolic music theory processing, encompassing three core submodules: rhythm, structure, and harmony. Each submodule focuses on specific aspects of musical information, facilitating efficient computation and analysis. We employ an object-oriented approach within these submodules. This paradigm defines classes that encapsulate musical entities along with their properties. Inheritance allows for the creation of specialized classes that inherit properties from more general ones. For instance, a major triad chord is represented as an object with properties including its major quality, a list of constituent intervals (third major and fifth perfect), and its classification as a triad. This object-oriented framework enables the construction of chords from individual intervals and scales from musical degrees and chords. This design approach promotes modularity, reusability, and extensibility within the symbolic music theory processing system.

Rhythm In this module, users have the capability to manipulate various time signatures and apply diverse quantization techniques to MIDI file data. The `musicaiz.rhythm` module encompasses classes and methods facilitating the organization of notes into measures, computation of note durations, and related functionalities. This module is structured into two submodules: `time` and `quantization`.

Structure he structure submodule of this library encompasses fundamental elements related to musical form or arrangement, including pieces, instruments, and notes.

Within this submodule, the primary elements include `Instrument`, `Bar` and `Note`:

- `Instrument`: This class facilitates the definition and grouping of notes and measures within their respective instruments.
- `Bar`: A class designed to define measures, aiding in the organization of symbolic musical data in alignment with traditional score representations.
- `Note`: The `structure.notes` component houses classes inspired by Pearce and Wiggins' framework [122]. These classes enable the construction of note objects for analysis or symbolic music generation, incorporating attributes such as pitch, time, and performance characteristics.

Harmony In addition to building the foundational blocks on the time axis of music, our framework also incorporates a `harmony` submodule to handle vertical aspects such as pitch. The key elements of this submodule include:

- `Intervals`: Definitions of common intervals based on established music theory principles [128].
- `Chords`: Definitions and representations of chords, which are characterized by their root note, quality, complexity (e.g., triads or sevenths), and the intervals relative to the root that define the chord.

- `Tonalities` and `Scales`: Definitions of keys spanning from natural to altered (sharps and flats). Each tonality and mode (major and minor) is accompanied by its corresponding scales, encompassing major, minor, and Greek modes.

These components within the `harmony` submodule provide the necessary tools to analyze and generate harmonic data, crucial for tasks ranging from chord progression analysis to harmonic composition in symbolic music representation.

Processing

Token representations have become pivotal in training deep learning sequence models for music generation tasks in recent years. To support these efforts, our framework includes the following submodules: `tokenizers` and `datasets`.

Tokenizers In `musicaiz`, we have implemented tokenization methods inspired by the Multitrack Music Machine model (MMM) [21], leveraging the GPT-2 model [130] for generating multi-track music. Token sequences can be exported in `.txt` format for compatibility with other software. Additionally, the submodule provides a `get_vocabulary` method to save the vocabulary used in a tokenization as a `.txt` file. We also implemented the Compound Word Transformer tokenizer [58].

Datasets To simplify the processing of popular open-source music generation datasets, this submodule utilizes functionalities from the `tokenizers` submodule. Specifically, it supports datasets for music generation such as MAESTRO [43], Lakh MIDI Dataset [132], and JSB Chorales [5]. We also include the Beethoven Piano Sonatas (BPS) [8] and Schubert Winterreise Dataset (SWD) for structure analysis [170]. The `MusicGenerationDataset` metaclass incorporates a `_prepare_tokenize` function inherited by datasets designed for symbolic music generation. Each dataset includes specific methods such as `get_metadata`, which accommodates variations in data organization, and the `tokenize` method. The latter leverages `get_metadata` to parse metadata and subsequently invokes `_prepare_tokenize` for tokenization of the dataset files.

Generation

For the specific task of symbolic music generation, `musicaiz` contains two submodules: `algorithms` and `models`.

Algorithms Within this submodule, we have implemented a harmonic transposition algorithm tailored for symbolic music data. This algorithm operates by transposing the music to target degrees and scales specified by the user. It utilizes information such as degrees per measure and the scale of the input music to perform these transformations effectively. This functionality is particularly useful as a data augmentation technique for training extensive deep learning models. By generating variations of the original music through harmonic transposition, the algorithm enhances the diversity of training examples. This approach

helps in improving the robustness and generalization capability of deep learning models designed for music generation tasks.

Models This submodule includes a nested submodule called `transformer_composers`, which features the implementation of a GPT-based model along with its corresponding modules for training and data loading in PyTorch [118]. This model is specifically designed for generating symbolic music[6]. By utilizing the aforementioned submodules, researchers and practitioners can efficiently evaluate results and compute necessary preprocessing steps required for the model. Additionally, the `models` submodule serves as a foundation that can be expanded to incorporate state-of-the-art model implementations for music generation. Examples include models like the Music Transformer [61] or the Pop Music Transformer [62], as it provides base classes defining each Transformer layer. This modular structure supports flexible experimentation and development of advanced deep learning architectures tailored for the creative domain of symbolic music generation.

Evaluation

Using our music theory classes and functions as a foundation, we created the `features` submodule to extract characteristics from symbolic music data, which are utilized within the `eval` submodule.

Features Within the `features` submodule, we have implemented functionalities to extract various pitch, rhythm, and harmonic features from symbolic music data. These features serve multiple purposes such as analysis, measurement, and encoding for conditioning deep learning sequence models. Key components of this submodule include:

- `Rhythm`: Implements methods inspired by Roig et al. [140], specifically focusing on time signature numerator estimation.
- `Harmony`: Contains methods related to harmonic analysis and features.
- `Pitch`: Provides fundamental pitch characteristics crucial for evaluating musical generation models within the `eval` submodule.
- `Self-Similarity`: Enables the creation of self-similarity matrices in the symbolic domain, following the methodology by Foote [23]. These matrices visually represent model performance in terms of inter-onset intervals (IOIs), note counts, and other metrics.
- `Graphs`: Implements a symbolic music representation akin to graphs, as introduced by Jeong et al. [70], utilizing the NetworkX package[7] for this purpose.

Eval This submodule facilitates the evaluation of music generation using both subjective and objective methods [51]. It specifically implements Yang and Lerch's evaluation approach [172] for objective assessment. The evaluation method involves cross-validating two or more datasets, typically the training set

[6]Note that in spite that we included the model's code in this package, we did not train it for research purposes.
[7]https://github.com/networkx/, accessed October 2024

and a generated set. Initially, it calculates Euclidean distances within each dataset (inter-set comparison) and subsequently compares samples between different datasets (intra-set comparison). Histograms of these distances are constructed, from which probability density functions (PDF) are derived. Key metrics such as overlap area (OA) and Kullback-Leibler divergence (KLD) are then computed from these PDFs. The submodule integrates with the `features` submodule to leverage various computed features, including:

- Pitch Features: *Pitch count* (PC), *Pitch class histogram* (PCH), *Pitch class transition matrix* (PCTM), *Pitch range* (PR), and *Average pitch interval* (PI).
- Rhythm-based Features: *Note count* (NC) or *Note Density*, *Average inter-onset-interval* (IOI), *Note length histogram* (NLH), and *Note length transition matrix* (NLTM).

These features enable comprehensive evaluation of music generation models, providing insights into both pitch and rhythm characteristics. By quantitatively assessing generated music against training data, this submodule supports rigorous evaluation and refinement of deep learning models in music generation tasks.

An example of the PDFs extracted from 2 different datasets and features are shown in Fig. 4.2. In the figure, JSB Fakes is a generated dataset built from JSB Chorales, and JSB Chorales is the JSB Chorales dataset [5]. When comparing both datasets (inter), we can observe how the pitch counts are similar between them, whereas the pitch class histogram presents some differences.

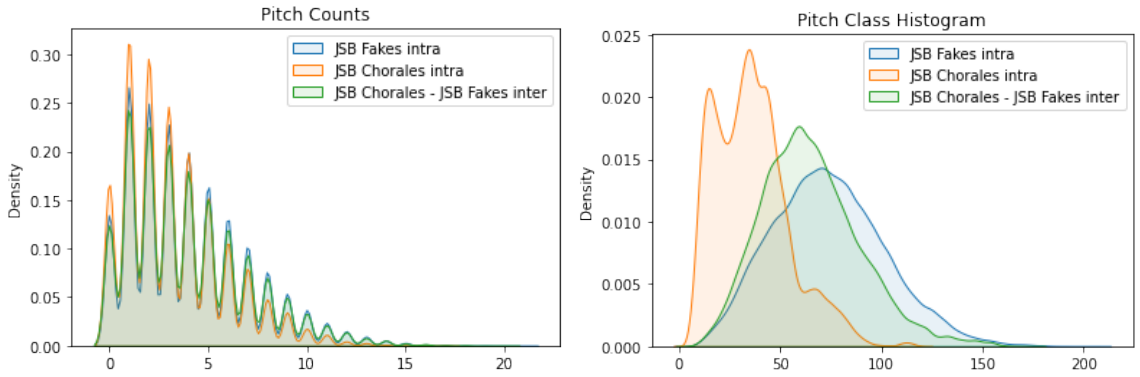

Figure 4.2: An example of objective evaluation of the *intra-set* PDFs of the JSB Fakes and the JSB Chorales datasets. From left to right: a) Pitch Counts (PC) b) Pitch class Histogram (PCH) PDFs.

Apart from the evaluation method that is implemented in the current version of the software, there are more evaluation metrics such as the *groove consistency* which are implemented in other packages such as `muspy` and that could be easily added in future versions of `musicaiz`.

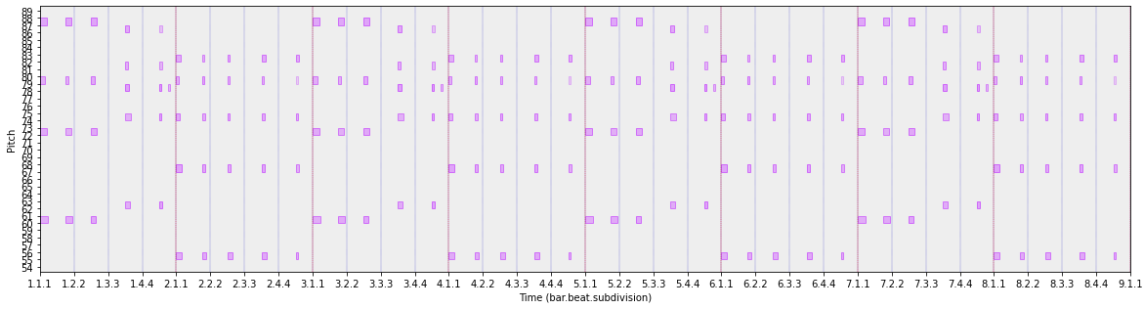

Figure 4.3: Pianoroll with subdivision's grid.

Visualization

The `musicaiz` library includes functionalities for visualizing symbolic music data as pianorolls using two classes capable of generating plots in `.png` format (via `matplotlib`) and `.html` format (via `plotly`). These plots are configured to align with digital audio workstations (DAWs), allowing users to customize the x-axis (time axis) with bars and subdivisions. We show an example of a pianoroll with the grid of subdivisions on the x-axis in Fig. 4.3.

Exports

The final functionality of the `musicaiz` software includes export capabilities designed to enhance its applicability in various contexts, including industrial applications. This functionality comprises two main components:

- Protocol Buffer Support: The software offers both a musicaiz to protocol buffer converter and a protocol buffer to musicaiz converter, utilizing the `protobuf` `protobuf` package[8]. This format ensures efficient serialization and deserialization of musical data, optimizing data transmission and storage.
- JSON Representation: Additionally, the software provides functionality to export `musicaiz` data into JSON format. This facilitates interoperability and integration with REST APIs, enabling seamless exchange of musical information across different systems and platforms.

These export functionalities extend the utility of `musicaiz` beyond local processing environments, making it well-suited for integration into larger-scale applications where interoperability, efficiency, and data exchange are critical. This capability supports diverse use cases, ranging from research and development to industrial applications in music technology and beyond.

[8] https://github.com/protocolbuffers/protobuf, accessed October 2024

4.3 Conclusions

In this chapter, we presented a new open source object-oriented library for symbolic music data that provides the key components to generate and evaluate music with AI. We hope that this library will be a useful tool for the community and will help to build music generation systems with (or without) AI approaches. Apart from that, further extensions can be made by adding new token representations that may be needed by future deep learning models or implementing new or other algorithms for feature extraction, either by us as original authors or by any other researcher or developer interested in it.

4.3.1 Limitations

The package only contains one model, GPT-2, which makes it limited for the music generation task. Also, it works with MIDI data but not with other symbolic representations such as MusicXML.

4.3.2 Benefits

Due to the OOP design and that the software is free and open source, the it can be easily extended to handle new tokenizers, representations such as MusicXML, or additional features for evaluating music generation such as structure pattern detection with self-similarity matrices [23, 45]. The software is tested with `pytest` to ensure its correct usability.

4.3.3 Future Work

The future work that can be done with this software is the addition of new music generation models and score-based representations such as MusicXML. Also, the software should be used it future research to prove its functionalities.

Part III

MUSIC PRODUCTION

Chapter 5

VOCAL RESTORATION VIA DIFFUSION POSTERIOR SAMPLING

ABOUT THIS CHAPTER

The work presented in this chapter has been presented at ICASSP 2024. This work was conducted entirely in Sony under the supervision of the Music Foundation Team and Generative AI teams in Tokyo. In particular, Koichi Saito set the main line of research using CQT-Diff for singing voice restoration, Naoki Murata and Chieh-Hsin Lai proposed applying RePaint, ΠGDM and FreeDOM, Marco Martinez-Ramirez the listening MUSHRA test and Weihsiang Liao supervised the work. I implemented the methods especially adapting the sampler, and conducted the experiments.

C. Hernandez-Olivan, K. Saito, N. Murata, C. H. Lai, M. A. Martinez-Ramirez, W. H. Liao, and Y. Mitsufuji.
Vrdmg: Vocal restoration via diffusion posterior sampling with multiple guidance.
In the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 596-600, 2024.

5.1 Introduction

Following the music creation process described in Chapter 1, we introduce the first audio production work of this thesis in this chapter. Audio restoration is a well known problem in audio processing tasks and

its applications cover from speech enhancement to music restoration. Restoring audio tracks that contains noise or degradations is a common problem in music production. Audio restoration aims to restore degraded signals to re-use the restored tracks in re-mixing, or to remove noise from old recordings.

The field of audio restoration encompasses a variety of tasks aimed at improving the quality of degraded audio signals. Common examples include inpainting (filling in missing audio segments), declipping (removing unwanted clipping distortion), bandwidth extension (enhancing the frequency range) and dereverberation [108] (reducing unwanted reverberation). These tasks present a unique challenge: they are inherently ill-posed, meaning there can be multiple equally valid restored versions of the original signal.

Traditionally, audio restoration methods have fallen into two categories: unsupervised signal processing approaches and supervised deep learning techniques. Unsupervised methods rely on signal processing algorithms to make assumptions about the clean signal based on the degraded version. Supervised methods, on the other hand, leverage paired data consisting of both degraded and clean signals to train deep neural networks (DNNs) for restoration [66, 81, 105, 182]. While both approaches have their strengths, they share limitations. Unsupervised methods might not generalize well when the target audio deviates from the underlying assumptions. Supervised methods require not only preparing these assumptions but also collecting significant amounts of paired training data, which can be a cumbersome and task-specific process.

A recent paradigm shift has seen the emergence of unsupervised audio restoration using deep generative models. These models are trained solely on clean audio data, allowing them to learn a prior distribution of what clean audio signals typically look like. This approach offers several advantages:

- Data efficiency: No degraded data is required during training, making the method more adaptable to new tasks without retraining.
- Flexibility: The model can be applied to various restoration tasks without modification.
- Data-driven assumptions: The model learns assumptions about clean audio directly from the data.

This paradigm does not require degraded signals during training, allowing for a data-driven acquisition of assumptions solely about the clean signals. It also offers the advantage of being adaptable to various tasks without the need for retraining. Among these, notably, unsupervised diffusion-based approaches have demonstrated significant performance [104, 107, 144, 175].

Among generative models, diffusion models have shown particular promise in unsupervised audio restoration. These models learn a process that gradually corrupts clean audio into a degraded version. The restoration process then reverses this corruption, effectively denoising or repairing the audio signal.

Our contributions are the following:

- Analyze how a DPS [9] piano restoration method, CQT-Diff [104], performs for singing voice restoration.
- Introduce Pseudoinverse-Guided Diffusion Models (ΠGDM) [157], RePaint [91] and FreeDOM [174] for singing voice restoration.

5.2 Diffusion Models for Inverse Problems

5.2.1 Diffusion Models and Score-based Modeling

Diffusion models draw inspiration from non-equilibrium thermodynamics. This approach involves a series of controlled steps where noise is gradually introduced to a clean data sample. The diffusion model then learns the inverse process taking highly noisy data and progressively denoising it to recover the original data.

Let's consider a collection of clean signals, each represented by a data point. We can describe this collection statistically using a probability distribution, which we will call p_{data}. Now, let's consider a process where we gradually add noise to these clean signals. We can think of it like a gradual *blurring* effect. We will start with a clean signal, denoted by $\boldsymbol{x}_0 \sim p_{\text{data}}$, which comes from our p_{data} distribution. Over time steps (represented by t, which goes from 0 to T), we progressively add noise to the signal.

The amount of noise which is added at each step (represented by σ_t) depends on the time steps. We start adding very high noise ($\sigma_T = \sigma_{\max}$), and gradually decrease it down to almost no noise ($\sigma_0 = \sigma_{\min}$) by the final time step T where we have Gaussian noise $\boldsymbol{x}_T \sim \mathcal{N}(0, \sigma_{\max}^2 \boldsymbol{I})$. By the end of this process (at time step T), the signal ($\boldsymbol{x}_T$) will be completely corrupted by noise and will resemble random Gaussian noise with zero mean and a specific level of variance $\boldsymbol{x}_T \sim \mathcal{N}(0, \sigma_{\max}^2 \boldsymbol{I})$.

Diffusion models [54, 72, 156], alternatively termed score-based models, generate a clean signal through the reverse diffusion process previously described. These models approximate the gradient of the logarithmic probability density relative to the score function $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ [65]. The reverse diffusion process can be articulated by the following Stochastic Differential Equation (SDE) [72]:

$$\mathrm{d}\boldsymbol{x}_t = \left(\frac{1}{2} g(t)^2 - \dot{\sigma}(t)\sigma(t) \right) \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) \mathrm{d}t + g(t) \mathrm{d}\boldsymbol{w}_t \tag{5.1}$$

where w denotes the standard Wiener process, $\dot{\sigma}(t)$ represents the time derivative of noise levels at time t, and $g(t)$ is *diffusion* coefficient of $\boldsymbol{x}_t$ [159].

For diffusion models, the score function is approximated by a neural network $D_{\boldsymbol{\theta}}(\boldsymbol{x}; \sigma)$, where $\boldsymbol{\theta}$ are the parameters of neural network. In Karras et al. (EDM) [72], the score function is estimated as $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t; \sigma_t) = (D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t) - \boldsymbol{x}_t)/\sigma_t^2$, where the neural network of the denoiser $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t)$ is trained with the following L_2 objective:

$$\mathbb{E}_{\boldsymbol{x}_{\text{data}} \sim p_{\text{data}}, \boldsymbol{n} \sim \mathcal{N}(0, \sigma_t^2 \boldsymbol{I})} \left[\| D_{\boldsymbol{\theta}}(\boldsymbol{x}_{\text{data}} + \boldsymbol{n}; \sigma_t) - \boldsymbol{x}_{\text{data}} \|_2^2 \right], \tag{5.2}$$

where $\boldsymbol{x}_{\text{data}} \sim p_{\text{data}}$ is a training data. These models can generate samples by solving the SDE where the estimated score function is substituted by the discrete sampling times t.

5.2.2 Diffusion Posterior Sampling

Diffusion models have emerged as a powerful paradigm in generative modeling, demonstrating exceptional capabilities in creating realistic and diverse data samples. However, their ability extends beyond just

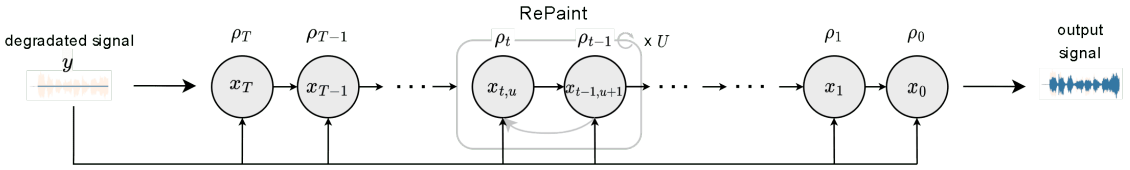

Figure 5.1: Inference process of our RG $\delta\rho$ + DC + RP method. ρ_t is the parameter that controls the gradients weighting and $\boldsymbol{x}_t$ is the prediction at each diffusion step t. U are the repainting steps.

generating new data. Recent advancements have unlocked their potential for tackling a crucial challenge in machine learning such as inverse problems. Diffusion posterior sampling (DPS) [9] leverages the strengths of diffusion models to solve these intricate problems.

5.2.3 Inverse Problem via Posterior Sampling

The objective of music inverse problem is to recover the clean music signal $\boldsymbol{x}_0$ from observed degraded signal $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}_0) + \boldsymbol{\epsilon}$, where $\mathcal{A}(\cdot)$ represents the degradation function, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_{\boldsymbol{y}}^2 I)$ is the measurement noise and $\sigma_{\boldsymbol{y}}^2$ is noise variance of measurement. We can substitute the score $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t)$ in Eq. 5.1 for the conditional score function $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{y})$ [9, 104] to solve inverse problems with diffusion models. Within the Bayesian framework, the *posterior* $p(\boldsymbol{x}|\boldsymbol{y})$ can be expressed as $p(\boldsymbol{x}|\boldsymbol{y}) \propto p(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x})$. Leveraging this relationship, the conditional score $\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{y})$ can be deduced as:

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{y}) = \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{y}|\boldsymbol{x}_t). \tag{5.3}$$

The first term in Eq. (5.3) is the score function which we can compute via $(D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t) - \boldsymbol{x}_t)/\sigma_t^2$.

The reconstruction guidance (RG) technique [56, 104] computes the likelihood (second term in Eq. (5.3)) as [9]:

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{y}|\boldsymbol{x}_t) \simeq -\rho(t) \nabla_{\boldsymbol{x}_t} \|\boldsymbol{y} - \mathcal{A}(D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t))\|^2. \tag{5.4}$$

The gradient in Eq. 5.4 is scaled by a time-dependent scaling function $\rho(t)$. In CQT-Diff [104], $\rho(t)$ is empirically determined so that $\rho(t) = \rho' \sqrt{L}/(t\|G\|^2)$, where $G = \nabla_{\boldsymbol{x}_t} \|\boldsymbol{y} - \mathcal{A}(D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t))\|^2$, where L is the sample length of audio signal and ρ' is a constant value.

5.3 Methods

5.3.1 Time-dependent Gradient Weighting

FreeDoM[174, Fig.3] analyzes the intermediate diffusion steps of the diffusion process and describes 3 main stages: chaotic, semantic and refinement stages. During the chaotic stage, the generation is "out of control", in other words, the noise level is high, and the generated image bears little resemblance to the target data distribution. The semantic stage is critical for conditional generation and thus, this is the stage where we can safely inject conditional information. The last stage is the refinement, when it is too late for conditioning.

Inspired by these insights, we can redefine the gradient weighting as a function which depends on the steps. We prove that redefining $\rho(t)$ as $\rho(t)\delta\rho$ exhibits superior performance compared to a static $\rho\prime$. We empirically found $\delta\rho = (T-t)/75$. Defining $\delta\rho$ in this manner, the scale of RG remains small during the initial stages of the diffusion process (or the chaotic phase) where the generation is out of control according to FreeDoM [174]. As the diffusion process progresses and the generation becomes more controllable, the scale of RG becomes large.

5.3.2 Pseudo-Inverse Guidance

The second term of Eq. (5.4) can alternatively be calculated by utilizing Pseudo-inverse Guidance (ΠGDM) [157] as:

$$\nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{y}|\boldsymbol{x}_t) \simeq \left((h^\dagger(\boldsymbol{y}) - h^\dagger(h(\hat{\boldsymbol{x}}_0)))^\top \frac{\partial \hat{\boldsymbol{x}}_0}{\partial \boldsymbol{x}_t} \right)^\top, \tag{5.5}$$

where $\hat{\boldsymbol{x}}_0$ is estimated clean signal by the denoiser network $D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t)$, h is a non-linear degradation function, and $h^\dagger$ need to adhere to the equation $h(h^\dagger(h(\boldsymbol{x}))) = h(\boldsymbol{x})$ for all $\boldsymbol{x}$. h and $h^\dagger$ can be even non-differentiable degradation function, rendering DPS inapplicable. Empirical evaluations by the authors [157] reveal ΠGDM outperform RG with even fewer diffusion steps, indicative of its computational efficiency.

5.3.3 RePaint

The DC method, which stands for Data Consistency, is a technique used during inference in certain deep learning models. It substitutes the intermediate prediction, denoted by $\hat{\boldsymbol{x}}_0 = D_{\boldsymbol{\theta}}(\boldsymbol{x}_t; \sigma_t)$, with the actual measurement, $\boldsymbol{y}$, at each step of the sampling process. This substitution can be represented by $\hat{\boldsymbol{x}}_0 = \boldsymbol{y}$.

In the context of image inpainting, the task involves reconstructing missing parts of an image. RePaint, a deep learning model for inpainting [91], highlights a critical issue with the DC method. When solely relying on the DC method, the predicted region tends to prioritize matching the texture of the surrounding areas rather than capturing the overall semantic meaning of the image. This leads to reconstructions that are visually similar to the surrounding regions but lack semantic coherence, resulting in inaccuracies.

The limitations of the DC method are not restricted to image inpainting. Similar issues might arise in applying this method to inverse problems within the music domain. When used for music restoration, for instance, the DC method could potentially lead to suboptimal results. The restored music might sound similar to the original but lack the nuances and details that contribute to the overall musical experience.

To alleviate this limitation, we advocate integrating the RePaint (RP) strategy [91] with the DC method.

The RP strategy is a resampling strategy that first diffuses output $\boldsymbol{x}_{t-1}$ back to $\boldsymbol{x}_t$ through the sampling as:

$$\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}_{t-1}, (\sigma_t^2 - \sigma_{t-1}^2)\mathbf{I}) \tag{5.6}$$

and then conducts a regular reverse diffusion step from $\boldsymbol{x}_t$, again. We designate this one cycle as the RP cycle.

Originally, the RP strategy is applied with every diffusion step during inference with multiple RP cycles (Please refer to [91, Algo. 1]). Even in the absence of the RP strategy, diffusion models generally need multiple sampling steps to generate samples, which is time-consuming. Hence, applying multiple RP cycles will result in an increased number of sampling steps.

To address this issue, we propose to apply the RP cycle exclusively during the phase which crucially affects the sampling outcome inspired by the categorized stages of diffusion sampling [174, Fig.3]. We define the number of the RP cycles U depending on the entire number of the diffusion steps T as:

$$U = u \cdot \mathbb{1}_{[\phi_1 T/3, \phi_2 T/3]}(t) \tag{5.7}$$

where ϕ_1 and ϕ_2 are terms that define when the RP cycle starts and ends, inspired by FreeDoM [174].

Algorithm in Appendix A shows the DPS algorithm that contains the techniques introduced before: RG, DC and RP.

5.4 Experiments and Results

5.4.1 Dataset

We use vocal data which do not contain audio effects (dry vocals) and contain audio effects (wet vocals) to evaluate the methods under the various types of vocals. We use the NHSS dataset [153], which only contains dry vocals, and an internal dataset of wet vocals. The NHSS dataset contains 100 English pop songs (20 unique songs) of ten different male and female singers. The total duration of the dataset is 95.76h for training and 0.94h for validation. For evaluation, we use the NUS dataset [20], which contains dry vocals, and the wet vocals from MUSDB18 dataset [135]. We dowmsample all the tracks to 22.05 kHz. Due to the time constraints, We selected 200 random segments of 2.97 seconds for both dry and wet vocals, ensuring that they encompassed a diverse range of vocalists.

5.4.2 Model Configurations

In our preliminary experiments, we observe that the CQT-Diff (CQT model) requires more time to output restored data compared to the SaShiMi-Diff (1D model), due to its inverse CQT implementation [104, 167]. Both models are referred in [104, Section 3.1]. Consequently, we train those two types of models and evaluate both in terms of performance and inference time. The difference between these two models is the input data representation for the neural network. In the 1D model, the neural network receives input in the form of a waveform representation. On the other hand, in the CQT model, the input data representation is CQT representation.

Both models have a U-Net architecture comprising the six layers with the dilated convolutions and the GELU activations. Both models have 15M parameters. We use Adam optimizer and an initial learning

rate of $2e-4$. The noise variance schedule at diffusion step t, σ_t, follows the expression:

$$\sigma_t = \left(\sigma_{\max}^{\frac{1}{\nu}} + \frac{t}{T-1}\left(\sigma_{\min}^{\frac{1}{\nu}} - \sigma_{\max}^{\frac{1}{\nu}}\right)\right)^{\nu} \quad t = T \dots 1 \tag{5.8}$$

We set $\nu = 13$, $\sigma_{\min} = 10^{-4}$, $\sigma_{\max} = 1$ and $S_{\text{churn}} = 5$ which controls the overall amount of stochasticity for the noise variance schedule [72]. For inference step, we utilize $T = 300$ diffusion steps.

5.4.3 Declipping

Table 5.1: Objective metrics for declipping with pre-trained CQT and 1D models. For each metric and test set, each result is highlighted in bold and second best result is underlined.

		SDR = 5dB				SDR = 10dB			
		FAD		SI-SDR		FAD		SI-SDR	
	Method	↓ dry	↓ wet	↑ dry	↑ wet	↓ dry	↓ wet	↑ dry	↑ wet
	Clipped	3.48	2.25	6.42	5.06	1.10	0.87	11.08	9.18
1D	RG [104]	1.84	1.55	9.79	5.36	0.66	1.19	14.39	7.76
	RG + DC	0.94	1.05	10.13	<u>6.15</u>	0.66	0.52	14.48	<u>8.69</u>
	RG $\delta\rho$ + DC	<u>0.88</u>	<u>1.00</u>	<u>10.97</u>	6.03	<u>0.35</u>	<u>0.32</u>	<u>14.79</u>	8.57
	ΠGDM + DC	2.76	1.73	5.44	3.18	1.02	0.76	10.31	7.01
	RG $\delta\rho$ + DC + RP	**0.47**	**0.65**	**11.83**	**7.03**	**0.15**	**0.21**	**15.42**	**9.05**
CQT	RG [104]	1.59	1.05	9.87	5.76	0.37	1.04	14.29	8.20
	RG + DC	1.16	<u>0.80</u>	10.09	6.24	0.37	0.39	14.33	<u>8.91</u>
	RG $\delta\rho$ + DC	**0.52**	**0.34**	<u>10.58</u>	<u>6.51</u>	<u>0.16</u>	**0.19**	<u>14.50</u>	**9.02**
	ΠGDM + DC	2.62	1.57	5.70	3.75	0.74	0.60	10.70	7.22
	RG $\delta\rho$ + DC + RP	<u>0.73</u>	1.02	**11.61**	**7.33**	**0.13**	<u>0.28</u>	**15.03**	8.75

Declipping an audio signal is restoring the signal after a distortion with a threshold c has been applied to it. The degradation function is formulated as $\mathcal{A}(\boldsymbol{x}_0) = (|\boldsymbol{x}_0 + c| - |\boldsymbol{x}_0 - c|)/2$. This is hard-clipping setting [66]. We perform both objective and subjective evaluations. The objective evaluation is measured in terms of the Frèchet Audio Distance (FAD) [74] and the Scale-Invariant Signal-to-Distortion Ratio (SI-SDR) [84].

We test both models with the different sampling methods: RG and RG + DC as the baseline methods proposed in [104], and our proposed methods: RG $\delta\rho$ + DC, which consists on applying RG with DC method and $\delta\rho$, RG $\delta\rho$ + DC + RP and DC + ΠGDM. With employing the RP strategy, Eq. 5.6 is invoked with $u = 10$, $\phi_1 = 1.5$ and $\phi_2 = 2.8$, leading to the $U = 10$ RP cycles between the diffusion steps $t = 150$ to $t = 20$. For the ΠGDM method, we define $h(\boldsymbol{x}) = \mathcal{A}(\boldsymbol{x})$ and $h^{\dagger}(\boldsymbol{x}) = \mathcal{A}(\boldsymbol{x})$. With this definition, $h(h^{\dagger}(h(\boldsymbol{x}))) = h(\boldsymbol{x})$ is satisfied for all x. We compare the models and methods with two Signal-to-Distortion Ratio (SDR) of applied clipping , 5dB and 10dB as it is reported in Table 5.1.

Table 5.2: Objective metrics for bandwidth extension with pre-trained CQT and 1D models

		f_c = 3KHz				f_c = 5KHz			
		FAD		LSD		FAD		LSD	
Model	Method	↓ dry	↓ wet	↓ dry	↓ wet	↓ dry	↓ wet	↓ dry	↓ wet
	LPF	4.12	3.16	4.26	4.55	2.53	1.79	3.61	3.86
1D	RG [104]	2.38	1.35	**1.87**	1.95	1.72	1.00	1.59	1.86
	RG + DC [104]	1.53	1.44	1.98	**1.84**	0.57	**0.44**	1.67	1.61
	RG $\delta\rho$ + DC	**1.13**	1.38	2.01	1.89	**0.46**	0.48	**1.57**	1.61
	ΠGDM + DC	1.69	**1.26**	2.14	1.86	0.67	0.48	1.93	1.70
	RG $\delta\rho$ + DC + RP	1.15	1.46	2.00	1.88	0.73	0.58	1.56	**1.60**
CQT	RG [104]	1.63	0.75	**1.95**	1.91	1.31	0.82	1.70	1.79
	RG + DC [104]	**1.06**	0.65	2.01	1.82	**0.39**	**0.30**	1.67	**1.59**
	RG $\delta\rho$ + DC	1.65	1.17	2.34	2.10	0.44	0.35	**1.64**	1.61
	ΠGDM + DC	**1.06**	**0.63**	2.00	**1.80**	1.23	0.32	1.68	1.66
	RG $\delta\rho$ + DC + RP	1.78	1.57	2.31	2.03	0.51	0.67	1.68	1.62

In terms of the 1D model, RG $\delta\rho$ + DC + RP consistently outperforms all of the other methods for all the experimental settings. As for the CQT model, upon overall consideration, RG $\delta\rho$ + DC + RP tends to show better performance than the other methods. From these observations, combining all the method, which are $\delta\rho$, the DC method, and the RP strategy, is effective to enhance the performance of the original RG method. Moreover, comparing with RG $\delta\rho$ + DC + RP of the 1D and the CQT model, it is remarkable that the 1D model tends to show better scores since the inference time of the 1D model is $\times$15 faster than the CQT model using the a RTX 3090 24Gb GPU. The actual inference time of the 1D model using RG $\delta\rho$ + DC + RP is three minutes and the CQT model takes 45 minutes.

We also conducted a listening test using the multiple stimuli with the hidden reference and anchor (MUSHRA) protocol [67]. We ask the 12 participants to rate the quality of the restored signals with the different methods. To avoid the listening fatigue, we only include the samples from the CQT models under SDR = 5dB, which is harsher setting as the objective evaluation proves. Also, we discarded the worst-performing methods (RG) and (RG + DC) according to the objective results in Table 5.1. The results of the listening test are illustrated in Fig 5.2. In both dry and wet configurations, RG $\delta\rho$ + DC + RP show the best score as we see in Table 5.1. Notably, within the wet setting, RG $\delta\rho$ + DC + RP is perceived by the listeners to be very close to the reference.

5.4.4 Bandwidth Extension

Bandwidth extension consists on restoring the frequencies that have been removed from the signal with a low pass filter (LPF). Within this configuration, the degradation function of bandwidth extension is

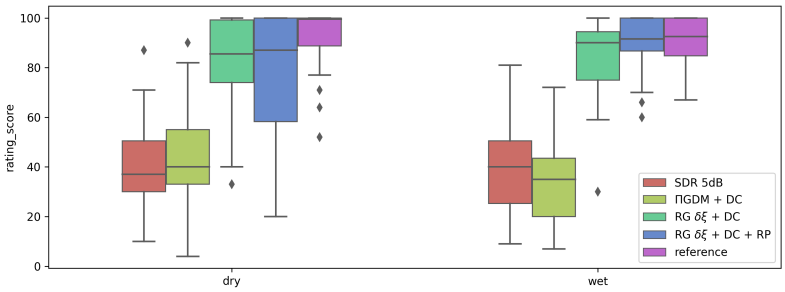

Figure 5.2: Results of the audio declipping listening test of CQT model for 5dB.

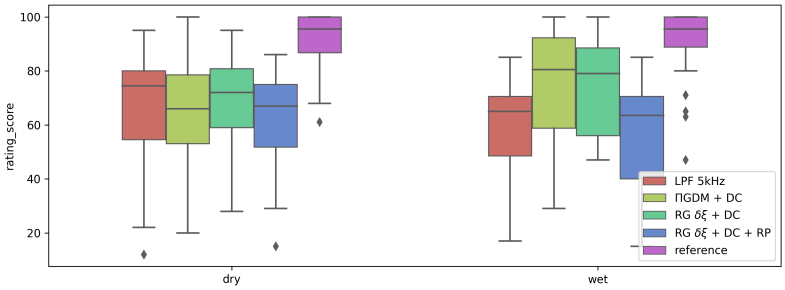

Figure 5.3: Results of the audio bandwidth extension listening test of CQT model for 5kHz.

formulated as $\mathcal{A}(\boldsymbol{x}) = \text{LPF}(\boldsymbol{x})$. We evaluate the methods with two settings of cutoff frequencies $f_c = 3$ and 5 kHz. For the ΠGDM method, same as the declipping setting, we define $h(\boldsymbol{x}) = \mathcal{A}(\boldsymbol{x})$ and $h^\dagger(\boldsymbol{x}) = \mathcal{A}(\boldsymbol{x})$. With this definition, if we assume that the LPF can cutoff all the frequency components lower than f_c, $h(h^\dagger(h(\boldsymbol{x}))) = h(\boldsymbol{x})$ is satisfied, as illustrated in Eq. 5.5 since applying consecutive the LPF with the same f_c to the signal is the same as applying it once. In this task, we set $\phi_1 = 2.5$, $\phi_2 = 2.8$ and $u = 5$ for the RP strategy.

In this case, the objective evaluation shows that the best-performing methods are RG + DC and ΠGDM + DC. In contrast with the declipping task, ΠGDM performs well and outperforms other techniques like RG. Is also worth noticing that RP does not work that well in this task. The reason why RP does not work for this task might be due to the diffusion trajectories that the time-traveling introduces itself which for bandwidth extension are not optimal.

In parallel to the declipping task, we conducted a listening test, engaging the same participants to evaluate the methods for the $f_c = 5$ kHz case. The results of the bandwidth extension listening test are presented in Fig. 5.3. The subjective evaluation reports that the best method for $f_c = 5$ kHz is RG $\delta\rho$ + DC. The listening test results show that none of the methods outperform the lowpassed signal for the dry vocals. The human auditory system is highly attuned to the formants of speech. Consequently, due to the absence of effects in dry signals containing only the voice, the inadequate reconstruction of these formants might be more perceptible to the human ear compared to wet signals where effects mask this imperfect formant reconstruction. In wet signals, we can observe how ΠGDM and RG $\delta\rho$ + DC perform better, which also happens in the objective evaluation.

5.5 Conclusions

In this chapter, we analyzed various diffusion sampling techniques for music restoration. We pointed out potential shortcomings in the current DPS-based methods and introduced the way to address them.

By incorporating the RP strategy, ΠGDM, and time-dependent weights for the guidance term, we enhanced restoration performance in both vocal declipping and bandwidth extension tasks, outclassing the baseline methods. Notably, the 1D model, when using the same sampling technique, achieved equivalent performance to the CQT model but was 15 times faster.

5.5.1 Limitations

The proposed methods are evaluated for singing voice restoration, however, it is not proven how they perform for other music signals or speech. During the experiments, we saw significant variations in the performance against the CQT-Diff model trained for piano. This sets the question about how sensible is the model to other signals.

5.5.2 Benefits

The methods proposed in this chapter allow to perform singing voice restoration without needing to re-train the model. We also prove that for voice signals, the 1-D model outperforms the CQT model and it is faster in inference.

5.5.3 Future Work

Future work should explore new sampling techniques, especifically for audio. Also, other audio inverse problems like inpainting could be addressed. According to the model inference time, other instruments could be tested and explore 1-D inputs instead of CQT which increases the inference time.

Chapter 6

BINAURAL TARGET SOUND EXTRACTION

ABOUT THIS CHAPTER

The work presented in this chapter has been presented at IWAENC 2024. This work was conducted entirely in NTT under the supervision of Dr. Marc Delcroix and the NTT CSL Keihanna laboratories research team. In particular, Marc Delcroix and Tsubasa Ochiai set the main line of research using Semantic Hearing model, they proposed the idea of using the Interaural time difference as a loss function and supervised the work. I implemented the losses and conducted the experiments.

C. Hernandez-Olivan, M. Delcroix, T. Ochiai, N. Tawara, T. Nakatani, and S. Araki.
Interaural time difference loss for binaural target sound extraction.
In the IEEE International Workshop on Acoustic Signal Enhancement (IWAENC), 2024.

6.1 Introduction

In Chapter 5, we explored various techniques related to music production tasks. This chapter diverges from the previous one in its focus, but the topic and methods discussed in this chapter remain applicable to music production. As we described in Chapter 1, target sound extraction can be used to extract a single instrument from a mixture to create song remixes. This is a common practice in music production.

This chapter covers binaural target sound extraction (TSE). TSE aims to isolate a specific sound signal belonging to a desired sound class from a mixture of diverse sounds [116]. For instance, TSE could extract

the sound of a siren from a recording that includes other noises like dog barking and car passing. This problem extends the concept of targeted speech extraction [180] to encompass arbitrary sounds. The practical applications of TSE are manifold. It could enable advancements in technologies such as hearables or hearing aids, allowing them to selectively focus on sounds of interest in everyday environments. Additionally, TSE could enhance smart audio post-production systems by facilitating targeted manipulation and enhancement of specific sounds within audio recordings.

A typical TSE system comprises a neural network designed to process a sound mixture and output the desired sound signal, effectively removing other sounds and noise. The network's operation relies on a conditioning clue that identifies the target sound within the mixture. Various types of clues are employed for this purpose, including a one-hot vector indicating the target sound class [13, 116], an audio recording resembling the desired sound [13, 31, 179], video information [177], onomatopoeic representations [117], specified spatial regions [40], and combinations of these clues [88]. In this context, our focus centers on TSE systems that are conditioned specifically on the target sound class. This approach utilizes categorical clues, such as one-hot vectors, to guide the network in accurately extracting the desired sound from complex audio environments.

Most TSE frameworks predominantly focus on single-channel processing [13, 168]. However, humans rely on binaural hearing to perceive spatial information about sounds. For instance, interaural level difference (ILD) and interaural time differences (ITD) are crucial cues for localizing sound sources [4, 106]. Therefore, it is crucial to develop TSE systems capable of extracting target sounds while preserving their binaural cues.

To achieve binaural processing, one approach involves extending a monaural TSE system to accept binaural input and produce binaural output signals. This concept initially emerged in the domain of speech enhancement [41] and has more recently been applied to TSE tasks [169]. These systems can be trained using a signal-level loss on each output channel, encouraging the model to replicate left and right signals as faithfully as possible to the references. This training approach implicitly encourages the preservation of spatial cues. However, prior research efforts [41, 169] have not explicitly utilized spatial loss functions, which could potentially limit the ability of TSE systems to accurately recover spatial cues.

This chapter explores whether incorporating an explicit spatial loss could enhance the preservation of spatial cues in binaural Target Sound Extraction (TSE). Defining a loss based on ILD, as described in Tokala et al. [160], involves computing the difference in the ratio of the norms of left and right signals, which is differentiable. However, computing ITD requires identifying the peak in the cross-correlation between left and right signals, a non-differentiable operation. Recently, Tokala et al. [160] proposed a loss based on interaural phase difference (IPD), which relates to time difference but faces challenges due to phase wrapping and may not effectively reduce ITD errors. Instead, we propose an alternative loss based on the errors between cross-correlation coefficients of estimated and reference signals, termed the ITD loss in this context. This approach more directly targets ITD computation compared to IPD loss and aims to improve ITD cue preservation.

In a previous study [160], ILD and IPD losses were proposed for enhancing binaural speech. However, exploring spatial losses for Target Sound Extraction (TSE) is crucial due to its broader application across

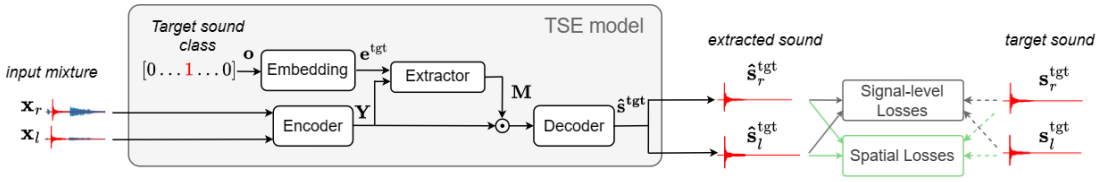

Figure 6.1: Our proposed binaural TSE system with signal-level and spatial losses.

diverse sound types compared to speech signals [13, 31, 116, 168].

To summarize, our contributions are the following:

- Introduce a novel spatial loss directly addressing ITD cues.
- Conduct a comparative analysis of three spatial losses (ITD, ILD, and the newly proposed ITD losses) for binaural TSE.

6.2 Binaural Target Sound Extraction

In this section, we begin by describing TSE and presenting a baseline binaural TSE model.

The goal of TSE is to isolate a sound signal belonging to a desired sound class from an observed signal $\mathbf{x}_m$, which comprises a mixture of sounds defined as:

$$\mathbf{x}_m = \sum_{i=1}^{I} \mathbf{s}_{i,m}, \tag{6.1}$$

where $\mathbf{s}_{i,m} \in \mathbb{R}^T$ is the sound signal of the i-th source at the m-th microphone, T is the number of samples, and I is the total number of sound sources. In the following, we define binaural mixtures as $\mathbf{x} = [\mathbf{x}_l, \mathbf{x}_r] \in \mathbb{R}^{T\times 2}$, where l and r represent the index of the left and right microphones, respectively. Similarly, we denote by $\mathbf{s}^{\text{tgt}} = \left[\mathbf{s}_l^{\text{tgt}}, \mathbf{s}_r^{\text{tgt}}\right] \in \mathbb{R}^{T\times 2}$ the binaural target signal, which corresponds to the desired sound class.

Binaural TSE aims to recover the binaural signal of the target by preserving the spatial information of the sound. Therefore, a TSE model follows the expression:

$$\hat{\mathbf{s}}^{\text{tgt}} = \text{TSE}(\mathbf{x}, \mathbf{o}; \theta), \tag{6.2}$$

where $\hat{\mathbf{s}}^{\text{tgt}}$ represents the extracted target sound signal, TSE$(\cdot)$ is a neural network with parameters θ, and $\mathbf{o} = [0, \ldots, 1, \ldots 0]^\top$ is a 1-hot vector representing the target sound class, i.e., it has a value of 1 for the index of the target class and 0 otherwise. The 1-hot vector has a dimension O corresponding to the number of classes the models can handle.

6.2.1 TSE Model

Our TSE model incorporates an encoder-decoder architecture with a mask estimation network, or extractor, shown in Fig. 6.1. In contrast to traditional single-channel TSE models [13, 168], our baseline model is Semantic Hearing [169], which extends the Waveformer model [168] to a binaural setting.

First, the input signal is processed by an encoder block:

$$\mathbf{Y} = \text{Encoder}(\mathbf{x}), \tag{6.3}$$

where Encoder$(\cdot)$ is the encoder layer, $\mathbf{Y} \in \mathbb{R}^{T' \times D}$ is the encoded representation of the input signal, D is the feature dimension and T' is the number of frames. The encoded mixture representation is then passed to an extractor, which predicts a mask $\mathbf{M} \in \mathbb{R}^{T' \times D}$ of the target sound to extract:

$$\mathbf{M} = \text{Extractor}(\mathbf{Y}, \mathbf{e}^{\text{tgt}}), \tag{6.4}$$

where Extractor$(\cdot)$ is an extractor neural network, and $\mathbf{e}^{\text{tgt}} \in \mathbb{R}^D$ is an embedding vector characterizing the target sound class. The embedding vector can be simply obtained from an embedding layer, which accepts the one-hot vector characterizing the desired sound class, $\mathbf{o}$, as input, i.e., $\mathbf{e}^{\text{tgt}} = \mathbf{W}\mathbf{o}$, where $\mathbf{W} \in \mathrm{R}^{D \times O}$ is an embedding matrix.

Finally, the estimate of the target sound is obtained by applying the decoder to the masked features as follows:

$$\hat{\mathbf{s}}^{\text{tgt}} = \text{Decoder}(\mathbf{M} \odot \mathbf{Y}), \tag{6.5}$$

where Decoder$(\cdot)$ is the decoder layer, and $\odot$ is the Hadamard product.

Training the TSE system involves using triplets consisting of binaural sound mixtures, target sound class labels, and binaural target sound signals. The model parameters are optimized by minimizing the training loss:

$$\theta = \underset{\theta}{\text{argmin}}\, \mathcal{L}\left(\mathbf{s}^{\text{tgt}}, \hat{\mathbf{s}}^{\text{tgt}}(\theta)\right). \tag{6.6}$$

Conventional TSE systems use only a signal-level training loss. Here, we investigate using a multi-task loss as:

$$\mathcal{L} = \alpha \mathcal{L}^{\text{signal}} + \beta \mathcal{L}^{\text{spatial}}, \tag{6.7}$$

where α and β are weights, and $\mathcal{L}^{\text{signal}}$, $\mathcal{L}^{\text{spatial}}$ are the signal-level and spatial losses, respectively, which we define below.

6.3 Methods

Now, we formulate the signal-level and spatial losses employed for training the TSE system presented in this chapter. Notably, we introduce our novel ITD loss in Section 6.3.2.

6.3.1 Signal-Level Losses

Conventional TSE systems typically employ a signal-level loss such as signal-to-noise ratio (SNR), scale-invariant SNR (SI-SNR) [84], or a combination of both [168, 169]. These losses can be adapted for binaural outputs by calculating a signal-level loss for each channel. For instance, the SNR loss for binaural outputs is formulated as:

$$\mathcal{L}^{\text{SNR}}(\mathbf{s}^{\text{tgt}}, \hat{\mathbf{s}}^{\text{tgt}}) = -\left(\frac{1}{2}\text{SNR}(\mathbf{s}_r^{\text{tgt}}, \hat{\mathbf{s}}_r^{\text{tgt}}) + \frac{1}{2}\text{SNR}(\mathbf{s}_l^{\text{tgt}}, \hat{\mathbf{s}}_l^{\text{tgt}})\right), \tag{6.8}$$

where $\text{SNR}(\mathbf{s}, \hat{\mathbf{s}}) = 10\log_{10}\left(\frac{||\mathbf{s}||_2^2}{||\mathbf{s}-\hat{\mathbf{s}}||_2^2}\right)$ is the SNR between the reference signal, $\mathbf{s}$, and the estimated target signal, $\hat{\mathbf{s}}$. We can define a similar loss for the SI-SNR.

Following [169], we use a weighted sum of SNR and SI-SNR losses as the signal-level loss:

$$\mathcal{L}^{\text{signal}} = 0.9\mathcal{L}^{\text{SNR}} + 0.1\mathcal{L}^{\text{SI-SNR}}. \tag{6.9}$$

6.3.2 Spatial Losses

To enhance the preservation of spatial cues in the TSE model, we explore integrating binaural metrics into the signal-level losses. To achieve this, we incorporate Inter-Aural Level Difference (ILD) and Inter-Aural Phase Difference (IPD) losses, which have been previously utilized in speech enhancement [160]. Additionally, we introduce a novel loss related to Inter-Aural Time Difference (ITD).

Interaural Level Difference Loss

ILD aims to measure the level difference between the channels of a signal. The ILD of a binaural signal is defined as:

$$\text{ILD} = 10\log_{10}\left(\frac{||\mathbf{s}_l||_2^2}{||\mathbf{s}_r||_2^2}\right), \tag{6.10}$$

where $\mathbf{s}_l \in \mathrm{R}^T$ and $\mathbf{s}_r \in \mathrm{R}^T$ are left and right signals.

Then, the ILD loss can be expressed as the mean of the absolute difference of the target and predicted ILDs:

$$\mathcal{L}^{\text{ILD}} = \left|\text{ILD}^{\text{tgt}} - \widehat{\text{ILD}}^{\text{tgt}}\right|, \tag{6.11}$$

where ILD^{tgt} and $\widehat{\text{ILD}}^{\text{tgt}}$ are the ILD of the reference and extracted target sound signals computed with Eq. (6.10). Note that prior work [160] defines this loss in the Short-Time Fourier Transform (STFT) domain, whereas we define it in the waveform domain.

Inter-Aural Phase Difference Loss

The Time Difference of Arrival (TDOA) between microphones results in phase differences between the received sound signals. To retain the spatial information of the sound source, a strategy is to prioritize the preservation of Inter-Aural Phase Difference (IPD) between the reference left and right signals in the extracted signals. The IPD between two signals can be calculated using the STFT of these signals.

However, the phase of a complex number has inherent periodicity (i.e., 0 and 2π are equivalent). This property, known as the circularity problem, makes direct subtraction of phases unreliable. To address this, we compute the IPD between the two channels of a signal as:

$$\mathrm{IPD}_{u,v} = \mathrm{atan}\left(\frac{\mathrm{Im}\left(S_{u,v,l}S^*_{u,v,r}\right)}{\mathrm{Re}\left(S_{u,v,l}S^*_{u,v,r}\right)}\right), \tag{6.12}$$

where $S_{u,v,r}$, $S_{u,v,l} \in \mathbb{C}$ represent the STFT of the right and left signals, u, v are the indexes of the time and frequency bins, * is the conjugate operation, $\mathrm{Im}(\cdot)$ and $\mathrm{Re}(\cdot)$ represent the imaginary and real parts of a complex number, and $\mathrm{atan}(\cdot)$ denotes the arctangent function.

To leverage the IPD as a loss function, we compute the MSE between the IPD of the target and predicted signals:

$$\mathcal{L}^{\mathrm{IPD}} = \frac{1}{UV}\sum_{u=1}^{U}\sum_{v=1}^{V}\left(\mathrm{IPD}^{\mathrm{tgt}}_{u,v} - \widehat{\mathrm{IPD}}^{\mathrm{tgt}}_{u,v}\right)^2, \tag{6.13}$$

where $\mathrm{IPD}^{\mathrm{tgt}}$ and $\widehat{\mathrm{IPD}}^{\mathrm{tgt}}$ are the IPD of the reference and extracted target sound computed with Eq. (6.12), and U and V are the number of time and frequency bins, respectively.

In [160], a similar IPD loss was used, where the IPD was computed directly as the angle of the ratio $S_{u,v,l}/S_{u,v,r}$[1]. This approach is mathematically equivalent to Eq. (6.12), but it handles phase wrapping differently. Eq. (6.12) defines the phase difference between $-\pi/2$ and $\pi/2$, corresponding to the smallest possible phase difference. Additionally, they utilized a binary mask to restrict the IPD computation to the regions where the source is active. In contrast, our experiments did not employ such a mask due to poorer performance, likely stemming from the challenge of defining a mask suitable for the diverse range of sounds covered by TSE.

Proposed Interaural Time Difference Loss

Our aim is to develop a TSE system that can preserve the ITD of the target sound since it is an important spatial cue used by humans to localize sounds [4, 106]. Therefore, we propose training the TSE model to output signals with an ITD close to that of the target sound.

ITD measures the difference in time of sound arrival between the left and right microphones. It can be obtained by finding the position of the highest peak in the cross-correlation between the left and right signals of the target sound.

We can compute the ITD as:

$$\mathrm{ITD} = \underset{t\in[-\tau,\tau]}{\mathrm{argmax}}\ c_t, \tag{6.14}$$

where c_t is the cross-correlation coefficient between the left and right signals at time step t. τ is a scalar limiting the predicted delay to be in a given range, which is related to the distance between the microphones.

Typically, the cross-correlation is computed using the generalized cross-correlation phase transform

[1] https://github.com/VikasTokala/BCCTN, accessed October 2024

Table 6.1: signal-level and spatial metrics for mixture, baseline TSE using only spatial loss and proposed systems using signal-level and spatial losses. We report ΔITD-GCC values (using GCC-PHAT as in Eq. (6.15)) and in parenthesis ΔITD values (using simple cross-correlation).

System		Signal-Level Metrics ↑ SI-SNR [dB]	↑ SNR [dB]	Spatial Metrics ↓ ΔILD [dB]	↓ ΔIPD [rad]	↓ ΔITD-GCC (ΔITD) [μs]	↓ FR [%]
(1)	Mixture	-0.74	-0.73	2.68	0.84	235.7 (263.0)	-
(2)	Baseline TSE w/ $\mathcal{L}^{\text{signal}}$	6.50	7.85	0.84	0.88	163.5 (86.3)	0.17
(3)	(2) + $\mathcal{L}^{\text{ILD}}$	6.72	8.10	**0.74**	0.83	168.5 (**74.8**)	**0.16**
(4)	(2) + $\mathcal{L}^{\text{IPD}}$	**6.76**	8.03	0.79	**0.49**	242.9 (80.1)	**0.16**
(5)	(2) + $\mathcal{L}^{\text{ITD}}$	6.74	**8.11**	0.78	0.84	**137.3** (79.0)	**0.16**

(GCC-PHAT) algorithm. [80] as follows:

$$\mathbf{c} = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(\mathbf{s}_l) \odot \mathcal{F}(\mathbf{s}_r)^*}{|\mathcal{F}(\mathbf{s}_l) \odot \mathcal{F}(\mathbf{s}_r)^*|}\right), \tag{6.15}$$

where $\mathbf{c} = [c_{t=-T}, \ldots, c_{t=0}, \ldots, c_{t=T}] \in \mathbb{R}^{2T+1}$ is the vector of the cross-correlation coefficients, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier transform (FT) and inverse FT (IFT), respectively.

A loss function for ITD should quantify the difference between the ITD of the reference target sound signals, $\mathbf{s}^{\text{tgt}}$, and that of the extracted signals, $\hat{\mathbf{s}}^{\text{tgt}}$. However, as indicated in Eq. (6.14), ITD computation involves the argmax operation, which is not differentiable. Therefore, we define the ITD loss as the mean squared error (MSE) between the cross-correlation functions of the reference and extracted signals:

$$\mathcal{L}^{\text{ITD}} = \frac{1}{2\tau+1}\sum_{t=-\tau}^{\tau}\left(c_t^{\text{tgt}} - \hat{c}_t^{\text{tgt}}\right)^2, \tag{6.16}$$

where c_t^{tgt} and $\hat{c}_t^{\text{tgt}}$ are the cross-correlation between the reference and extracted signals computed with Eq. (6.15). Note that the proposed ITD loss is differentiable since all operations required to compute the cross-correlations, including FT and IFT, are differentiable.

6.4 Experiments and Results

6.4.1 Experimental Settings

Datasets

In our experiments, we used an openly available dataset of binaural sound mixtures [169]. This dataset comprises simulated reverberant mixtures consisting of three to four sound events added to urban background noise. It leverages 20 sound classes sourced from FSD50K [22] (general-purpose), ESC-50 [127] (environmental sounds), MUSDB18 [135], and noise files from the DISCO dataset [28].

The background sounds were sourced from TAU Urban Acoustic Scenes 2019 [102]. Binaural mixtures were created by convolving the sound source signals with room impulse responses (RIRs) and head-related

transfer functions (HRTFs). We utilized HRTFs from 43 subjects from the CIPIC corpus [1], along with real and simulated RIRs from three corpora [68, 69, 148]. The sampling frequency of all signals was 44.1 kHz.

We dynamically mixed the data using Scaper [145]. From each mixture, we consistently extracted one foreground source (the same across all experiments) [169]. The training utilized 6-second mixtures, with 100,000 training samples, 1,000 validation samples, and 10,000 testing samples.

System Configuration

We employed the same configuration for the TSE system as used in [169].[2] The system performs online processing with a latency of 20 msec and comprises a total of 1.74M trainable parameters.

The model architecture includes a 1-D convolutional encoder and a 1-D transposed convolutional decoder. The encoder's 1-D convolutional layer operates with a stride of $L = 32$ samples and a kernel size of $K = 3L$. Following the mask extraction process by the extractor, the 1-D transposed convolutional layer reconstructs the waveform of the target sound.

The extractor features an encoder-decoder architecture conditioned on the target class embedding, achieved by multiplying the output of dilated convolution with the target sound embedding, $\mathbf{e}^{\text{tgt}}$. The encoder comprises a series of 1-D dilated convolution (DCC) layers followed by a Transformer-like layer. The DCC layers employ a kernel size of $K = 3$ and dilation factors $\{2^0, 2^1, ..., 2^9\}$. The Transformer-like layer consists of two multi-head attention (MHA) layers with 8 heads.

We trained all models for 80 epochs using a batch size of 32 and a learning rate of 5×10^{-4}. For the IPD loss, we computed the complex short-time Fourier transform (STFT) using a Hann window of size 1024, a hop length of 256, and an FFT size of 1024 samples. All models utilized the signal-level loss defined in Eq. (6.9), combined with spatial losses according to Eq. (6.7) with a fixed weight $\alpha = 1$.

Hyperparameters, including loss weights β and the number of epochs, were selected based on the best SI-SNR metric values from the validation set. Specifically, β values were set to 0.1, 1, and 1 for ILD, IPD, and ITD losses, respectively. For the ITD loss, we set $\tau = 1$ ms.

Evaluation Metrics

Following Semantic Hearing [169], we evaluated performance using both signal-level metrics (SI-SNR and SNR) and spatial metrics, computed as differences between the reference and extracted signals: ΔILD, ΔIPD, and ΔITD. ΔILD and ΔIPD were computed using Eq. (6.11) and Eq. (6.13), respectively. ΔITD was obtained as the absolute difference between the ITD of the reference and extracted signals, where ITD is computed with Eq. (6.14). For ITD computation, we utilized GCC-PHAT, known for its robustness to reverberation. Additionally, we also computed ITD using simple cross-correlation for comparison with prior work [169]. These metrics are named ΔITD-GCC and ΔITD, respectively.

Finally, we also report the failure rate (FR) as an estimate of how often TSE failed to correctly identify the target sound in the mixture. FR is defined as the percentage of test samples for which the SI-SNR

[2] We used the model variant $D = 256$ proposed by Semantic Hearing https://github.com/vb000/SemanticHearing

improvement is below 1 dB [12].

6.4.2 Experimental Results

Table 6.1 compares the performance of the proposed losses with that of the unprocessed mixture and a baseline TSE that only uses the signal-level loss defined in Eq. (6.9) [169]. We observe that the baseline TSE significantly reduces ILD and ITD errors compared to the mixture but shows little improvement in IPD. In fact, most experiments did not show reduced IPD errors, indicating the challenge of using IPD metrics effectively for TSE.

Using the spatial losses in addition to the signal-level loss (systems (3)-(5)) outperforms the baseline TSE system (system (2)) for most metrics. These results indicate that the multi-task loss does not degrade extraction performance in terms of signal-level metrics and failure rate (FR); it even slightly improves them.

Specifically, we confirm that adding a spatial loss enhances performance on the corresponding spatial metric: (i) using the ILD loss (system (3)) achieves the best performance in terms of ΔILD but does not consistently improve ΔITD, (ii) employing the IPD loss (system (4)) significantly reduces ΔIPD and slightly improves ΔILD. However, this improvement does not extend to ITD errors, suggesting limitations of the IPD loss. (iii) In contrast, our newly proposed ITD loss (system (5)) improves all spatial metrics compared to the baseline while achieving comparable SI-SNR, SNR, and FR values. Particularly noteworthy is the substantial improvement in ΔITD by 26.2 μs (more than 1 sample) or a relative improvement of 16. Regarding the FR values, there is no change between the systems since the SI-SNR does not present a high variation between them.

In Fig. 6.2, we show the ITD performance of the baseline system compared to the mixture and our proposed ITD loss.

6.5 Conclusions

In this chapter, we introduced a novel loss function centered around ITD, designed to enhance binaural characteristics and reduce ITD discrepancies between left and right channels. Our proposed ITD loss surpasses traditional IPD methods by enhancing the binaural cue restoration of TSE across ILD, IPD, and ITD metrics, while maintaining strong performance in signal-level metrics.

6.5.1 Limitations

The limitations of this work can be summarized in the following: (i) the improvement of 26.2μs in the ITD metric might not perceptually noticeable, (ii) only the 20 classes in the dataset were explored in this work, therefore, more classes could be evaluated.

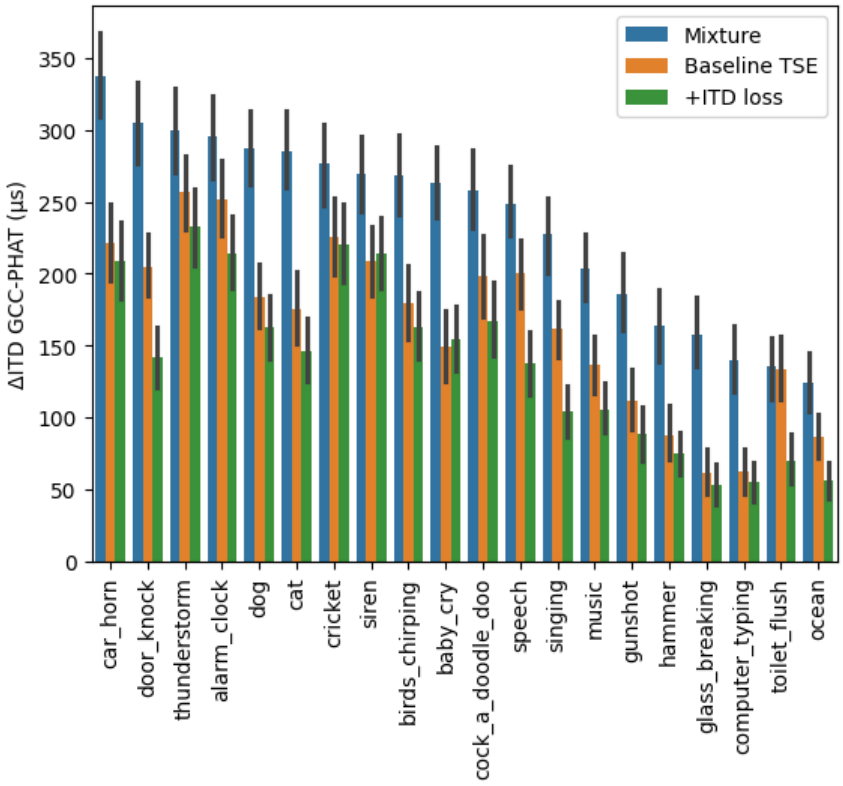

Figure 6.2: ITD results per class.

6.5.2 Benefits

The versatility of our proposed ITD loss extends beyond TSE and holds potential for application in other speech and audio processing tasks, such as binaural speech enhancement and speech separation.

6.5.3 Future Work

Following past research in binaural processing [4, 15], this work could be extended by comparing the actual results with the effect of including a low-pass filter around 1.5KHz in ITD and IPD metrics and losses, and a high-pass filter of 3KHz in the ILD formulation. Since the spatial cues formulation is defined differently across different research studies, this work has not addressed such nuances.

Chapter 7

TARGET SOUND EXTRACTION WITH AUDIO FOUNDATION MODEL

ABOUT THIS CHAPTER

The work presented in this chapter has been submitted at ICASSP 2025. This work was conducted entirely in NTT CSL laboratories under the supervision of Dr. Marc Delcroix and the NTT CSL Keihanna laboratories research team. In particular, Marc Delcroix proposed the idea of integrating the audio foundation model into a speaker extraction model and adapt it to target sound extraction, Daisuke Niizumi provided the audio foundation model for the experiments and the other members supervised the work. I implemented the model and conducted the experiments.

C. Hernandez-Olivan, M. Delcroix, T. Ochiai, D. Niizumi, N. Tawara, T. Nakatani, and S. Araki.
SoundBeam meets M2D: Target Sound Extraction with Audio Foundation Model.
Submitted to the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2025.

7.1 Introduction

This chapter is related to the previous one since it is also related to Target sound extraction (TSE). However, in this chapter, we aim to improve the general performance of TSE in a single-channel model. Following

up the previous chapter, target sound extraction (TSE) is a technology that isolates a desired source signal in a mixture of arbitrary sounds. This technology has potential applications in sound production, hearable devices, and related fields [168]. TSE can be implemented using a neural network that processes the sound mixture as input and outputs the desired sound signal, based on target sound embeddings that differentiate the target from other sounds in the mixture. These target sound embeddings can be derived from class labels [13, 116], audio enrollments (queries of the target sound) [13, 31], or a combination of both [13]. While other studies have explored additional cues such as video [177] and language [93, 117], these are beyond the scope of the work described in this chapter.

Class label clues are represented as one-hot vectors, which are mapped into an embedding space corresponding to different sound classes through an embedding layer. These class label-based TSE systems can directly learn target sound embeddings optimized for TSE. However, they are limited in their ability to extract sounds that were not encountered during training. In contrast, enrollment-based TSE systems map audio enrollments into the embedding space using a clue encoder network, enabling the extraction of sounds that share similar characteristics with the enrollment sample without relying explicitly on sound class labels. As a result, these systems can generalize to sounds not seen during training. Recently, we introduced SoundBeam [13], a TSE system trained by alternating between class label and enrollment clues, thereby combining the advantages of both approaches and improving extraction performance due to the multi-task training framework.

Training a TSE system presents significant challenges, as it must learn a sound representation that allows for the identification of specific sounds within a mixture, while simultaneously acquiring the ability to extract the target sound signal. This task becomes particularly difficult when learning the problem from scratch, as it necessitates training the system on a vast number of sound combinations to ensure the model generalizes effectively to a wide range of sound classes.

Recently, foundation models, including those based on self-supervised learning (SSL), have received increased attention for facilitating the training of complex tasks. These models develop rich, general-purpose feature representations from extensive datasets, which can subsequently be utilized to train task-specific models. For instance, pre-trained SSL models, such as Whisper [129], have been effectively applied in various domains, including automatic speech recognition (ASR) [103], diarization [7], and speech enhancement [124]. Beyond speech processing, foundation models have also been explored for general audio signal representations, demonstrating promising results in tasks such as sound event classification [32, 78, 114]. Notably, the recently proposed masked-modeling duo (M2D) model [115], trained on AudioSet [30] using a combination of a *modified masked auto-encoder* (MAE) [44] and *sound label classification* objectives, has achieved state-of-the-art performance across various audio processing tasks [115].

In this chapter, we investigate the potential of leveraging the robust audio foundation model, M2D, for Target Sound Extraction (TSE). We hypothesize that the M2D model, with its dual training objectives, is particularly well-suited for TSE tasks. Specifically, TSE involves: 1) identifying the target sound within a mixture, which aligns with the M2D model's sound label classification objective, and 2) extracting the target sound signal, which corresponds to the Modified Masked Auto-Encoder (MAE) training objective.

We propose a novel TSE system that integrates SoundBeam and M2D. Our approach utilizes the pre-trained M2D model to 1) generate target sound embeddings from enrollment clues, and 2) produce a representation of the mixture that serves as additional input for SoundBeam. To achieve this, we introduce an Adaptive Input Enhancer (AIE) module that combines the feature representations from both the TSE encoder and the M2D model.

We conduct TSE experiments with noisy and reverberant sound mixtures and demonstrate that incorporating representations from the M2D model for both enrollment and mixture significantly enhances performance. This improvement is particularly notable when using enrollment clues, as it leads to more discriminative target sound class embeddings. These findings underscore the efficacy of employing general audio foundation models, such as M2D, for TSE tasks.

While our primary experiments are conducted with SoundBeam, an offline TSE system, we also address the need for online processing in many applications. To this end, we perform additional experiments showing that M2D can benefit causal and lightweight TSE systems, such as Waveformer [168].

7.2 Related Work

Pre-trained foundation models have been utilized in various downstream tasks, including speech enhancement [63], target speech extraction [124], and Target Sound Extraction (TSE) [93, 178]. Concurrent with our work, recent pre-prints [93, 178] have also investigated the application of audio foundation models for TSE. In [93], the Contrastive Language-Audio Pre-trained (CLAP) model [171] was employed to obtain target sound embeddings from text descriptions (audio captions) and enrollment clues, in addition to processing the input mixture. Similarly, [178] utilized a self-supervised learning (SSL) model trained with a Modified Masked Auto-Encoder (MAE) loss to encode the audio mixture, while a pre-trained sound event detection (SED) system was used to derive target sound embeddings from enrollment.

In contrast, our work explores the use of an audio foundation model for TSE systems that can incorporate both class labels and enrollment clues. Moreover, we streamline the system by employing the same pre-trained foundation model to encode both the mixture and the enrollment, thereby eliminating the need for an SED system and simplifying the overall approach.

Our study also builds on our previous research [124], which investigated SSL models for target speech extraction. Here, we examine whether similar concepts can be effectively applied to the extraction of arbitrary sounds using a combined target sound class/enrollment TSE approach, applicable to both online and offline models.

7.3 TSE system

Single-channel TSE aims to extract a desired sound $\mathbf{s}^{\mathrm{tgt}} \in \mathbb{R}^{1\times T}$ from a sound mixture $\mathbf{x} \in \mathbb{R}^{1\times T}$ using a clue $\mathbf{o}$;

$$\hat{\mathbf{s}}^{\mathrm{tgt}} = \mathrm{TSE}(\mathbf{x}, \mathbf{o}; \theta), \tag{7.1}$$

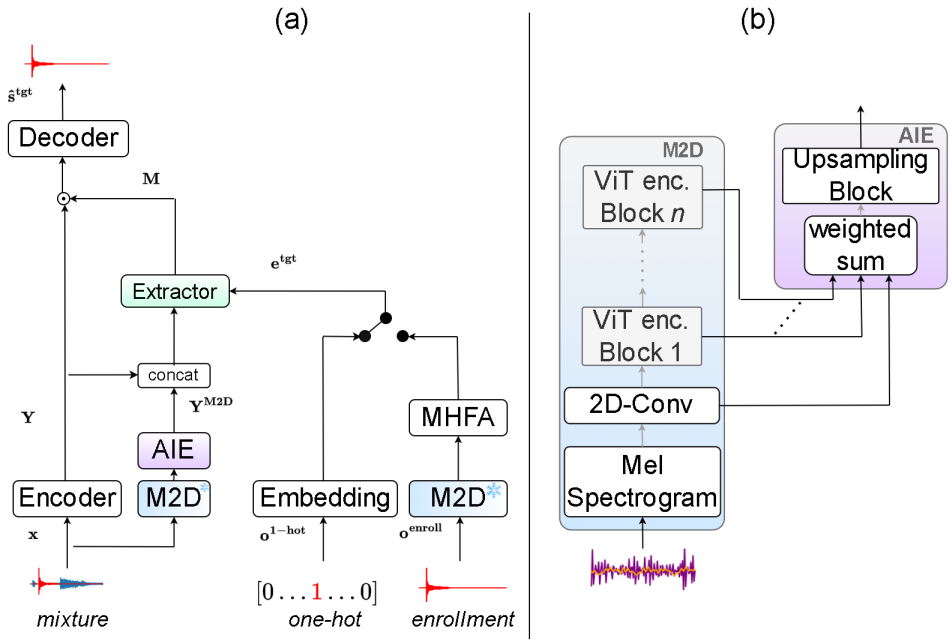

Figure 7.1: a) Generic TSE system with SSL model and b) M2D model and AIE module.

where TSE$(\cdot)$ is a neural network with parameters θ. The clue $\mathbf{o}$ is either a one-hot vector, $\mathbf{o}^{\text{1-hot}}$ representing the target sound class or an enrollment of the desired sound to extract, $\mathbf{o}^{\text{enroll}}$. Class label clues, $\mathbf{o}^{\text{1-hot}} \in \mathbb{R}^C$, are restricted to the classes in the training set. However, with enrollment clues, we can extend the extraction to sound classes unseen during training [13].

7.3.1 Model Architecture

As we explained in the previous chapter, a typical neural TSE system consists of an encoder-decoder model with a mask estimation network or extractor and a clue encoder module, which derives target sound embeddings from the clues. The encoder processes the input signal $\mathbf{Y} = \text{Encoder}(\mathbf{x})$, where $\mathbf{Y} \in \mathbb{R}^{T' \times D}$ is the encoded mixture and T' is the time dimension after the encoding, and D the feature dimension. Then, the extractor predicts a mask conditioned by the target sound embedding vector $\mathbf{M} = \text{Extractor}(\mathbf{Y}, \mathbf{e}^{\text{tgt}})$, where $\mathbf{M} \in \mathbb{R}^{T' \times D}$ is the estimated mask and $\mathbf{e}^{\text{tgt}}$ is the target sound embedding. The encoded mixture $\mathbf{Y}$ is multiplied by the mask and converted back into a signal $\hat{\mathbf{s}}^{\text{tgt}} \in \mathbb{R}^{1 \times T}$ with the decoder $\hat{\mathbf{s}}^{\text{tgt}} = \text{Decoder}(\mathbf{M} \odot \mathbf{Y})$, to obtain the extracted signal.

We obtain the target sound embedding, $\mathbf{e}^{\text{tgt}}$, by processing the target sound clue with the clue encoder. For class label clues, the clue encoder consists of an embedding layer, and for enrollment clues, it is a neural network combined with an average pooling layer.

7.3.2 Training Loss

All network parameters, including the encoder, decoder, extractor, and clue encoder modules are jointly trained from scratch using the following loss:

$$\begin{aligned}\mathcal{L}(\theta) =& \alpha \mathcal{L}^{\text{ext}} \left(\text{TSE}(\mathbf{x}, \mathbf{o}^{\text{1-hot}}; \theta), \mathbf{s}^{\text{tgt}} \right) \\ &+ (1-\alpha) \mathcal{L}^{\text{ext}} \left(\text{TSE}(\mathbf{x}, \mathbf{o}^{\text{enroll}}; \theta), \mathbf{s}^{\text{tgt}} \right), \end{aligned} \tag{7.2}$$

where $\mathcal{L}^{\text{ext}}$ is the extraction loss, which here is the weighted sum of the signal-to-noise ratio (SNR) and scale-invariant SNR (SI-SNR) losses following [49, 168]. α is a multi-task weight that we fix to $\alpha = 0.5$ in all experiments following our prior works [13]. This loss enables us to jointly train a model for class label and enrollment clues, ensuring good performance thanks to the multi-task training effect and generalization to unseen classes with enrollment clues.

7.3.3 Baseline Systems

In this chapter, we use two models as our baseline systems, SoundBeam [13], and Waveformer [168], which mainly differ by the implementation of the extractor. Both systems use 1D convolution and deconvolution layers for the encoder and decoders. SoundBeam is an offline system based on Conv-TasNet[92]. Waveformer [168] is a lightweight online TSE model that uses dilated convolution and multi-head attention layers for the extractor. The original Waveformer used only class label clues. We train it here using both class label and enrollment clues as SoundBeam. For both SoundBeam and Waveformer models, we use one convolution block of TasNet for the enrollment clue encoder, followed by an average pooling layer.

7.4 M2D Model

Let us briefly review the M2D model, before explaining how to use it for TSE in the next Section. We use the M2D model pre-trained on AudioSet [30] as a pre-trained foundation model. We use the M2D variant, denoted M2D-AS in [115], which integrates a self-supervised learning (SSL) objective with an additional supervised loss for label prediction of AudioSet audio clips. The SSL objective is based on an enhanced Modified Masked Auto-Encoder (MAE) training approach [115]. In this approach, the model learns to predict features of the masked portion of the input signal using only the unmasked portion. This contrasts with conventional training methods, which use the entire input signal as the reference for prediction, potentially leading to contamination of the reference by the prediction input. M2D's methodology, by using only the masked portion for training references, mitigates this contamination and thus simplifies the training process.

This approach encourages the model to develop a more meaningful local representation of the signal, which can be advantageous for Target Sound Extraction (TSE). The dual training objectives of M2D are particularly well-suited for TSE, as the task involves both reconstructing the signal (related to the MAE objective) and identifying the source (related to the label prediction loss).

The network architecture of M2D is based on the Vision Transformer (ViT) [19], as illustrated in Fig. 7.1-(b). It is composed by a 2D convolutional (2D-Conv) layer followed by encoder blocks, which we refer to as ViT encoder blocks. The input to the M2D model consists of mel spectrogram features, which are extracted using a window length of 25 milliseconds and a shift of 10 milliseconds. The 2D-Conv layer processes the input spectrogram and projects it into tokens. For this model, we use a patch size of 80×2, resulting in a segmentation of the input every 20 milliseconds, corresponding to a frame rate of 50 frames per second. The model is trained on a dataset of 2,005,132 audio clips, each 10 seconds in length, sourced from AudioSet (totaling 5,569 hours), with clips cropped into 6-second segments.

7.5 SoundBeam Meets M2D

Figure 7.1 shows the proposed Target Sound Extraction (TSE) system utilizing the pre-trained M2D model. The M2D model is employed to process both the audio mixture and the enrollment in a manner similar to that described in [124].

7.5.1 M2D-based Enrollment Clue Encoder

In this system, we leverage the pre-trained M2D model to obtain target sound embeddings. Specifically, we use an advanced attentive pooling technique known as Multi-Head Factorized Attentive Pooling (MHFA), which was initially introduced for speaker verification tasks [125].

MHFA performs attentive pooling to transform the enrollment into a fixed-dimension enrollment vector. It generates two distinct weighted sums from the internal layers of the M2D model: one serves as values and the other as keys to compute attention weights for the pooling layer. The output of the pooling layer is then projected into a lower-dimensional space through a linear layer. MHFA has demonstrated promising results in both speaker verification [125] and target speech extraction [124].

7.5.2 M2D for Mixture Encoder

To exploit the rich feature representation of the SSL model, we concatenate the encoded mixture, $\mathbf{Y}$, with features obtained from the pre-trained M2D model, $\mathbf{Y}^{\mathrm{M2D}}$. Different layers of a pre-trained SSL model capture different types of information, consequently, we use a weighted sum of the output of all intermediate processing blocks of the M2D model (the first CNN block and each ViT encoder block). In addition, the time resolution of the M2D model and that of the encoder typically used for TSE differ. We thus use a deconvolution layer to upsample the feature representation obtained from the SSL model. The AIE performs these two operations, weighted-sum and upsampling with deconvolution, as:

$$\mathbf{Y}^{\mathrm{M2D}} = \mathrm{DeConv}\left(\sum_i w_i \mathbf{Z}_i\right), \tag{7.3}$$

where $\mathbf{Z}_i$ are the output of the i-th processing block of the SSL model and w_i are learnable weights constraint so that $\sum_i w_i = 1$. Note that [124] used a cascade of deconvolution layers to handle the different time-resolutions of the convolutional layers of the speech SSL models [7, 59]. Here, since the M2D model downsamples the signal using a single convolutional layer, we can upsample all layers with the same deconvolution layers. Another distinction is that the M2D model adds a learnable [CLS] token at the beginning of the sequence to learn global embeddings. We thus pad the output sequence of the CNN layer with a zero to adjust its length with that of the output of the transformer layers.

7.5.3 Causal Extension

Most of our investigations are conducted using the offline SoundBeam model. For online processing, while an online version of SoundBeam could be developed following the online ConvTasNet configuration [92], we opted instead for the lightweight Waveformer model, which has demonstrated promising results for online TSE [168, 169].

It is important to note that the M2D model is non-causal due to its attention and layer normalization modules, making it unsuitable for online processing. To adapt the M2D model for online TSE, we propose modifying the network architecture by replacing the attention mechanisms and layer normalization in the ViT encoder blocks with causal alternatives, such as masked attention and cumulative layer normalization [92]. This approach involves modifying an already trained non-causal M2D model to avoid the need for retraining. While this adjustment may not yield optimal results and could potentially lead to weaker representations, we anticipate that the TSE system will still benefit from the meaningful information captured by the feature representations of the causal M2D model.

7.6 Experimental Settings

7.6.1 Datasets

Same as in the previous chapter, we used the dataset proposed by Semantic Hearing [168], which consists of simulated reverberant binaural mixtures of three to four sound events added to urban background noise. This dataset leverages 20 sound classes from FSD50K [22] (general-purpose), ESC-50 [127] (environmental sounds), MUSDB18 [135] and noise files for the DISCO dataset [27]. The background sounds were taken from TAU Urban Acoustic Scenes 2019 [102]. Binaural mixtures were generated using sound source signals with room impulse responses (RIRs) and head-related transfer functions (HRTFs). We used HRTFS from 43 subjects from the CIPIC corpus[1], and real and simulated RIRs from three corpora [68, 69, 148]. The sampling frequency of all signals is 16 kHz. To train single-channel models in this work, we average the right and left channels of the mixtures and enrollments.

For training, the data is mixed on the fly with Scaper [145]. From each mixture, we extracted two foreground sources. The training was done with 6-sec mixtures. The number of training, validation, and testing mixtures were 100K, 1K, and 10K, respectively. During training, the enrollment cues are chosen by

randomly sampling audio files from the target sound class, different from the ones used in each mixture. For validation and test, the enrollments are fixed for each mixture. Note that the enrollments are also reverberant.

7.6.2 System Configuration

We experiment with the following four models:

Baseline SoundBeam (Offline Model)

We used one 1D-Conv and 1D-DeConv layers for the encoder and decoder, respectively, with a kernel size of 16 and a stride of 8. For the extractor, we used 512 filters, 8 blocks, 3 repeats, and a global layer norm. For the class label clue embedding network, we used 2 fully connected layers with layer normalization following [168]. For the enrollment embedding network, we used a kernel size of 40, a stride of 20, and an embedding size $D = 256$.

SoundBeam with M2D

We used the same configuration as the baseline SoundBeam model but replaced the enrollment clue encoder with the M2D-based one as explained in Subsection 7.5.1. In addition, we also experimented using the M2D model to process the mixtures. In that case, we used the AIE module described in Subsection 7.5.2, where we used two 1D-DeConv layers of kernels $\{2, 25\}$ and strides of $\{2, 20\}$ to implement the deconvolution operation of Eq. (7.3).

Baseline Waveformer (Online Model)

We used the same setting as [169]. The encoder and decoder consist 1D-Conv and 1D-DeConv layers with stride of $L = 32$ samples and a kernel size of $K = 3L$; therefore, the lookahead of the model is 32 frames, and the downsampling factor is 8. For the extractor, we used 10 DCC layers with a kernel size of K=3 and dilation factors set to $\{2^0, 2^1, ..., 2^9\}$. The masker decoder consisted of one MHA layer with 8 heads. We used the same clue networks and parameters as for SoundBeam.

Waveformer with M2D

We used the same configuration as the baseline Wavefomer model but replaced the enrollment clue encoder with the M2D-based one as explained in Subsection 7.5.1. Processig the enrollment can be performed offline, and we thus used the original non-causal M2D model to process the enrollment.

To keep the model online, we used the causal implementation of M2D, explained in Section 7.5.3, to process the mixture. Here, we implemented the deconvolutional layer of Eq. (7.3) with four 1D-DeConv layers of kernels $\{2, 2, 2, 1\}$ and strides of $\{2, 2, 2, 5\}$. Then, we added a bottleneck CNN layer of kernel and stride 1 that processes the encoded mixture concatenated with the output of the AIE CNN block.

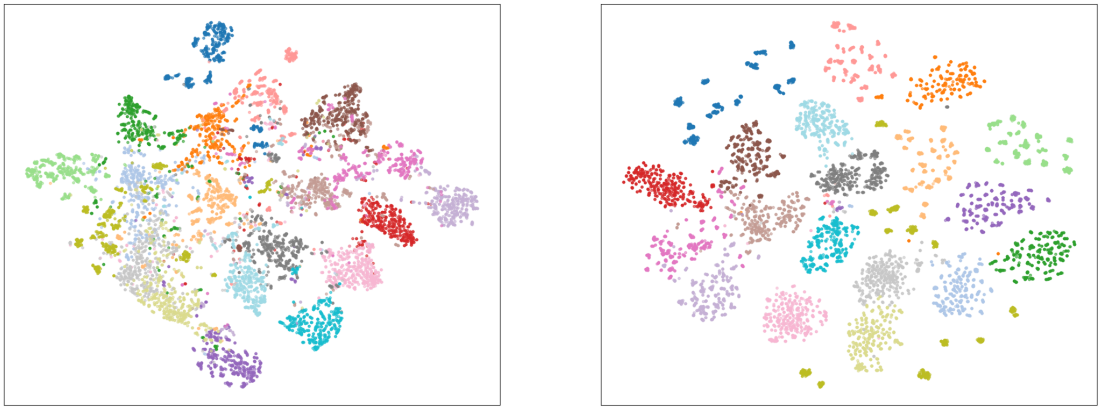

Figure 7.2: TSNE of the enrollment clue inference for: SoundBeam model (left) and SoundBeam model with M2D for both the mixture and enrollment (right).

Table 7.1: TSE performance in terms of SNRi for offline SoundBeam model with and without M2D. The SNR of the mixture is -0.4 dB. ✓indicates that the M2D model is used for the enrollment or the mixture.

Use M2D model for enrollment	Use M2D model for mixture	Clue used for inference	SNRi ↑ [dB]	FR ↓ [%]
-	-	class label	9.48	0.05
		enrollment	7.72	0.19
✓	-	class label	9.52	0.05
		enrollment	9.13	0.09
✓	✓	class label	10.49	0.03
		enrollment	9.86	0.08

Other Settings

We trained all models for 80 epochs with four GPUs using a batch size of 24. We chose the models that achieved the best SI-SNR score on the cross-validation set.

During inference, we perform extraction using either class label or enrollment clues. For each mixture, we extract two target sounds for the evaluation. We report results in terms of SNR and failure rate (FR), which is the percentage of samples with SNR improvement (SNRi) below 1 dB.

7.6.3 Experiments with SoundBeam (Offline TSE)

In Fig. 7.2 we show the TSNE embeddings of the enrollment clue inference for SoundBeam and SoundBeam trained with the M2D model for both mixture and enrollment embeddings. We can observe how the M2D learns more discriminative embeddings and thus, helps the system to identify the correct source to extract.

Table 7.1 compares the baseline SoundBeam model with two variants that use the M2D model to

Table 7.2: TSE performance in terms of SNRi for online Waveformer model with and without M2D.

Use M2D model for enrollment	Use M2D model for mixture	Clue used for inference	SNRi ↑ [dB]	FR ↓ [%]
-	-	class label	7.75	0.07
		enrollment	6.01	0.23
✓	-	class label	7.43	0.09
		enrollment	7.01	0.12
✓	✓	class label	7.73	0.07
		enrollment	7.17	0.10

process either the enrollment or the mixture, depending on whether class label or enrollment clues are used for inference. We observe that incorporating the M2D model significantly improves extraction performance when using enrollment clues for inference, with an improvement of over 2 dB. This improvement can be partially attributed to better target sound identification, as evidenced by the reduction in false rejection rates (FR). The optimal performance was achieved when the M2D model was employed for both the enrollment clue encoder and the mixture processing.

Using the M2D model to process the mixture also improves the performance of class label clue-based inference by 1 dB. These results validate the efficacy of the M2D model for the Target Sound Extraction (TSE) task. Figure 7.3 presents the performance metrics for each of the 20 target sound classes, demonstrating that the use of M2D consistently yields improvements across nearly all target sound classes.

7.6.4 Experiments with Waveformer (Online TSE)

Table 7.2 presents the extraction performance for experiments conducted with the Waveformer online model. Similar to the SoundBeam experiments, we observe that incorporating M2D improves performance when using enrollment clues for inference, although no such improvement is noted with class label clues. Despite this, the model offers advantages by supporting both class label and enrollment clue-based TSE with comparable performance levels.

It is noteworthy that the lack of performance improvement with class label clues may be attributed to the suboptimal representation of the mixture provided by the causal version of the M2D model. This issue could potentially be addressed by either pre-training a causal M2D model directly or by jointly retraining its parameters alongside the TSE task. These avenues will be explored in our future research.

7.7 Conclusions

In this chapter, we proposed a new TSE system that combines the SoundBeam model with the strong M2D audio foundation model. We showed that the M2D model can significantly improve performance using class label and enrollment clues, by providing more discriminative target sound embeddings and better representation of the mixture. In future work, we would like to explore causal implementation or retraining

of M2D as well as lightweight configuration to boost the performance of online TSE.

7.7.1 Limitations

The main limitation of this work is the performance of the causal system evaluated with one-hot clues. As we mentioned, the performance is similar to the causal system when not using the M2D model, therefore, the M2D model could be retrained or fine-tuned for the TSE task to address this issue. Also, similar to the previous chapter, we only validated the performance of the systems across 20 classes. It would be interesting to see how the systems perform when adding more classes and also in other fields like music.

7.7.2 Benefits

Our approach improves the overall performance of TSE when using enrollment clues. This can benefit few-shot learning when adding classes that the model has not seeing during training.

7.7.3 Future Work

Future work should rely on the fine-tuning of the M2D momdel to improve the overall performance and also exploring causal TSE systems that can benefit from the M2D model when using it in the enrollment embedding.

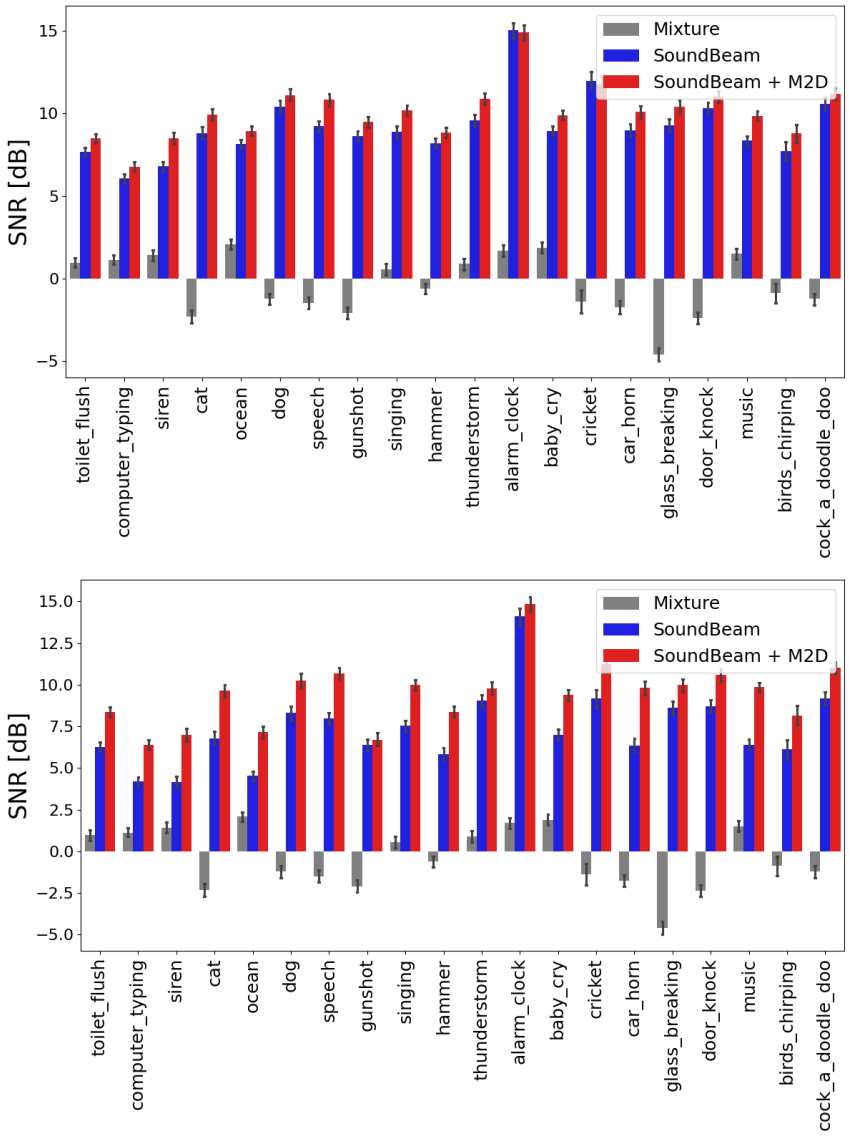

Figure 7.3: SNR for different target sound classes using class label (top figure) or enrollment (bottom figure) clues.

Chapter 8

CONCLUSIONS

ABOUT THIS CHAPTER

In this chapter, we present the conclusions and future research that benefit from the work presented in this thesis.

8.1 Summary and Conclusions

In this thesis, we presented different techniques to address a wide variety of tasks in the music and audio processing fields, from a symbolic music package for music generation to target sound extraction in audio mixtures with general audio sounds. In spite that the topics presented in this thesis belong to different fields, we believe that all of them could make part, in the future, of a single multi-modal model that could be used by music practitioners.

We introduced a system for audio boundary detection that segments songs by their structure. The system uses a convolutional neural network and self-similarity lag matrices computed from different audio features.

For symbolic music generation, we developed an open source package that contains several state-of-the-art tokenizers, evaluation methods and a model based on GPT. To our known, is the first package that includes all the necessary steps to generate symbolic music. However, there is a limitation in the number of models that we implemented, which is only one. We hope to add more models in the future and that the software could help the research community to safe time regarding data processing.

In music production, we conducted research on audio restoration and target sound extraction. Both works can be used by music producers for restoring recorded vocals and remixing purposes. Regarding audio restoration, we used a diffusion model previously developed for piano restoration with different posterior

sampling techniques already introduced in the image domain. We addressed bandwidth extension and declipping. In spite that the declipping results with some of the presented techniques outperform state-of-the-art ones and can be used in music studios, bandwidth extension task could be further improved. According to target sound extraction, we presented two works. The first one improves interaural time difference with a new loss. The advantage of that is the fact that the improvement of spatial metrics does not degrade signal-metrics. However, it is unclear how traditional binaural signal processing techniques, such the addition of high and low pass filters for the different spatial metrics can change the results, whether the filters are only used for evaluation or also in the loss computation. We reserve this research as part of future work. In our second work, we improve the performance of state-of-the-art single-channel target sound extraction models by using an audio foundation model in the enrollment clue and mixture. The audio foundation model helps the overall system discriminating sound classes which leads to the improvement of SNR. Also, the causal system benefits from the foundation model which, in case we use it only in the enrollment clue, can keep the full model causal and be useful in applications that require online processing.

8.2 Future Work

In the audio boundary detection task, there are many way of improving the work that we currently presented in this thesis and the current works. Music structure analysis is not a high researched topic in MIR, therefore, not many solutions have been proposed after we published our work. One possible future research would be to improve the performance of current systems with better models, such as Transformer-based models as is has been done recently [123]. However, as we mentioned previously, it is necessary to propose new datasets, or annotate current ones, for this task. In a recent work, we propose another work for MSA in the symbolic domain to address this issue with graph representations and unsupervised changepoint methods [50]. In that way, music practitioners could take advantage of such systems and use it for educational or production purposes.

According to the symbolic music generation field, we believe that our package can help the music research community, from novel researchers that aim to develop new systems to well -established researchers that can use the package to save time on data pre-processing and evaluation. The software can be extended in many ways, such as implementing more state-of-the-art models for music generation or handle more symbolic music formats.

Our work in audio restoration can inspire new works to explore diffusion sampling techniques specifically for audio tasks. We proved how such techniques, initially developed and tested with images, worked in the audio domain. However, how other diffusion posterior sampling techniques work for different audio processing tasks remain unknown. We also believe that newer sampling techniques, apart from the ones developed for images, can be tested with audio samples across different tasks.

In target sound extraction, our findings confirm that there is still a gap in improving performance of current systems, specially for online models. We believe that, in spite that source separation is more popular in the music field, both general audio and music sounds could be addressed by the same model. However, music differs from environmental sounds in the sense that, depending on the genres, some of them present

a well-structured form or have predictable rhythms, they are composed by harmonic and tonal sounds, they do not have background noise and timbre or the tone quality is more controlled than environmental sounds. In such way, hearables and portable systems could be used in real-word scenarios where sounds may contain music, speech or environmental audio and not just a specific group of sounds for a specific application. Also, source tracking applications could be combined with separation models to better model the human hearing system and thus, improve life of people with hearing diseases.

New research trends towards developing larger models or foundation models. In spite that there are some foundation models for speech such as WavLM [7] and general sounds [115], to our known there are not many foundation models for music [94]. As we mentioned in Chapter 2, we believe that it is feasible to build a general foundation model for speech, music and general sounds that could help downstream tasks in each of these domains of the machine listening topic (see Chapter 2). Also, one single model could be built to address different machine listening tasks where the works of this thesis are included. An example of this general model is Fugatto by NVIDIA [164]. The model is capable of perform different tasks such as audio enhancement, audio generation, audio synthesis with text-to-audio or midi-to-audio, and for both music, general sounds and speech signals. The model uses a Composable Audio Representation Transformation (ComposableART) which authors define to generate and manipulate audio based on instructions. This is a new technique that combines Classifier-Free Guidance (CFG) [55], which is used in diffusion models to generate high-quality samples based on specific conditions, with Composable Guidance, which extends CFG by allowing users to combine different elements such as audio attributes, tasks, or models when generating audio. Additionally, the model supports Task Composition, enabling the combination of tasks that were trained separately such as generating speech with a specific audio event in the background, and Temporal Composition, which allows varying the weights for each frame of audio over time.

The tasks addressed in this thesis could be merged in a Fugatto-like model by doing:

- Attribute Composition: The model can combine different audio attributes such as melody, harmony, rhythm, and instrumentation allowing users to generate music based on specific stylistic or thematic instructions.
- Task Composition: a Fugatto-like model could integrate multiple tasks such as symbolic music generation, audio enhancement, and music structure analysis. For example, given a prompt, the model can generate symbolic music, transform it into audio, enhance the audio quality, analyze its structure, and even separate different sources within the audio (e.g., isolating the piano from the orchestral background). This combination of tasks helps streamline workflows and provides comprehensive audio generation and manipulation capabilities.
- Model Composition: integrating different domain-specific models for each task. One model could handle symbolic music generation, like creating piano rolls or MIDI files. Another could focus on audio enhancement, improving the clarity or reducing noise in synthesized or real-world audio recordings. A third model could specialize in music source separation, isolating individual instruments or voices from a complex audio mix. Using Compositional Guidance, these models can work together seamlessly, with Fugatto offering fine-grained control over how each model or task contributes to the final audio

output.

- Temporal Composition: since Fugatto supports varying the influence of different tasks over time by assigning unique weights for each audio frame, this would allow controlling the composition dynamically, such as emphasizing certain instrumental sections or adjusting the level of enhancement applied at specific moments. For example, we could infer dynamics by gradually increase the prominence of a melody line over time or apply more enhancement during louder sections of a piece.

An example of the model's workflow is the following:

- Input: The user provides a prompt, such as "Generate a classical piano piece with a string quartet background, remove noise, and separate the piano sound."
- Symbolic Music Generation: Fugatto first uses a symbolic music generation model to create the piano score and string quartet parts based on the user's input.
- Audio Synthesis: The generated symbolic music is then transformed into audio using an appropriate model for sound synthesis, turning the piano and string quartet parts into actual sound.
- Audio Enhancement: The synthesized audio is enhanced to improve quality, reduce noise, or refine the overall sound.
- Music Structure Analysis: Fugatto analyzes the structure of the music, identifying different sections such as movements, transitions, and key changes.
- Target Sound Extraction: The model isolates different elements in the audio, such as separating the piano from the string quartet parts.
- Output: The final result is clean, enhanced audio, with isolated piano and string quartet tracks, and structural annotations detailing the piece's organization.

This framework would contain all the aspects addressed in this thesis: music structure analysis, symbolic music processing, audio enhancement, and music source separation and combine them in a unique and coherent model which would work as a human would do in the music creation process described in Chapter 1. As for the engineering side, it would provide an efficient, customizable way to integrate different tasks to offer users powerful control over the music creation process.

Appendix A

DIFFUSION POSTERIOR SAMPLING ALGORITHM

ABOUT THIS APPENDIX

In this Appendix, we show the sampling algorithm used in Chapter 5 for vocal resstoration.

A.1 SAMPLING ALGORITHM

As we mentioned in Chapter 5, we designed the sampling algorithm according to EDM [72]. Here is a summary of the variables that we previously introduced in Chapter 5.

- $\widetilde{\sigma}_i$: Increased noise level in diffusion steps.
- $\widetilde{x}_i$: Noisy sample after applying noise in the diffusion process.
- $\hat{x}_i$: Denoised sample from the model $D_{\boldsymbol{\theta}}$ at step i.
- x'_i: Refined sample after evaluating $\mathrm{d}x/\mathrm{d}\sigma$.
- $\widetilde{w}_i$: Weighted term combining the denoised sample and input y for data consistency.
- γ_i: Step size adjustment for the noise scale in the diffusion process.
- x_i: Sample at step i in the diffusion process.
- $\boldsymbol{\epsilon}_i$: Noise sampled from a Gaussian distribution $\mathcal{N}(0, S_{noise}^2\ \mathbf{I})$.
- $D_{\boldsymbol{\theta}}$: Denoising model parameterized by $\boldsymbol{\theta}$.
- S_{noise}: The noise scale used in the algorithm.
- S_{churn}: A parameter controlling the noise level adjustment.

- U: Number of repainting steps in the diffusion process.
- y: The observed data or measurement to which the algorithm applies data consistency.
- $\mathcal{A}(\hat{x}_i)$: Degradation function applied to the denoised sample.
- d_i: Gradient term representing the change of the sample with respect to noise scale.
- ξ: Small parameter used in noise adjustment.
- L: Lipschitz constant used in refinement steps.
- N: Total number of diffusion steps.
- $S_{\text{tmin}}, S_{\text{tmax}}$: Time window for adjusting noise.
- σ_i: Noise scale at step i.

Algorithm 1 Diffusion Sampling Algorithm

procedure StochasticSampler($y, D_{\boldsymbol{\theta}}(x;\sigma)$, $\sigma_{i\in\{0,\dots,N\}}$, $\gamma_{i\in\{0,\dots,N-1\}}$, S_{noise})
  sample $x_0 \sim \mathcal{N}(0, \sigma_0^2\ \mathbf{I})$
  $\gamma_i = \begin{cases} min(S_{\text{churn}}/N, \sqrt{2}-1) & \text{if } t \in [S_{\text{tmin}}, S_{\text{tmax}}] \\ \sqrt{2}-1 & \text{otherwise} \end{cases}$
  for $i \in 0,\dots,N-1$ **do** ▷ diffusion steps
    U $\leftarrow$ Eq. 5.6 ▷ schedule repainting steps
    for $u \in 0,\dots,U-1$ **do** ▷ repainting steps
      $\widetilde{\sigma}_i = \sigma_i + \gamma_i\ \sigma_i$ ▷ increase noise level
      sample $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, S_{noise}^2\ \mathbf{I})$
      $\widetilde{x}_i = x_i + \sqrt{\widetilde{\sigma}_i^2 - \sigma_i^2}\ \boldsymbol{\epsilon}_i$ ▷ New noise
      $\hat{x}_i = D_{\boldsymbol{\theta}}(\widetilde{x}_i;\ \widetilde{\sigma}_i)$ ▷ Denoise
      if rec_guidance **then** ▷ guidance term
        $g(\widetilde{x}_i,\widetilde{\sigma}_i) \leftarrow$ Eq. 5.4
      else if pi_gdm **then**
        $g(\widetilde{x}_i,\widetilde{\sigma}_i) \leftarrow$ Eq. 5.5
      else
        $\hat{x}_i \leftarrow (\hat{x}_i - x_i)/\sigma_i^2$
      end if
      $\hat{x}_i = \hat{x}_i - g$ ▷ get score
      if data_consistency **then** ▷ apply DC
        $\widetilde{w}_i = \hat{x}_i\ \widetilde{\sigma}_i^2 + \widetilde{x}_i$
        $\widetilde{w}_i = \widetilde{w}_i + y + \mathcal{A}(\hat{x}_i)$
        $\hat{x}_i = (\widetilde{w}_i - \widetilde{x}_i)/\sigma_i^2$
      end if
      $d_i = -\widetilde{\sigma}_i \hat{x}_i$ ▷ Evaluate $\mathrm{d}x/\mathrm{d}\sigma$ at $\widetilde{\sigma}_i$
      $x_i' = \widetilde{x}_i + (\sigma_{i+1} - \widetilde{x}_i)\ d_i$
      if $\sigma_{i+1} \neq 0$ or order $= 2$ **then**
        $\hat{x}_i = D_{\boldsymbol{\theta}}(x_i';\ \widetilde{\sigma}_{i+1})$
        $\hat{x}_i = \hat{x}_i - \xi\sqrt{L}\ /\ (\widetilde{\sigma}_{i+1}\nabla_{\widetilde{x}_{i+1}}\ \|y - \mathcal{A}(\hat{x}_i)\|_2^2)$
        if data_consistency **then**
          $\widetilde{w}_i = \hat{x}_i\ \widetilde{\sigma}_i^2 + \widetilde{x}_i$
          $\widetilde{w}_i = \widetilde{w}_i + y + \mathcal{A}(\hat{x}_i)$
          $\hat{x}_i = (\widetilde{w}_i - \widetilde{x}_i)/\sigma_i^2$
        end if
      else
        $x_i = x_i'$
      end if
      if repaint and $u < U$ and i > 0 **then**
        $x_i \leftarrow$ Eq. 5.6
      end if
    end for
  end for
end procedure

Appendix B

CONCLUSIONES

SOBRE ESTE CAPÍTULO

En este capítulo, presentamos las conclusiones y las futuras líneas de investigación que se benefician del trabajo presentado en esta tesis.

B.1 Resumen y Conclusiones

En esta tesis, presentamos diferentes técnicas para abordar una amplia variedad de tareas en los campos del procesamiento de música y audio, desde un software de música simbólica para generación musical hasta la extracción de sonidos en mezclas de audio con sonidos generales. A pesar de que los temas presentados en esta tesis pertenecen a diferentes campos, todos ellos podrían formar parte, en el futuro, de un único modelo multimodal que podría ser utilizado por los profesionales de la música.

Se ha introducido un sistema para la detección de estructura en audio que segmenta las canciones según su estructura. El sistema utiliza una red neuronal convolucional y matrices de autosemejanza calculadas a partir de diferentes características de audio.

Para la generación de música simbólica, se ha desarrollado un paquete de código abierto que contiene varios tokenizadores, métodos de evaluación y un modelo basado en GPT. Hasta donde sabemos, es el primer paquete que incluye todos los pasos necesarios para generar música simbólica. Sin embargo, existe una limitación en la cantidad de modelos que implementamos, que es solo uno. Esperamos agregar más modelos en el futuro y que el software pueda ayudar a la comunidad investigadora a ahorrar tiempo en el procesamiento de datos.

En producción musical, se han realizado trabajos sobre restauración de audio y extracción de sonidos objetivo. Ambos trabajos pueden ser utilizados por productores musicales para restaurar voces grabadas y

para fines de remezcla. En cuanto a la restauración de audio, utilizamos un modelo de difusión previamente desarrollado para la restauración de piano con diferentes técnicas de muestreo posterior ya introducidas en el dominio de las imágenes. Abordamos la extensión de ancho de banda y el *clipping*. A pesar de que los resultados de *declipping* con algunas de las técnicas presentadas superan a los del estado del arte y pueden ser utilizados en estudios de música, la tarea de extensión de ancho de banda podría mejorarse aún más. En cuanto a la extracción de sonidos objetivo, presentamos dos trabajos. El primero mejora la diferencia de tiempo interaural con una nueva función de pérdida. La ventaja de esto es que la mejora de las métricas espaciales no degrada las métricas de la señal. Sin embargo, no está claro cómo las técnicas tradicionales de procesamiento binaural de señales, como la adición de filtros paso-altos y paso-bajos para las diferentes métricas espaciales, pueden cambiar los resultados, ya sea utilizando los filtros solo para la evaluación o también en el cálculo de la pérdida. Esta investigación es parte del trabajo futuro. En nuestro segundo trabajo, mejoramos el rendimiento de los modelos de extracción de sonidos de un solo canal utilizando un modelo de fundación de audio en pistas de referencia y en la mezcla. El modelo de fundación de audio ayuda al sistema general a discriminar clases de sonido, lo que lleva a la mejora de la relación señal-ruido (SNR). Además, el sistema causal se beneficia del modelo de fundación, que, en caso de que lo usemos solo en la pista de referencia, puede mantener el modelo completo causal y ser útil en aplicaciones que requieren procesamiento en tiempo real.

B.2 Trabajo Futuro

Existen muchas formas de mejorar el trabajo que presentamos actualmente en esta tesis y los trabajos actuales. El análisis de la estructura musical no es un tema ampliamente investigado en MIR, por lo tanto, no se han propuesto muchas soluciones después de la publicación de nuestro trabajo. Una posible línea de investigación futura sería mejorar el rendimiento de los sistemas actuales con modelos mejores, como los basados en Transformer, tal como se ha hecho recientemente [123]. Sin embargo, como mencionamos anteriormente, es necesario proponer nuevos conjuntos de datos o anotar los actuales para esta tarea. En un trabajo reciente, proponemos otra investigación para el análisis de la estructura musical en el dominio simbólico para abordar este problema con representaciones gráficas y métodos no supervisados de detección de puntos de cambio [50]. De esa manera, los profesionales de la música podrían aprovechar estos sistemas y utilizarlos para fines educativos o de producción.

En cuanto al campo de la generación de música simbólica, creemos que nuestro software puede ayudar a la comunidad investigadora de música, desde investigadores noveles que buscan desarrollar nuevos sistemas hasta investigadores bien establecidos que pueden utilizar el software para ahorrar tiempo en el preprocesamiento de datos y evaluación. El software puede ampliarse de muchas maneras, como implementando más modelos de última generación para la generación de música o incluyendo más formatos de música simbólica.

Nuestro trabajo en restauración de audio puede inspirar nuevos trabajos para explorar técnicas de muestreo de difusión específicamente para tareas de audio. Demostramos cómo tales técnicas, inicialmente desarrolladas y probadas con imágenes, funcionan en el dominio del audio. Sin embargo, el rendimiento de otras técnicas de muestreo posterior de difusión para diferentes tareas de procesamiento de audio sigue

siendo desconocido. También creemos que se pueden probar técnicas de muestreo más recientes, además de las desarrolladas para imágenes, con muestras de audio a través de diferentes tareas.

En la extracción de sonidos objetivo, nuestros resultados confirman que todavía existe una brecha en la mejora del rendimiento de los sistemas actuales, especialmente para modelos en tiempo real. Creemos que, a pesar de que la separación de fuentes es más popular en el campo de la música, tanto los sonidos generales como los musicales podrían ser abordados por el mismo modelo. Sin embargo, la música difiere de los sonidos ambientales en el sentido de que, dependiendo de los géneros, algunos presentan una forma bien estructurada o tienen ritmos predecibles, están compuestos por sonidos armónicos y tonales, no tienen ruido de fondo y el timbre o la calidad tonal es más controlada que en los sonidos ambientales. De esta manera, los sistemas de audífonos podrían ser utilizados en escenarios del mundo real donde los sonidos puedan contener música, voz o audio ambiental y no solo un grupo específico de sonidos para una aplicación específica. Además, las aplicaciones de detección de fuentes podrían combinarse con modelos de separación para modelar mejor el sistema auditivo humano y, por lo tanto, mejorar la vida de las personas con discapacidades auditivas.

Las nuevas tendencias de investigación se dirigen al desarrollo de modelos más grandes o modelos de fundación. A pesar de que existen algunos modelos de fundación para el habla, como WavLM [7] y sonidos generales [115], hasta donde sabemos, no existen muchos modelos de fundación para la música [94]. Como mencionamos en el Capítulo 2, creemos que es factible construir un modelo de fundación general para el habla, la música y los sonidos generales que pueda ayudar en tareas posteriores en cada uno de estos dominios del tema de la escucha de máquinas (ver Capítulo 2). Además, se podría construir un solo modelo para abordar diferentes tareas donde se incluyen los trabajos de esta tesis. Un ejemplo de este modelo general es Fugatto de NVIDIA [164]. El modelo es capaz de realizar diferentes tareas como mejora de audio, generación de audio, síntesis de audio con texto a audio o MIDI a audio, y para música, sonidos generales y señales de voz. El modelo utiliza una Transformación Composicional de Representación de Audio (ComposableART) que los autores definen para generar y manipular audio según instrucciones. Esta es una nueva técnica que combina la Guía Libre de Clasificadores (CFG) [55], que se utiliza en modelos de difusión para generar muestras de alta calidad basadas en condiciones específicas, con la Guía Composicional, que extiende la CFG permitiendo a los usuarios combinar diferentes elementos como atributos de audio, tareas o modelos al generar audio. Además, el modelo admite la Composición de Tareas, lo que permite la combinación de tareas que se entrenaron por separado, como generar habla con un evento de audio específico en el fondo, y la Composición Temporal, que permite variar los pesos del modelo a lo largo del tiempo.

Las tareas abordadas en esta tesis podrían fusionarse en un modelo similar a Fugatto haciendo lo siguiente:

- Composición de Atributos: El modelo puede combinar diferentes atributos de audio como melodía, armonía, ritmo e instrumentación, permitiendo a los usuarios generar música basada en instrucciones estilísticas o temáticas específicas.
- Composición de Tareas: Un modelo similar a Fugatto podría integrar múltiples tareas como generación de música simbólica, mejora de audio y análisis de estructura musical. Por ejemplo, dada una descripción en texto, el modelo puede generar música simbólica, transformarla en audio, mejorar

la calidad del audio, analizar su estructura e incluso separar diferentes fuentes dentro del audio (por ejemplo, aislando el piano del fondo orquestal). Esta combinación de tareas ayuda a agilizar los flujos de trabajo y proporciona capacidades integrales de generación y manipulación de audio.

- Composición de Modelos: Integrando diferentes modelos específicos de dominio para cada tarea. Un modelo podría encargarse de la generación de música simbólica, como la creación de partituras de piano o archivos MIDI. Otro podría centrarse en la mejora de audio, mejorando la claridad o reduciendo el ruido en grabaciones de audio sintetizado o reales. Un tercer modelo podría especializarse en la separación de fuentes musicales, aislando instrumentos o voces individuales de una mezcla de audio compleja. Usando la Guía Composicional, estos modelos pueden trabajar juntos de manera fluida, con Fugatto ofreciendo un control detallado sobre cómo cada modelo o tarea contribuye al resultado final de audio.
- Composición Temporal: Dado que Fugatto admite variar la influencia de diferentes tareas a lo largo del tiempo asignando pesos únicos para cada cuadro de audio, esto permitiría controlar dinámicamente la composición, como enfatizar ciertas secciones instrumentales o ajustar el nivel de mejora aplicado en momentos específicos. Por ejemplo, podríamos inferir dinámicas aumentando gradualmente la prominencia de una línea melódica a lo largo del tiempo o aplicar mejoras.

Un ejemplo del flujo de trabajo del modelo es el siguiente:

- Entrada: El usuario proporciona una entrada de texto, como "Generar una pieza de piano clásico con un cuarteto de cuerdas de fondo, eliminar ruido y separar el sonido del piano."
- Generación de Música Simbólica: Fugatto primero utiliza un modelo de generación de música simbólica para crear la partitura de piano y las partes del cuarteto de cuerdas según la entrada del usuario.
- Síntesis de Audio: La música simbólica generada se transforma en audio utilizando un modelo apropiado para la síntesis de sonido, convirtiendo las partes de piano y cuarteto de cuerdas en sonido real.
- Mejora de Audio: El audio sintetizado se mejora para mejorar la calidad, reducir el ruido o refinar el sonido general.
- Análisis de Estructura Musical: Fugatto analiza la estructura de la música, identificando diferentes secciones como movimientos, transiciones y cambios de clave.
- Extracción de Sonidos Objetivo: El modelo aísla diferentes elementos en el audio, como separar el piano de las partes del cuarteto de cuerdas.
- Salida: El resultado final es un audio limpio y mejorado, con pistas aisladas de piano y cuarteto de cuerdas, y anotaciones estructurales detallando la organización de la pieza.

Este marco contendría todos los aspectos tratados en esta tesis: análisis de la estructura musical, procesamiento de música simbólica, mejora de audio y separación de fuentes musicales, combinándolos en un modelo único y coherente que funcionaría como lo haría un ser humano en el proceso de creación musical descrito en el Capítulo 1. En cuanto a la parte relativa a la ingeniería y desarrollo, proporcionaría una forma eficiente y personalizable de integrar diferentes tareas para ofrecer a los usuarios un control potente sobre el proceso de creación musical.

Bibliography

[1] V Ralph Algazi, Richard O Duda, Dennis M Thompson, and Carlos Avendano. The CIPIC HRTF database. In *Proc. WASPAA*, pages 99–102, 2001.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, 2015.

[3] Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xlstm: Extended long short-term memory. *CoRR*, abs/2405.04517, 2024.

[4] Jens Blauert. Spatial hearing: The psychophysics of human sound localization. *Spatial hearing*, 1997.

[5] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *International Conference on Machine Learning (ICML)*. icml.cc / Omnipress, 2012.

[6] Chris Cannam, Emmanouil Benetos, Matthias Mauch, Matthew EP Davies, Simon Dixon, Christian Landone, Katy Noland, and Dan Stowell. Mirex 2015: Vamp plugins from the centre for digital music. *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2015.

[7] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE J. Sel. Top. Signal Process.*, 16(6):1505–1518, 2022.

[8] Tsung-Ping Chen and Li Su. Functional harmony recognition of symbolic music data with multi-task recurrent neural networks. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 90–97, 2018.

[9] Hyungjin Chung, Jeongsol Kim, Michael T. McCann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *CoRR*, abs/2209.14687, 2022.

[10] Alice Cohen-Hadria and Geoffroy Peeters. Music structure boundaries estimation using multiple self-similarity matrices as input depth of convolutional neural networks. In *International Conference Semantic Audio (AES)*, 2017.

[11] Michael Scott Cuthbert and Christopher Ariza. Music21: A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pages 637–642, 2010.

[12] Marc Delcroix, Keisuke Kinoshita, Tsubasa Ochiai, Katerina Zmolikova, Hiroshi Sato, and Tomohiro Nakatani. Listen only to me! How well can target speech extraction handle false alarms? In *Proceedings Interspeech*, pages 216–220, 2022.

[13] Marc Delcroix, Jorge Bennasar Vázquez, Tsubasa Ochiai, Keisuke Kinoshita, Yasunori Ohishi, and Shoko Araki. Soundbeam: Target sound extraction conditioned on sound-class labels and enrollment clues for increased performance and continuous learning. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 31:121–136, 2023.

[14] Irène Deliège and Geraint A. Wiggins. *Musical Creativity: Multidisciplinary Research in Theory and Practice*. Psychology Press, 1st edition, 2006.

[15] Mathias Dietz, Stephan D Ewert, and Volker Hohmann. Auditory model based direction estimation of concurrent speakers from binaural signals. *Speech Communication*, 53(5):592–605, 2011.

[16] Hao-Wen Dong, Ke Chen, Julian J. McAuley, and Taylor Berg-Kirkpatrick. Muspy: A toolkit for symbolic music generation. In *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, pages 101–108, 2020.

[17] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Conference on Artificial Intelligence, (AAAI)*, pages 34–41. AAAI Press, 2018.

[18] Hao-Wen Dong, Wen-Yi Hsiao, and Yi-Hsuan Yang. Pypianoroll: Open source python package for handling multitrack pianoroll. *19th International Society for Music Information Retrieval Conference, (ISMIR) late breaking and demo papers*, 2018.

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[20] Zhiyan Duan, Haotian Fang, Bo Li, Khe Chai Sim, and Ye Wang. The NUS sung and spoken lyrics corpus: A quantitative comparison of singing and speech. In *APSIPA*, pages 1–9, 2013.

[21] Jeffrey Ens and Philippe Pasquier. MMM : Exploring conditional multi-track music generation with the transformer. *CoRR*, abs/2008.06048, 2020.

[22] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra. FSD50K: an open dataset of human-labeled sound events. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852, 2022.

[23] Jonathan Foote. Visualizing music and audio using self-similarity. In *Proceedings of the 7th ACM International Conference on Multimedia*, pages 77–80. ACM, 1999.

[24] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *2000 IEEE International Conference on Multimedia and Expo, ICME 2000, New York, NY, USA, July 30 - August 2, 2000*, page 452. IEEE Computer Society, 2000.

[25] Nathan Fradet, Jean-Pierre Briot, Fabien Chhel, Amal El Fallah-Seghrouchni, and Nicolas Gutowski. Miditok: A python package for midi file tokenization. In *22nd International Society for Music Information Retrieval Conference, (ISMIR) late breaking and demo papers*, 2021.

[26] Kunihiko Fukushima. Neocognitron: A new hierarchical neural network architecture for pattern recognition. In *Proceedings of the 1988 IAPR International Workshop on Syntactic and Structural Pattern Recognition*, pages 250–255. IAPR, 1988.

[27] Nicolas Furnon. Noise files for the disco dataset. (2020), 2020.

[28] Nicolas Furnon, Romain Serizel, Slim Essid, and Irina Illina. Dnn-based mask estimation for distributed speech enhancement in spatially unconstrained microphone arrays. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 29:2310–2323, 2021.

[29] Lillian Hemingway Gallay. *Understanding and treating creative block in professional artists.* Alliant International University, 2013.

[30] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Proc. ICASSP*, pages 776–780, 2017.

[31] Beat Gfeller, Dominik Roblek, and Marco Tagliasacchi. One-shot conditional audio filtering of arbitrary sounds. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 501–505, 2021.

[32] Yuan Gong, Yu-An Chung, and James Glass. AST: Audio Spectrogram Transformer. In *Proceedings Interspeech*, pages 571–575, 2021.

[33] Ian Goodfellow. Deep learning, 2016.

[34] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Advances in Neural Information Processing Systems (NeurIPS), 2014.

[35] Masataka Goto. A chorus-section detecting method for musical audio signals. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 437–440. IEEE, 2003.

[36] Thomas Grill and Jan Schlüter. Music boundary detection using neural networks on combined features and two-level annotations. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, pages 531–537, 2015.

[37] Thomas Grill and Jan Schlüter. Music boundary detection using neural networks on spectrograms and self-similarity lag matrices. In *23rd European Signal Processing Conference, EUSIPCO 2015, Nice, France, August 31 - September 4, 2015*, pages 1296–1300. IEEE, 2015.

[38] Thomas Grill and Jan Schlüter. Structural segmentation with convolutional neural networks mirex submission. *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, page 3, 2015.

[39] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023.

[40] Rongzhi Gu and Yi Luo. Rezero: Region-customizable sound extraction. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 32:2576–2589, 2024.

[41] Cong Han, Yi Luo, and Nima Mesgarani. Real-time binaural speech separation with preserved spatial cues. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6404–6408, 2020.

[42] Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition. *CoRR*, abs/1412.5567, 2014.

[43] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International Conference on Learning Representations*, 2019.

[44] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15979–15988, 2022.

[45] Carlos Hernandez-Olivan, Jose R. Beltran, and David Diaz-Guerra. Music boundary detection using convolutional neural networks: A comparative analysis of combined input features. *International Journal of Interactive Multimedia and Artificial Intelligence*, 7(2):78–88, 2021.

[46] Carlos Hernandez-Olivan and José Ramón Beltrán. Music composition with deep learning: A review, 2021.

[47] Carlos Hernandez-Olivan and José Ramón Beltrán. Musicaiz: A python library for symbolic music generation, analysis and visualization. *SoftwareX*, 22:101365, 2023.

[48] Carlos Hernandez-Olivan, Marc Delcroix, Tsubasa Ochiai, Daisuke Niizumi, Naohiro Tawara, Tomohiro Nakatani, and Shoko Araki. Soundbeam meets m2d: Target sound extraction with audio foundation model. *submitted to IEEE International Conference on Acoustics, Speech, and Signal Processing. ICASSP*, 2024.

[49] Carlos Hernandez-Olivan, Marc Delcroix, Tsubasa Ochiai, Naohiro Tawara, Tomohiro Nakatani, and Shoko Araki. Interaural time difference loss for binaural target sound extraction. *IEEE International Workshop on Acoustic Signal Enhancement. IWAENC*, 2024.

[50] Carlos Hernandez-Olivan, Sonia Rubio Llamas, and José Ramón Beltrán. Symbolic music structure analysis with graph representations and changepoint detection methods. *CoRR*, abs/2303.13881, 2023.

[51] Carlos Hernandez-Olivan, Jorge Abadias Puyuelo, and Jose R. Beltran. Subjective evaluation of deep learning models for symbolic music composition. *Workshop on Generative AI and CHI*, abs/2203.14641, 2022.

[52] Carlos Hernandez-Olivan, Koichi Saito, Naoki Murata, Chieh-Hsin Lai, Marco A. Martínez-Ramírez, Wei-Hsiang Liao, and Yuki Mitsufuji. Vrdmg: Vocal restoration via diffusion posterior sampling with multiple guidance. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 596–600, 2024.

[53] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[54] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[55] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications.

[56] JonathaNn Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. In *Advances in Neural Inf. Proc. Systems*, volume 35, pages 8633–8646, 2022.

[57] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.

[58] Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs. In *Conference on Artificial Intelligence, (AAAI)*, pages 178–186. AAAI Press, 2021.

[59] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[60] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations (ICLR)*, 2019.

[61] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations (ICLR)*, 2019.

[62] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *MM '20: The 28th ACM International Conference on Multimedia*, pages 1180–1188. ACM, 2020.

[63] Zili Huang, Shinji Watanabe, Shu-Wen Yang, Paola García, and Sanjeev Khudanpur. Investigating self-supervised learning for speech enhancement and separation. In *Proc. ICASSP*, pages 6837–6841, 2022.

[64] David Huron. Music information processing using the humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal*, 26(2):11–26, 2002.

[65] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005.

[66] Johannes Imort, Giorgio Fabbro, Marco A. Martínez Ramírez, Stefan Uhlich, Yuichiro Koyama, and Yuki Mitsufuji. Distortion Audio Effects: Learning How to Recover the Clean Signal. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[67] International Telecommunication Union. Method for the subjective assessment of intermediate quality level of audio systems. ITU-R Recommendation ITU-R BS.1534-3, 2015.

[68] IoSR-Surrey. Iosr-surrey/realroombrirs: Binaural impulse responses captured in real rooms. https://github.com/IoSR-Surrey/RealRoomBRIRs, 2016.

[69] IoSR-Surrey. Simulated room impulse responses. https://iosr.uk/software/index.php, 2023. Accessed: April 24, 2024.

[70] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *International Conference on Machine Learning*, pages 3060–3070. PMLR, 2019.

[71] Florian Kaiser and Geoffroy Peeters. Multiple hypotheses at multiple scales for audio novelty computation within music. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 231–235. IEEE, 2013.

[72] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[73] Emmanouil Karystinaios and Gerhard Widmer. Roman numeral analysis with graph neural networks: Onset-wise predictions from note-wise features, 2023.

[74] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms. In *Proceedings Interspeech*, pages 2350–2354. ISCA, 2019.

[75] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ODE trajectory of diffusion. In *International Conference on Learning Representations (ICLR)*, 2024.

[76] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[77] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[78] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D. Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[79] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations (ICLR)*, 2021.

[80] Jeffrey L. Krolik, M. Joy, Subbarayan Pasupathy, and Moshe Eizenman. A comparative study of the LMS adaptive filter versus generalized correlation method for time delay estimation. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 652–655, 1984.

[81] Volodymyr Kuleshov, S. Zayd Enam, and Stefano Ermon. Audio super resolution using neural networks. *International Conference on Learning Representations (ICLR)*, 2017.

[82] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[83] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[84] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey. SDR–half-baked or well done? In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 626–630, 2019.

[85] Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nat.*, 521(7553):436–444, 2015.

[86] Fred Lerdahl and Ray Jackendoff. An overview of hierarchical structure in music. *Music Perception: An Interdisciplinary Journal*, 1(2):229–252, 1983.

[87] Mark Levy and Mark B. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Speech and Audio Processing*, 16(2):318–326, 2008.

[88] Chenda Li, Yao Qian, Zhuo Chen, Dongmei Wang, Takuya Yoshioka, Shujie Liu, Yanmin Qian, and Michael Zeng. Target sound extraction with variable cross-modality clues. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

[89] Beth Logan and Stephen M. Chu. Music summarization using key phrases. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 749–752. IEEE, 2000.

[90] Lie Lu, Muyuan Wang, and HongJiang Zhang. Repeating pattern discovery and structure analysis from acoustic music data. In *Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval*, pages 275–282. ACM, 2004.

[91] Andreas Lugmayr, Martin Danelljan, Andrés Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11451–11461. IEEE, 2022.

[92] Yi Luo and Nima Mesgarani. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 27(8):1256–1266, 2019.

[93] Hao Ma, Zhiyuan Peng, Xu Li, Mingjie Shao, Xixin Wu, and Ju Liu. CLAPSep: Leveraging contrastive pre-trained model for multi-modal query-conditioned target sound extraction, 2024.

[94] Yinghao Ma, Anders Øland, Anton Ragni, Bleiz MacSen Del Sette, Charalampos Saitis, Chris Donahue, Chenghua Lin, Christos Plachouras, Emmanouil Benetos, Elona Shatri, et al. Foundation models for music: A survey. *arXiv preprint arXiv:2408.14340*, 2024.

[95] Marco A Martínez Ramírez. *Deep learning for audio effects modeling.* PhD thesis, Queen Mary University of London, 2021.

[96] Matthias Mauch, Katy C. Noland, and Simon Dixon. Using musical structure to enhance automatic chord transcription. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 231–236, 2009.

[97] Matthew C. McCallum. Unsupervised learning of deep features for music segmentation. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 346–350. IEEE, 2019.

[98] Brian McFee and Juan Pablo Bello. Structured training for large-vocabulary chord recognition. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, pages 188–194, 2017.

[99] Brian McFee and DP Ellis. Dp1, mp1, mp2 entries for mirex 2013 structural segmentation and beat tracking. *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2013.

[100] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, pages 18–25, 2015.

[101] Cory McKay, Julie Cumming, and Ichiro Fujinaga. JSYMBOLIC 2.2: Extracting features from symbolic music for use in musicological and MIR research. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 348–354, 2018.

[102] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. A multi-device dataset for urban acoustic scene classification. In *Proc. DCASE*, pages 9–13, 2018.

[103] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. Self-supervised speech representation learning: A review. *IEEE J. Sel. Top. Signal Process.*, 16(6):1179–1210, 2022.

[104] Eloi Moliner, Jaakko Lehtinen, and Vesa Välimäki. Solving audio inverse problems with a diffusion model. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1–5, 2023.

[105] Eloi Moliner and Vesa Välimäki. BEHM-GAN: bandwidth extension of historical music using generative adversarial networks. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 31:943–956, 2023.

[106] David R Moore. Anatomy and physiology of binaural hearing. *Audiology*, 30(3):125–134, 1991.

[107] Naoki Murata, Koichi Saito, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Yuki Mitsufuji, and Stefano Ermon. GibbsDDRM: A partially collapsed gibbs sampler for solving blind inverse problems with denoising diffusion restoration. In *International Conference on Machine Learning (ICML)*, 2023.

[108] Vivek Sivaraman Narayanaswamy, Jayaraman J. Thiagarajan, and Andreas Spanias. On the Design of Deep Priors for Unsupervised Audio Restoration. In *Proceedings Interspeech*, pages 2167–2171, 2021.

[109] Gerhard Nierhaus. *Algorithmic Composition*. Springer Vienna, 1 edition, 2009.

[110] Oriol Nieto. Mirex: Msaf v0. 1.0 submission. 2016.

[111] Oriol Nieto and Juan P Bello. Mirex 2014 entry: 2d fourier magnitude coefficients. *Proceedings of the Music Information Retrieval Evaluation eXchange (MIREX)*, 2014.

[112] Oriol Nieto, Morwaread M. Farbood, Tristan Jehan, and Juan Pablo Bello. Perceptual analysis of the f-measure to evaluate section boundaries in music. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 265–270, 2014.

[113] Oriol Nieto, Gautham J. Mysore, Cheng-i Wang, Jordan B. L. Smith, Jan Schlüter, Thomas Grill, and Brian McFee. Audio-based music structure analysis: Current trends, open challenges, and applications. *Transactions of the International Society for Music Information Retrieval*, 3(1):246–263, 2020.

[114] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino. BYOL for Audio: Exploring Pre-trained General-purpose Audio Representations. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 31:137–151, 2023.

[115] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino. Masked modeling duo: Towards a universal audio pre-training framework. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, 32:2391–2406, 2024.

[116] Tsubasa Ochiai, Marc Delcroix, Yuma Koizumi, Hiroaki Ito, Keisuke Kinoshita, and Shoko Araki. Listen to What You Want: Neural Network-Based Universal Sound Selector. In *Proceedings Interspeech*, pages 1441–1445, 2020.

[117] Yuki Okamoto, Shota Horiguchi, Masaaki Yamamoto, Keisuke Imoto, and Yohei Kawaguchi. Environmental sound extraction using onomatopoeic words. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 221–225, 2022.

[118] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[119] Jouni Paulus and Anssi Klapuri. Music structure analysis by finding repeated parts. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, page 59–68, New York, NY, USA, 2006. Association for Computing Machinery.

[120] Jouni Paulus and Anssi Klapuri. Music structure analysis with a probabilistic fitness function in MIREX2009. In *Proceedings of the Fifth Annual Music Information Retrieval Evaluation eXchange*, October 2009. Extended abstract.

[121] Jouni Paulus, Meinard Müller, and Anssi Klapuri. State of the art report: Audio-based music structure analysis. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pages 625–636, 2010.

[122] Marcus T Pearce and Geraint A Wiggins. Expectation in melody: The influence of context and learning. *Music Perception*, 23(5):377–405, 2006.

[123] Geoffroy Peeters. Self-similarity-based and novelty-based loss for music structure analysis. In *Proceedings of the 24th International Society for Music Information Retrieval Conference (ISMIR)*, pages 749–756, 2023.

[124] Junyi Peng, Marc Delcroix, Tsubasa Ochiai, Oldrich Plchot, Shoko Araki, and Jan Cernocký. Target speech extraction with pre-trained self-supervised learning models. In *Proc. ICASSP*, 2024.

[125] Junyi Peng, Oldrich Plchot, Themos Stafylakis, Ladislav Mosner, Lukás Burget, and Jan Cernocký. An attention-based backend allowing efficient fine-tuning of transformer models for speaker verification. In *Proc. SLT*, pages 555–562, 2022.

[126] Karol J. Piczak. Environmental sound classification with convolutional neural networks. In *25th IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6, 2015.

[127] Karol J. Piczak. ESC: dataset for environmental sound classification. In *Proceedings Annual ACM Conference on Multimedia*, pages 1015–1018. ACM, 2015.

[128] W Piston. *Harmony*. WW Norton y Company, Inc, 1941.

[129] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning (ICML)*, volume 202, pages 28492–28518, 2023.

[130] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[131] Alec Radford, Karthik Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[132] Colin Raffel. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. PhD thesis, Columbia University, 2016.

[133] Colin Raffel and Daniel PW Ellis. Intuitive analysis, creation and manipulation of midi data with pretty_midi. In *15th International Society for Music Information Retrieval Conference, (ISMIR) late breaking and demo papers*, pages 84–93, 2014.

[134] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. Mir_eval: A transparent implementation of common MIR metrics. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 367–372, 2014.

[135] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. The MUSDB18 corpus for music separation, December 2017.

[136] Pedro Ramoneda, Nazif Can Tamer, Vsevolod Eremenko, Xavier Serra, and Marius Miron. Score difficulty analysis for piano performance education based on fingering. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 201–205. IEEE, 2022.

[137] Christopher Raphael. Automatic segmentation of acoustic musical signals using hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):360–370, 1999.

[138] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1530–1538. JMLR.org, 2015.

[139] Adam Roberts, Jesse H. Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 4361–4370, 2018.

[140] Carles Roig, Lorenzo J Tardón, Isabel Barbancho, and Ana M Barbancho. Automatic melody composition based on a probabilistic model of music style and harmonic rules. *Knowledge-Based Systems*, 71:419–434, 2014.

[141] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015.

[142] Eleanor Row and György Fazekas. Composers' evaluations of an ai music tool: Insights for human-centred design. *arXiv preprint arXiv:2412.10968*, 2024.

[143] Koichi Saito, Dongjun Kim, Takashi Shibuya, Chieh-Hsin Lai, Zhi Zhong, Yuhta Takida, and Yuki Mitsufuji. Soundctm: Uniting score-based and consistency models for text-to-sound generation. *CoRR*, abs/2405.18503, 2024.

[144] Koichi Saito, Naoki Murata, Toshimitsu Uesaka, Chieh-Hsin Lai, Yuhta Takida, Takao Fukui, and Yuki Mitsufuji. Unsupervised vocal dereverberation with diffusion-based generative models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1–5, 2023.

[145] Justin Salamon, Duncan MacConnell, Mark Cartwright, Peter Li, and Juan Pablo Bello. Scaper: A library for soundscape synthesis and augmentation. In *Proc. WASPAA*, pages 344–348, 2017.

[146] Gabriel Sargent, Frédéric Bimbot, and Emmanuel Vincent. A regularity-constrained viterbi algorithm and its application to the structural segmentation of songs. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 483–488, 2011.

[147] Gabriel Sargent, Stanislaw Andrzej Raczynski, Frédéric Bimbot, Emmanuel Vincent, and Shigeki Sagayama. A music structure inference algorithm based on symbolic data analysis. MIREX - ISMIR 2011, October 2011. Poster.

[148] D. Satongar, Y. Lam, and C. Pike. The Salford BBC spatially-sampled binaural room impulse response dataset, 2015.

[149] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[150] Jan Schlüter and Sebastian Böck. Improved musical onset detection with convolutional neural networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6979–6983. IEEE, 2014.

[151] Jan Schlüter, Karen Ullrich, and Thomas Grill. Structural segmentation with convolutional neural networks mirex submission. *Tenth running of the Music Information Retrieval Evaluation eXchange (MIREX 2014)*, 2014.

[152] Joan Serrà, Meinard Müller, Peter Grosche, and Josep Lluís Arcos. Unsupervised detection of music boundaries by time series structure features. In *Conference on Artificial Intelligence (AAAI)*. AAAI Press, 2012.

[153] Bidisha Sharma, Xiaoxue Gao, Karthika Vijayan, Xiaohai Tian, and Haizhou Li. NHSS: A speech and singing parallel database. *Speech Commun.*, 133:9–22, 2021.

[154] Jordan Bennett Louis Smith, John Ashley Burgoyne, Ichiro Fujinaga, David De Roure, and J. Stephen Downie. Design and creation of a large-scale database of structural annotations. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 555–560, 2011.

[155] Jascha Sohl-Dickstein, Eric Weiss, Narayanan Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, pages 2256–2265. PMLR, 2015.

[156] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, pages 2256–2265. PMLR, 2015.

[157] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations (ICLR)*, 2023.

[158] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 32211–32252. PMLR, 2023.

[159] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[160] Vikas Tokala, Eric Grinstein, Mike Brookes, Simon Doclo, Jesper Jensen, and Patrick A. Naylor. Binaural speech enhancement using deep complex convolutional transformer networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 681–685. IEEE, 2024.

[161] Christopher J. Tralie and Brian McFee. Enhanced hierarchical music structure annotations via feature level similarity fusion. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 201–205. IEEE, 2019.

[162] Douglas Turnbull, Gert R. G. Lanckriet, Elias Pampalk, and Masataka Goto. A supervised approach for detecting boundaries in music using difference features and boosting. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 51–54, 2007.

[163] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 417–422, 2014.

[164] Rafael Valle, Rohan Badlani, Zhifeng Kong, Sang gil Lee, Arushi Goel, Sungwon Kim, Joao Felipe Santos, Shuqi Dai, Siddharth Gururani, Aya AIJa'fari, Alex Liu, Kevin Shih, Wei Ping, Huck Yang,

and Bryan Catanzaro. Fugatto 1 - foundational generative audio transformer opus 1. *Under review at ICLR*, 2024. A versatile audio synthesis and transformation model capable of following free-form text instructions with optional audio inputs.

[165] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop, SSW 2016, Sunnyvale, CA, USA, September 13-15, 2016*, page 125. ISCA, 2016.

[166] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[167] Gino Angelo Velasco, Nicki Holighaus, Monika Doerfler, and Thomas Grill. Constructing an invertible constant-q transform with nonstationary gabor frames. In *Proc. DAFx*, 09 2011.

[168] Bandhav Veluri, Justin Chan, Malek Itani, Tuochao Chen, Takuya Yoshioka, and Shyamnath Gollakota. Real-time target sound extraction. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

[169] Bandhav Veluri, Malek Itani, Justin Chan, Takuya Yoshioka, and Shyamnath Gollakota. Semantic hearing: Programming acoustic scenes with binaural hearables. In *Proceedings Symposium on User Interface Software and Technology (UIST)*, pages 89:1–89:15. ACM, 2023.

[170] Christof Weiß, Frank Zalkow, Vlora Arifi-Müller, Meinard Müller, Hendrik Vincent Koops, Anja Volk, and Harald G. Grohganz. Schubert winterreise dataset: A multimodal scenario for music analysis. *ACM Journal on Computing and Cultural Heritage*, 14(2):25:1–25:18, 2021.

[171] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-Scale Contrastive Language-Audio Pretraining with Feature Fusion and Keyword-to-Caption Augmentation. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

[172] Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music. *Neural Computing and Applications*, 32(9):4773–4784, 2020.

[173] Botao Yu, Peiling Lu, Rui Wang, Wei Hu, Xu Tan, Wei Ye, Shikun Zhang, Tao Qin, and Tie-Yan Liu. Museformer: Transformer with fine- and coarse-grained attention for music generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[174] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. *CoRR*, abs/2303.09833, 2023.

[175] Chenshuang Zhang, Chaoning Zhang, Sheng Zheng, Mengchun Zhang, Maryam Qamar, Sung-Ho Bae, and In So Kweon. A survey on audio diffusion models: Text to speech synthesis and enhancement in generative AI. *CoRR*, abs/2303.13336, 2023.

[176] Tong Zhang and C.-C. Jay Kuo. Heuristic approach for generic audio data segmentation and annotation. In *Proceedings of the 7th ACM International Conference on Multimedia*, pages 67–76. ACM, 1999.

[177] Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh H. McDermott, and Antonio Torralba. The sound of pixels. In *Proc. Computer Vision - ECCV*, volume 11205, pages 587–604. Springer, 2018.

[178] Junqi Zhao, Xubo Liu, Jinzheng Zhao, Yi Yuan, Qiuqiang Kong, Mark D. Plumbley, and Wenwu Wang. Universal sound separation with self-supervised audio masked autoencoder, 2024.

[179] Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Tsubasa Ochiai, Tomohiro Nakatani, Lukáš Burget, and Jan Černockỳ. Speakerbeam: Speaker aware neural network for target speaker extraction in speech mixtures. *IEEE Journal of Selected Topics in Signal Processing*, 13(4):800–814, 2019.

[180] Katerina Zmolíková, Marc Delcroix, Tsubasa Ochiai, Keisuke Kinoshita, Jan Cernocký, and Dong Yu. Neural target speech extraction: An overview. *IEEE Signal Process. Mag.*, 40(3):8–29, 2023.

[181] Yong Zou, Marco Thiel, M. Carmen Romano, and Jürgen Kurths. Analytical description of recurrence plots of dynamical systems with nontrivial recurrences. *International Journal of Bifurcation and Chaos*, 17(12):4273–4283, 2007.

[182] Pavel Záviška, Pavel Rajmic, Alexey Ozerov, and Lucas Rencker. A survey and an extensive evaluation of popular audio declipping methods. *IEEE Journal of Sel. Topics in Signal Proc.*, 15(1):5–24, 2021.