DAFx 2024 Workshop • 3 September 2024

AI for Multitrack Music Mixing

Marco A. Martínez-Ramírez[1]

Soumya Sai Vanka[2]

[1] Sony AI, Tokyo, Japan

[2] Centre for Digital Music, Queen Mary University of London

Previous Editions of the Workshop

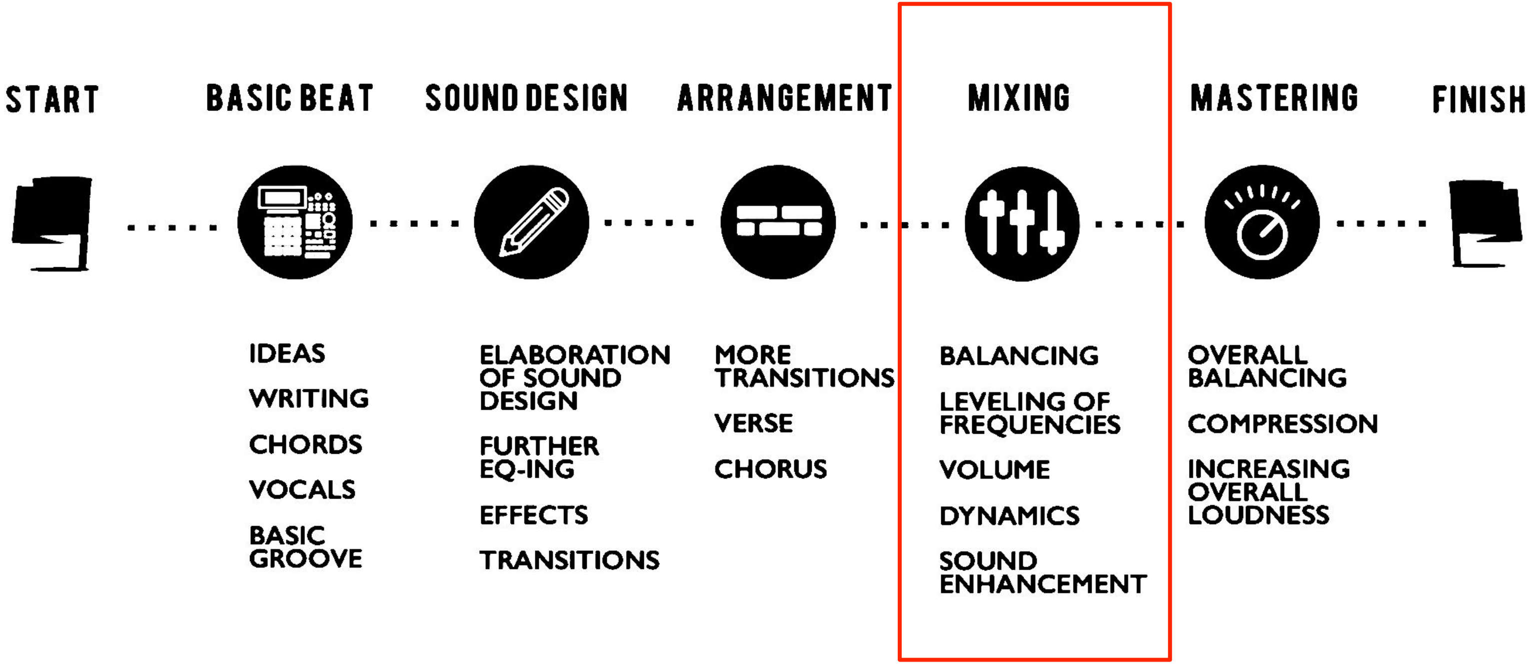
Outline

Part 1	**Introduction**	*3 min*	1 hr
Part 2	**Background**	*15 min*	
Part 3	**Work-so-Far**	*25 min*	
Part 4	**Evaluation**	*7 min*	
	Quick Pause	*10 min*	
Part 5	**Implementation**	*45 min*	1 hr
Part 6	**Future Directions**	*5 min*	
	Q&A	*10 min*	

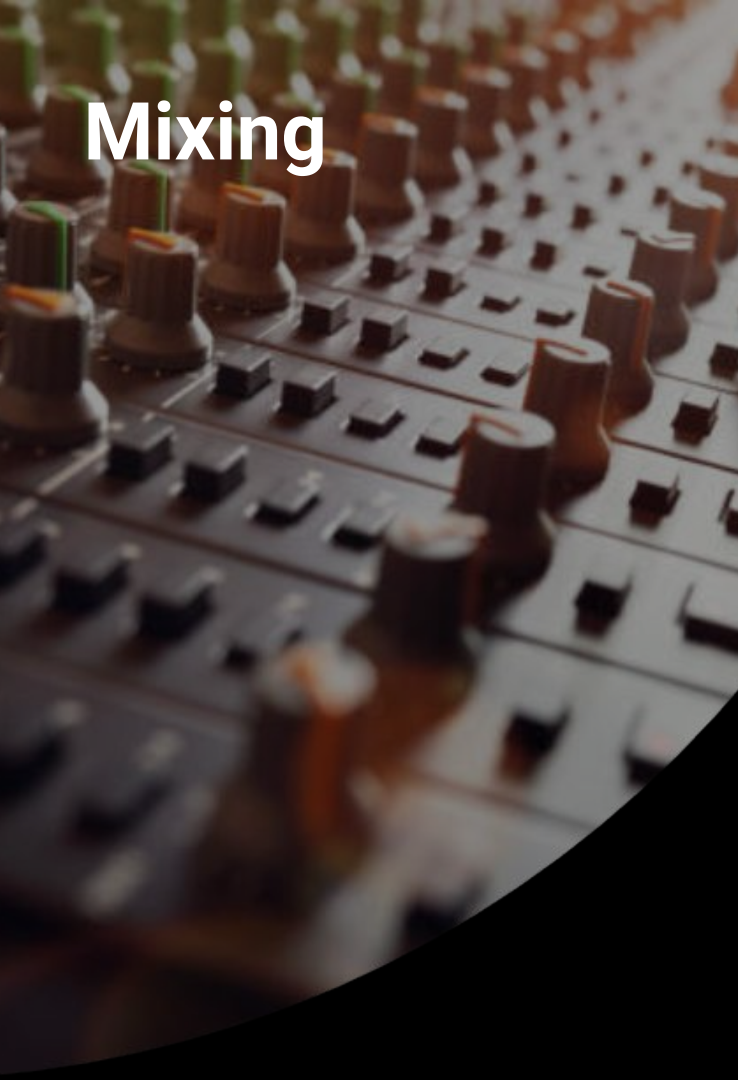
Introduction

Part 1

Audio Production



Mixing



Audio mixing is the process of blending multitrack recordings

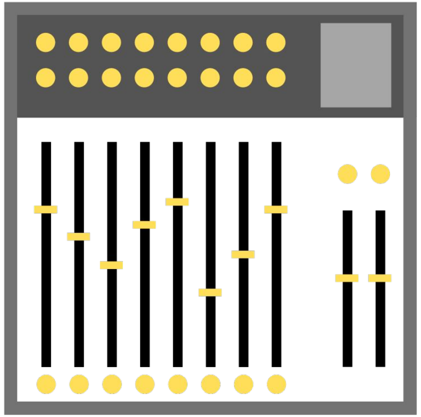
- Technical considerations together with creative, artistic or aesthetic decisions

Achieved with audio effects

- Gain
- Panning
- Equalization (EQ)
- Dynamic range compression (DRC)
- Artificial reverberation

Difficulties with Multitrack Mixing

- High level *engineering task*.
- Project can have *large number* of tracks and varieties of instruments.
- *Time consuming*: lot of repetitive tasks.
- Requires skills *developed over years*.
- Requires understanding of sound, music, and audio.



Camera has automatic face detection, autofocus, red eye, etc.

Mixing consoles aren’t yet smart enough to understand the incoming signal

We need smart mixing consoles.

More people are creating **audio** content

Music

Podcasts

Short-form content

Sound for Video

Demand for **high quality audio**

Producing **high quality audio** requires expertise

Intelligent Multitrack Mixing

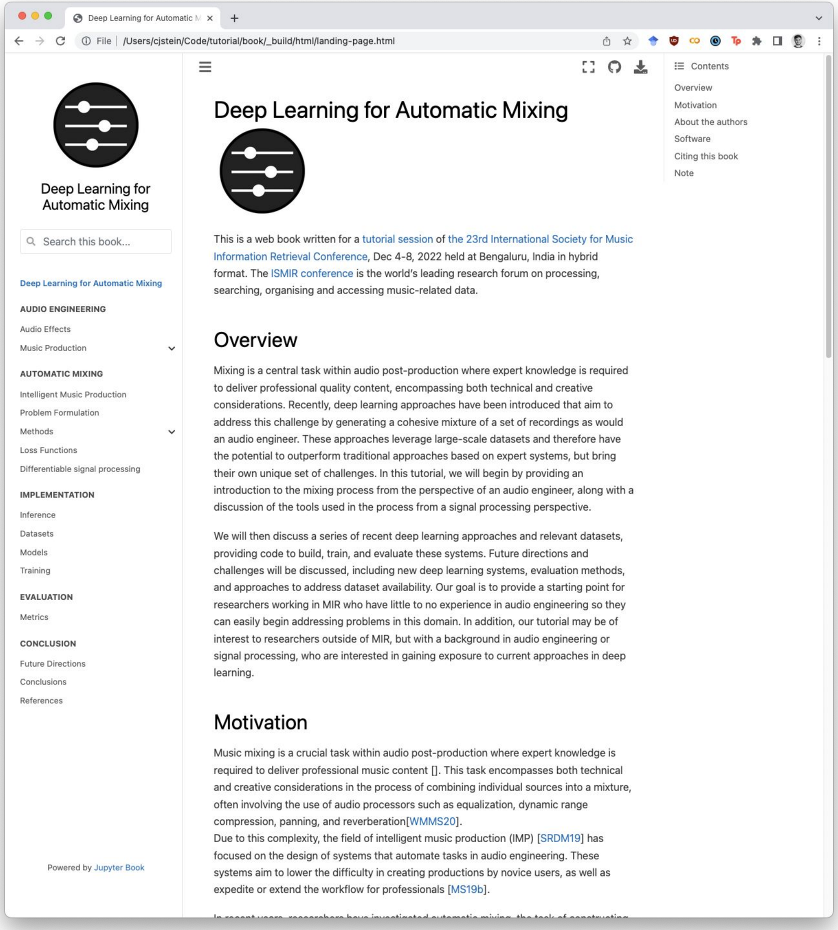
Intelligent tools that automate the complicated task of music mixing to produce technically sound and interpretable mixes.

Goals

1. What is mixing and what should we consider for automix systems?
2. Framework for understanding and designing automix systems
3. Technical understanding of **current deep learning automix** models
4. How to **implement**, **train**, and **evaluate** these models
5. Ideas for future research directions

Book

https://dl4am.github.io/tutorial



Background

Part 2

Automatic Microphone Mixing*

DAN DUGAN

San Francisco, Calif. 94108

A method of analysis of sound reinforcement problems by means of active and passive speech zones is outlined. The need for automatic control of multimicrophone systems is defined, along with the problems associated with the use of voice-operated switches (VOX). Adaptive threshold gating is proposed as the best solution to the problem of active microphone detection. The development and performance of two effective automatic control systems is described.

A ZONAL THEORY OF SOUND REINFORCEMENT

A designer, engineer or contractor who works with sound equipment every day naturally tends to think only about the technical details when approaching a new problem. It is usual to start with deciding where to put the speakers and microphones, and what models will be best for the job. In most cases, this approach is completely valid. There is always a danger that our preoccupation with equipment and specifications will make us miss the real purpose of our efforts. A reinforcement system may have −1 dB frequency response and still not fill the needs of its users.

This paper describes some new inventions which promise to make the craft of sound reinforcement easier and more satisfying. Before getting into the details, I would like to make a short philosophical excursion into a sketch for a general theory of sound reinforcement. This theory is subject to much clarification and improvement.

Each person is the center of a zone in which he can communicate verbally. The size of this zone depends on the acoustical properties of the environment and on the person's ability as a speaker. The variables affecting the size of a person's speech zone may be tabulated:

1) effort
2) vocal ability
3) hearing acuity
4) ambient noise
5) reverberation.

Items 1) – 3) are human variables, 5) and 6) are environmental variables.

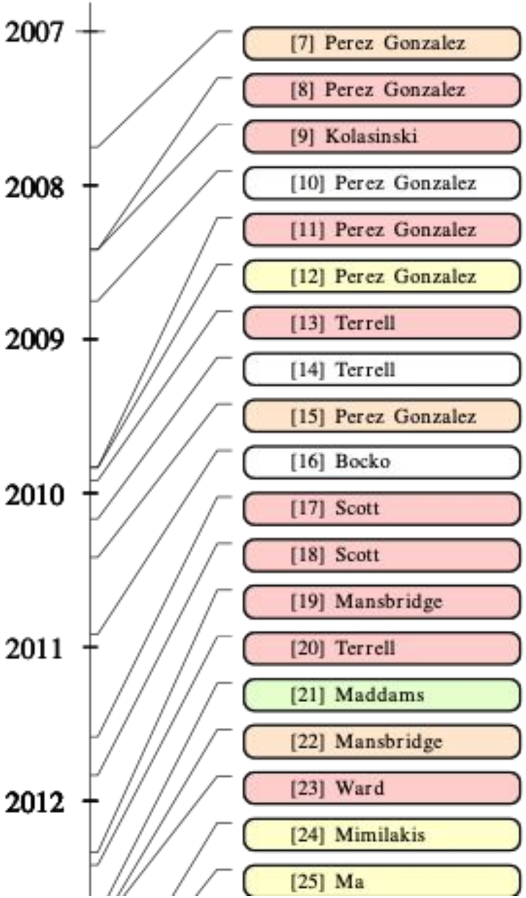
The border of this zone is not clearly defined, as all the variables change constantly, and the human ones are difficult to measure. If typical ranges of values are assigned to the variables, however, the design of environments will become possible in which speech will be relatively easy for almost all people, just as a door is designed to be high enough for people to pass without bumping their heads.

A frustrating thing about working in sound reinforcement is the lack of a direct and positive measurement of the effectiveness of communication transmitted through a system. The best available measurement is the articulation loss for consonants, AL_{cons} [2]. Measurement of AL_{cons} requires a group of observers whose responses can be treated statistically; this is too complex a procedure for daily use. AL_{cons} can be predicted from room data, but verification of these predictions is rare. Nevertheless, AL_{cons} is the best measurement available for speech transmission, and we will use the proposed 15% criterion.

* Presented May 14, 1975, at the Convention of the Audio Engineering Society, Los Angeles.

Dugan, 1975

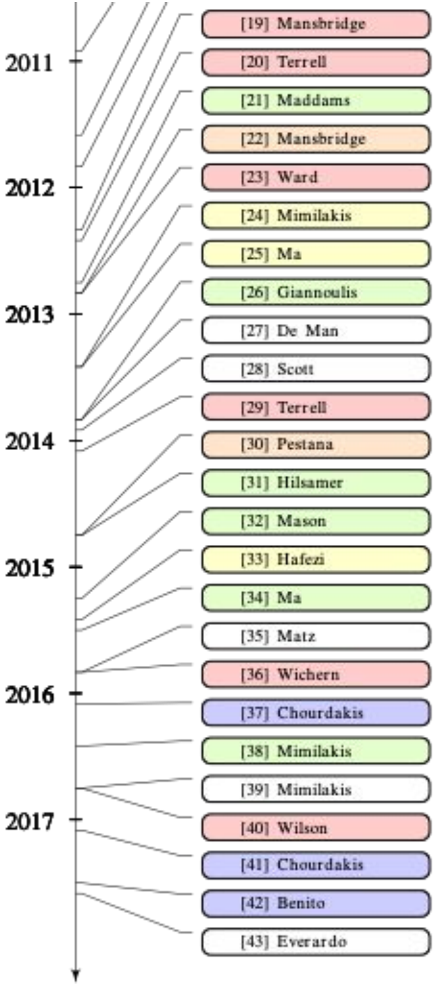
History 2007-2012



(14)

Brecht De Man, Ryan Stables and Joshua D. Reiss, “Ten Years of Automatic Mixing,” Proceedings of the 3rd Workshop on Intelligent Music Production, Salford, UK, 15 September 2017.

History 2012-2017

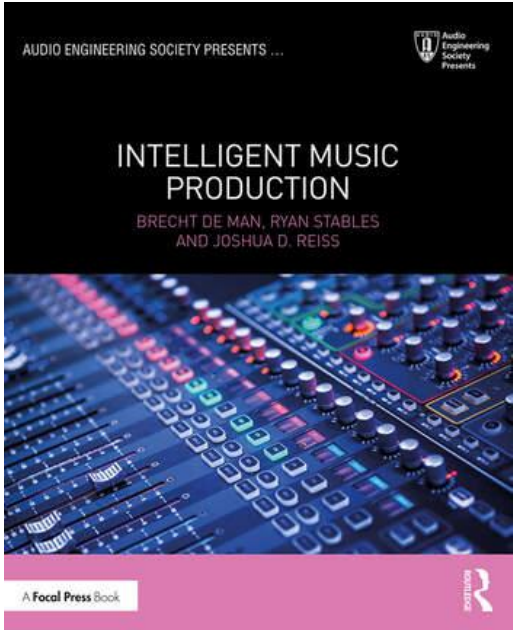


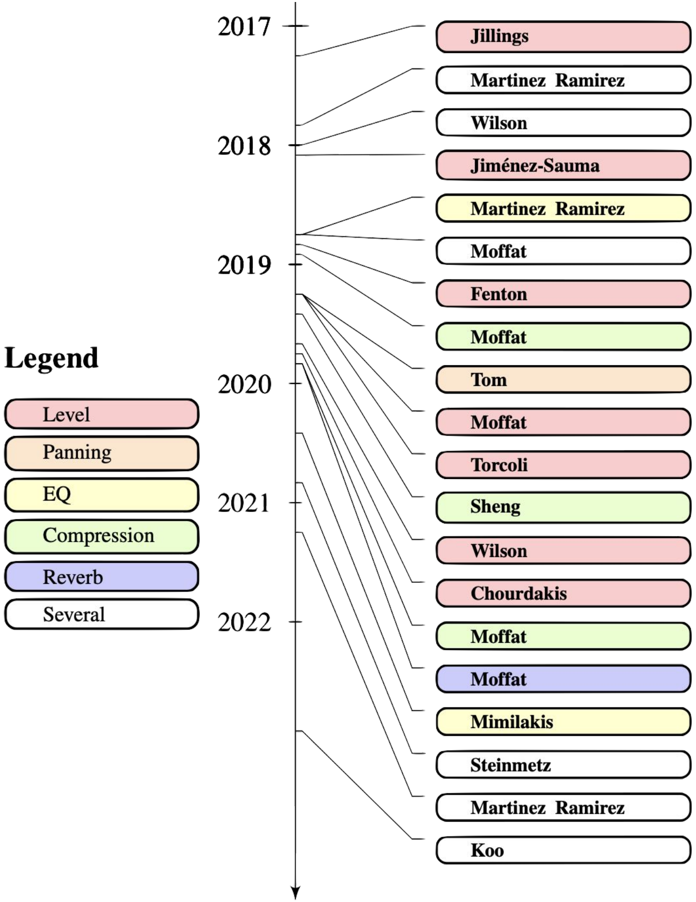
(23)

Brecht De Man, Ryan Stables and Joshua D. Reiss, “Ten Years of Automatic Mixing,” Proceedings of the 3rd Workshop on Intelligent Music Production, Salford, UK, 15 September 2017.

History 2017-2023

https://csteinmetz1.github.io/AutomaticMixingPapers/



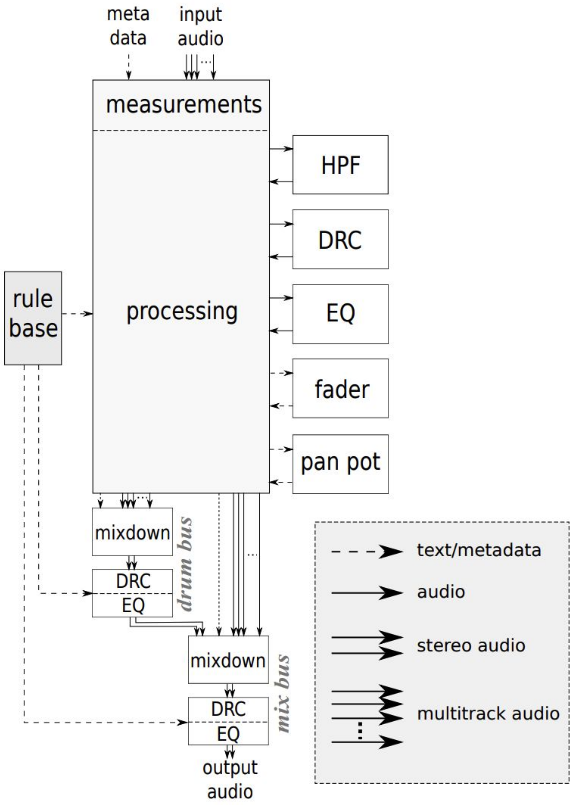


Knowledge-based or **Expert systems**

Design a set of rules based to create a mix based on analysis of the inputs.

Pro: Explainable decisions

Con: Often lacks sufficient complexity



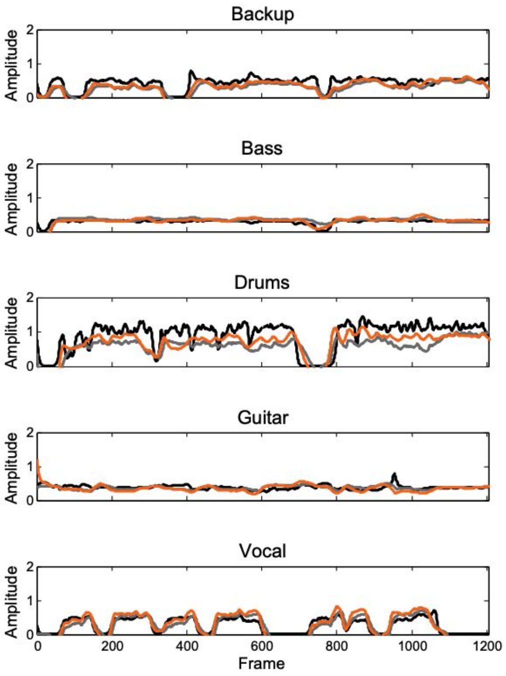
A knowledge-engineered autonomous mixing system
Brecht De Man, Joshua D. Reiss AES 2013

Machine Learning*

Learn to create a mix by leveraging parametric data collected from pros.

Pro: Greater model flexibility

Con: Requires data (parametric)



Analysis of acoustic features for automated multitrack mixing
Jeffrey J. Scott. Youngmoo E. Kim ISMIR 2011

*Approaches that use classical machine learning techniques

Proceedings of the 3rd Workshop on Intelligent Music Production, Salford, UK, 15 September 2017

TEN YEARS OF AUTOMATIC MIXING

Brecht De Man and Joshua D. Reiss
Centre for Digital Music
Queen Mary University of London
{b.deman, joshua.reiss}@qmul.ac.uk

Ryan Stables
Digital Media Technology Lab
Birmingham City University
ryan.stables@bcu.ac.uk

ABSTRACT

Reflecting on a decade of Automatic Mixing systems for multitrack music processing, this paper positions the topic in the wider field of Intelligent Music Production, and seeks to motivate the existing and continued work in this area. Tendencies such as the introduction of machine learning and the increasing complexity of automated systems become apparent from examining a short history of relevant work, and several categories of applications are identified. Based on this systematic review, we highlight some promising directions for future research for the next ten years of Automatic Mixing.

1. MOTIVATION

The democratisation of audio technology has enabled music production on limited budgets, putting high-quality results within reach of anyone who has access to a laptop, a microphone and the abundance of free software on the web. Similarly, musicians are able to share their own content at very little cost and effort, again due to high availability of cheap technology. Despite this, a skilled mix engineer is often still needed in order to deliver professional-standard material. Raw, recorded tracks almost always require a considerable amount of processing before being ready for distribution, such as balancing, panning, equalisation (EQ), dynamic range compression and artificial reverberation, to name a few. Furthermore, an amateur music producer will

Meanwhile, professional audio engineers are often under pressure to produce high-quality content quickly and at low cost [3]. While they may be unlikely to relinquish control entirely to autonomous mix software, assistance with tedious, time-consuming tasks would be highly beneficial. This can be implemented via more powerful, intelligent, responsive, intuitive algorithms and interfaces [4].

Throughout the history of technology, innovation has traditionally been met with resistance and scepticism, in particular from professional users who fear seeing their roles disrupted or made obsolete. Music production technology may be especially susceptible to this kind of opposition, as it is characterised by a tendency towards nostalgia, skeuomorphisms and analogue workflows [1], and it is concerned with aesthetic value in addition to technical excellence and efficiency. However, the evolution of music is intrinsically linked to the development of new instruments and tools, and essentially utilitarian inventions such as automatic vocal riding, drum machines, electromechanical keyboards and digital pitch correction have been famously used and abused for creative effect. These advancements have changed the nature of the sound engineering profession from primarily technical to increasingly expressive. Generally, there is economic, technological and artistic merit in exploiting the immense computing power and flexibility that today's digital technology affords, to venture away from the rigid structure of the traditional music production toolset.

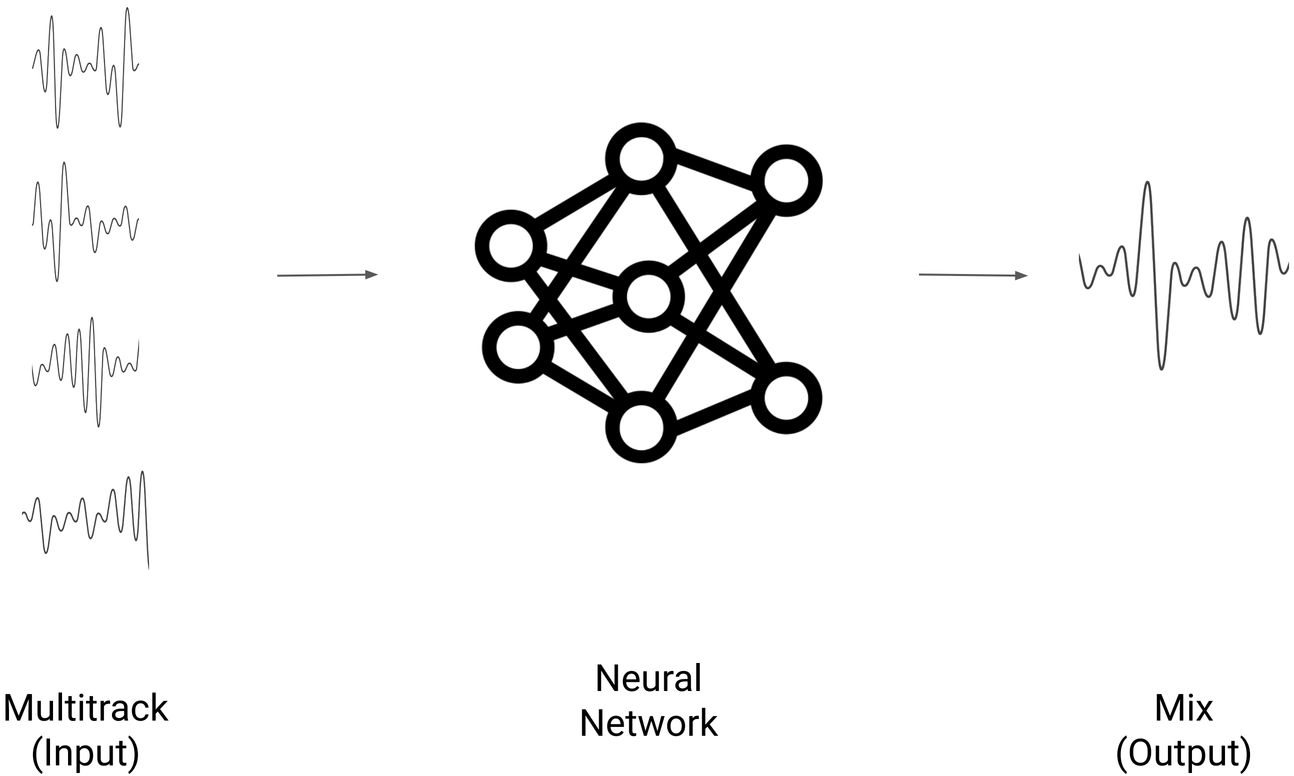
De Man et al., 2017

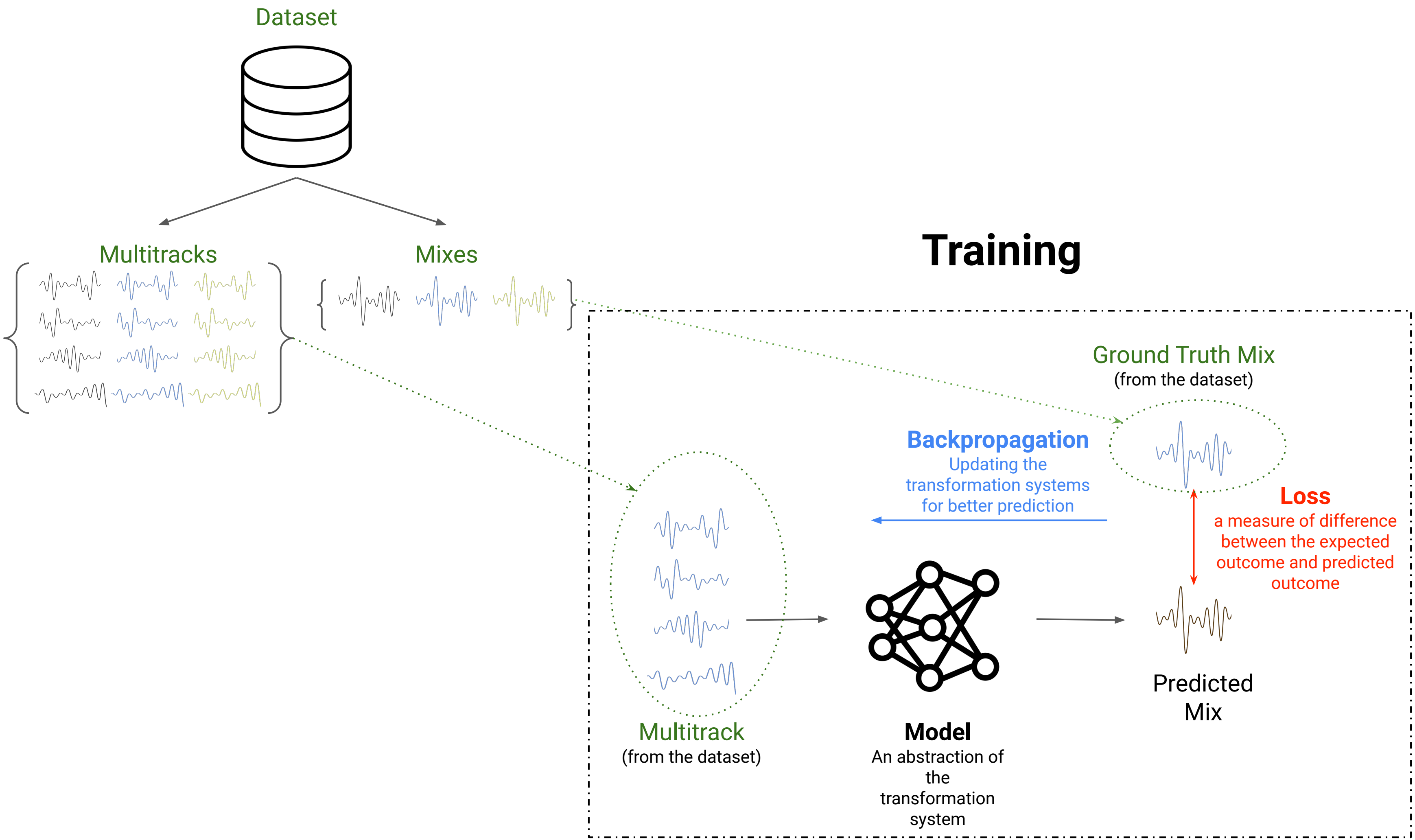
1. Knowledge-based Systems
Gonzalez et al. 2007, De Man et al. 2013,

2. Classical ML-based Systems
Scott and Kim, 2011

3. Deep Learning-based Systems
Martinez Ramirez et al., 2021, 2022; Steinmetz et al. 2020; Koo et al, 2023; Vanka et al, 2024

What we want? (at Inference)





Dataset
Multitracks
Mixes
Training
Ground Truth Mix
(from the dataset)
Backpropagation
Updating the transformation systems for better prediction
Loss
a measure of difference between the expected outcome and predicted outcome
Multitrack
(from the dataset)
Model
An abstraction of the transformation system
Predicted Mix

Datasets

Popular Multitrack Datasets

ENST-Drums

- 8 channels of drum components
- Recordings by 3 drummers
- Accessible on request
- Size: 1.25 hrs

MedleyDB and Mixing Secrets

- Complete songs with varied number of channels and instruments
- Different Genres
- Medley (7.2hrs) + Mixing Secrets (~50hrs)

MUSDB18

- Stems have audio effects applied
- Four stems: Vocals, Bass, Drums, and Others
- Mostly rock, pop, and metal
- ~10hrs

We have very limited open source, time-aligned, real multi-track data capturing various genres and types of music.

Speech recognition: >300 hrs data
Music sequence classification: 280 GB worth data

MoisesDB

MoisesDB is a comprehensive multitrack dataset for source separation beyond 4-stems, comprising 240 previously unreleased songs by 47 artists spanning twelve high-level genres. The total duration of the dataset is 14 hours, 24 minutes and 46 seconds, with an average recording length of 3:36 seconds. MoisesDB is offered free of charge for non-commercial research use only and includes baseline performance results for two publicly available source separation methods.

More datasets

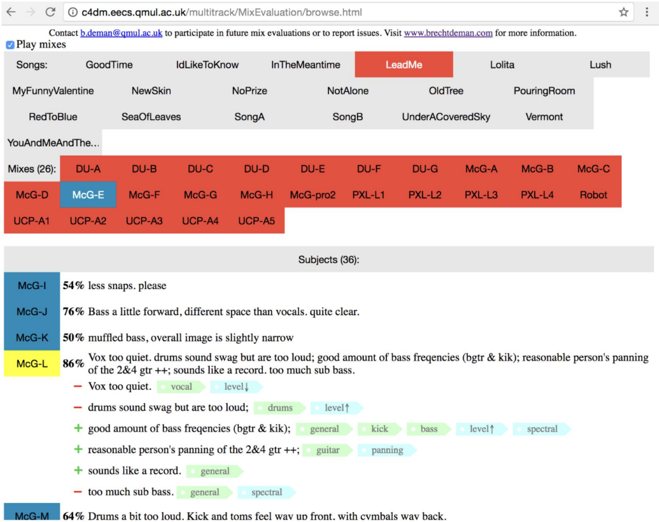
Slakh2100

Manilow, Ethan[1]; Wichern, Gordon[2]; Seetharaman, Prem[1]; Le Roux, Jonathan[2]

Show affiliations

Introduction:

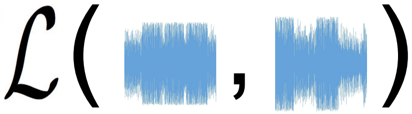
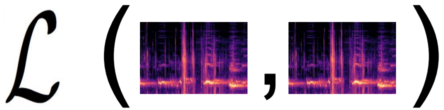
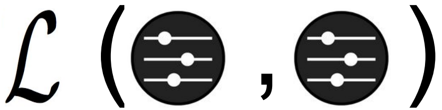
The Synthesized Lakh (Slakh) Dataset is a dataset of multi-track audio and aligned MIDI for music source separation and multi-instrument automatic transcription. Individual MIDI tracks are synthesized from the Lakh MIDI Dataset v0.1 using professional-grade sample-based virtual instruments, and the resulting audio is mixed together to make musical mixtures. This release of Slakh, called Slakh2100, contains 2100 automatically mixed tracks and accompanying, aligned MIDI files, synthesized from 187 instrument patches categorized into 34 classes, totaling 145 hours of mixture data.



Open Multitrack testbed

Loss

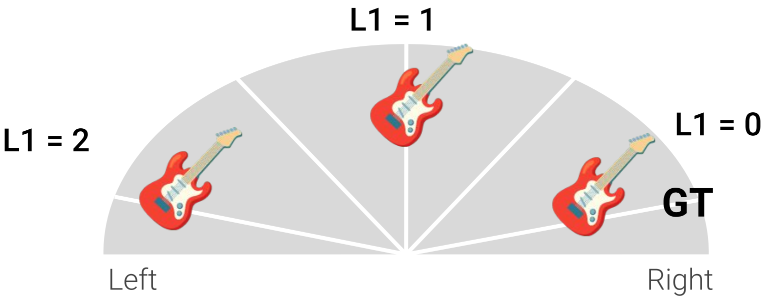
Loss functions

Time domain (Audio Loss)	Frequency domain (Audio Loss)	Parameter Loss
		
Audio needs to be time aligned	Need to choose proper scaling that can capture perceptual qualities of sound	Multiple parameter combinations can lead to same result, may penalise the model unnecessarily

Stereo loss function

Loss function to encourage realistic mixes

Panning here is more perceptually similar but gives a higher L1 loss



L1 and L2 loss on stereo signals encourage panning all elements to the center.

$$y_{\text{sum}} = y_{\text{left}} + y_{\text{right}}$$

$$y_{\text{diff}} = y_{\text{left}} - y_{\text{right}}$$

$$\ell_{\text{Stereo}}(\hat{y}, y) = \ell_{\text{MR-STFT}}(\hat{y}_{\text{sum}}, y_{\text{sum}}) + \ell_{\text{MR-STFT}}(\hat{y}_{\text{diff}}, y_{\text{diff}})$$

Achieves invariance to stereo (left-right) orientation

auraloss

A collection of audio-focused loss functions in PyTorch

[PDF]

Setup

```
pip install auraloss
```

Usage

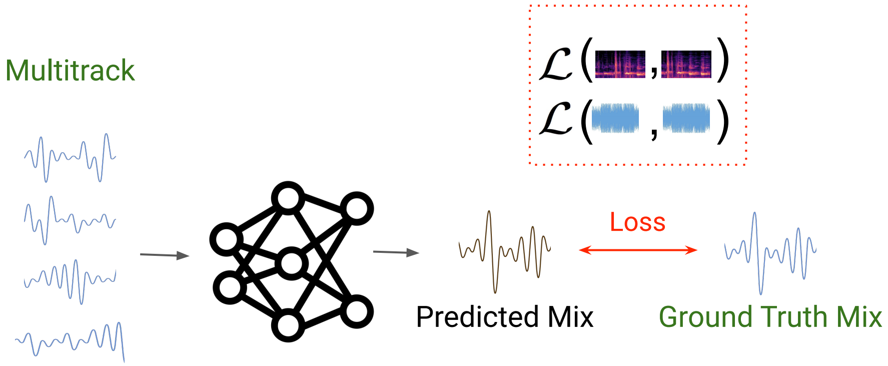
```python
import torch
import auraloss

mrstft = auraloss.freq.MultiResolutionSTFTLoss()

input = torch.rand(8,1,44100)
target = torch.rand(8,1,44100)

loss = mrstft(input, target)
```

https://github.com/csteinmetz1/auraloss

Loss function	Interface	Reference
Time domain		
Error-to-signal ratio (ESR)	`auraloss.time.ESRLoss()`	Wright & Välimäki, 2019
DC error (DC)	`auraloss.time.DCLoss()`	Wright & Välimäki, 2019
Log hyperbolic cosine (Log-cosh)	`auraloss.time.LogCoshLoss()`	Chen et al., 2019
Signal-to-noise ratio (SNR)	`auraloss.time.SNRLoss()`	
Scale-invariant signal-to-distortion ratio (SI-SDR)	`auraloss.time.SISDRLoss()`	Le Roux et al., 2018
Scale-dependent signal-to-distortion ratio (SD-SDR)	`auraloss.time.SDSDRLoss()`	Le Roux et al., 2018
Frequency domain		
Aggregate STFT	`auraloss.freq.STFTLoss()`	Arik et al., 2018
Aggregate Mel-scaled STFT	`auraloss.freq.MelSTFTLoss(sample_rate)`	
Multi-resolution STFT	`auraloss.freq.MultiResolutionSTFTLoss()`	Yamamoto et al., 2019*
Random-resolution STFT	`auraloss.freq.RandomResolutionSTFTLoss()`	Steinmetz & Reiss, 2020
Sum and difference STFT loss	`auraloss.freq.SumAndDifferenceSTFTLoss()`	Steinmetz et al., 2020
Perceptual transforms		
Sum and difference signal transform	`auraloss.perceptual.SumAndDifference()`	
FIR pre-emphasis filters	`auraloss.perceptual.FIRFilter()`	Wright & Välimäki, 2019

Model Design

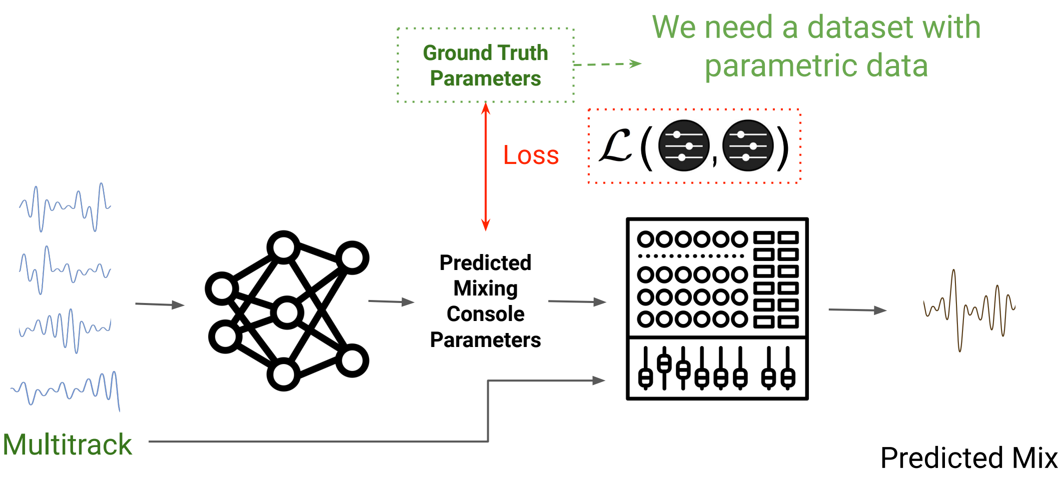
Model Types



Direct Transformation

Black box system that lacks interpretability and controllability (context not incorporated)

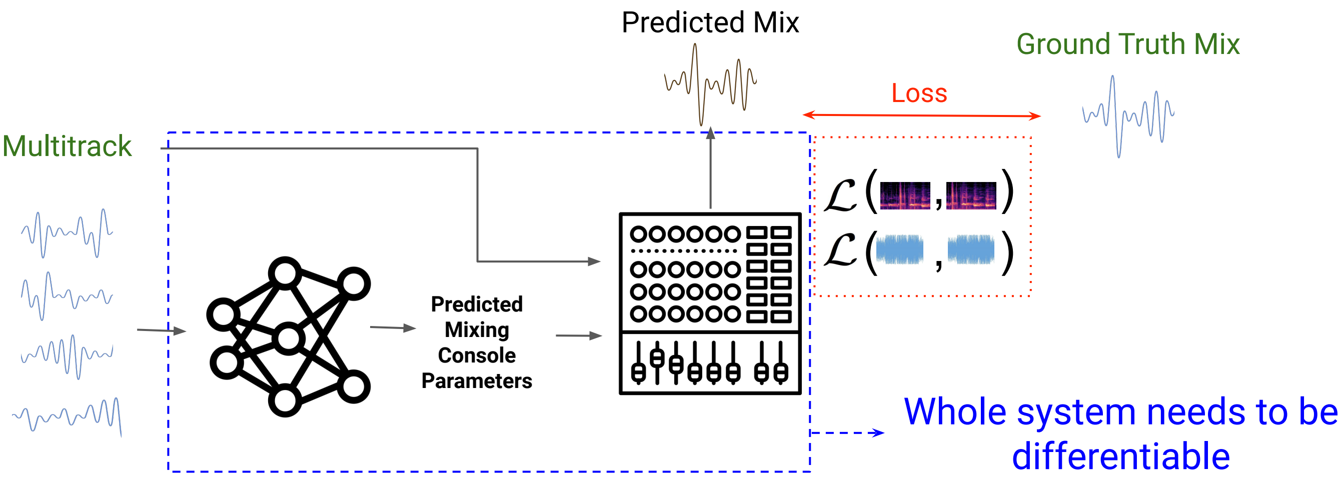
Model Types



Parameter Estimation
(Parameter Loss)

Black box system that allows interpretability and controllability (context not incorporated)

Model Types

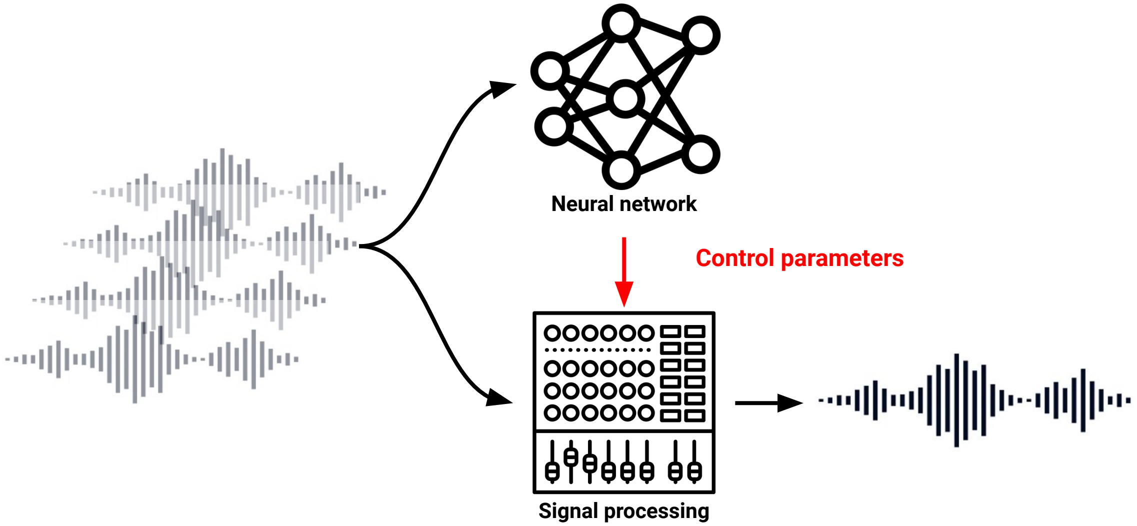


Parameter Estimation
(Audio Loss)

Black box system that allows interpretability and controllability (context not incorporated)

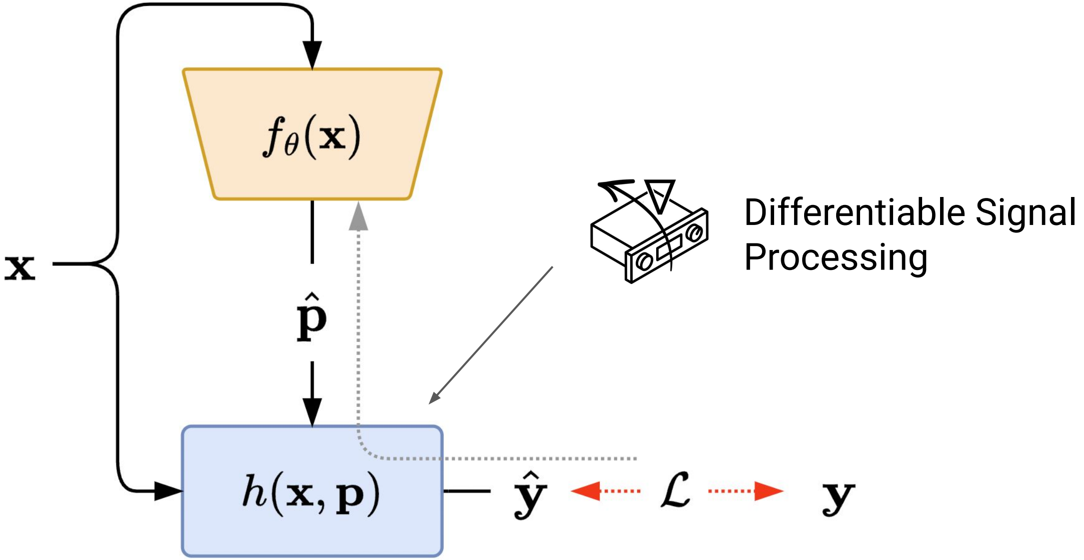
DDSP: Differentiable Digital Signal Processing

Neural networks that control DSP



- High-fidelity with minimal risk of introducing artifacts
- Audio processing is visible and controllable by end users
- Significantly more efficient enabling operation on CPU

Neural networks that control DSP



...but this requires harmonization of signal processing and **gradient-based learning**

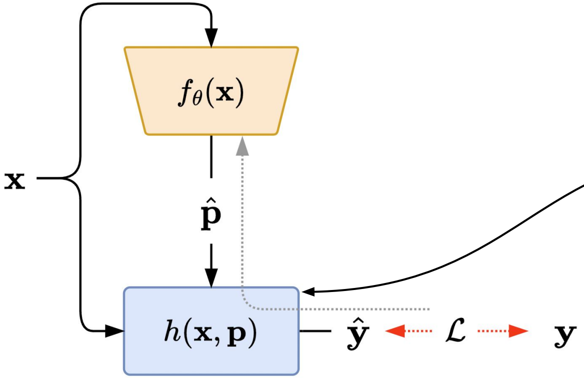
Techniques

1. **Automatic differentiation (AD)**
   Engel et al. 2020

2. **Neural proxies and hybrids (NP)**
   Steinmetz et al. 2020, Steinmetz et al. 2022

3. **Numerical gradient approximation (NGA)**
   Martínez Ramírez et al. 2021

Automatic Differentiation



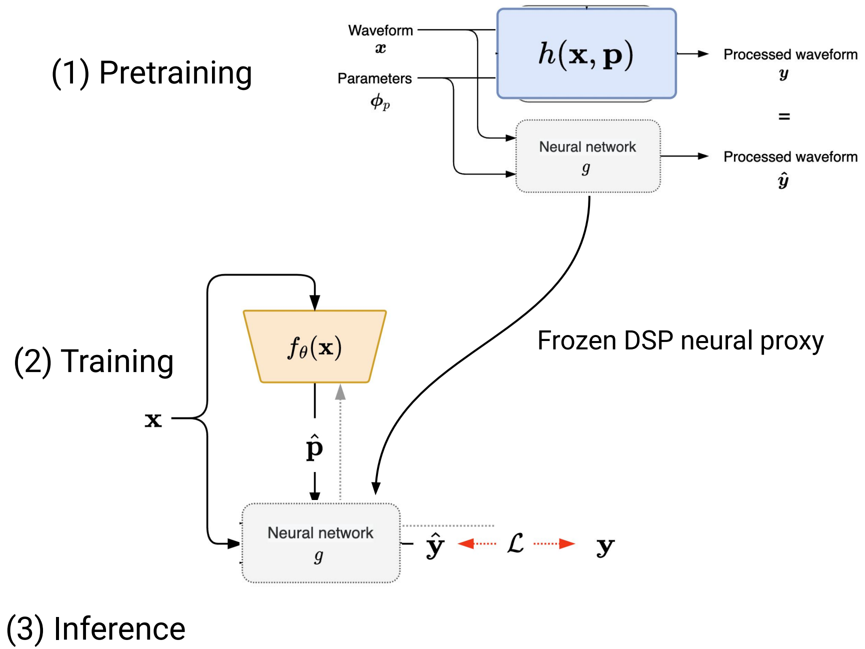
PyTorch TensorFlow

```
A = 10 ** (gain_dB / 40.0)
w0 = 2 * math.pi * (cutoff_freq / sample_rate)
alpha = torch.sin(w0) / (2 * q_factor)
cos_w0 = torch.cos(w0)
sqrt_A = torch.sqrt(A)

if filter_type == "high_shelf":
    b0 = A * ((A + 1) + (A - 1) * cos_w0 + 2 * sqrt_A * alpha)
    b1 = -2 * A * ((A - 1) + (A + 1) * cos_w0)
    b2 = A * ((A + 1) + (A - 1) * cos_w0 - 2 * sqrt_A * alpha)
    a0 = (A + 1) - (A - 1) * cos_w0 + 2 * sqrt_A * alpha
    a1 = 2 * ((A - 1) - (A + 1) * cos_w0)
    a2 = (A + 1) - (A - 1) * cos_w0 - 2 * sqrt_A * alpha
```

Explicitly define signal processing operations in autodiff framework

Engel, Jesse, et al. "DDSP: Differentiable digital signal processing." *ICLR* (2021).

Neural Proxy



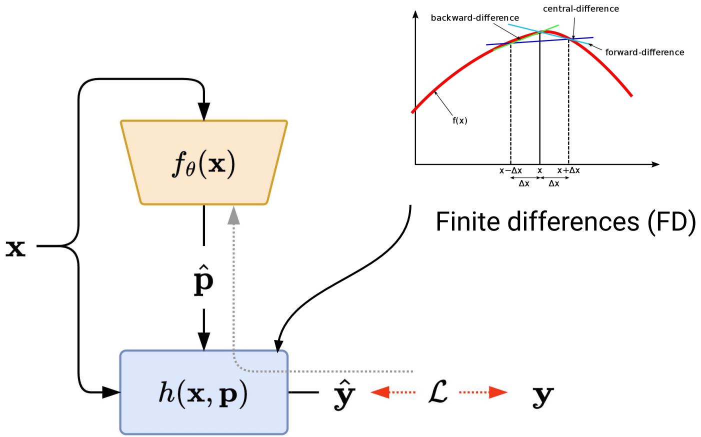
Steinmetz, Christian J., et al. "Automatic multitrack mixing with a differentiable mixing console of neural audio effects." ICASSP, 2021.

Gradient Approximation

$$\frac{\hat{h}(x, p_i)}{p_i} = \frac{h(x, p + \varepsilon\Delta^P) - h(x, p - \varepsilon\Delta^P)}{2\varepsilon\Delta_i^P}, \qquad (2)$$

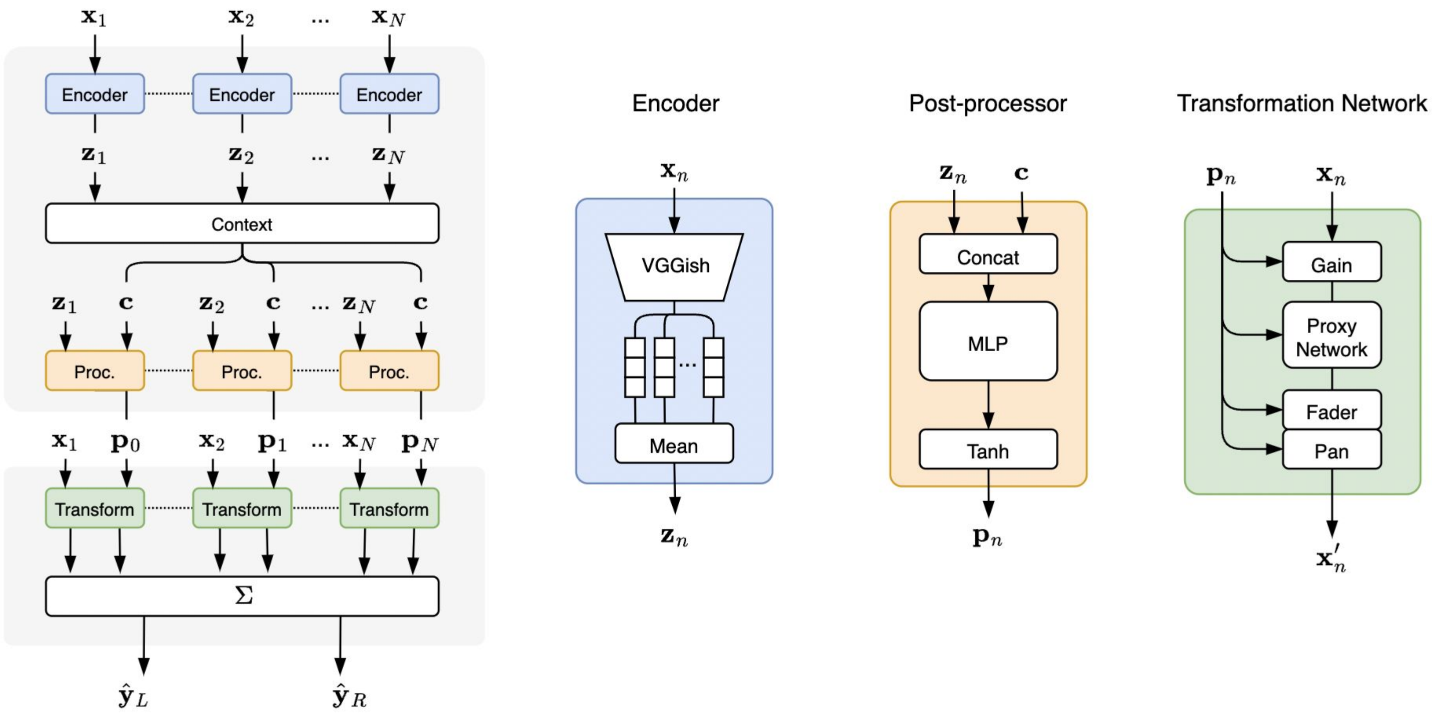
where ε is a small, non-zero value and $\Delta^P \in \mathbb{R}^P$ is a random vector sampled from a symmetric Bernoulli distribution ($\Delta_i^P = \pm 1$) [46].

Simultaneous perturbation stochastic approximation (SPSA)



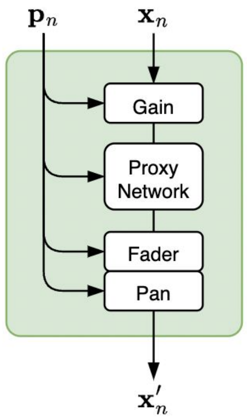
Martínez Ramírez, Marco A., et al. "Differentiable signal processing with black-box audio effects." ICASSP, 2021.

Creating a differentiable mixing console

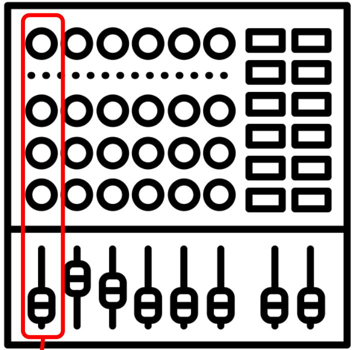


Steinmetz, Christian J., et al. "Automatic multitrack mixing with a differentiable mixing console of neural audio effects." ICASSP, 2021.

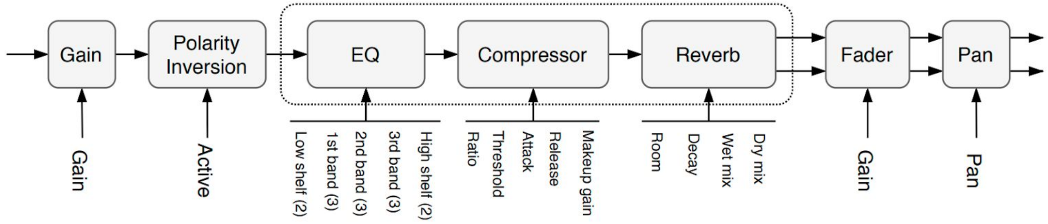
Creating a differentiable mixing console



Proxy network

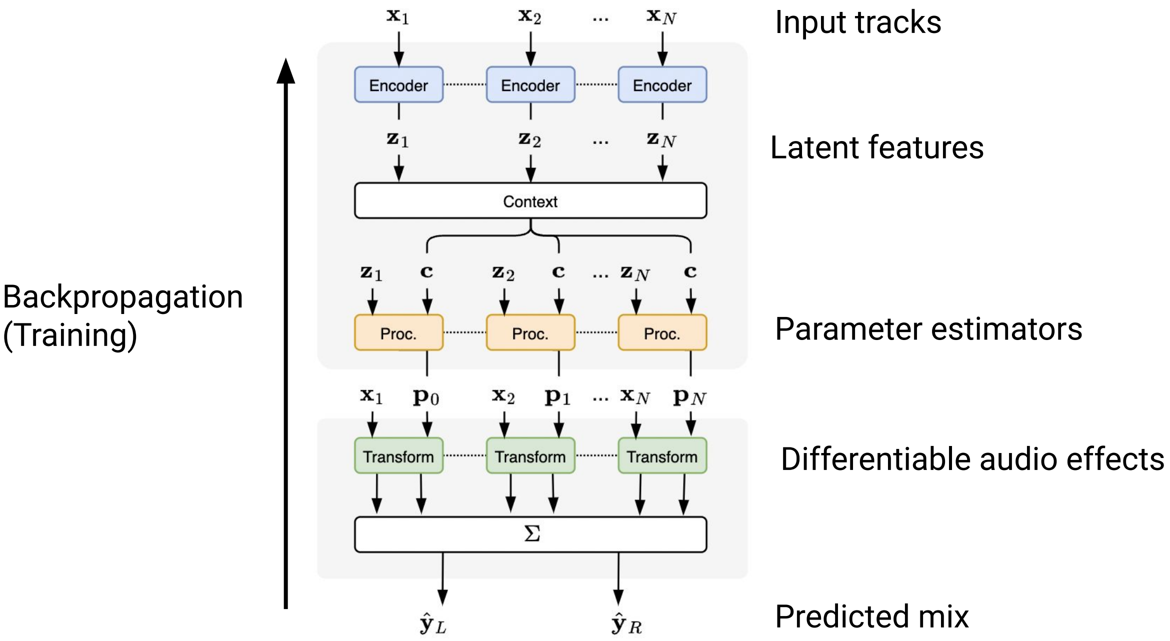


Differentiable channel strip



Steinmetz, Christian J., et al. "Automatic multitrack mixing with a differentiable mixing console of neural audio effects." ICASSP, 2021.

Creating a differentiable mixing console



Steinmetz, Christian J., et al. "Automatic multitrack mixing with a differentiable mixing console of neural audio effects." ICASSP, 2021.

DASP

Differentiable audio signal processors

in PyTorch

Reverberation

Compressor / Expander

Parametric Equalizer

Distortion

Stereo Widener

Stereo Panner

DASP

Differentiable audio signal processors

in PyTorch

$f(x)$ Pure functional interface for each audio processor

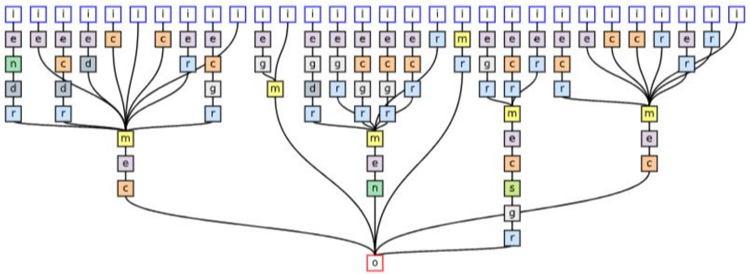
Differentiable implementations enable backprop

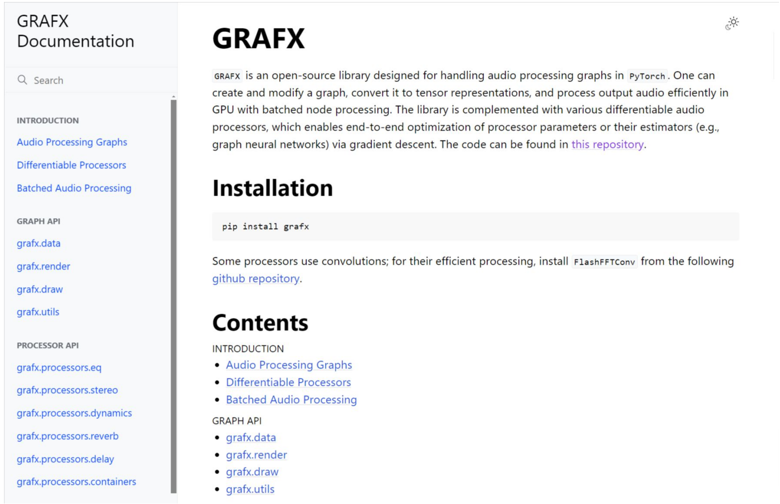
Can target CPU or GPU with support for batching

Permissive open source license (Apache 2.0)

GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch

`pip install grafx`




GRAFX
Documentation

Search

INTRODUCTION
Audio Processing Graphs
Differentiable Processors
Batched Audio Processing

GRAPH API
grafx.data
grafx.render
grafx.draw
grafx.utils

PROCESSOR API
grafx.processors.eq
grafx.processors.stereo
grafx.processors.dynamics
grafx.processors.reverb
grafx.processors.delay
grafx.processors.containers

GRAFX

`GRAFX` is an open-source library designed for handling audio processing graphs in `PyTorch`. One can create and modify a graph, convert it to tensor representations, and process output audio efficiently in GPU with batched node processing. The library is complemented with various differentiable audio processors, which enables end-to-end optimization of processor parameters or their estimators (e.g., graph neural networks) via gradient descent. The code can be found in this repository.

Installation

```
pip install grafx
```

Some processors use convolutions; for their efficient processing, install `FlashFFTConv` from the following github repository.

Contents

INTRODUCTION
- Audio Processing Graphs
- Differentiable Processors
- Batched Audio Processing

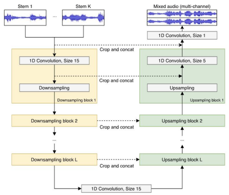
GRAPH API
- grafx.data
- grafx.render
- grafx.draw
- grafx.utils

GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch, Lee et al. (DAFx24, Sep 2024)

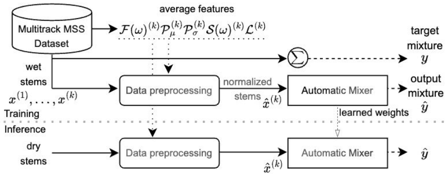
Work-so-Far

Part 3

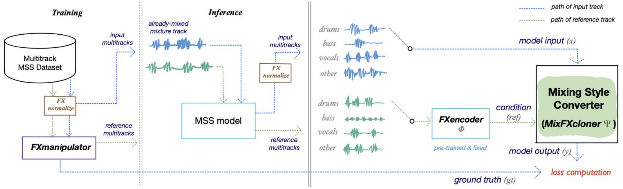
Direct Transformation



Wave-U-Net for drum mixing [a]

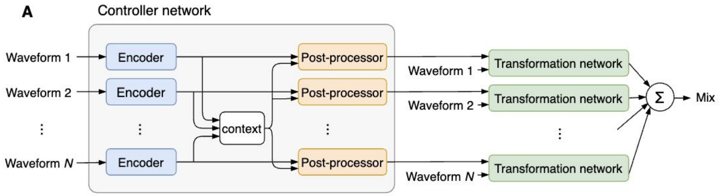


Mixing with out-of-domain data [c]

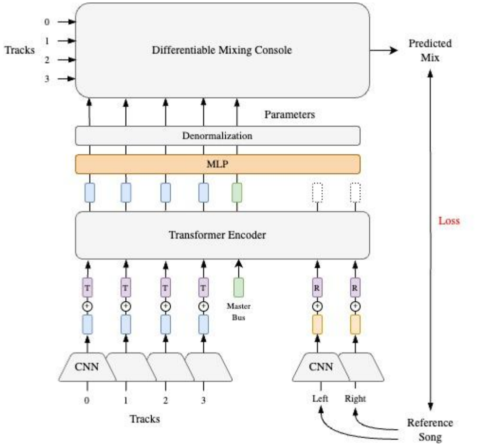


Mixing style transfer [d]

Parameter Estimation



Mixing with neural mixing console [b]



Mixing style transfer with differentiable mixing console [e]

[a] A Deep Learning Approach to Intelligent Drum Mixing With the Wave-U-Net, Martinez-Ramirez et al. (JAES Mar, 2021)
[b] Automatic multitrack mixing with a differentiable mixing console of neural audio effects, Steinmetz et al. (ICASSP 2021)
[c] Automatic music mixing with deep learning and out-of-domain data, Martinez-Ramirez et al. (ISMIR 2022)
[d] Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects, Koo et al. (ICASSP 2023)
[e] Diff-MST: Differentiable Mixing Style Transfer, Vanka et al. (ISMIR 2024)

First Attempt (2021)

Mix-Wave-U-Net

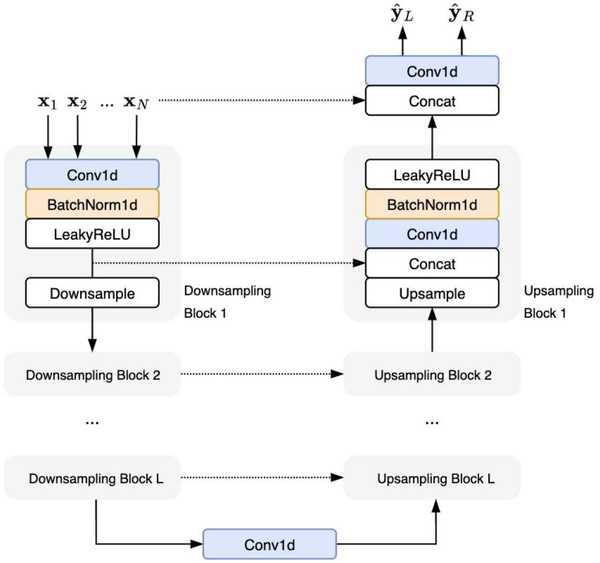
A Deep Learning Approach to Intelligent Drum Mixing with the Wave-U-Net

Marco A. Martínez Ramírez[1*], **Daniel Stoller**[1*], **AND David Moffat**[2], *AES Student Member*
(m.a.martinezramirez@qmul.ac.uk) (d.stoller@qmul.ac.uk) (david.moffat@plymouth.ac.uk)

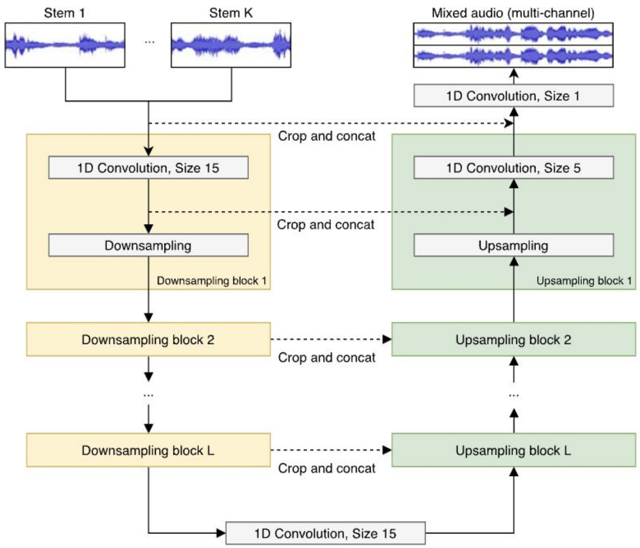
[1]*Centre for Digital Music, Queen Mary University of London, London, United Kingdom*
[2]*University of Plymouth, Plymouth, United Kingdom*

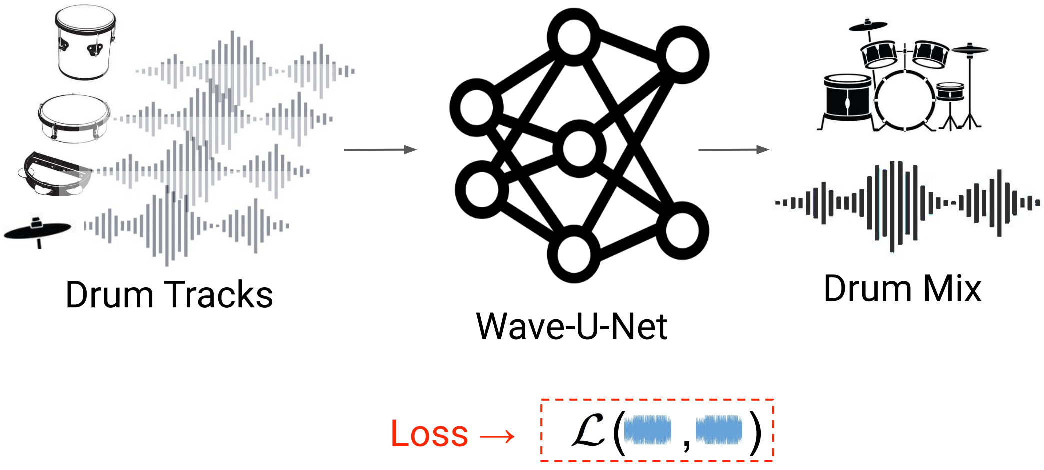
[*] *These authors contributed equally to this work.*

The development of intelligent music production tools has been of growing interest in recent years. Deep learning approaches have been shown as being a highly effective method for approximating individual audio effects. In this work, we propose an end-to-end deep neural network based on the Wave-U-Net to perform automatic mixing of drums. We follow an end-to-end approach, where raw audio from the individual drum recordings is the input of the system and the waveform of the stereo mix is the output. We compare the system to existing machine learning approaches to intelligent drum mixing. Through a subjective listening test, we explore the performance of these systems when processing various types of drum mixes. We report that the mixes generated by our model are virtually indistinguishable from professional human mixes, while also outperforming previous intelligent mixing approaches.



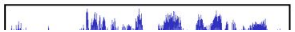
A Deep Learning Approach to Intelligent Drum Mixing With the Wave-U-Net

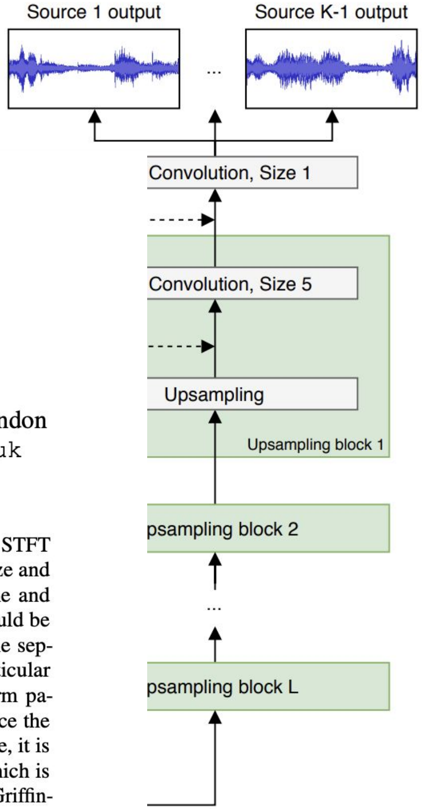




- Pros: directly learns the audio transformation
- Limitations: **Only drum mixing**, number of tracks is fixed

Wave-U-Net





WAVE-U-NET: A MULTI-SCALE NEURAL NETWORK FOR END-TO-END AUDIO SOURCE SEPARATION

Daniel Stoller
Queen Mary University of London
d.stoller@qmul.ac.uk

Sebastian Ewert
Spotify
sewert@spotify.com

Simon Dixon
Queen Mary University of London
s.e.dixon@qmul.ac.uk

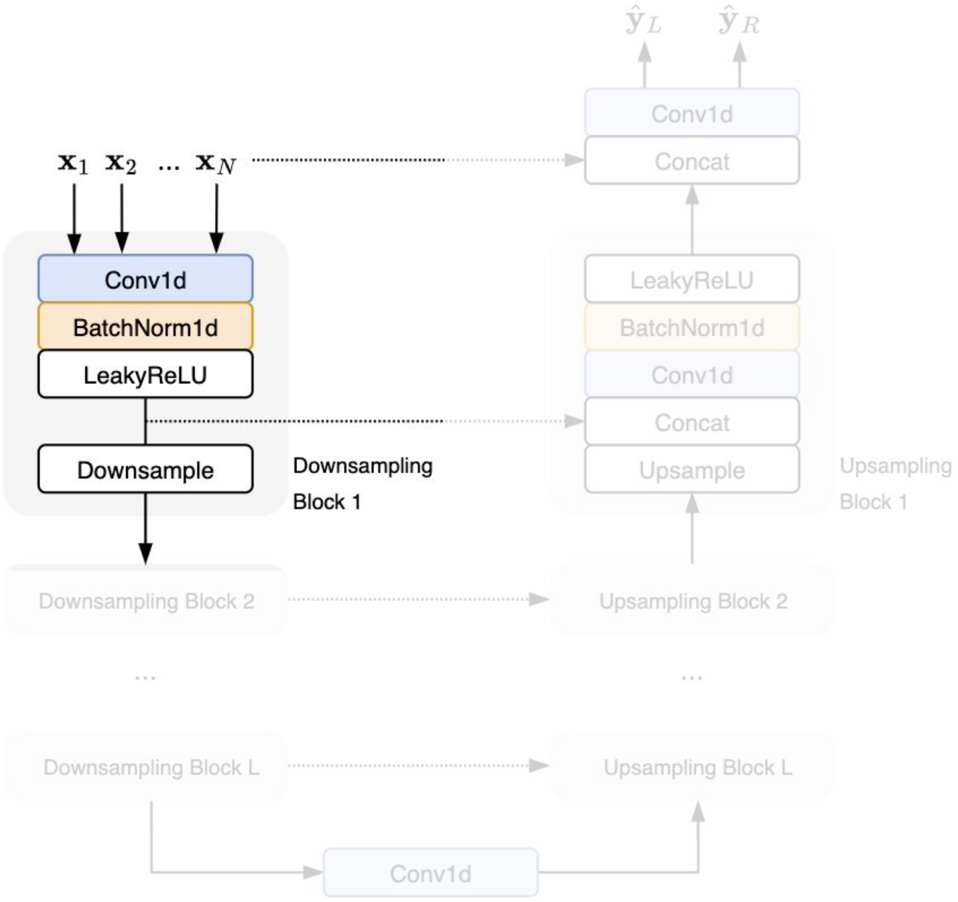
ABSTRACT

Models for audio source separation usually operate on the magnitude spectrum, which ignores phase information and makes separation performance dependant on hyper-parameters for the spectral front-end. Therefore, we investigate end-to-end source separation in the time-domain, which allows modelling phase information and avoids fixed spectral transformations. Due to high sampling rates for audio, employing a long temporal input context on the sample level is difficult, but required for high quality separation results because of long-range temporal correlations. In this context, we propose the Wave-U-Net, an adaptation of the U-Net to the one-dimensional time domain, which repeatedly resamples feature maps to compute and com-

This approach has several limitations. Firstly, the STFT output depends on many parameters, such as the size and overlap of audio frames, which can affect the time and frequency resolution. Ideally, these parameters should be optimised in conjunction with the parameters of the separation model to maximise performance for a particular separation task. In practice, however, the transform parameters are fixed to specific values. Secondly, since the separation model does not estimate the source phase, it is often assumed to be equal to the mixture phase, which is incorrect for overlapping partials. Alternatively, the Griffin-Lim algorithm can be applied to find an approximation to a signal whose magnitudes are equal to the estimated ones, but this is slow and often no such signal exists [8]. Lastly, the mixture phase is ignored in the estimation of sources,

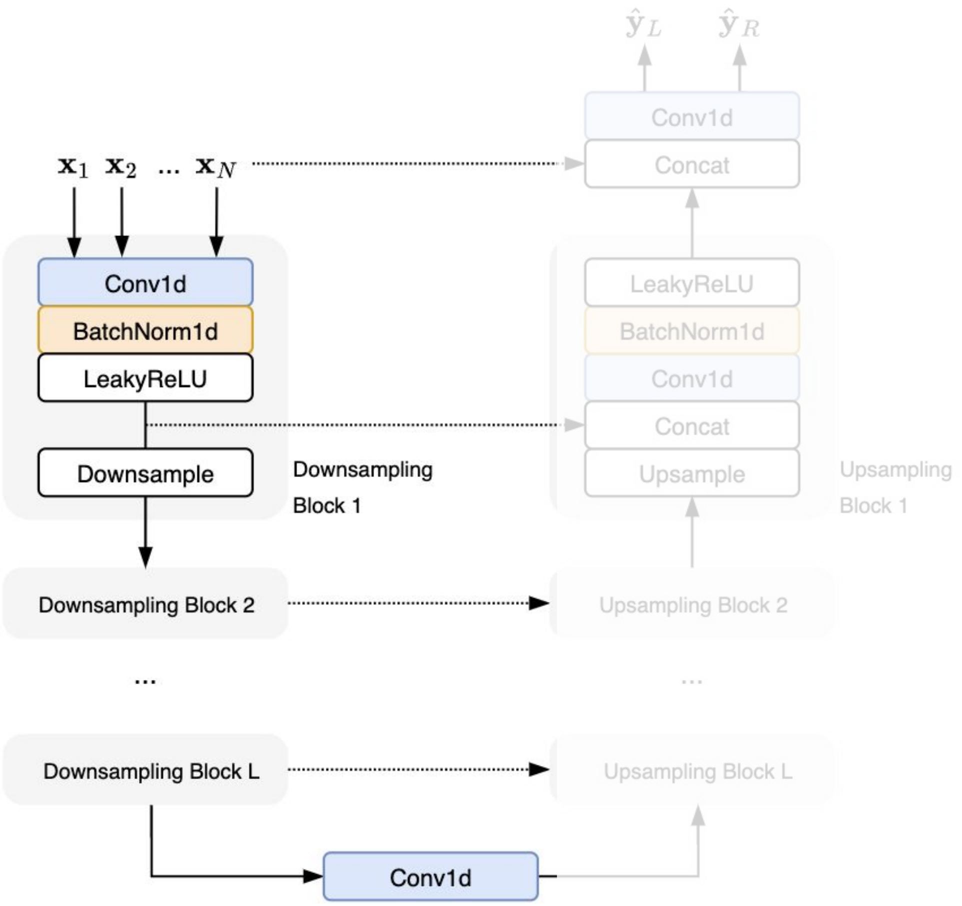
Mix-Wave-U-Net

Downsampling block



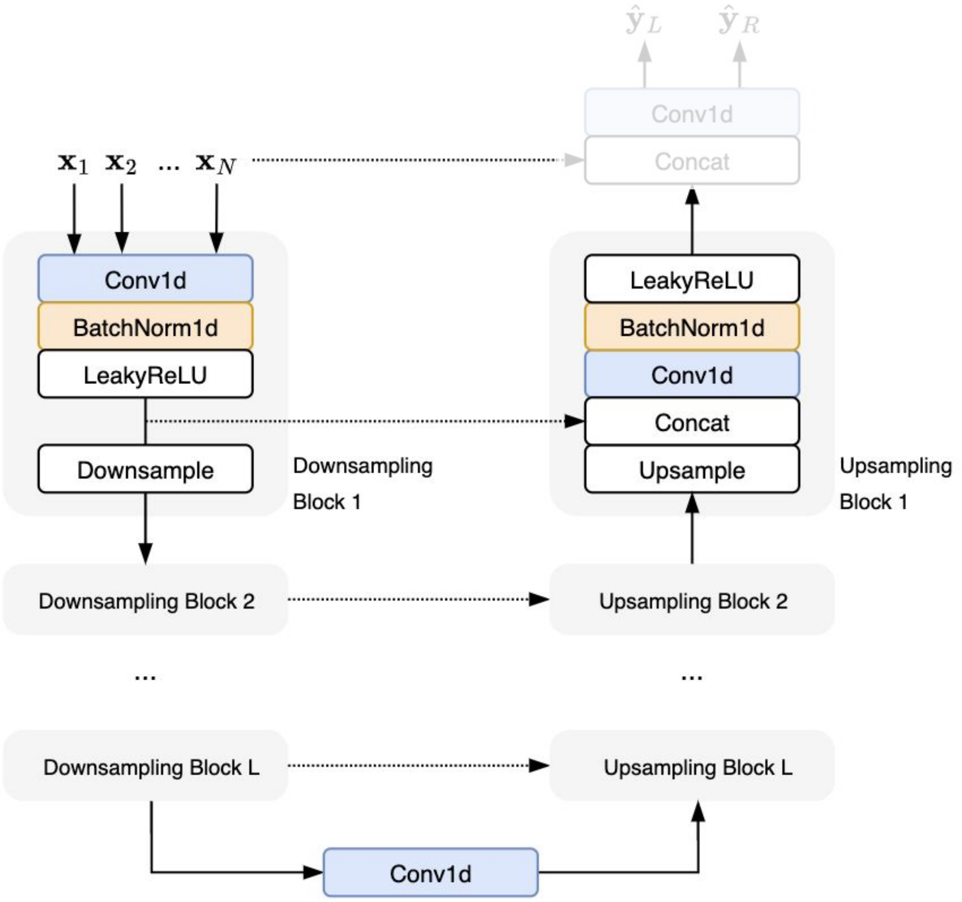
Mix-Wave-U-Net

Downsampling block



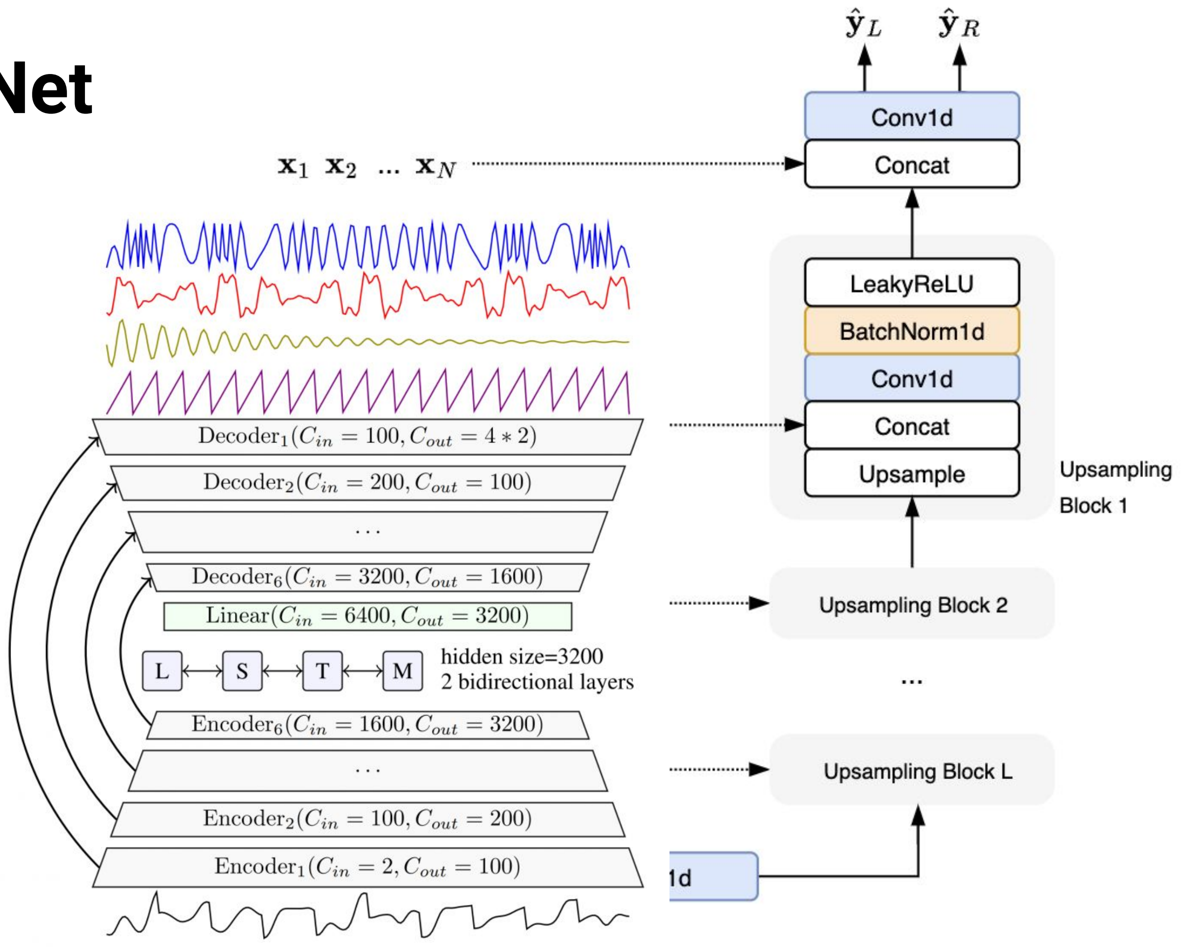
Mix-Wave-U-Net

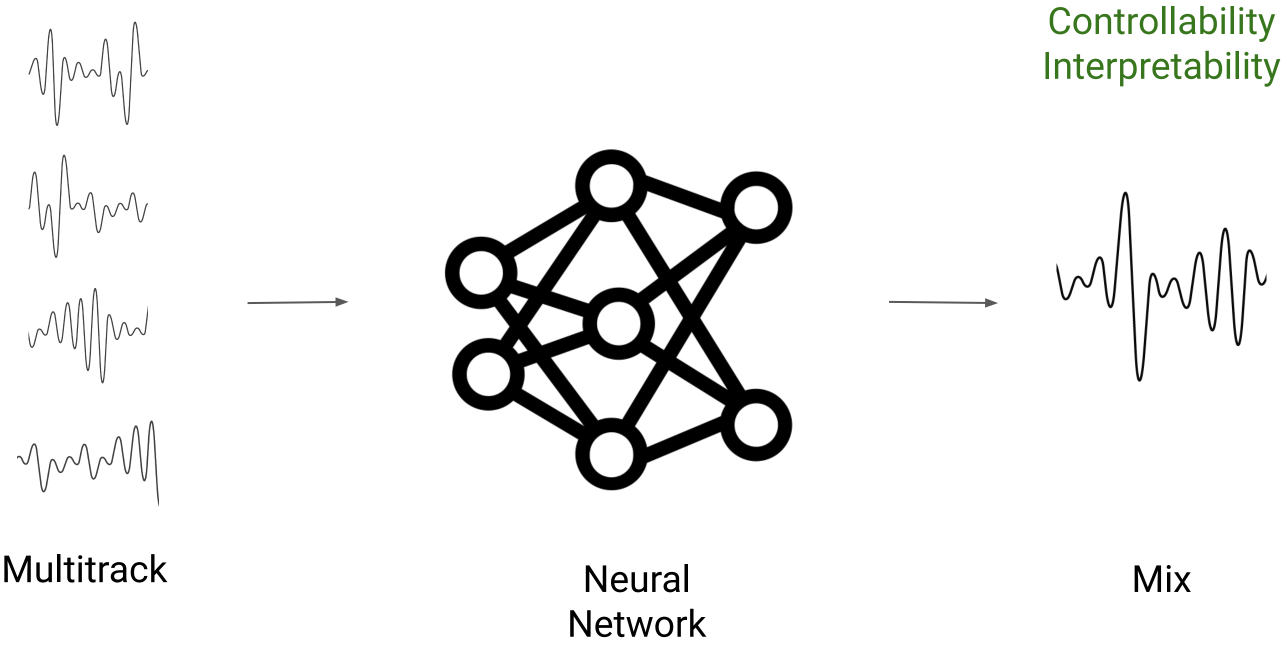
Upsampling block



Mix-Wave-U-Net

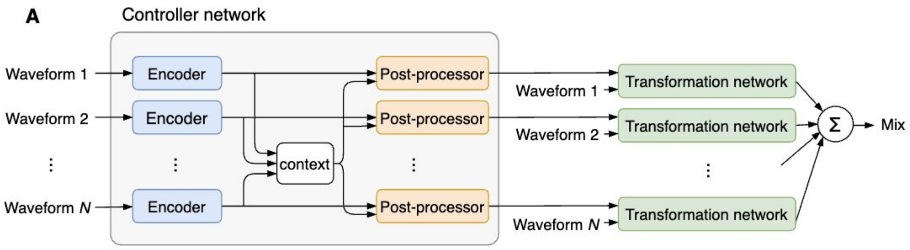
Output layer



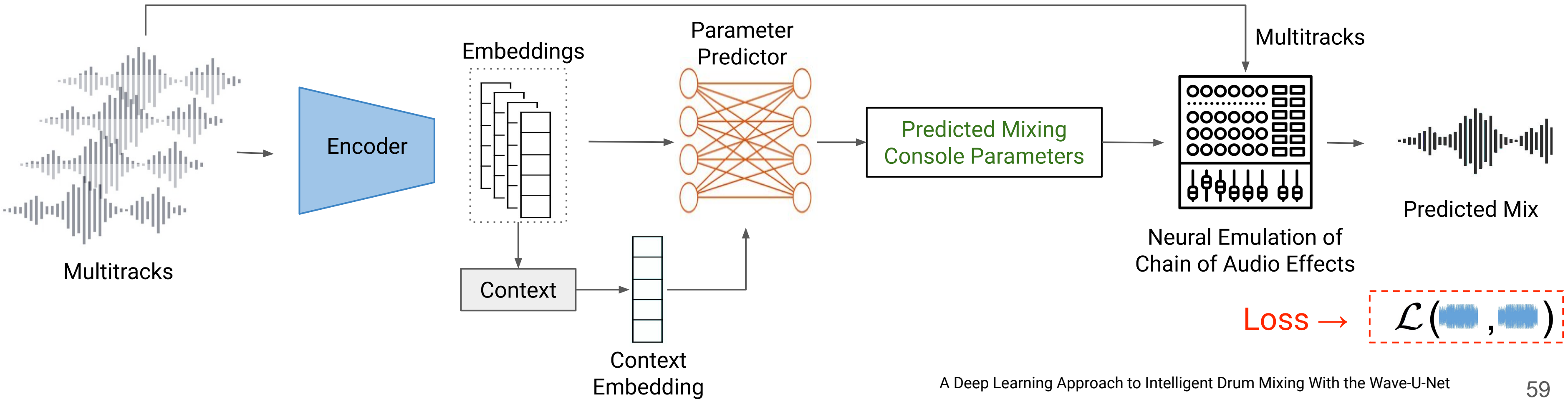


Can we make it controllable? (2021)

Automatic multitrack mixing with a differentiable mixing console of neural audio effects



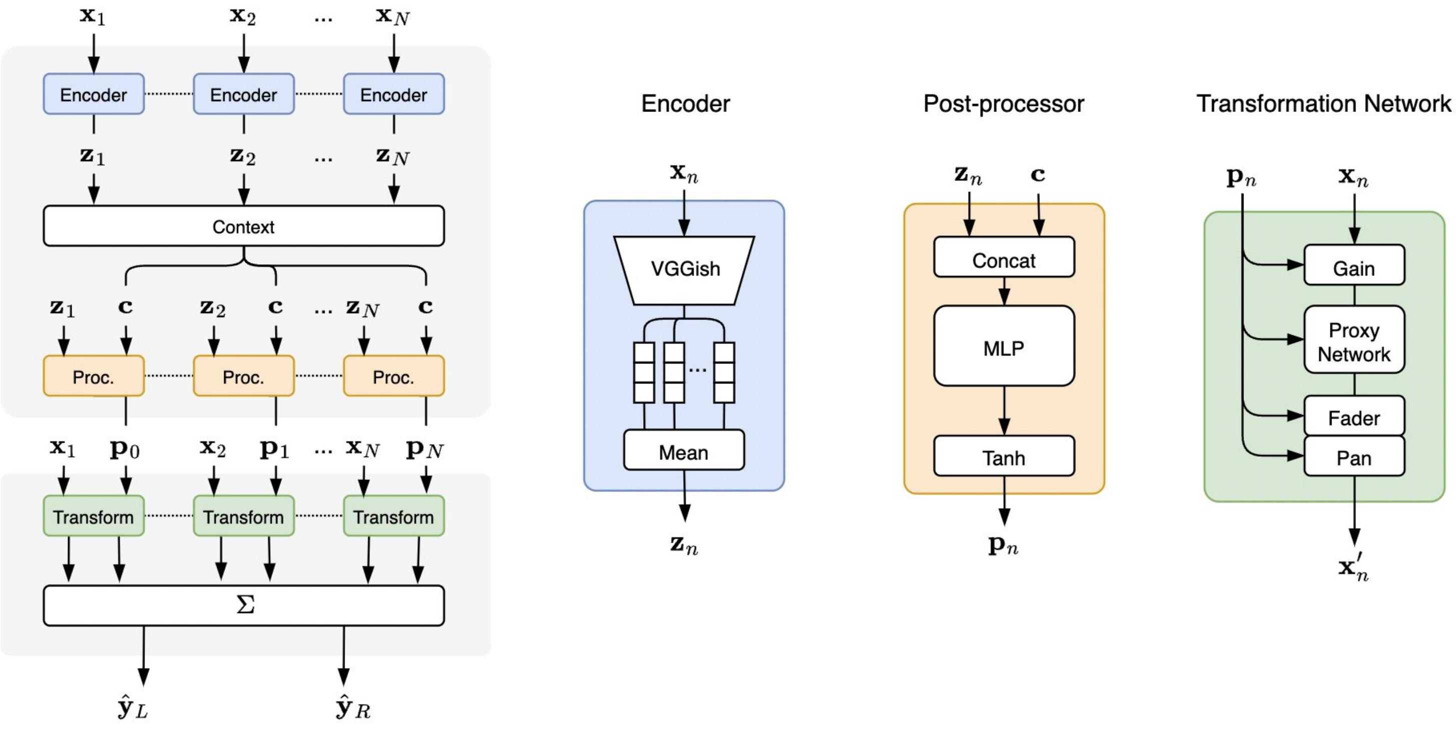
- Pros: Permutation invariant, works for any number of tracks, allows multitrack mixing
- Limitations: neural emulation of effects are difficult to train, **doesn't work well for all cases (Could be due to lack of enough data)**



A Deep Learning Approach to Intelligent Drum Mixing With the Wave-U-Net

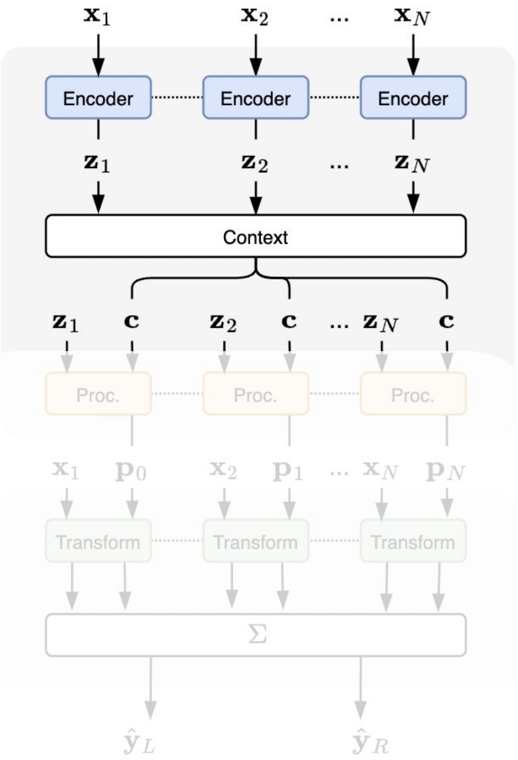
Differentiable Mixing Console

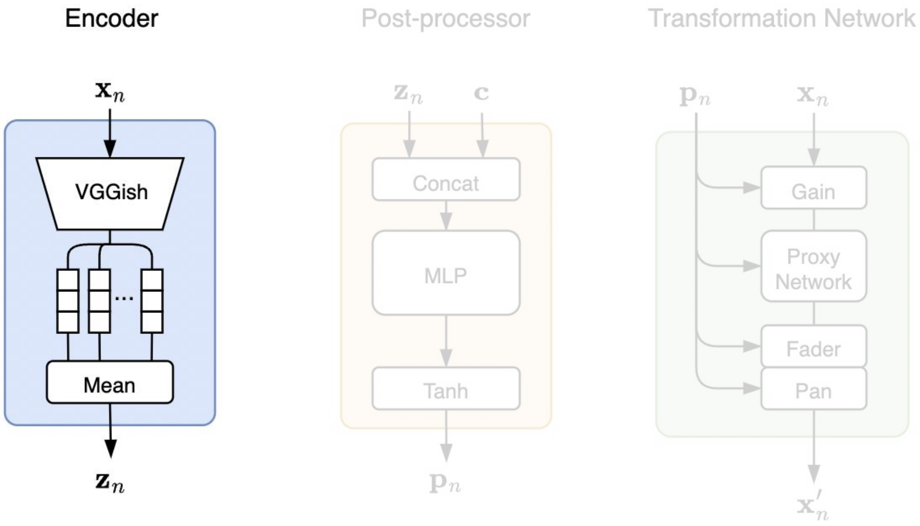
Parameter estimation



Differentiable Mixing Console

Encoder

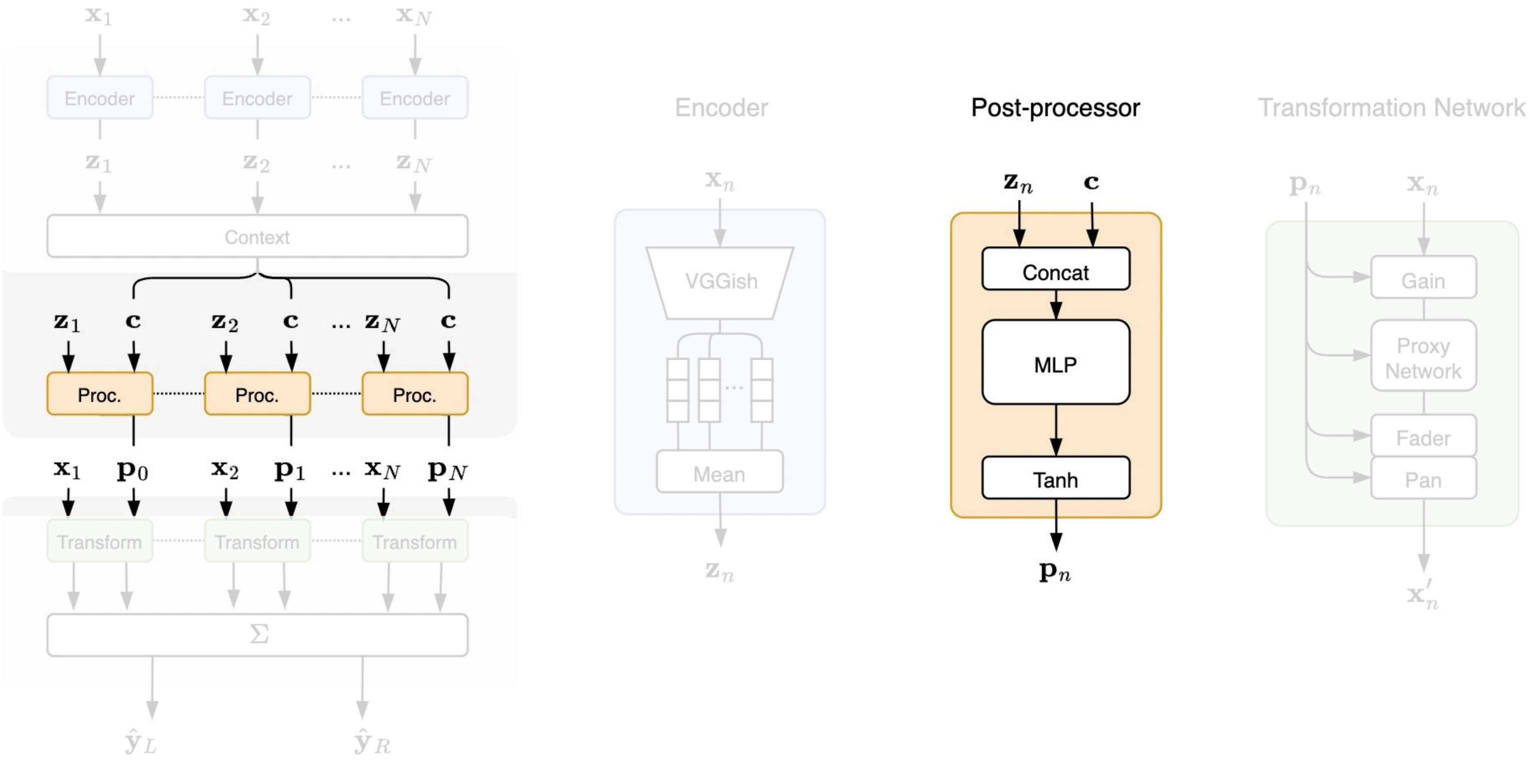




Weight sharing

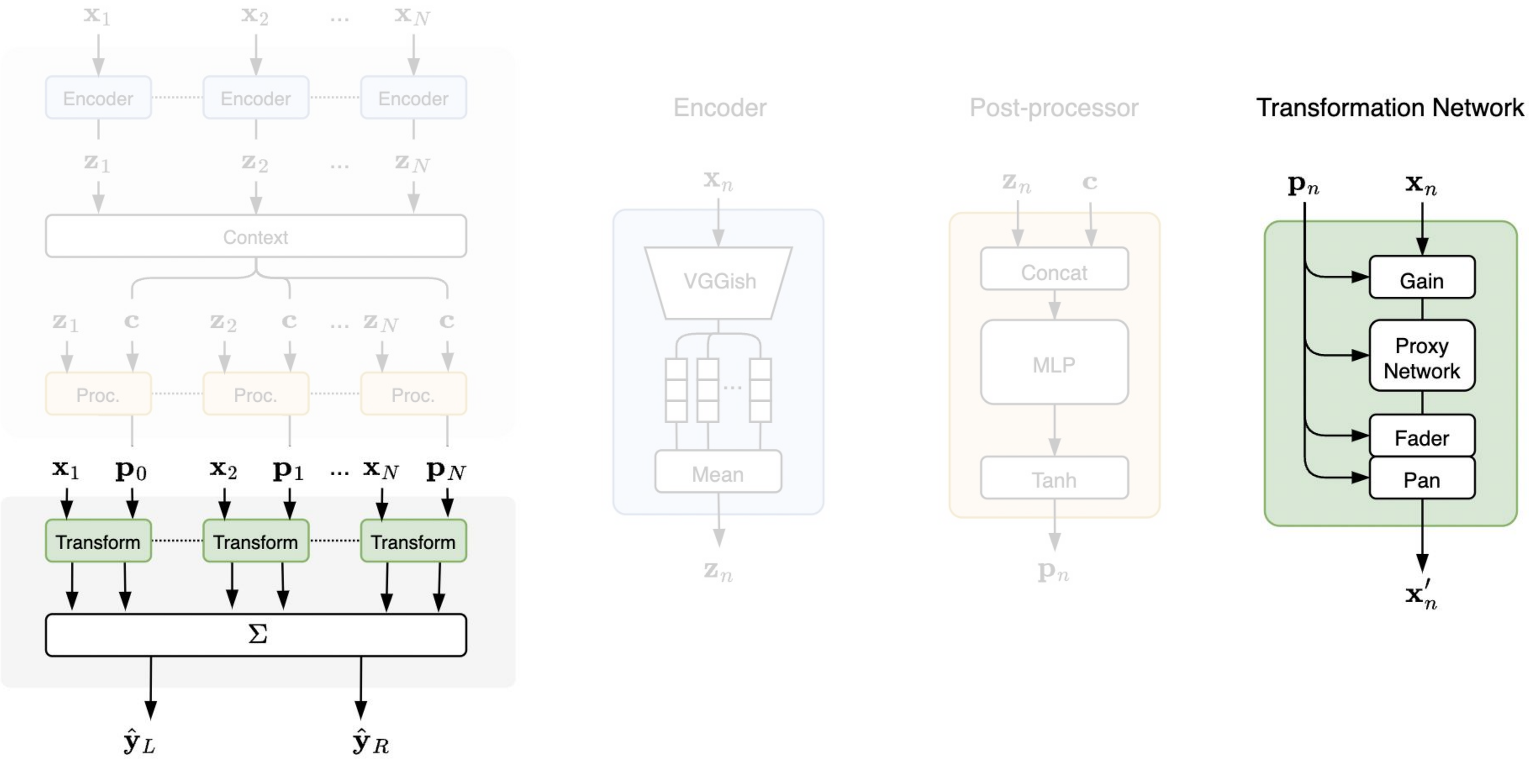
Differentiable Mixing Console

Post-processor



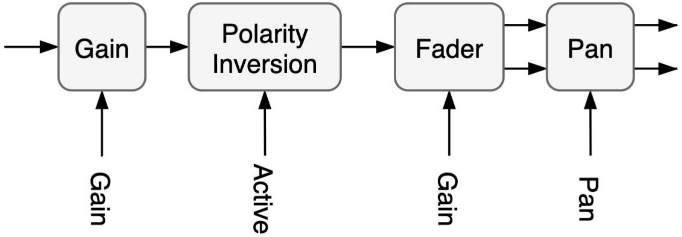
Differentiable Mixing Console

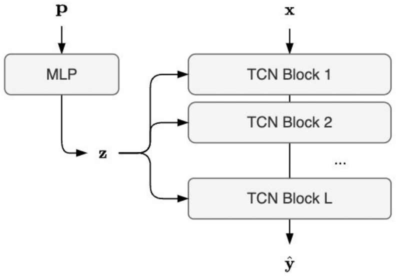
Transformation Network



Differentiable Mixing Console

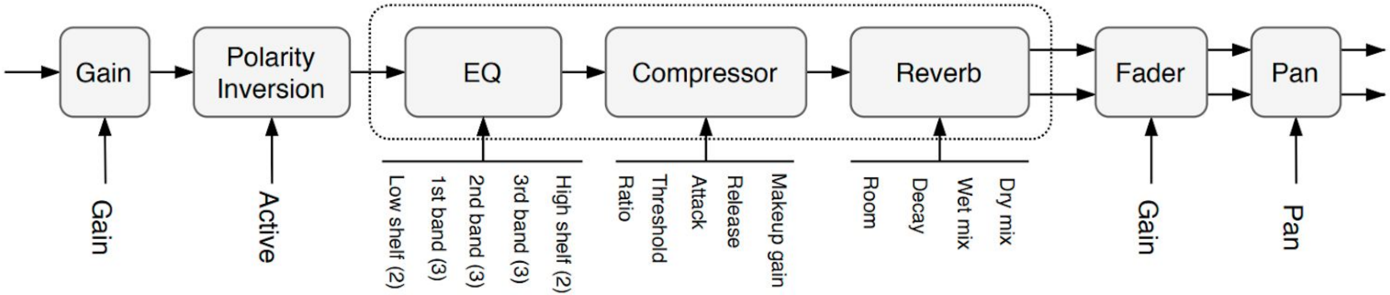
Gain + Panning (Proxy network is not used)





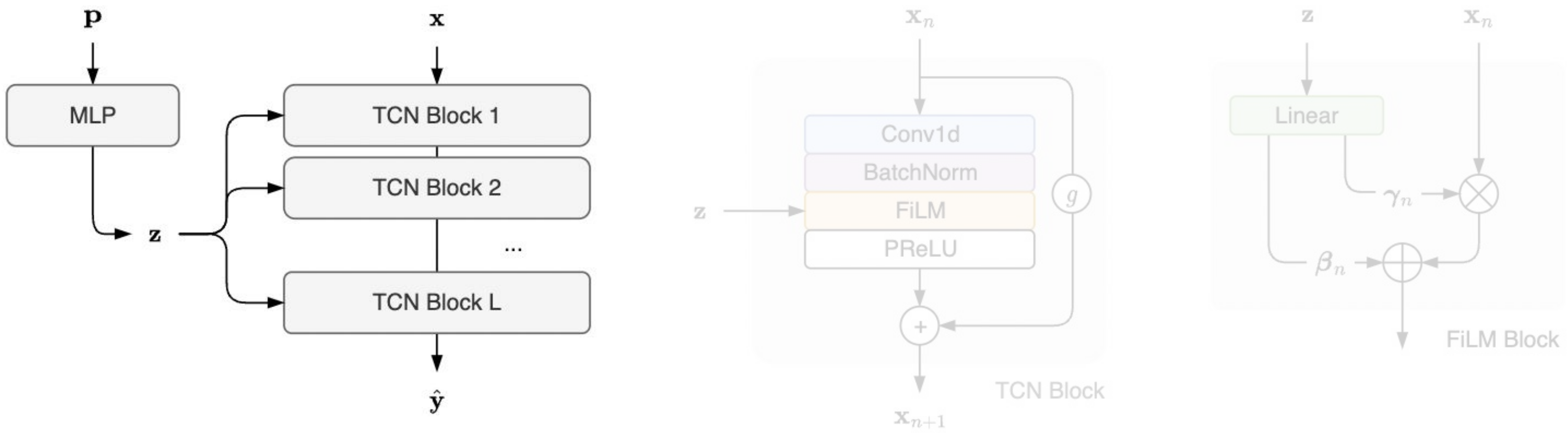
Proxy network

Gain + EQ + Compressor + Reverb + Panning



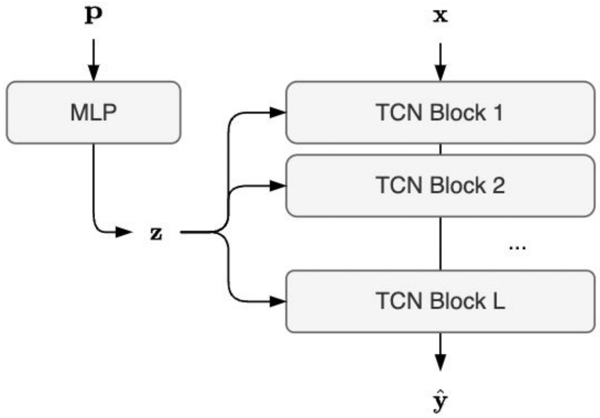
Differentiable Mixing Console

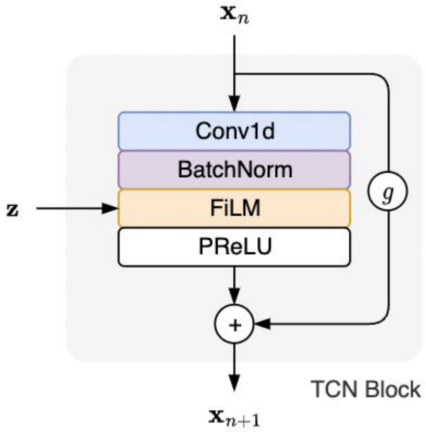
Proxy Networks



Differentiable Mixing Console

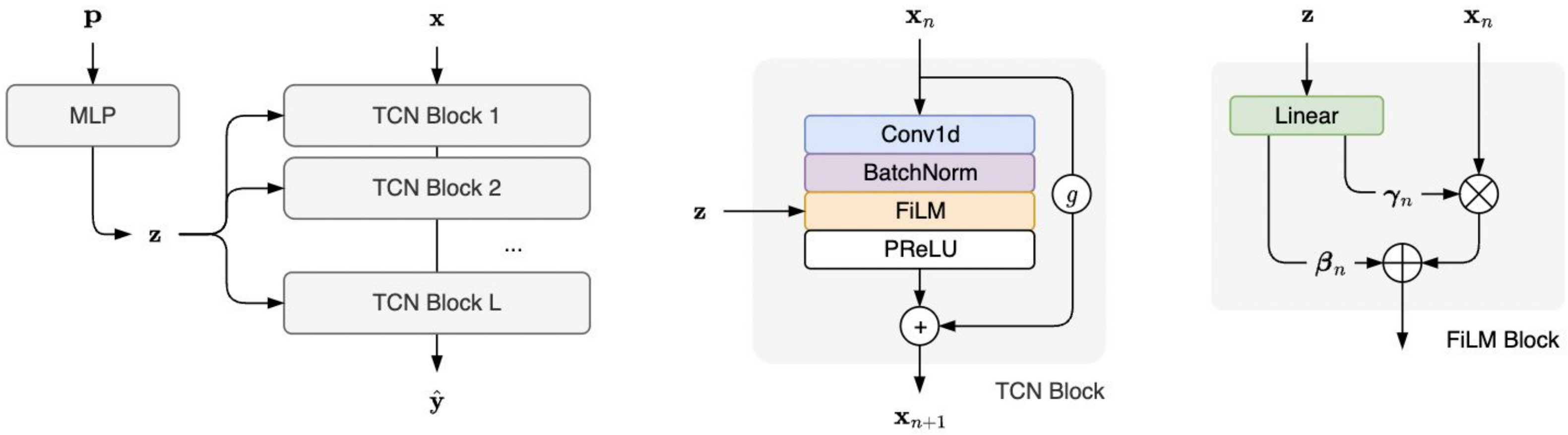
Proxy Networks





Differentiable Mixing Console

Proxy Networks



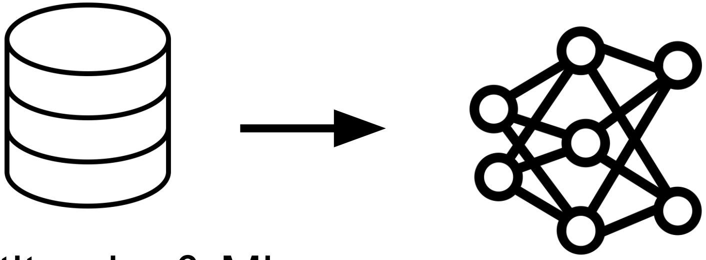
Limitations so far

- Previous methods have not yet achieved the level of professional audio engineers mixes

- It has been hypothesized that the **bottleneck of performance can be resolved with a large enough dataset**

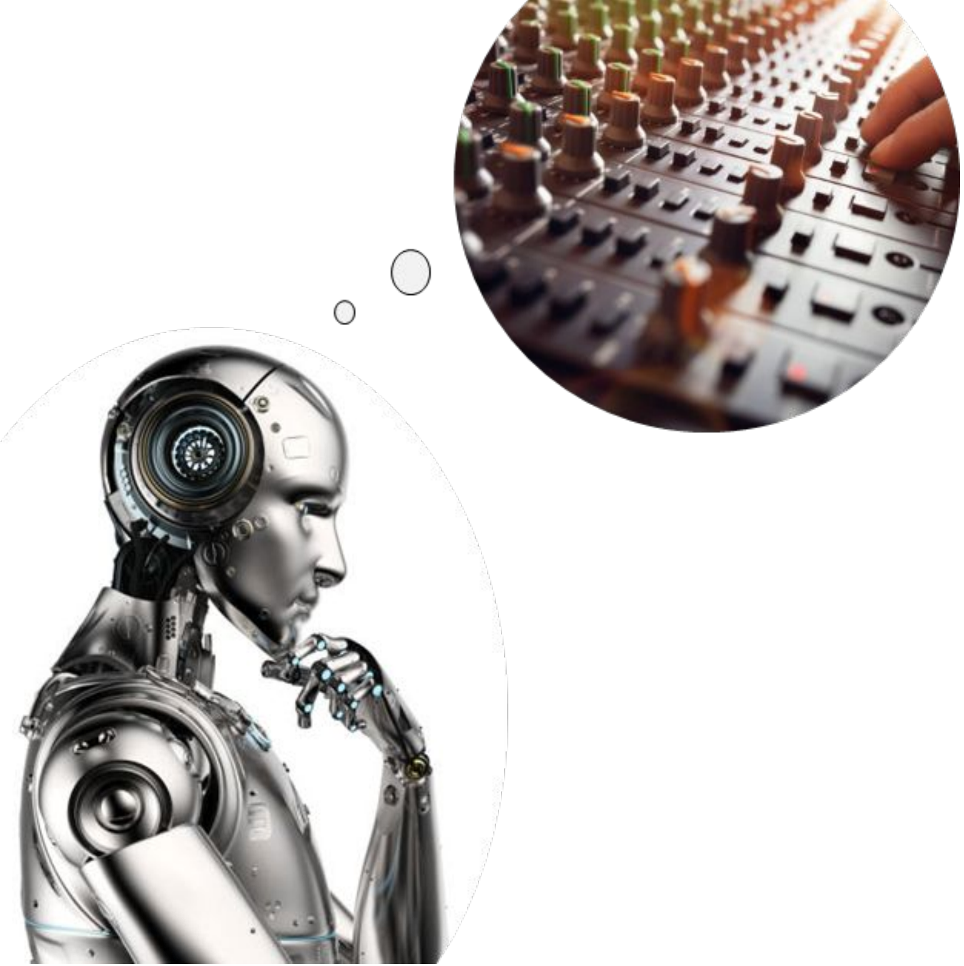
How can we address data bottleneck? (2022)

Challenging



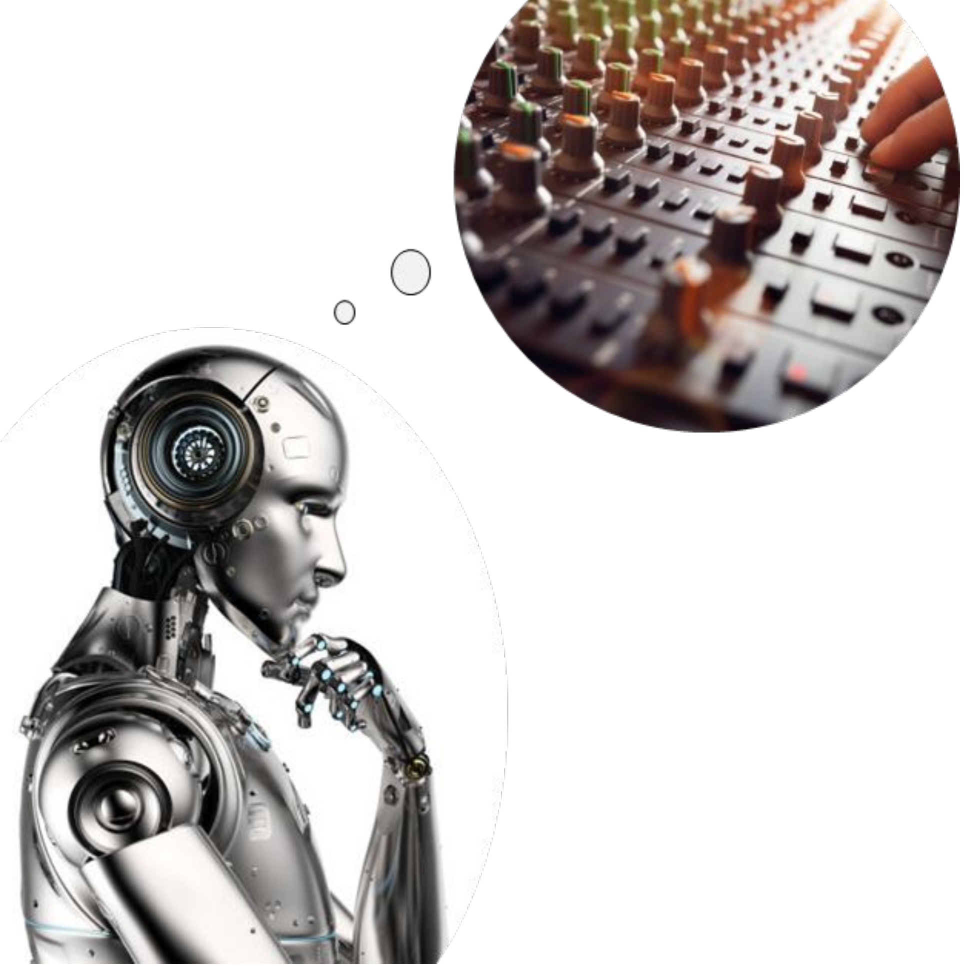
Dry multitracks & Mixes

Data driven approaches need data,
however, ***collecting dry data is difficult***



Research Question

- ***Can we use wet multitrack music data*** *and* ***repurpose it*** *to train deep learning models that perform automatic music mixing?*



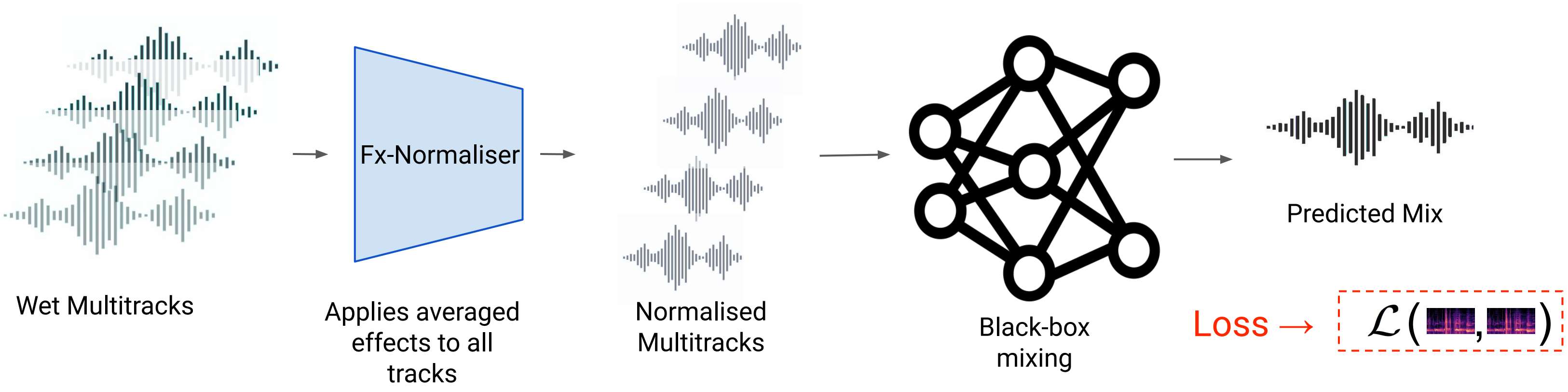
How ?

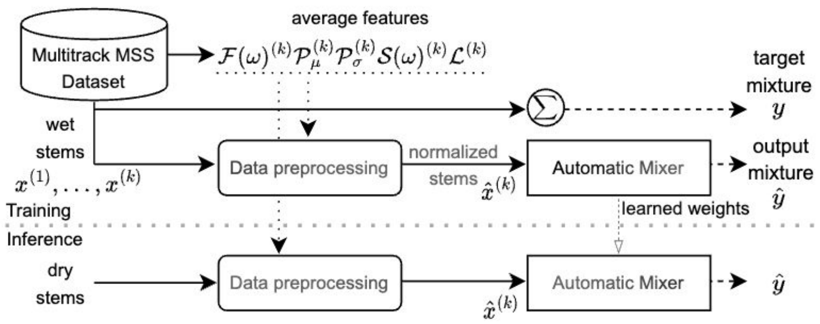
➢ ***Wet multitracks already contain the desired mixing effects****, which are what the networks need to learn*

🤔

Fx Normalization !

Automatic music mixing with deep learning and out-of-domain data



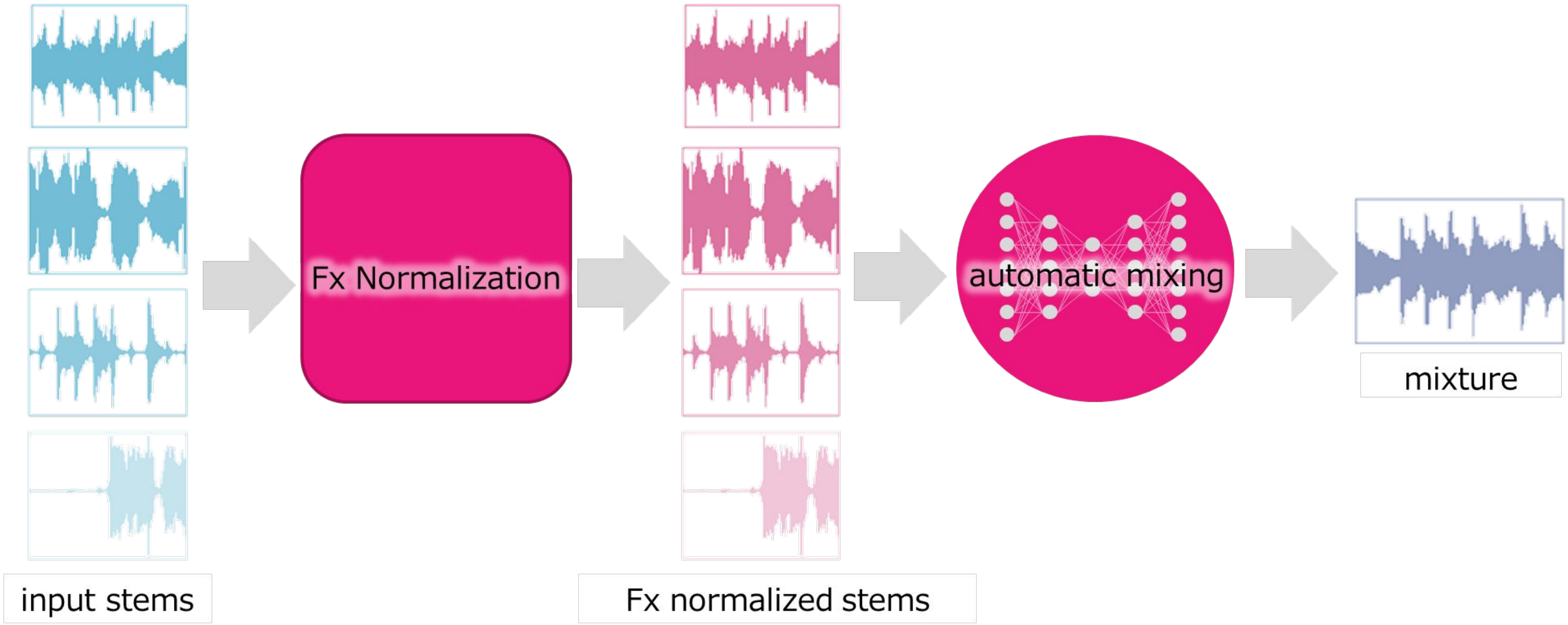


- Pros: uses of wet/processed stems to train, creates possibility for using extensive source separation datasets with wet stems
- Limitations: lacks interpretability and controllability, works for 4 stems

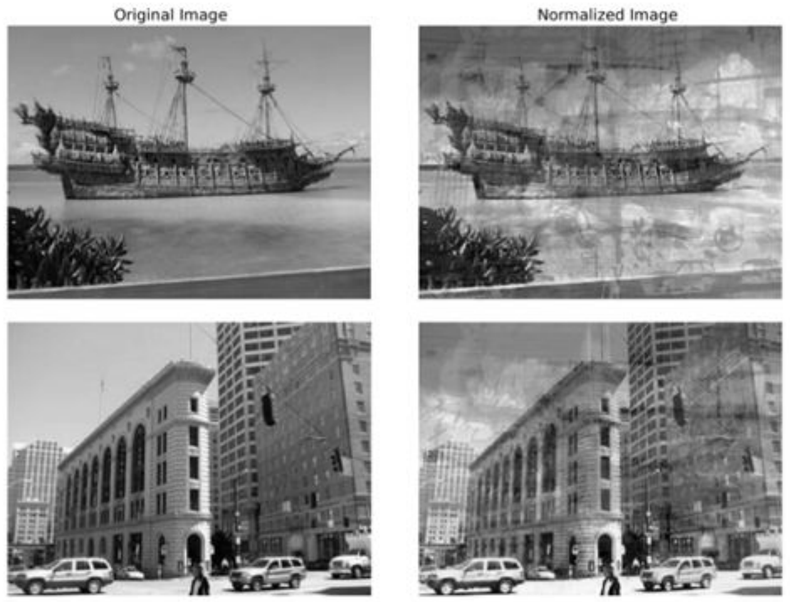
Fx-Normalization

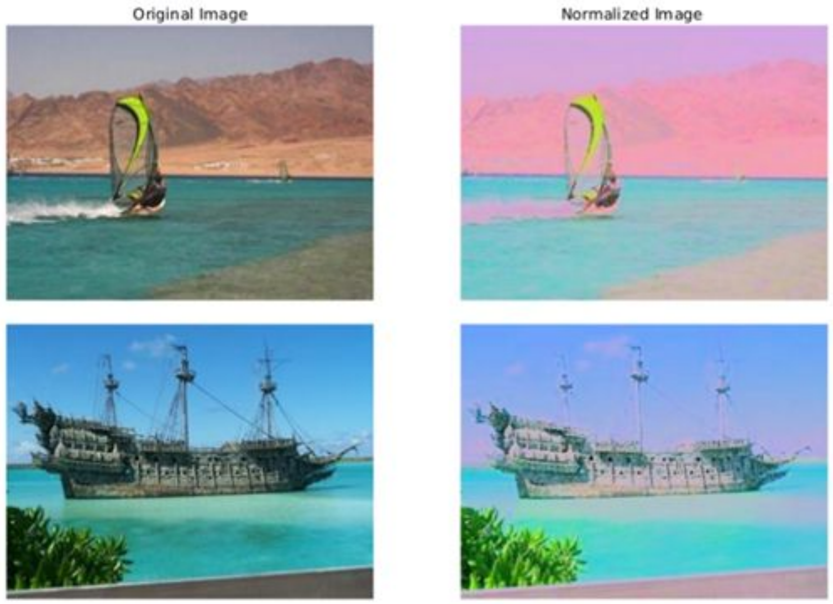
Direct transformation

Fx Normalization



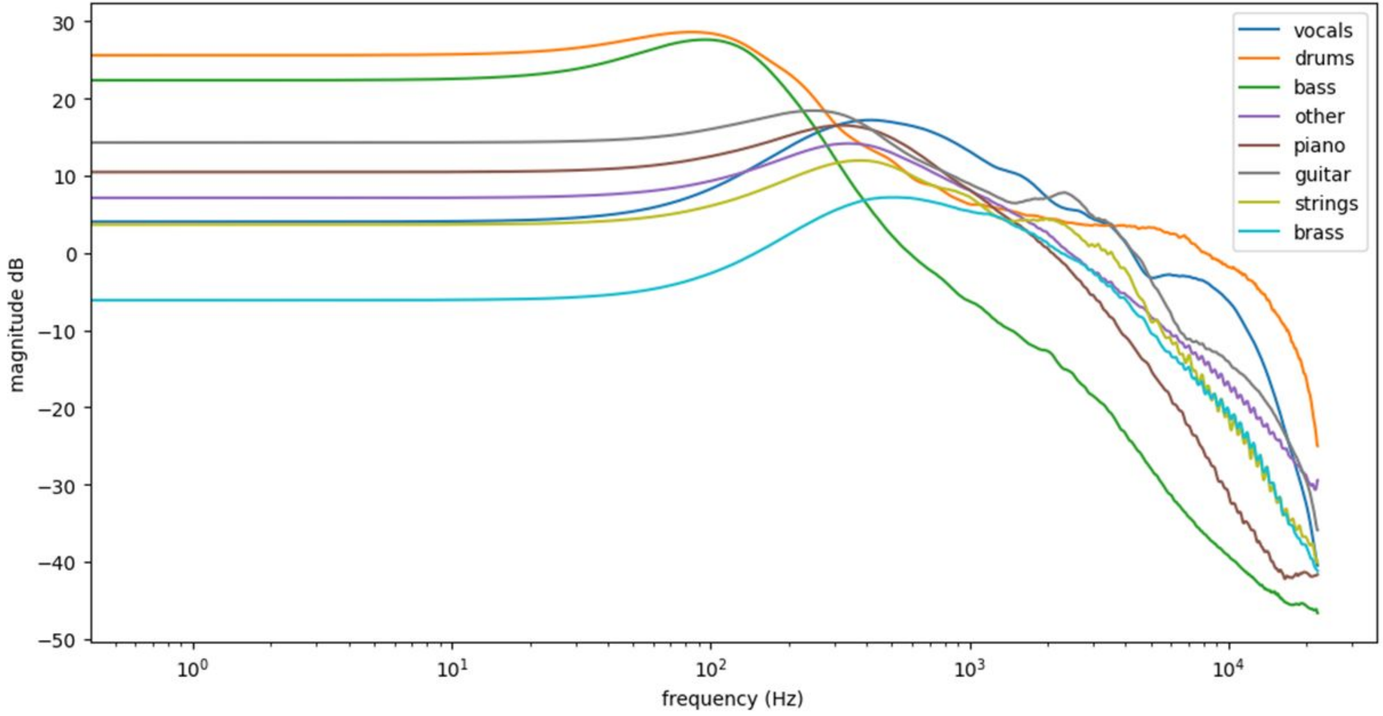
Data Normalization



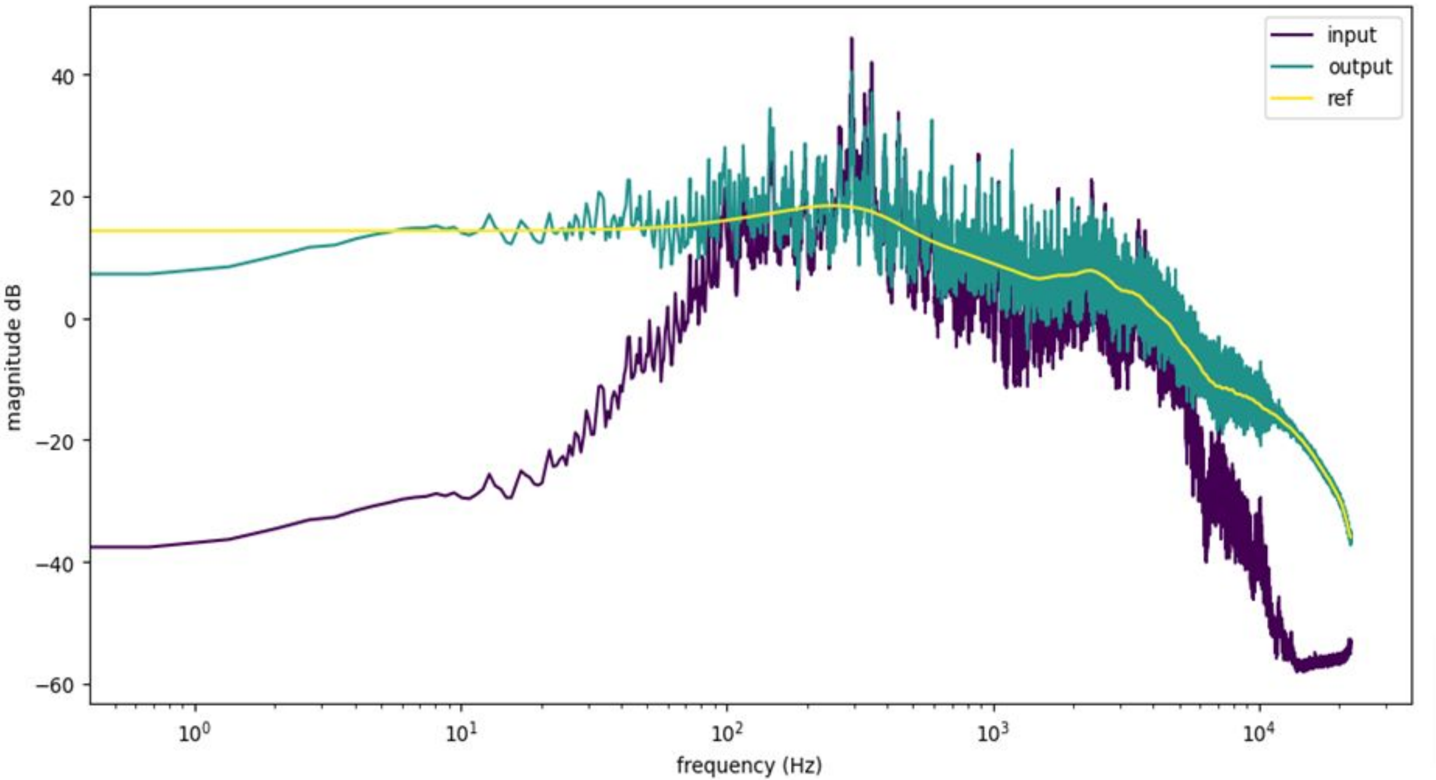


We apply the same to audio effects !

Fx Normalization–EQ average features

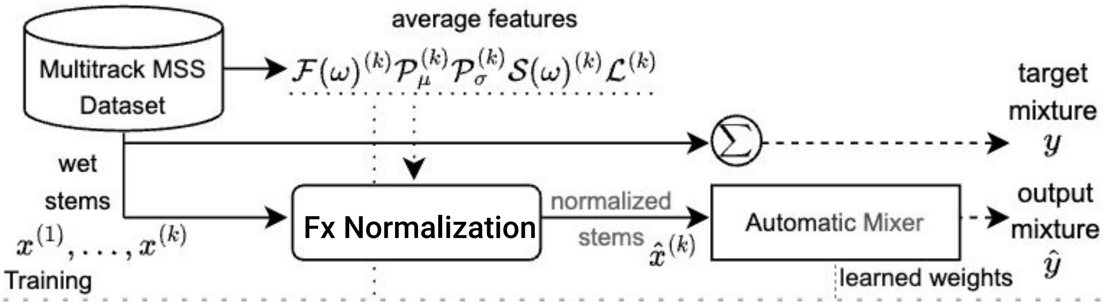


EQ Normalization



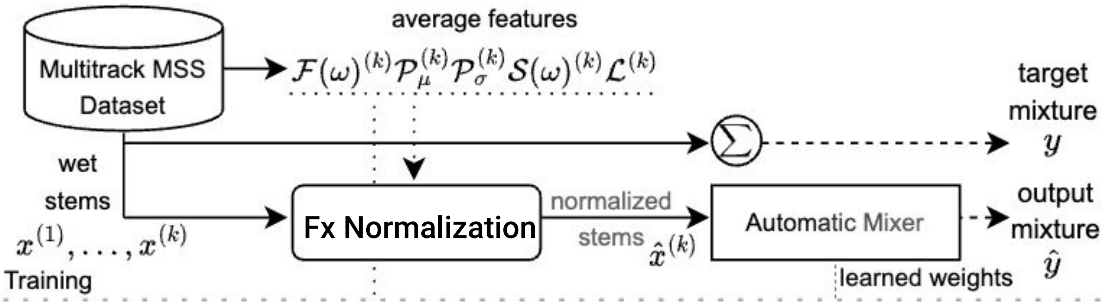
We propose loudness, EQ, panning, compression and reverberation normalization procedures

Method



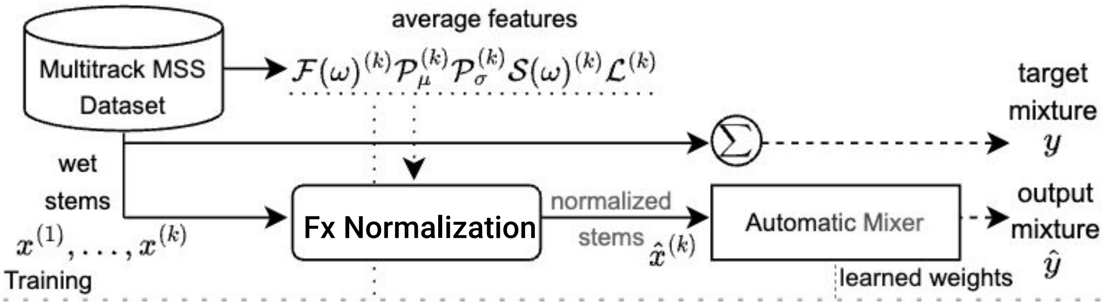
- We use data preprocessing that calculates average features related to audio effects on a music source separation dataset

Method



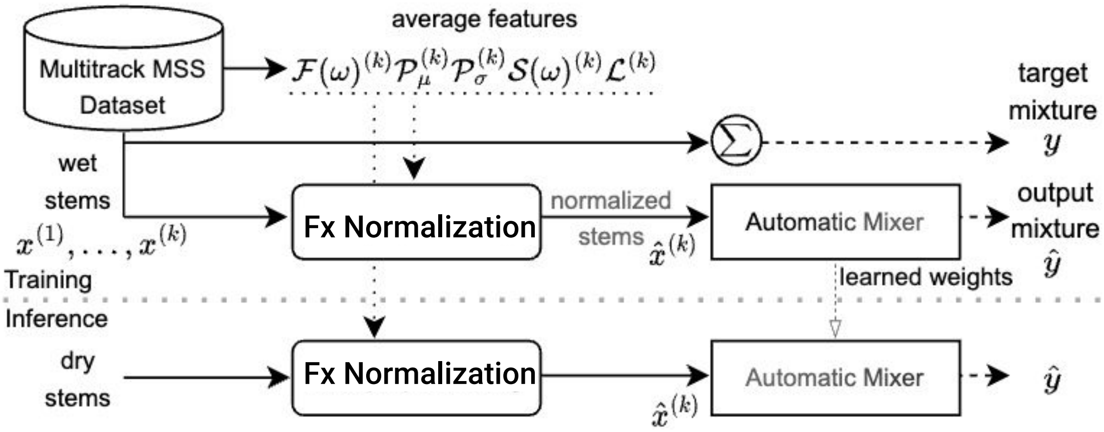
- Based on these features, we "effect-normalize" the wet stems and then train an automatic mixing network

Method



- During training, the model learns how to denormalize the input stems and thus approximate the original mix

Method



- At inference, t**he same preprocessing is applied to dry data**

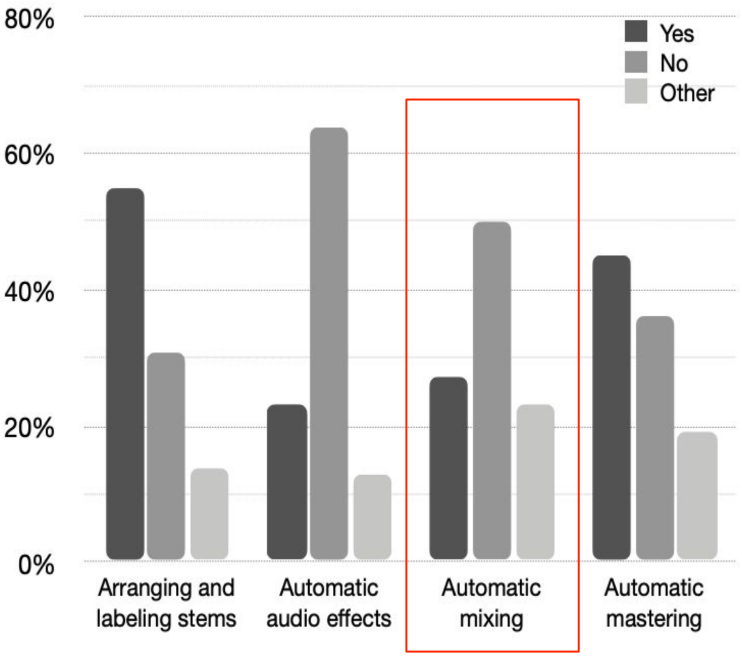
Conclusion

- We developed a method that performs automatic loudness, EQ, panning, compression and reverberation music mixing

- Fx Normalization works !—Our approach leverages on wet data

- Resulting mixes compared to professional mixes scored higher in terms of Clarity and are indistinguishable in terms of Production Value and Excitement

Context-Aware Systems (2023-24)



Why such a huge percentage is saying no?

Audio Engineering Society

Convention Paper

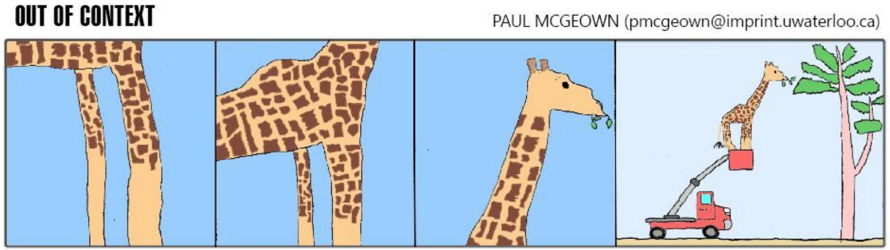
Presented at the 154th Convention
2023 May 13–15, Espoo, Helsinki, Finland

Adoption of AI Technology in the Music Mixing Workflow: An Investigation

Soumya Sai Vanka[1], Maryam Safi[2], Jean-Baptiste Rolland[2], and György Fazekas[1]

[1] *Queen Mary University of London, London, UK*
[2] *Steinberg Media Technologies GmbH, Hamburg, Germany*

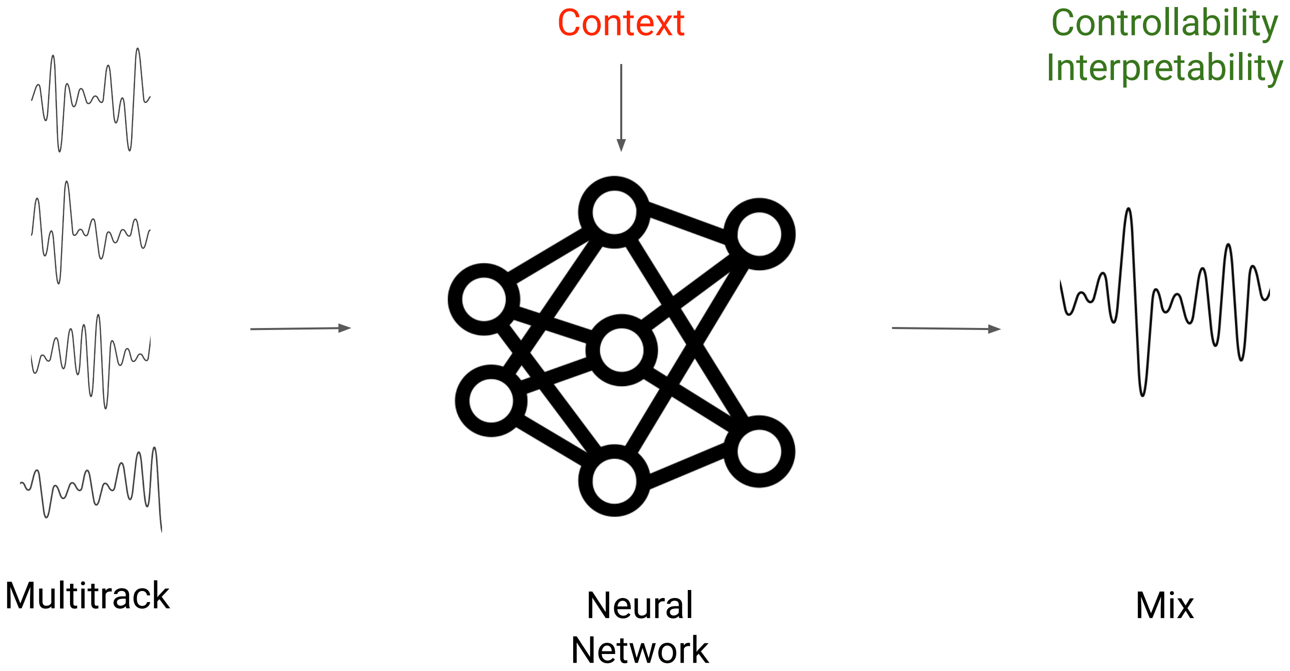
Correspondence should be addressed to Soumya Sai Vanka (s.s.vanka@qmul.ac.uk)



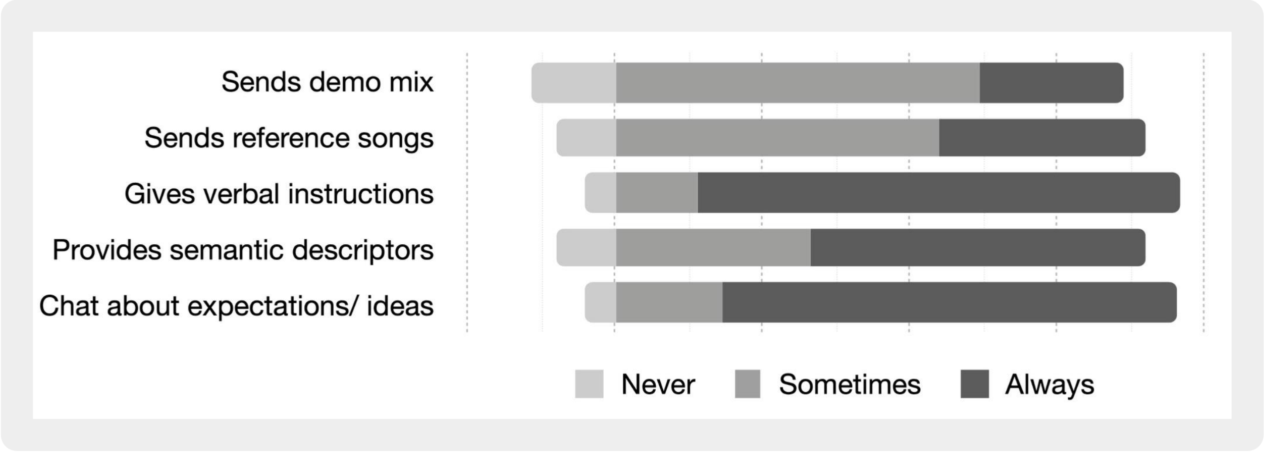
Results are generic and do not understand the context

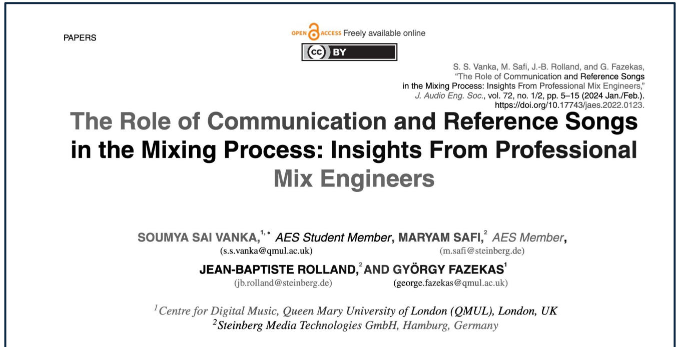
Black box systems: limiting control and interpretability.

What engineers want?



Various media used by artists to
communicate their expectations of the mix




PAPERS

OPEN ACCESS Freely available online

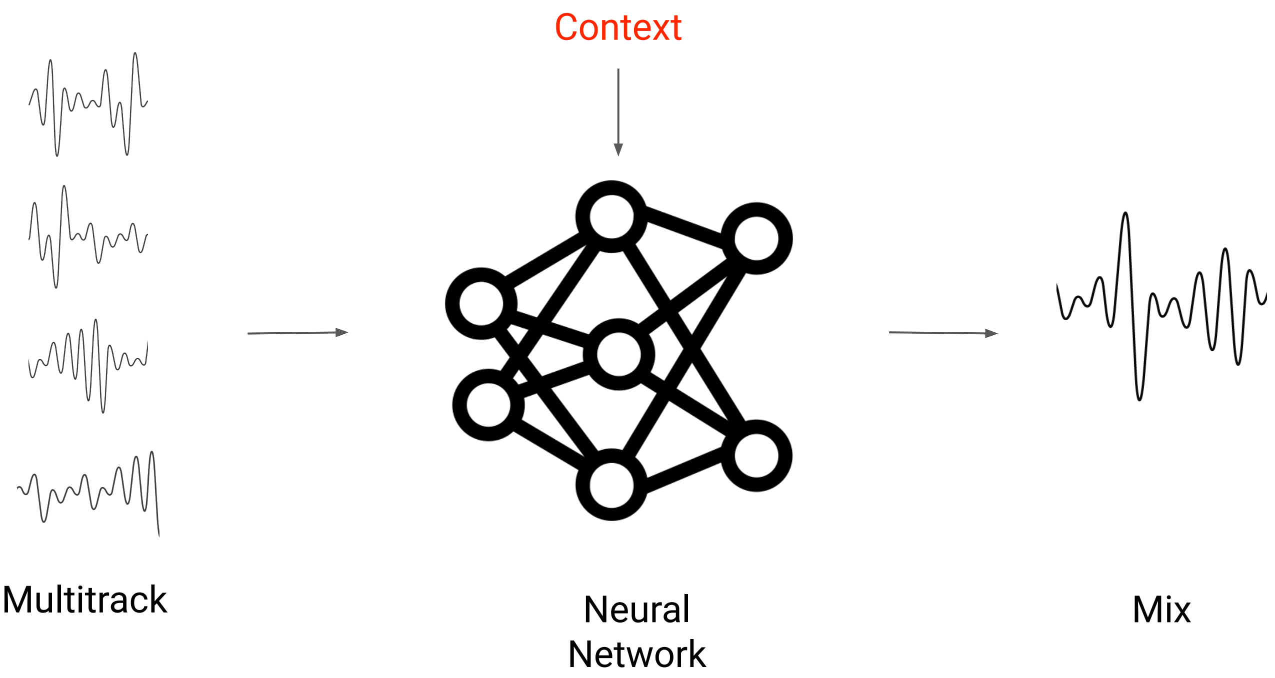
S. S. Vanka, M. Safi, J.-B. Rolland, and G. Fazekas,
"The Role of Communication and Reference Songs
in the Mixing Process: Insights From Professional Mix Engineers,"
J. Audio Eng. Soc., vol. 72, no. 1/2, pp. 5–15 (2024 Jan./Feb.).
https://doi.org/10.17743/jaes.2022.0123.

The Role of Communication and Reference Songs in the Mixing Process: Insights From Professional Mix Engineers

SOUMYA SAI VANKA,[1,*] *AES Student Member*, MARYAM SAFI,[2] *AES Member*,
(s.s.vanka@qmul.ac.uk) (m.safi@steinberg.de)
JEAN-BAPTISTE ROLLAND,[2] AND GYÖRGY FAZEKAS[1]
(jb.rolland@steinberg.de) (george.fazekas@qmul.ac.uk)

[1] *Centre for Digital Music, Queen Mary University of London (QMUL), London, UK*
[2] *Steinberg Media Technologies GmbH, Hamburg, Germany*

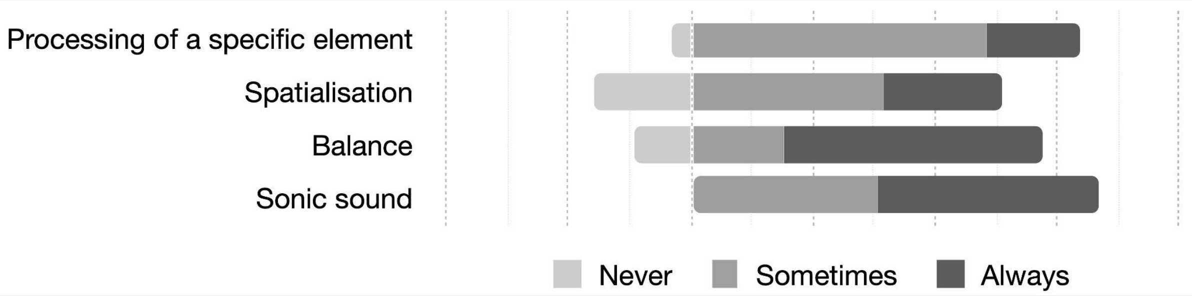
How is context communicated?



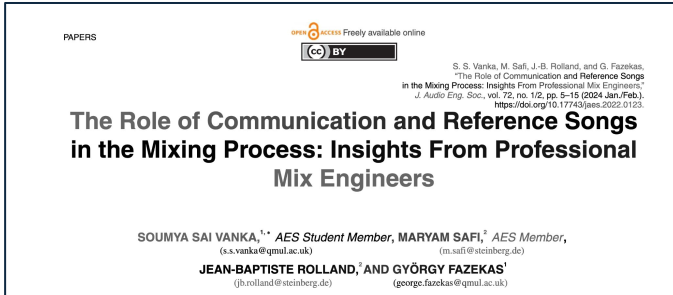
Can we build a system that incorporates context? (2023)

Reference Song

Information derived from Reference Song



Acts as a pointer for the sound of the final mix


PAPERS

OPEN ACCESS Freely available online

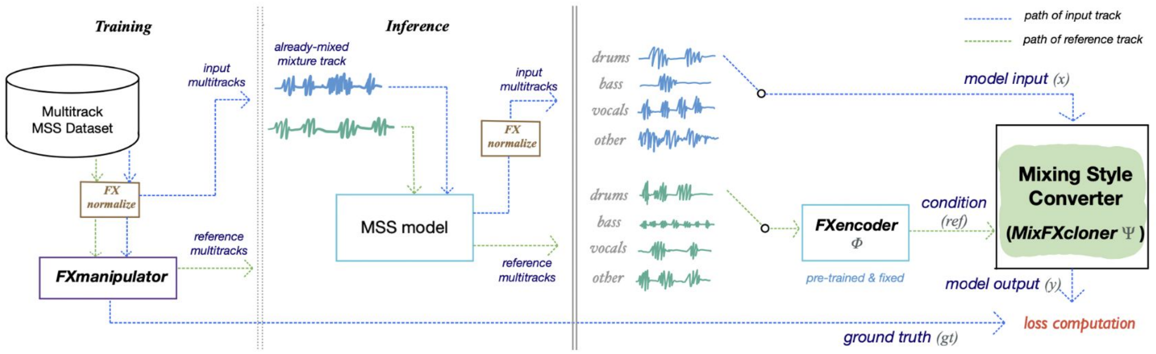
CC BY

S. S. Vanka, M. Safi, J.-B. Rolland, and G. Fazekas, "The Role of Communication and Reference Songs in the Mixing Process: Insights From Professional Mix Engineers," *J. Audio Eng. Soc.*, vol. 72, no. 1/2, pp. 5–15 (2024 Jan./Feb.). https://doi.org/10.17743/jaes.2022.0123.

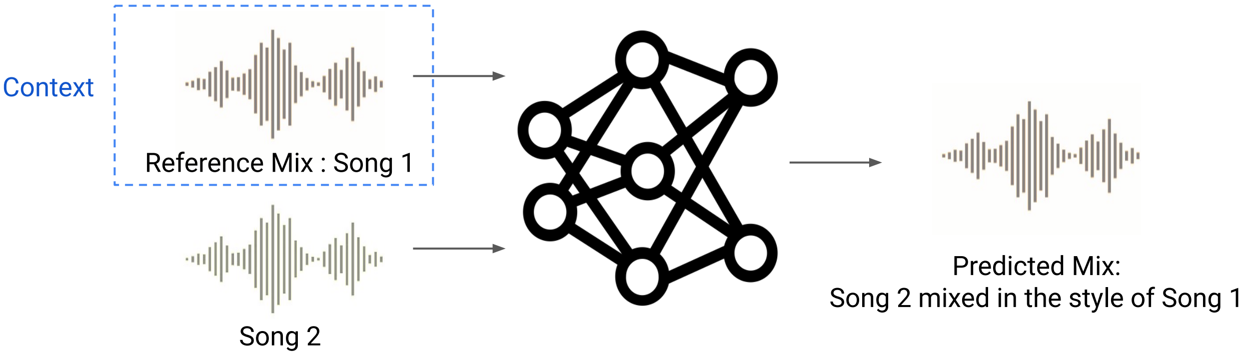
The Role of Communication and Reference Songs in the Mixing Process: Insights From Professional Mix Engineers

SOUMYA SAI VANKA,[1,*] *AES Student Member*, MARYAM SAFI,[2] *AES Member*,
(s.s.vanka@qmul.ac.uk) (m.safi@steinberg.de)
JEAN-BAPTISTE ROLLAND,[2] AND GYÖRGY FAZEKAS[1]
(jb.rolland@steinberg.de) (george.fazekas@qmul.ac.uk)

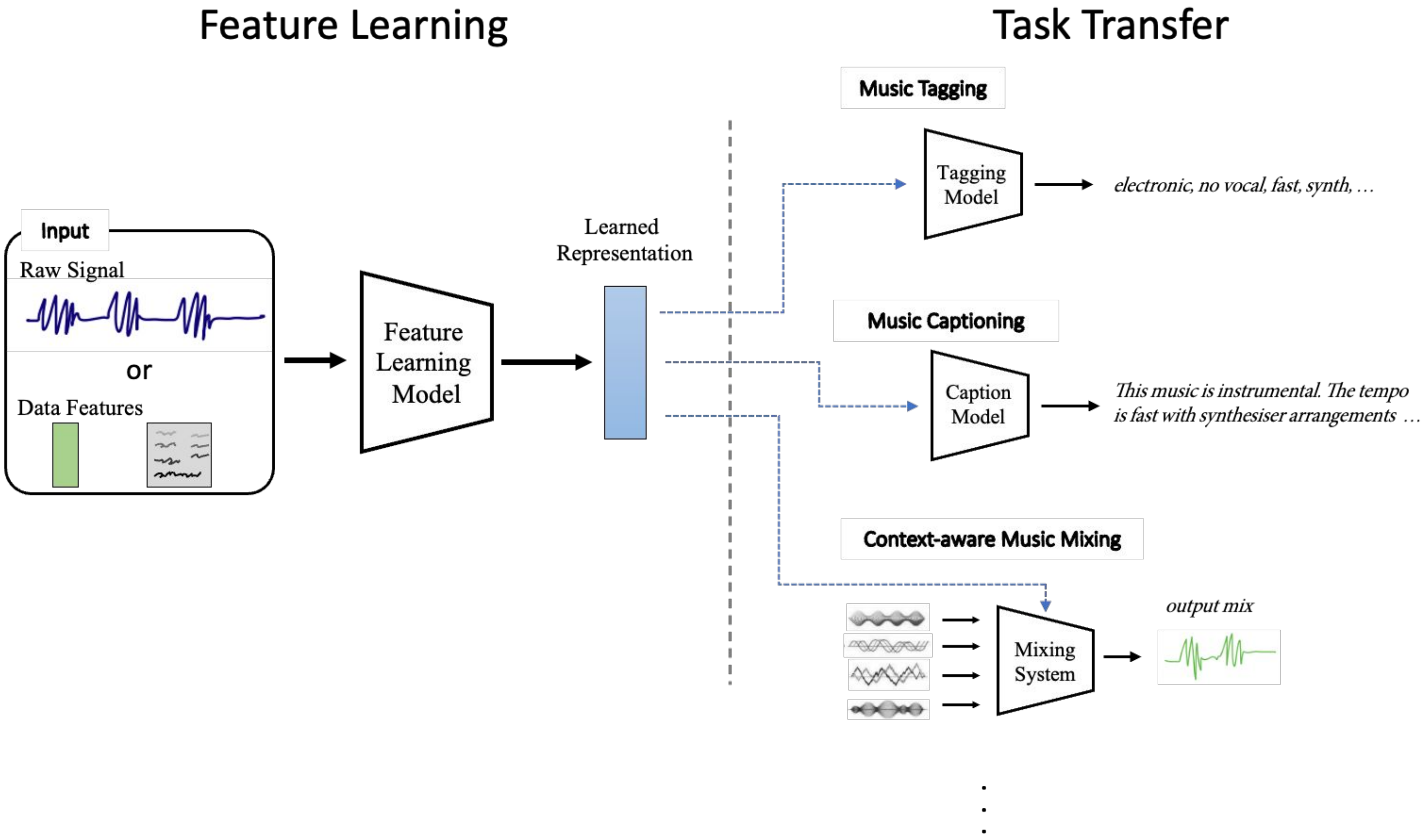
Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects



- Pros: incorporates context through reference
- Limitations: mix to mix transfer, lacks interpretability



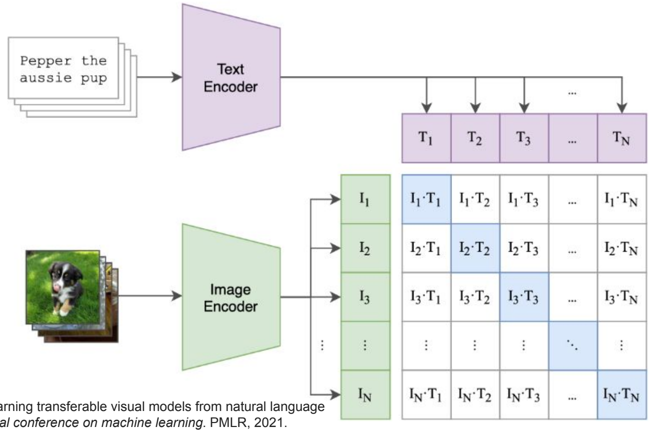
What is Feature Learning?



Contrastive Learning - Recent Applications

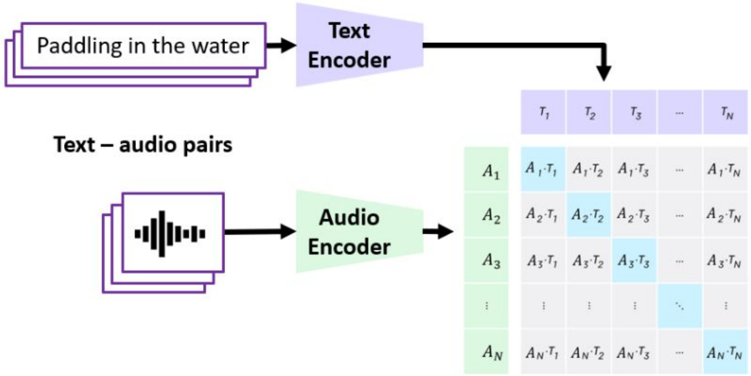
Contrastive Pre-training

Image



Radford, Alec, et al. "Learning transferable visual models from natural language supervision." *International conference on machine learning*. PMLR, 2021.

Audio



Elizalde, Benjamin, et al. "Clap learning audio concepts from natural language supervision." *ICASSP 2023*. IEEE, 2023.

Text Prompt Generative Models

Text-to-Image

DALL·E

Stable Diffusion

Text-to-Audio/Music

Google MusicLM

Stable Audio

AudioLDM

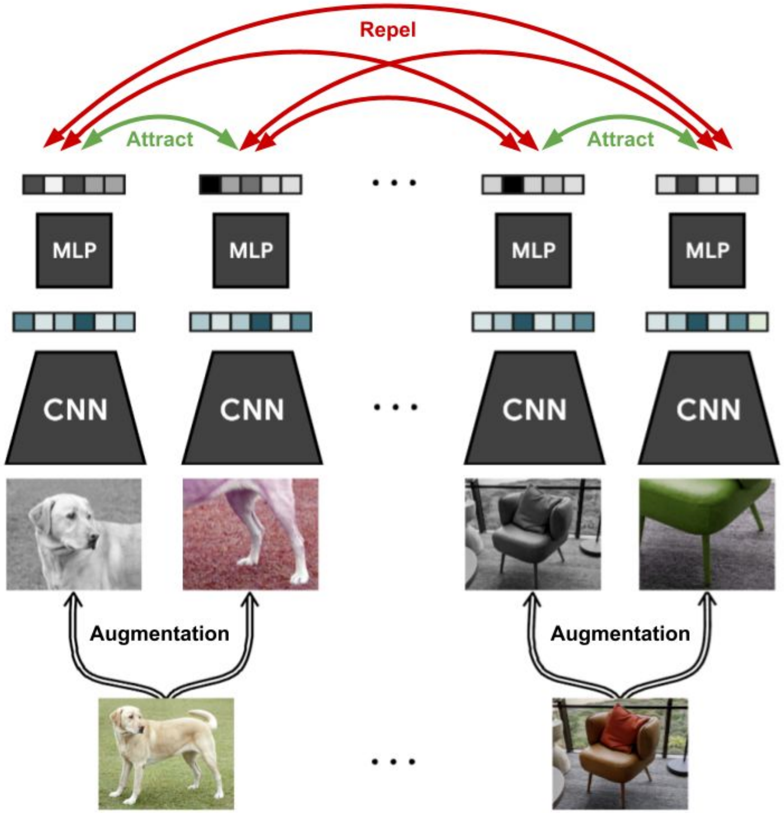
AudioGen

Meta MusicGen AI

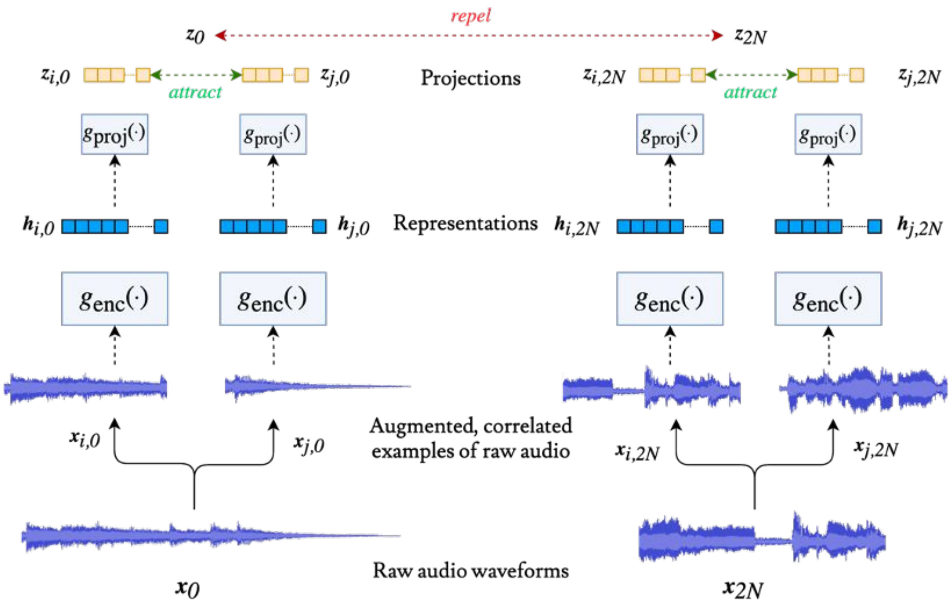
Mubert

Contrastive Learning - Training Method

SimCLR



CLMR



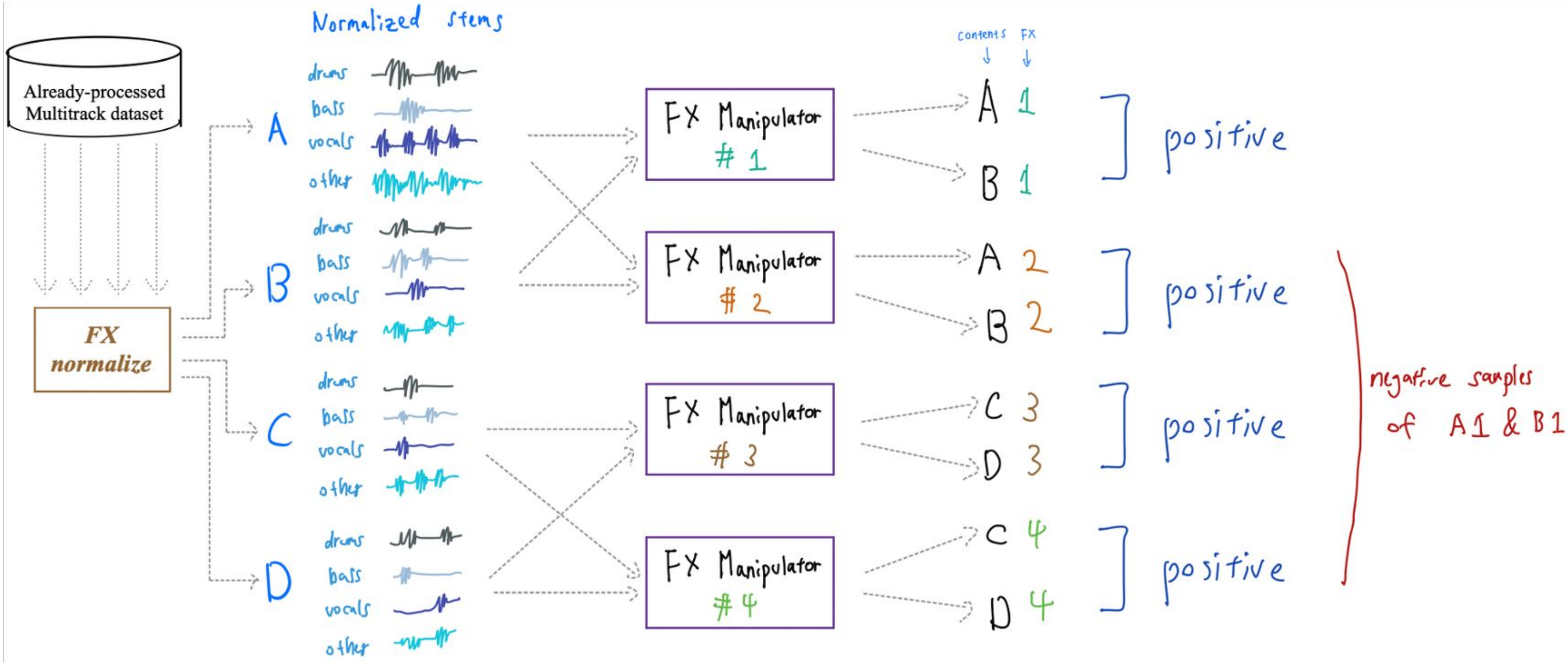
Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." *International conference on machine learning*. PMLR, 2020.

Spijkervet, Janne, and John Ashley Burgoyne. "Contrastive learning of musical representations." *ISMIR* 2021.

Contrastive Learning on Audio Effects

- Utilizes contrastive learning to understand audio effects.
- Objective: to disentangle mixing styles from musical content.
- Apply learnt representation to downstream task such as mixing style transfer.

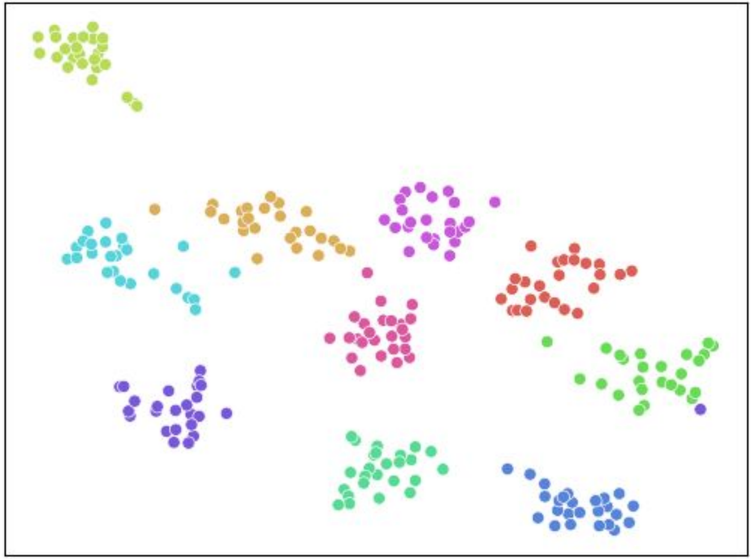
Training Procedure of the FXencoder



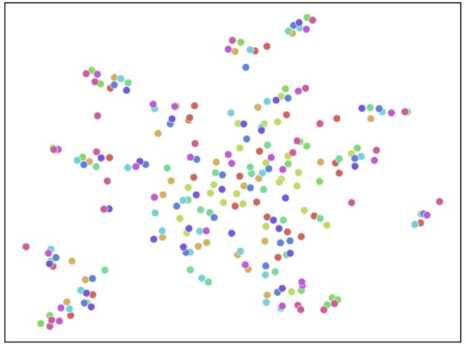
Koo, Junghyun, et al. "Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects." *ICASSP 2023*. IEEE, 2023.

Disentangled Representation

- t-SNE visualization on FXencoder
  - dimensional reduction on feature space
- 10 different random FX manipulation (color) on 25 different songs (point dot)



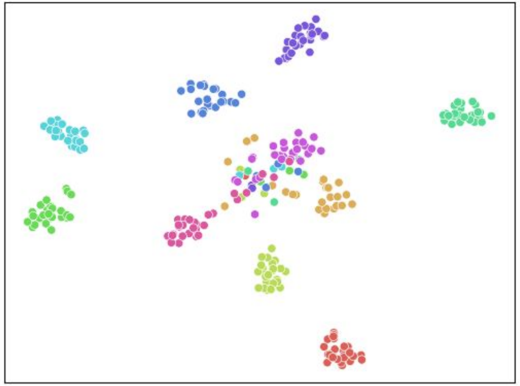
FXencoder



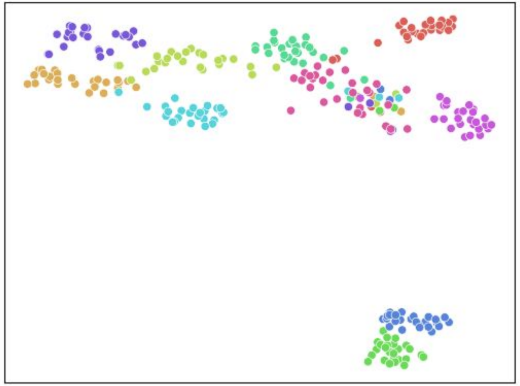
MEE
(model trained with standard approach)

Disentangled Representation - Individual Instrument

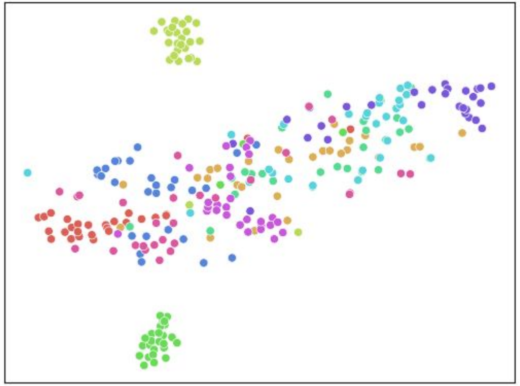
drums



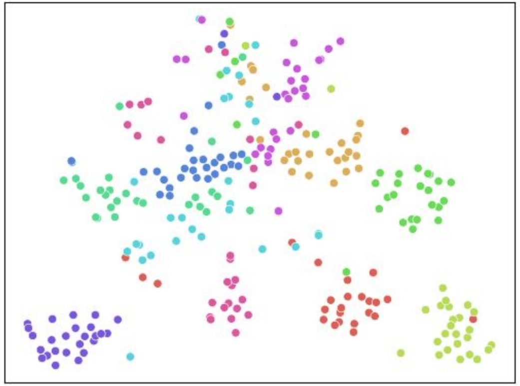
vocals



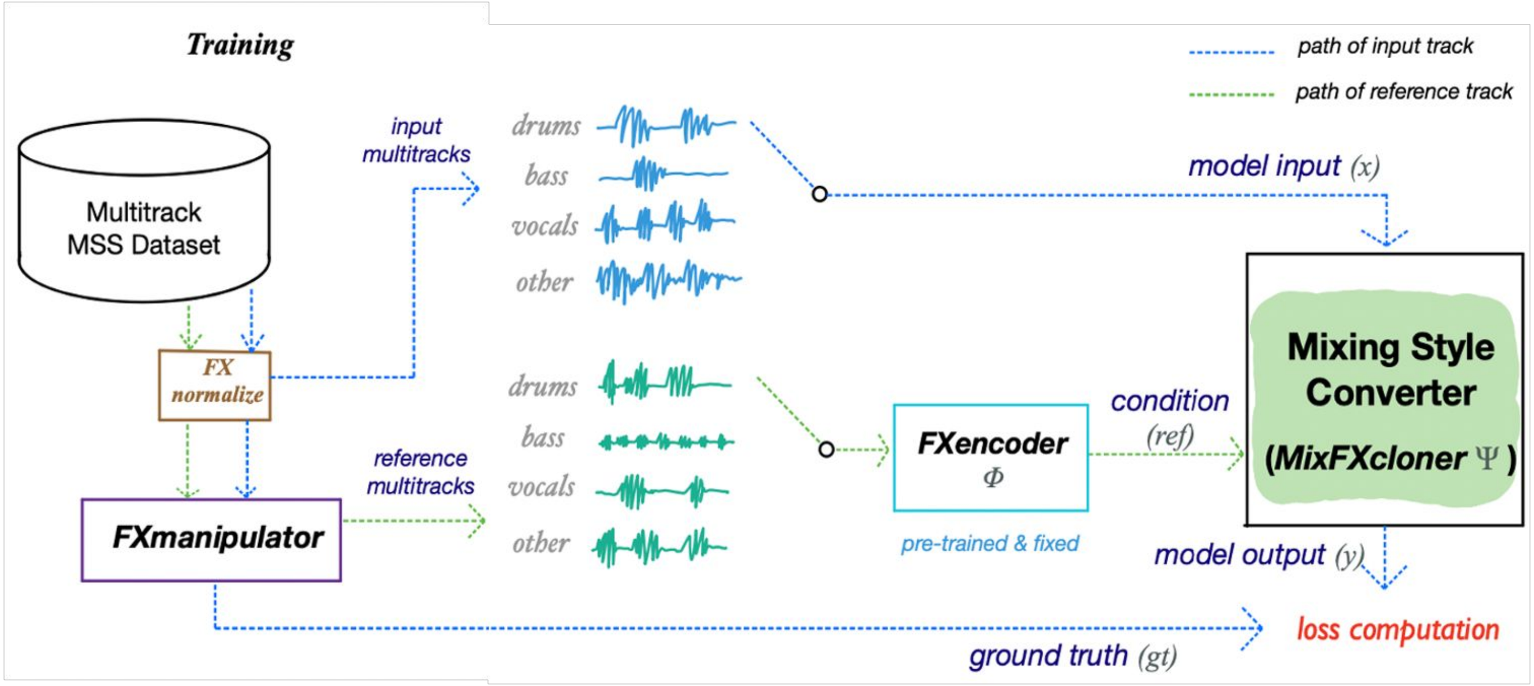
bass



other

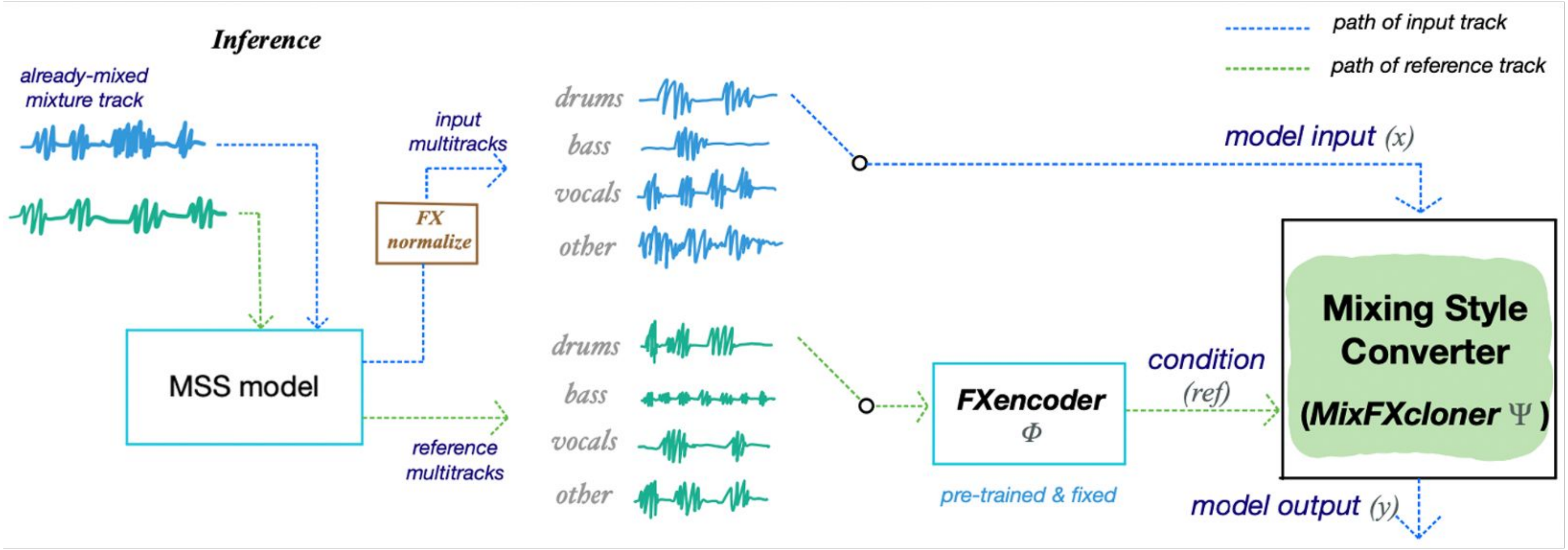


Music Mixing Style Transfer with FXencoder



- Training the mixing style converter is performed by utilizing the representation extracted with already-trained FXencoder

Music Mixing Style Transfer with FXencoder



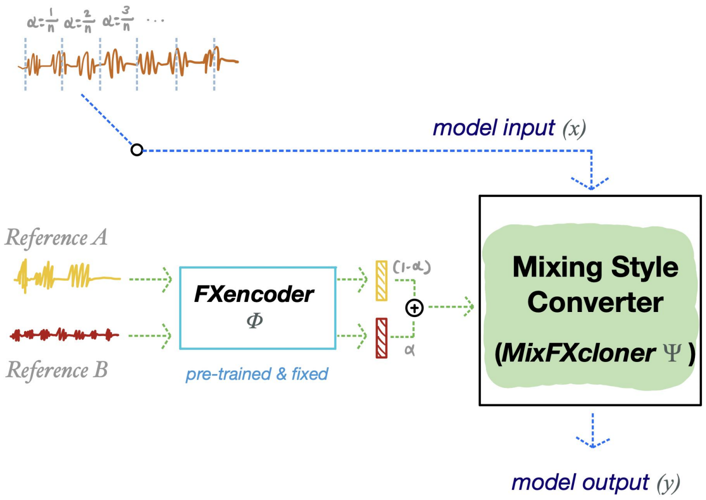
- During inference stage, we can transfer mixing style of mixture-wise inputs using a music source separation (MSS) model

Demo - Mixing Style Transfer

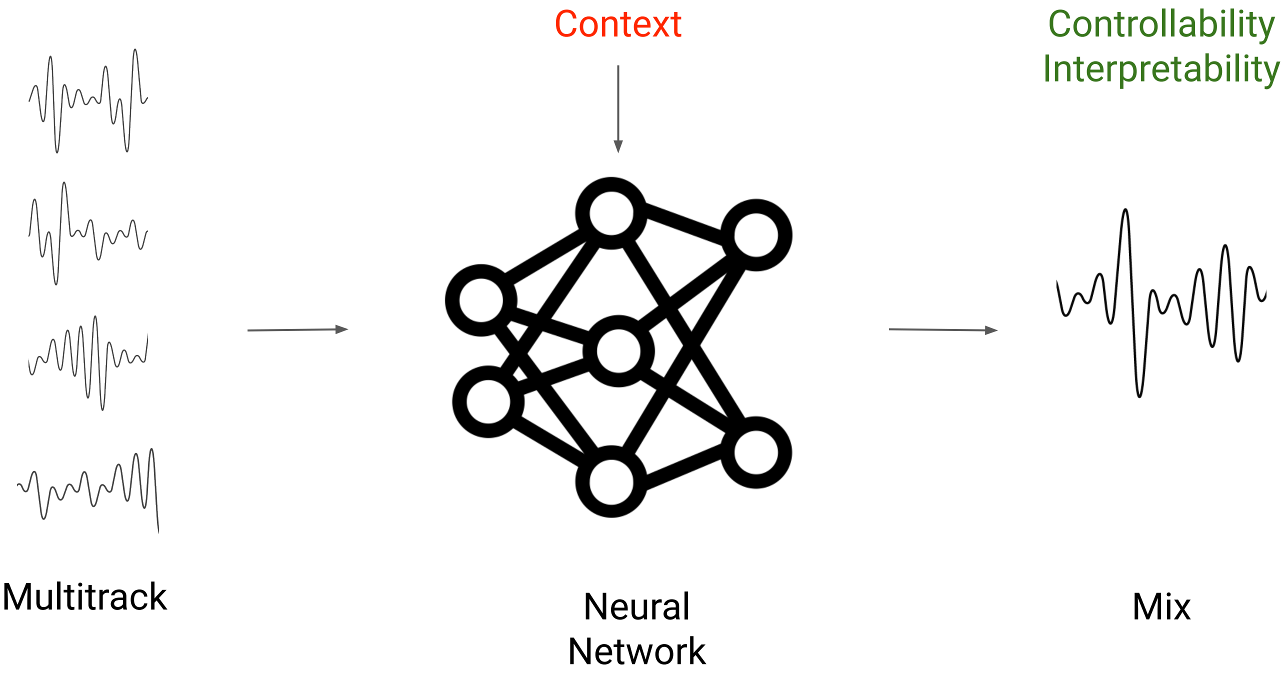
Input Mix:

	Reference A	Reference B
Target Style Mix		
Individual Output		

Interpolated Output



Try with your samples!

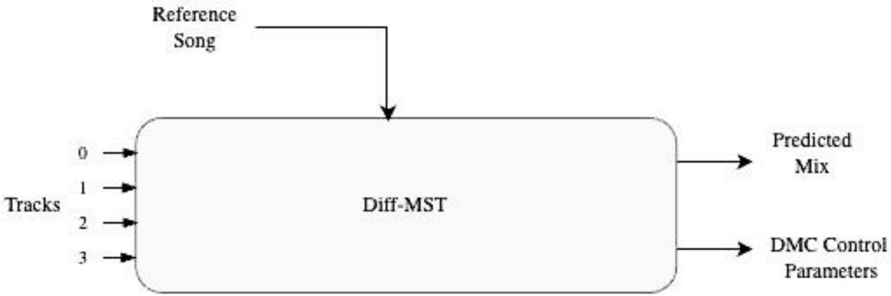


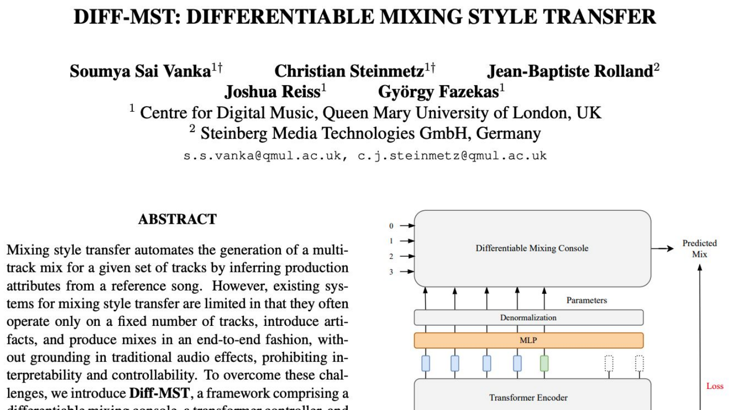
Can we make a context-aware system controllable? (2024)

Diff-MST: Differentiable Mixing Style transfer

Inputs: Tracks (8- 20) and a stereo reference song

Output: Mixing console parameters and predicted mix



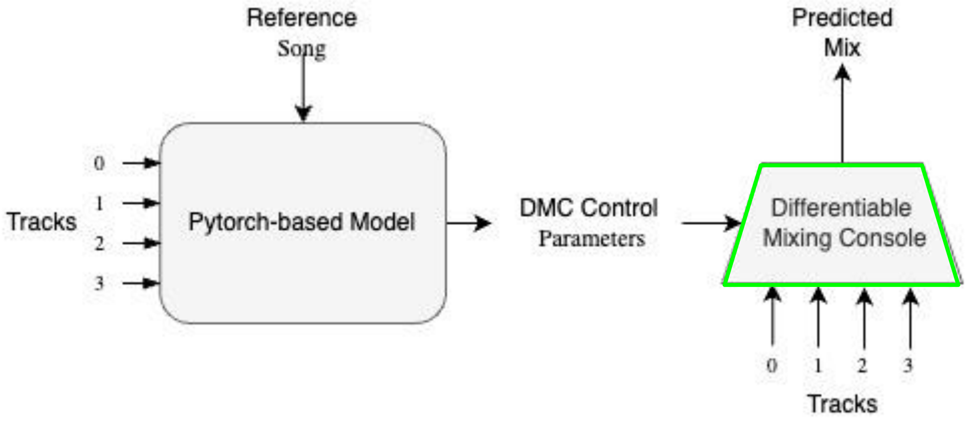


DIFF-MST: DIFFERENTIABLE MIXING STYLE TRANSFER

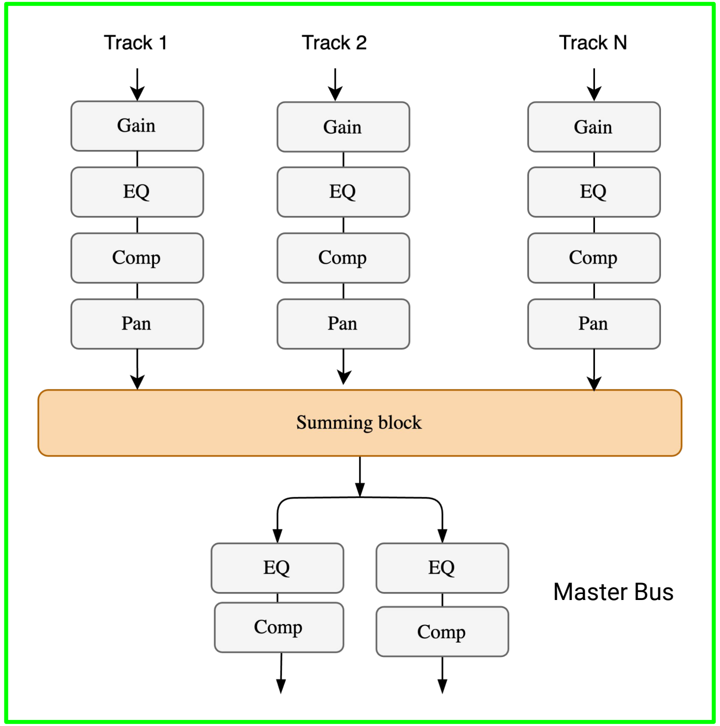
Soumya Sai Vanka[1†] Christian Steinmetz[1†] Jean-Baptiste Rolland[2]
Joshua Reiss[1] György Fazekas[1]
[1] Centre for Digital Music, Queen Mary University of London, UK
[2] Steinberg Media Technologies GmbH, Germany
s.s.vanka@qmul.ac.uk, c.j.steinmetz@qmul.ac.uk

ABSTRACT

Mixing style transfer automates the generation of a multitrack mix for a given set of tracks by inferring production attributes from a reference song. However, existing systems for mixing style transfer are limited in that they often operate only on a fixed number of tracks, introduce artifacts, and produce mixes in an end-to-end fashion, without grounding in traditional audio effects, prohibiting interpretability and controllability. To overcome these challenges, we introduce **Diff-MST**, a framework comprising a

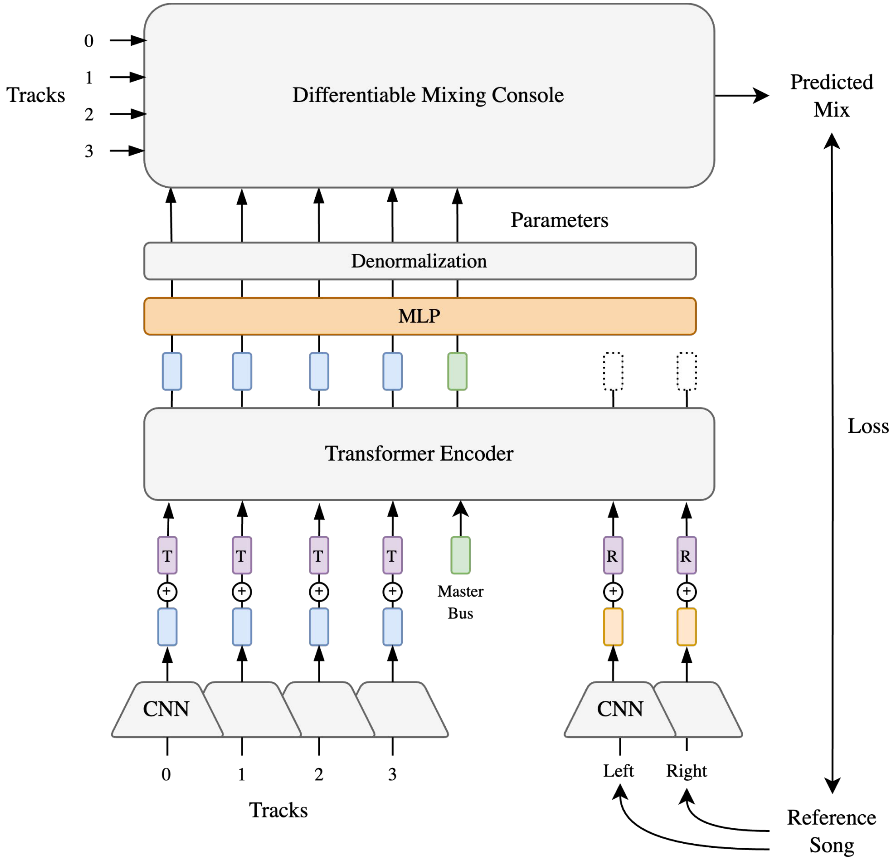


The mixing console is required to be differentiable, so that we can do end-to-end training of the system. Differentiable basically means we can backpropagate and calculate gradients which allows to learn the weights (learn a transformation).



Implemented using

DASP
Differentiable audio signal processors
in PyTorch



https://sai-soum.github.io/projects/diffmst/

Datasets

Multitracks: MedleyDB and Mixing Secrets

- Complete songs with varied number of channels and instruments
- Different Genres
- Medley (7.2hrs) + Mixing Secrets (~50hrs)

Reference Songs: MTG Jamendo

- 55k songs in MP3 format
- Different Genres

Losses

- MR-STFT: Multi-resolution STFT loss from `auraloss`
- AF-Loss: handcrafted weighted average of MSE loss of MIR features specific to mix (from literature)
    - Dynamics: Root mean square (RMS) and Crest factor (CF)
    - Spatialisation: Stereo width (SW) and Stereo imbalance (SI)
    - Spectral: Bark spectrum (BS)

$$T_1(\mathbf{x}) = \mathrm{RMS}(\mathbf{x}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2} \quad ; w_1 = 0.1 \tag{3}$$

$$T_2(\mathbf{x}) = \mathrm{CF}(\mathbf{x}) = 20\log_{10}\left(\frac{\max(|x_i|)}{\mathrm{RMS}(\mathbf{x})}\right) \quad ; w_2 = 0.001 \tag{4}$$

$$T_3(\mathbf{x}) = \mathrm{BS}(\mathbf{x}) = \log(\mathbf{FB}\cdot|\mathrm{STFT}(\mathbf{x})| + \epsilon) \quad ; w_3 = 0.1 \tag{5}$$

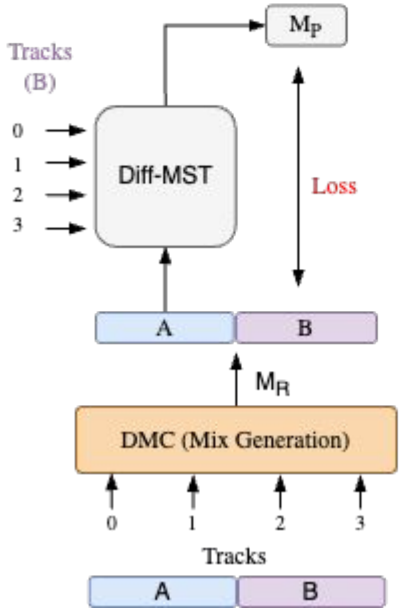
$$T_4(\mathbf{x}) = \mathrm{SW}(\mathbf{x}) = \frac{\frac{1}{N}\sum_{i=1}^{N}(x_{Li} - x_{Ri})^2}{\frac{1}{N}\sum_{i=1}^{N}(x_{Li} + x_{Ri})^2} \quad ; w_4 = 1.0 \tag{6}$$

$$T_5(\mathbf{x}) = \mathrm{SI}(\mathbf{x}) = \frac{\frac{1}{N}\sum_{i=1}^{N} x_{Ri}^2 - \frac{1}{N}\sum_{i=1}^{N} x_{Li}^2}{\frac{1}{N}\sum_{i=1}^{N} x_{Ri}^2 + \frac{1}{N}\sum_{i=1}^{N} x_{Li}^2} \quad ; w_5 = 1.0 \tag{7}$$

$$\boxed{\mathrm{Loss}(\mathbf{M_p}, \mathbf{M_r}) = \frac{1}{2}\sum_{i=1}^{2}\sum_{j=1}^{5} w_j \cdot \mathrm{MSE}\left(\mathrm{T}_j(\mathbf{M}_{\mathbf{p}_i}), \mathrm{T}_j(\mathbf{M}_{\mathbf{r}_i})\right) \tag{8}}$$

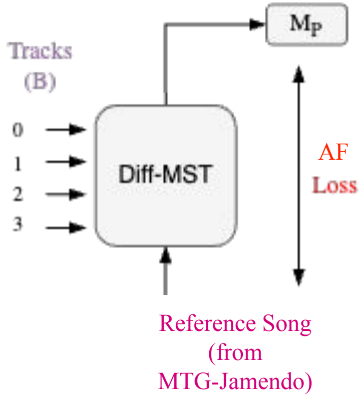
Training Method 1

- A random mix is created using tracks and random DMC parameters
- The random mix is split into equal halves
- One half is used as reference, the other as ground truth
- Losses tested: MR-STFT and MR-STFT plus fine tuning with AF loss
- Pros:
    - Ground truth is available
    - MR-STFT loss can be used
- Drawbacks:
    - During training, model see a lot of diversity
    - Most often really bad sounding mixes
- Performance:
    - MR-STFT only: fails to learn panning and compression
    - MR-STFT plus fine tuning with AF loss : Improves panning performance, not the best yet



Training Method 2
(Best Performance)

- Input:
  - Multitracks from MedleyDB and Cambridge
  - Reference Songs from MTG-Jamendo
- AF-Loss computed between reference and predicted mix
  - Non reference-based loss
- Performance
  - Best performed
  - MIR-based loss forces to learn crucial features of the reference mix.

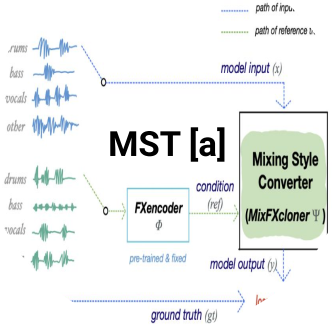


Overview of Diff-MST models

Model	Training Data			Loss (between predicted mix and ground truth)
	Multitrack	Reference	Ground Truth	
Diff-MST-MRSTFT	MedleyDB + Cambridge Multitrack	Random Mix	Unreferenced section of Random Mix	MRSTFT
Diff-MST-MRSTFT+AF	MedleyDB + Cambridge Multitrack	Random Mix	Unreferenced section of Random Mix	MRSTFT and then fine-tuned on AF loss
Diff-MST-AF	MedleyDB + Cambridge Multitrack	Songs from Jamendo	Referenced song from Jamendo	AF loss

Baselines

loudness normalise the tracks to -48.0 dBFS and take the mean among the tracks to generate the mix which is then normalised



model performs a mix-to-mix transformation, we make use of the equal loudness mix of input tracks as the input to be transformed by the model.

[a] Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects, koo. et al, ICASSP 2023

Objective Evaluation

Method	RMS ↓	CF ↓	SW ↓	SI ↓	BS ↓	AF Loss ↓
Equal Loudness	3.11	0.51	3.16	0.21	33.3	33.389
MST [16]	3.15	0.45	4.64	**0.13**	**0.09**	<u>0.185</u>
Diff-MST						
MRSTFT-8	3.63	1.44	1.97	4.29	0.17	0.379
MRSTFT-16	3.40	0.98	1.91	1.99	0.19	0.328
MRSTFT+AF-8	3.12	0.86	1.29	0.76	0.13	0.237
MRSTFT+AF-16	3.15	0.43	**0.89**	2.20	0.11	<u>0.186</u>
AF-16	**2.39**	**0.07**	1.60	0.97	0.13	**0.168**
Human 1	3.02	0.26	2.05	0.46	0.17	0.218
Human 2	3.21	0.14	3.63	2.29	0.11	<u>0.180</u>

Table 1. Average of metrics computed across the same section of three songs from three different genres. RMS is reported in e-04, CF in e-01, SW in e-02, and SI in e-02. We have provided audio examples as supplementary material.

Method	RMS ↓	CF ↓	SW ↓	SI ↓	BS ↓	AF loss ↓	FAD ↓
Equal Loudness	**2.31e-04**	2.11	6.03	1.41	32.7	6.55e+00	17.6
MST [16]	4.07e-04	1.72	5.84	0.89	0.31	<u>7.85e-02</u>	17.9
Diff-MST							
MRSTFT-8	3.08e+06	3.91	4.55	3.38	7.06	6.15e+05	51.3
MRSTFT-16	2.23e+03	4.07	5.00	1.97	1.81	4.47e+02	65.9
MRSTFT+AF-8	2.00e+05	1.79	4.58	2.86	6.89	4.00e+04	48.3
MRSTFT+AF-16	2.46e+00	1.14	**4.29**	3.44	0.92	6.92e-01	51.1
AF-16	4.24e-04	**0.67**	4.78	**0.22**	**0.11**	**3.26e-02**	**15.1**

Table 2. Average of metrics using unseen tracks from Cambridge dataset and mixes from MUSDB18 [25]. CF in e-02, SW in e-02, SI in e-02.

*-8 and *-16 are trained on maximum 8 and 16 tracks, respectively

Conclusions

- Improved metrics observed with training on more tracks.
- AF loss outperforms MRSTFT loss, especially in enhancing spatialization and dynamics.
    - Diff-MST-MRSTFT models underperform due to unrealistic training data; fine-tuning with AF loss improves results
- Training on real-world songs enhances performance, emphasizing the need for high-quality data.

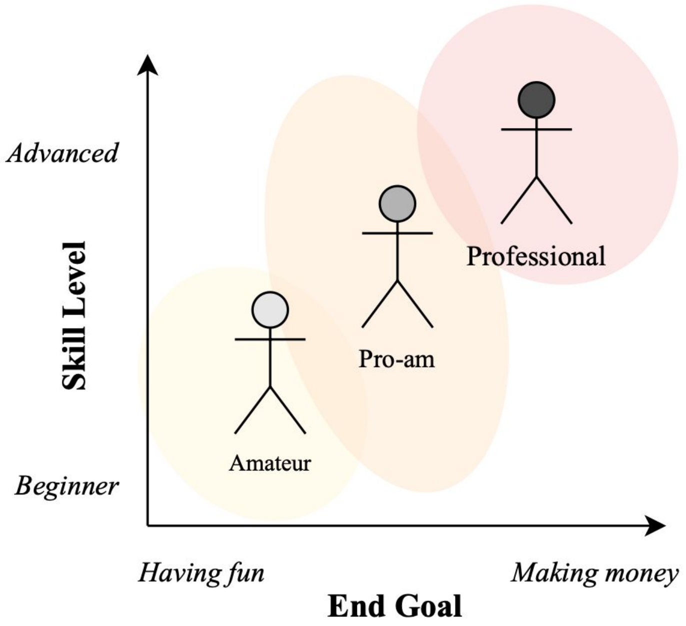
Limitations

- Challenges with increased input tracks and lack of a reverb module.
- Decline in performance for longer songs due to sparse embeddings.
- Human mixes capture creative elements that our system metrics may not fully assess.
- FAD metric may miss nuances like frequency masking and balance.
- System struggles with fully modeling mixing context but uses a reference input as a proxy.
- Currently limited to static mixing configurations, unlike the dynamic adjustments in real-world mixing.
- No subjective evaluation :/

Summary

Model	System Type	Controllability	Context	Interpretability	Input Taxonomy
Wave-U-Net for drum mixing	Direct transformation	No	No	No	Drums only
Mixing with neural mixing console	Parameter estimation	Yes	No	Yes	Multitrack, permutation and number of tracks invariant
Mixing with out-of-domain data	Direct transformation	No	No	No	Wet stems, limited on number of tracks
Mixing style transfer	Direct transformation	No	Yes (reference song)	Yes	Mix and style reference mix
Diff-MST	Parameter estimation	Yes	Yes (reference song)	Yes	Raw tracks and style reference mix

User-Centric Design



User of the tools

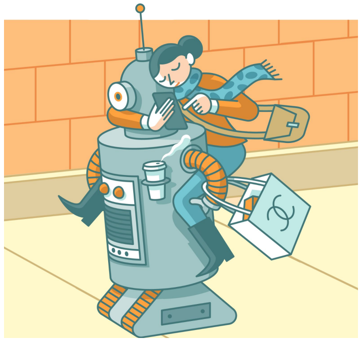
(not accurate but gives a sense of where each category of user fits)

Amateurs

Limited knowledge of music mixing

Primarily create and **compose music**

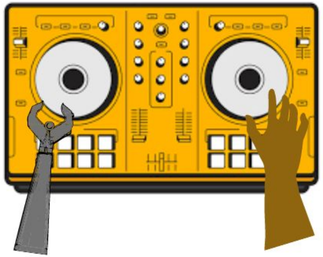
Mixing: biggest **hurdle** to releasing music



- Expectations: **highly autonomous** mixing system
- Not expecting high quality output
- Using AI mixing systems: **produce a decent mix with minimal effort**
- **Positively embracing** the emerging technology

Pro-Ams

Higher technical skills than amateurs but less experience than professionals.



- **Use cases**:
  - **Improve their skills** and work towards becoming professionals
  - quickly **achieve a certain sound** or style in their mixes.
- Aware of the **limitations** of technology - willing to put tools to best use.
- **Cautiously optimistic** about the future of these tools.

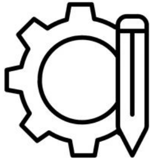
Professionals [Positive]

save time on repetitive tasks

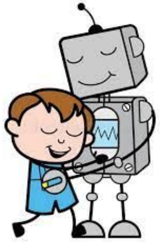
experiment with new sounds

tasks like filtering, peak detection, pitch detection, mastering, equalization, and sound enhancement

Accurate and precise



customizable



assistive and co-creative technologies that enable collaboration

Use Case

Expectations

Professionals [Negative]

Cannot fully replace the human touch and creativity required in the process.

Traditional methods of mixing are superior - learning by trial and error best way to master mixing.

Leads to a loss of control and precision in the final product

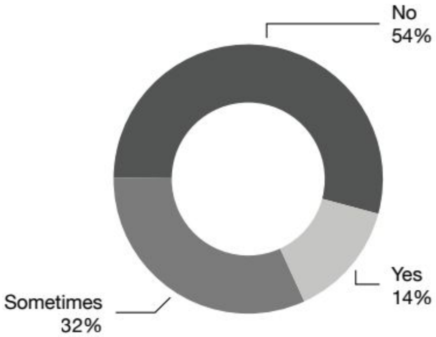


Fig. 2: Responses to the use of AI-powered tools in mixing workflow as reported by pro-ams and pros.

Smart tools\User	**Amateurs**	**Pro-ams**	**Professionals**
Use-case	Create decent mix	Learning and exploratory tool	Automate repetitive and time-consuming tasks
	As a learning tool	To find a starting point	Co-creation and assistance
		Automate technical tasks	To find a starting point/direction for mix
		Creativity and inspiration	Creativity and inspiration
Expectations	Autonomous with less control	Advanced and more control	Highly advanced and wide range of control option
	Cost-effective	Accurate and precise	Accurate and precise
	Easy to use	Assistive	Assistive
		Cost effective	Easy integration in current workflow
		Easy integration into current workflow	Context-aware
Sentiment	Positive	Cautiously positive	Mixed

Table 1: Comparison of use-case, expectations, and sentiment amongst different categories of users of AI technology in mixing workflows

Seamless Integration

Amateurs: may not be familiar or well-versed with DAW

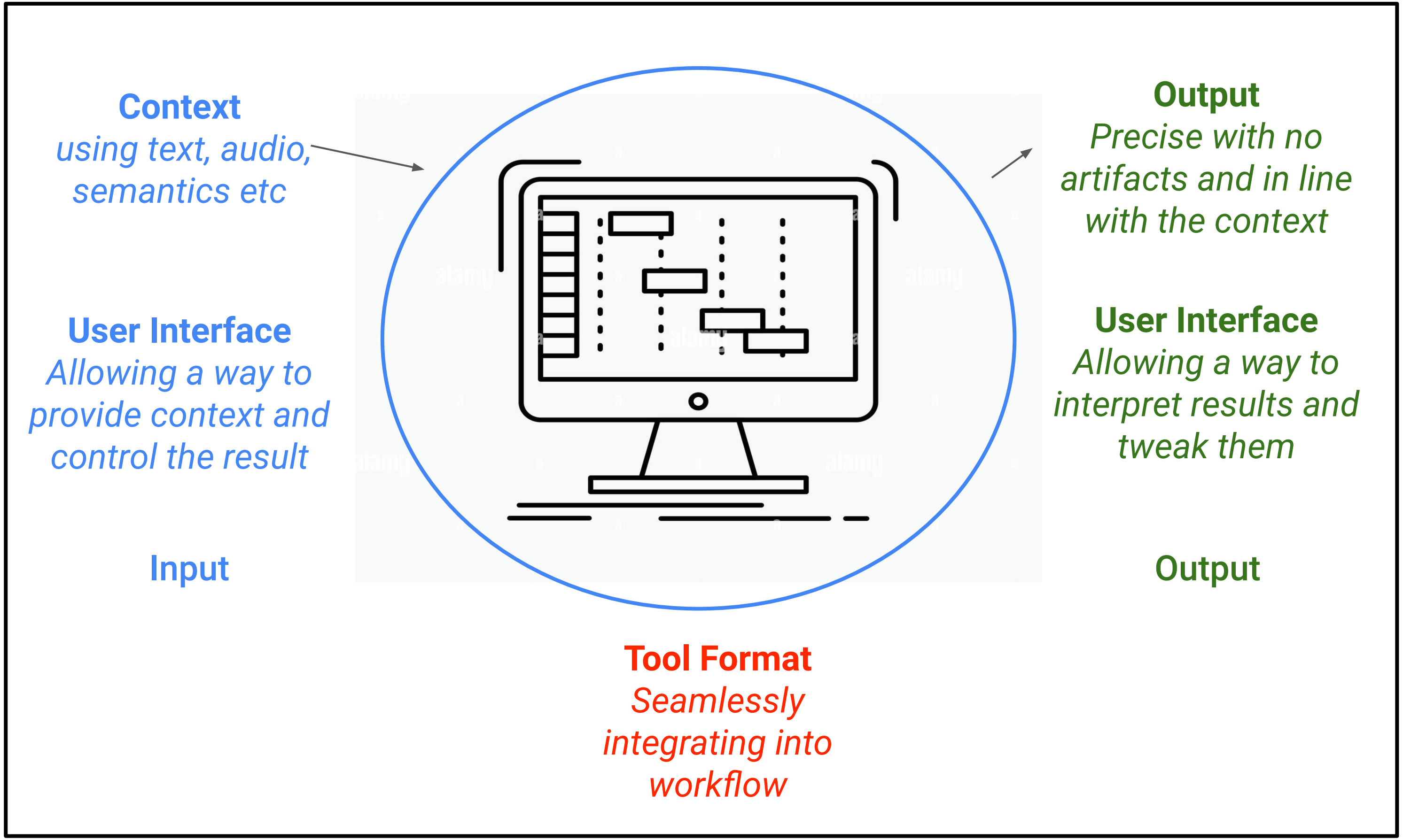
- Autonomous mixing tools hosted on web
- Tools with simpler interface and less options to control

Pro-ams: may have established workflows but are open and curious to try new tech

- Web-based interfaces or tools that are simple to use
- Tools that will integrate into their workflow

Professionals: established workflows and familiar tools

- Should integrate into their existing workflow
- build tools that have similar formats and configurations to what these users are familiar with

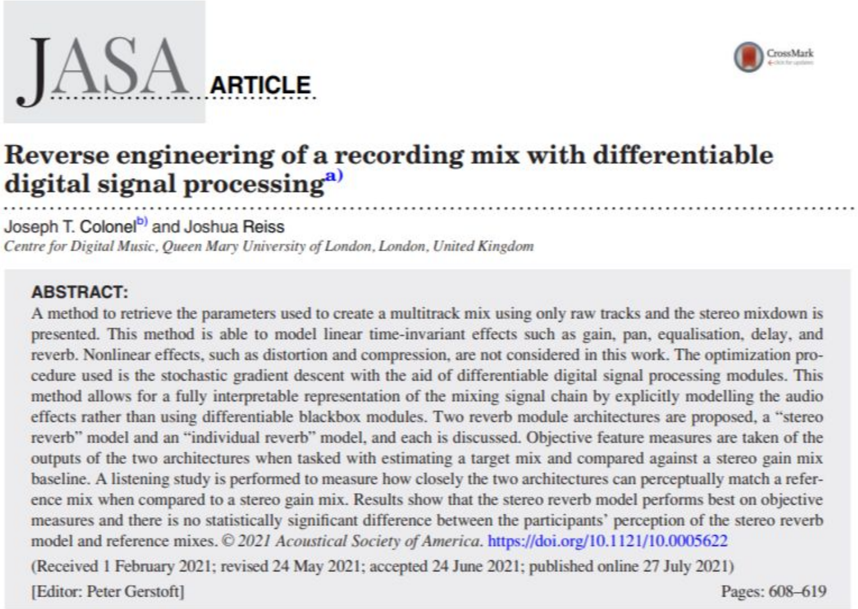


Ideal design for an automatic mixing system

Given a music mixture and its multitrack recordings, can we reverse-engineer the Fx graph?

Reverse Engineering (2021, 2024)

Reverse engineering of a mix



JASA ARTICLE

CrossMark

Reverse engineering of a recording mix with differentiable digital signal processing[a)]

Joseph T. Colonel[b)] and Joshua Reiss
Centre for Digital Music, Queen Mary University of London, London, United Kingdom

ABSTRACT:
A method to retrieve the parameters used to create a multitrack mix using only raw tracks and the stereo mixdown is presented. This method is able to model linear time-invariant effects such as gain, pan, equalisation, delay, and reverb. Nonlinear effects, such as distortion and compression, are not considered in this work. The optimization procedure used is the stochastic gradient descent with the aid of differentiable digital signal processing modules. This method allows for a fully interpretable representation of the mixing signal chain by explicitly modelling the audio effects rather than using differentiable blackbox modules. Two reverb module architectures are proposed, a "stereo reverb" model and an "individual reverb" model, and each is discussed. Objective feature measures are taken of the outputs of the two architectures when tasked with estimating a target mix and compared against a stereo gain mix baseline. A listening study is performed to measure how closely the two architectures can perceptually match a reference mix when compared to a stereo gain mix. Results show that the stereo reverb model performs best on objective measures and there is no statistically significant difference between the participants' perception of the stereo reverb model and reference mixes. https://doi.org/10.1121/10.0005622

(Received 1 February 2021; revised 24 May 2021; accepted 24 June 2021; published online 27 July 2021)

[Editor: Peter Gerstoft] Pages: 608–619

Reverse engineering of a recording mix with differentiable digital signal processing, Colonel et al. (JASA, July 2021)

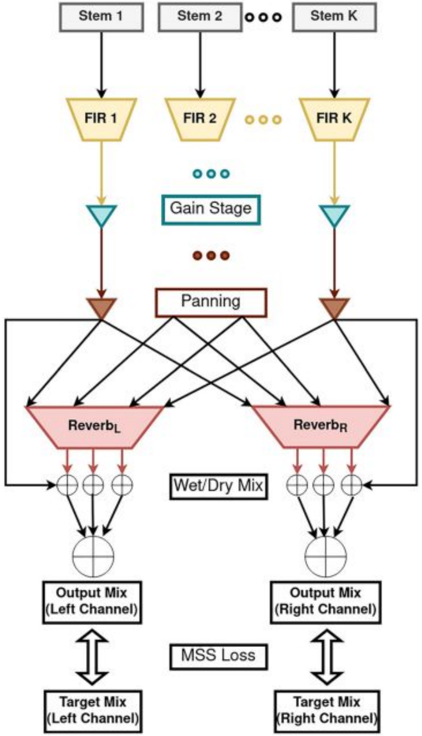


FIG. 1. (Color online) The mixing chain diagram for the "stereo bus" architecture.

Searching for Mixing Graphs: A Pruning Approach

Proceedings of the 27th International Conference on Digital Audio Effects (DAFx24), Guildford, United Kingdom, 3 - 7 September 2024

SEARCHING FOR MUSIC MIXING GRAPHS: A PRUNING APPROACH

Sungho Lee[†*], *Marco A. Martínez-Ramírez*[♮], *Wei-Hsiang Liao*[♮], *Stefan Uhlich*[♯], *Giorgio Fabbro*[♯], *Kyogu Lee*[†], *and Yuki Mitsufuji*[♮♭]

[†]Department of Intelligence and Information, Seoul National University, Seoul, South Korea
[♮]Sony AI, Tokyo, Japan [♯]Sony Europe B.V., Stuttgart, Germany [♭]Sony Group Corporation, Tokyo, Japan

ABSTRACT

Music mixing is *compositional* — experts combine multiple audio processors to achieve a cohesive mix from dry source tracks. We propose a method to reverse engineer this process from the input and output audio. First, we create a mixing console that applies all available processors to every chain. Then, after the initial console parameter optimization, we alternate between removing redundant processors and fine-tuning. We achieve this through differentiable implementation of both processors and pruning. Consequently, we find a sparse mixing graph that achieves nearly identical matching quality of the full mixing console. We apply this procedure to dry-mix pairs from various datasets and collect graphs that also can be used to train neural networks for music mixing applications.

1. INTRODUCTION

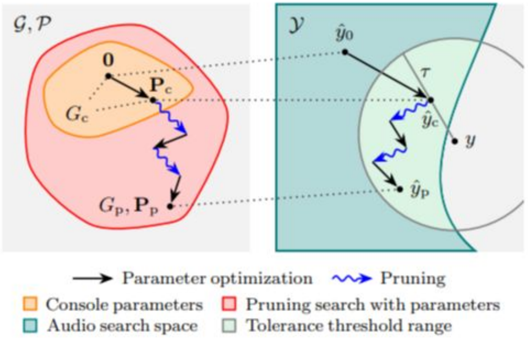
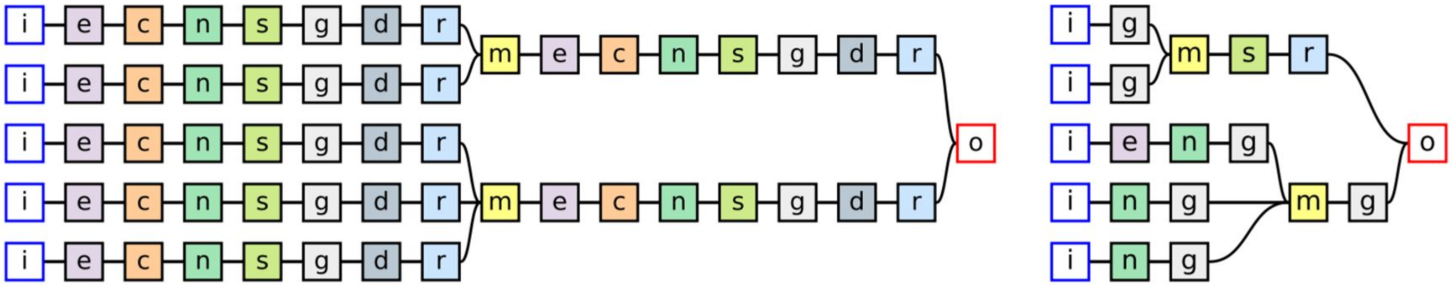


Figure 1: *Music mixing graph search via iterative pruning.*

Searching for Mixing Graphs: A Pruning Approach



(a) Full mixing console (before pruning)

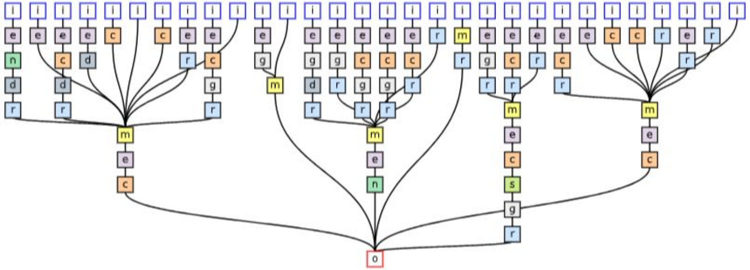
(b) Pruned graph

Searching for Mixing Graphs: A Pruning Approach

- → To assist engineers in music production applications
- → To collect graphs that can be used to train music AI models
- → To make black-box models interpretable

GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch

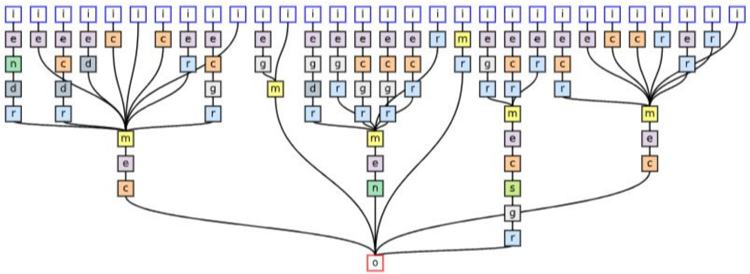
```
pip install grafx
```

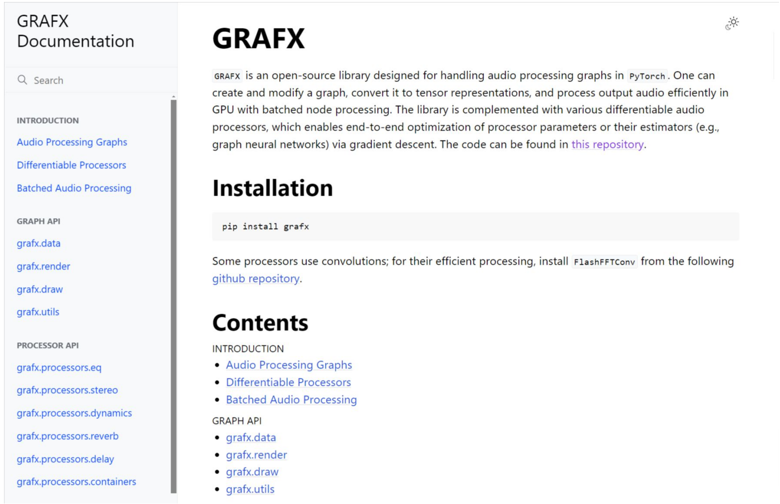


GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch, Lee et al. (DAFx24, Sep 2024)

GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch

```
pip install grafx
```





GRAFX Documentation

Search

INTRODUCTION

Audio Processing Graphs

Differentiable Processors

Batched Audio Processing

GRAPH API

grafx.data

grafx.render

grafx.draw

grafx.utils

PROCESSOR API

grafx.processors.eq

grafx.processors.stereo

grafx.processors.dynamics

grafx.processors.reverb

grafx.processors.delay

grafx.processors.containers

GRAFX

`GRAFX` is an open-source library designed for handling audio processing graphs in `PyTorch`. One can create and modify a graph, convert it to tensor representations, and process output audio efficiently in GPU with batched node processing. The library is complemented with various differentiable audio processors, which enables end-to-end optimization of processor parameters or their estimators (e.g., graph neural networks) via gradient descent. The code can be found in this repository.

Installation

```
pip install grafx
```

Some processors use convolutions; for their efficient processing, install `FlashFFTConv` from the following github repository.

Contents

INTRODUCTION

- Audio Processing Graphs
- Differentiable Processors
- Batched Audio Processing

GRAPH API

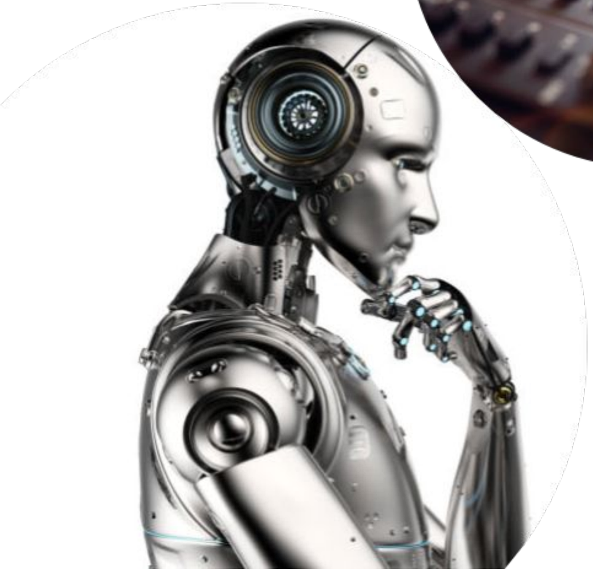
- grafx.data
- grafx.render
- grafx.draw
- grafx.utils

GRAFX: An Open-Source Library for Audio Processing Graphs in Pytorch, Lee et al. (DAFx24, Sep 2024)

Part-4

Evaluation

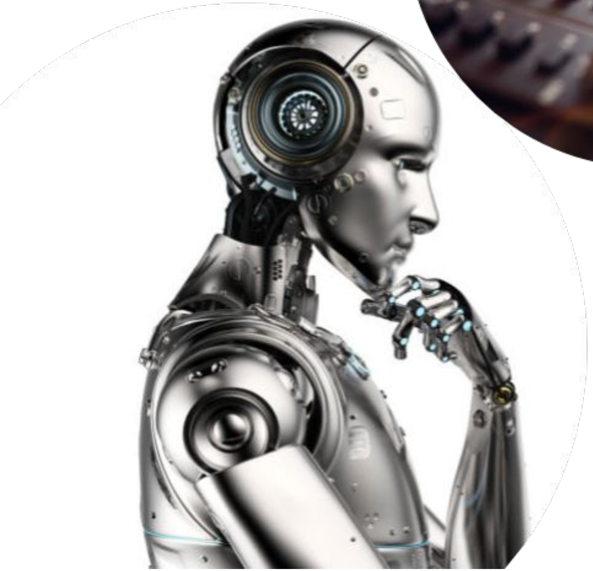
Evaluation



Music mixing is inherently a creative process and therefore a highly subjective task

It cannot be categorized as correct or incorrect

Evaluation



There is not a single metric that will fully encompass the production quality of a generated mix

The use of a professional mix as the ground truth can be an indicator of performance

However, a mix that deviates from the ground truth is not always an aesthetically unpleasant or “bad” mix.

Objective Metrics

- **Objective evaluation of music production tasks remains an open field of research**

- Audio features, loss function or deep learning embeddings to fully represent solely the mixing processing

- Also, we can use audio features related to mixing audio effects as a way to numerically approximate the evaluation of mixes

Objective Metrics

- **Objective evaluation of music production tasks remains an open field of research**
- No audio feature, loss function or deep learning embedding have yet been found that fully represent solely the mixing processing
- We can use audio features related to mixing audio effects as a way to numerically approximate the evaluation of mixes

Shortcomings

- Cannot capture production quality or aesthetic improvements
- Cannot evidence artifacts within the mix
- Ill-posed problem; deviating from the ground truth does not always mean the mix is incorrect

Audio Features

Spectral features

- EQ and reverberation
- Spectral centroid, bandwidth, contrast, flatness, and roll-off

Spatialisation features

- Panning
- Panning Root Mean Square (RMS)

Dynamic features

- DRC
- RMS level, dynamic spread and crest factor

Loudness features

- The integrated loudness level (LUFS) and peak loudness

Listening Test

Perceptual listening tests have become the conventional way to evaluate these systems

There is no standardized test type or platform

We can design tests based on a set of best practices

Adjust them to the specific characteristics of the automatic mixing system

Platforms for multi-stimuli tests

Platform	Multi-stimuli test	Features	Usage
Web Audio Evaluation Tool (Jillings et al., 2015)	-MUSHRA -APE -Discrete -Reference is optional	-Training stage -Loudness normalization -Synchronized playback -Randomization	-Requires server -PHP support has not been updated -Customization with effort
webMUSHRA (Schoeffler et al., 2018)	-MUSHRA	-Training stage -Fade-in/out -Synchronized playback -Randomization	-Requires server -Customization with effort
goListen (Barry et al., 2021b)	-MUSHRA -Reference is optional	-Synchronized playback -Randomization	-Requires account -Does not require server -Customization with effort -Ease-of-use

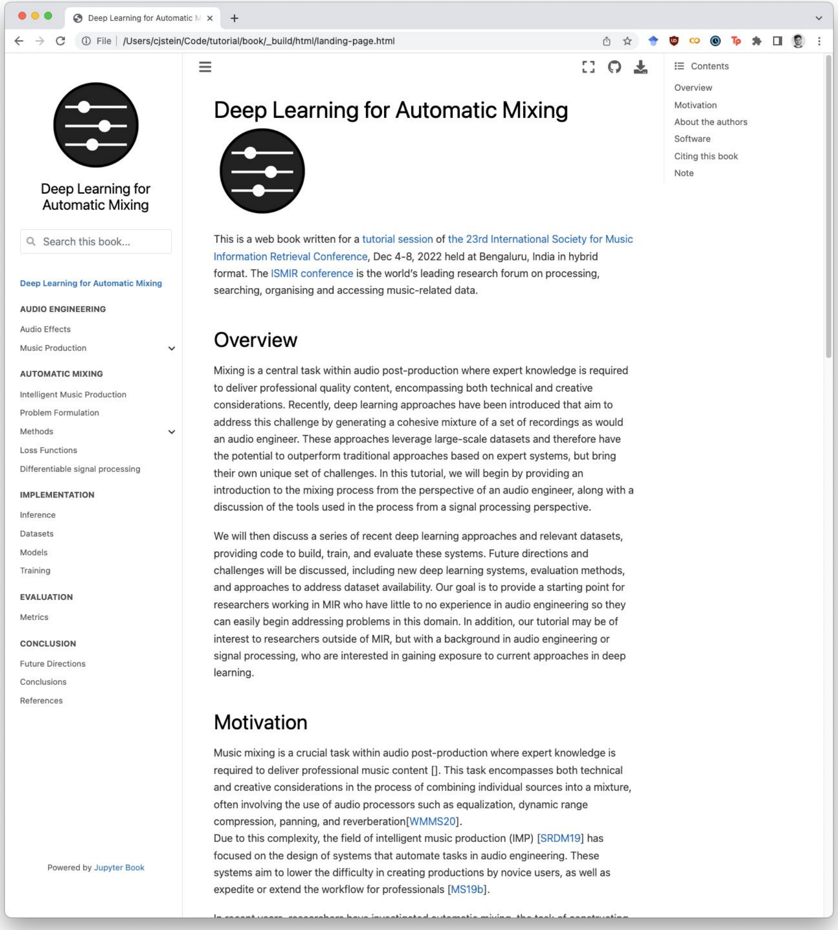
Listening Test

Several design decisions must be taken into account

- Type of test
- Number of stimuli
- Duration of the stimuli
- Criteria to be rated
- Requirements for the participants
- Listening environment

Book

https://dl4am.github.io/tutorial



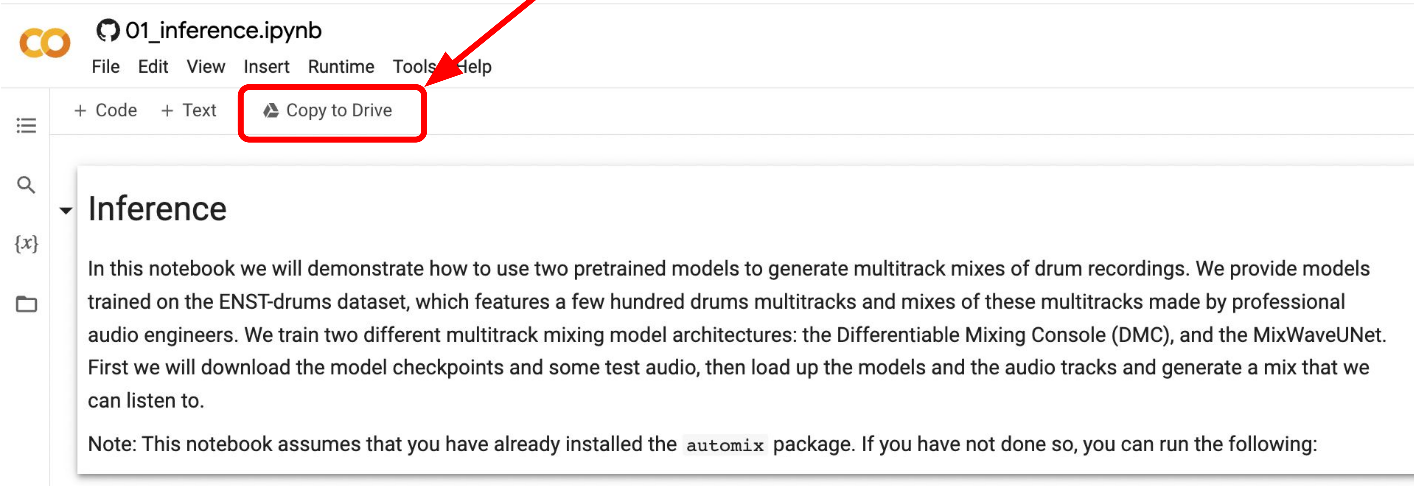
Break

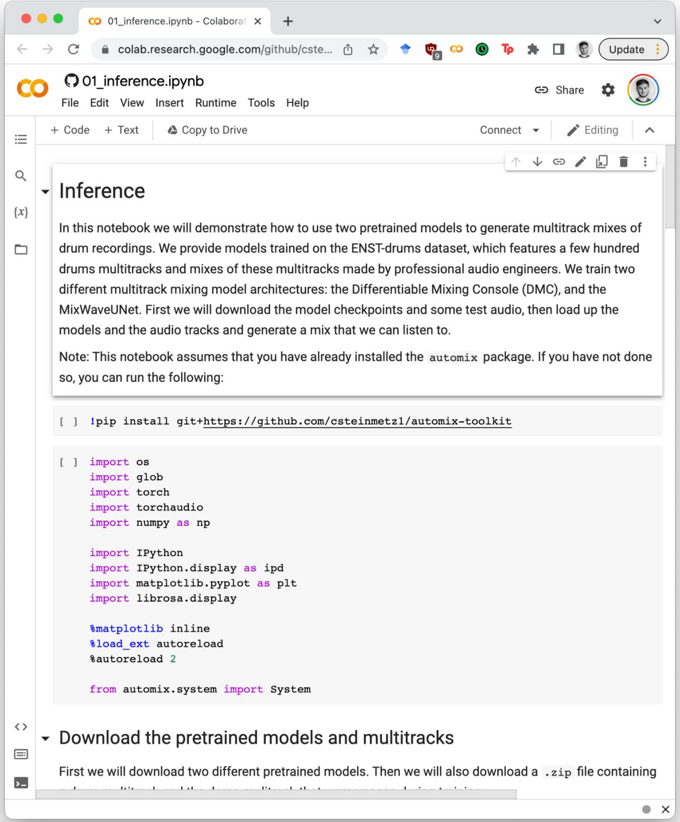
(15min)

Implementation

Part 5

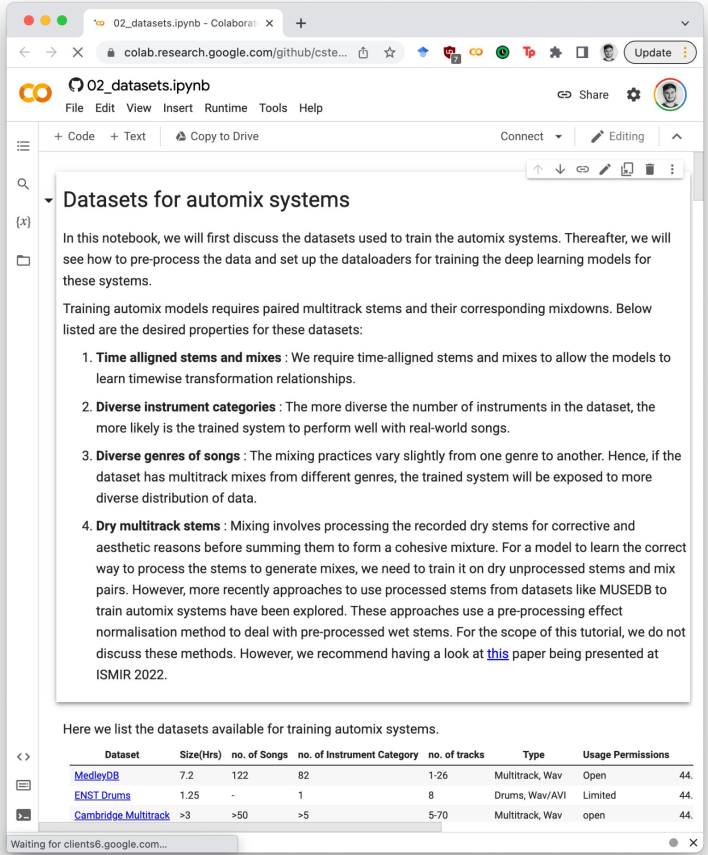
You can save your results and come back later if you click “Copy to Drive”





Inference

[Link]()



Datasets for automix systems

In this notebook, we will first discuss the datasets used to train the automix systems. Thereafter, we will see how to pre-process the data and set up the dataloaders for training the deep learning models for these systems.

Training automix models requires paired multitrack stems and their corresponding mixdowns. Below listed are the desired properties for these datasets:

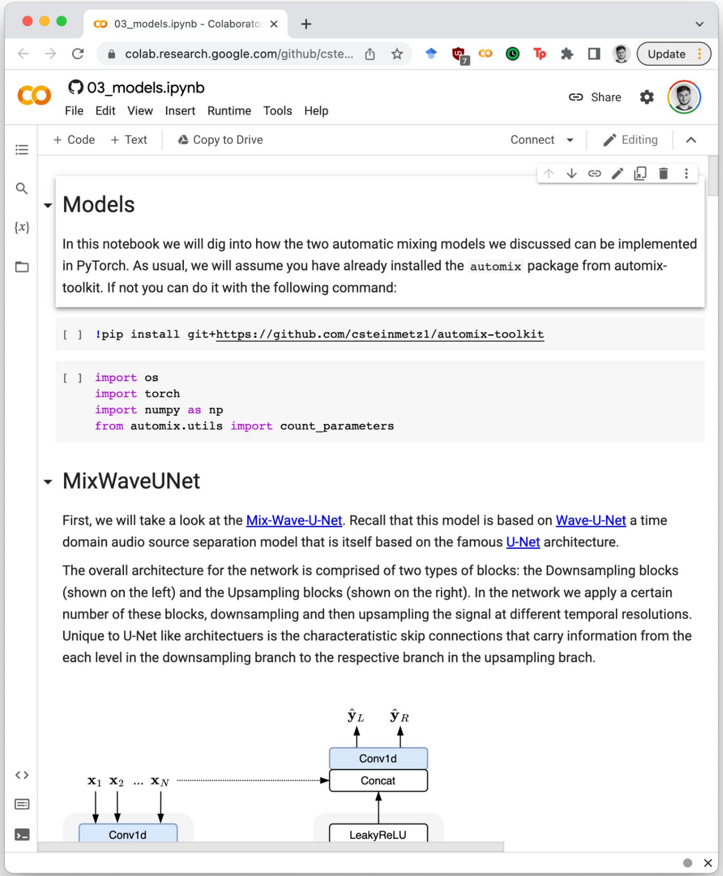
1. **Time alligned stems and mixes** : We require time-alligned stems and mixes to allow the models to learn timewise transformation relationships.
2. **Diverse instrument categories** : The more diverse the number of instruments in the dataset, the more likely is the trained system to perform well with real-world songs.
3. **Diverse genres of songs** : The mixing practices vary slightly from one genre to another. Hence, if the dataset has multitrack mixes from different genres, the trained system will be exposed to more diverse distribution of data.
4. **Dry multitrack stems** : Mixing involves processing the recorded dry stems for corrective and aesthetic reasons before summing them to form a cohesive mixture. For a model to learn the correct way to process the stems to generate mixes, we need to train it on dry unprocessed stems and mix pairs. However, more recently approaches to use processed stems from datasets like MUSEDB to train automix systems have been explored. These approaches use a pre-processing effect normalisation method to deal with pre-processed wet stems. For the scope of this tutorial, we do not discuss these methods. However, we recommend having a look at [this](#) paper being presented at ISMIR 2022.

Here we list the datasets available for training automix systems.

Dataset	Size(Hrs)	no. of Songs	no. of Instrument Category	no. of tracks	Type	Usage Permissions	
MedleyDB	7.2	122	82	1-26	Multitrack, Wav	Open	44.
ENST Drums	1.25	-	1	8	Drums, Wav/AVI	Limited	44.
Cambridge Multitrack	>3	>50	>5	5-70	Multitrack, Wav	open	44.

Datasets

Link


03_models.ipynb - Colaboratory
colab.research.google.com/github/cste...
03_models.ipynb
File Edit View Insert Runtime Tools Help
Share
+ Code + Text Copy to Drive
Connect Editing

Models

In this notebook we will dig into how the two automatic mixing models we discussed can be implemented in PyTorch. As usual, we will assume you have already installed the `automix` package from automix-toolkit. If not you can do it with the following command:

```
!pip install git+https://github.com/csteinmetz1/automix-toolkit
```

```
import os
import torch
import numpy as np
from automix.utils import count_parameters
```

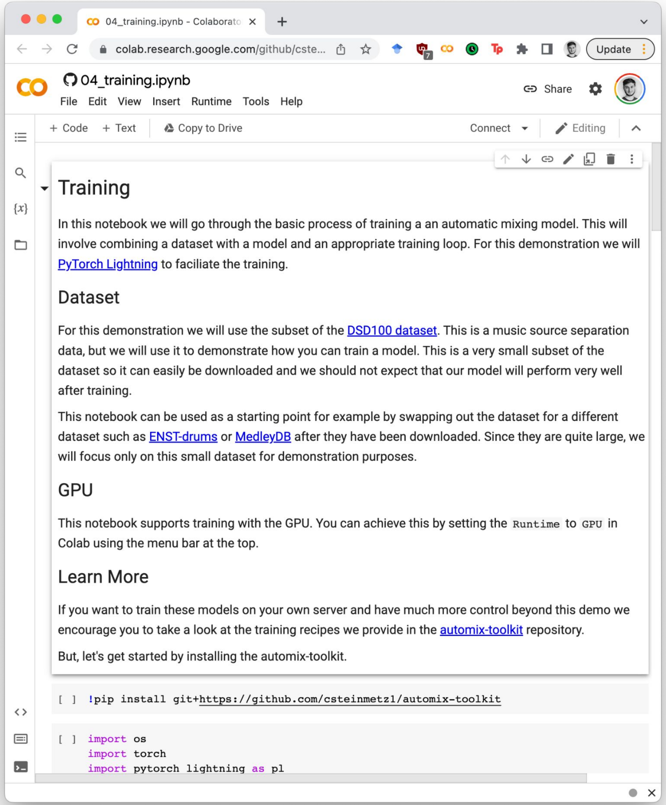
MixWaveUNet

First, we will take a look at the Mix-Wave-U-Net. Recall that this model is based on Wave-U-Net a time domain audio source separation model that is itself based on the famous U-Net architecture.

The overall architecture for the network is comprised of two types of blocks: the Downsampling blocks (shown on the left) and the Upsampling blocks (shown on the right). In the network we apply a certain number of these blocks, downsampling and then upsampling the signal at different temporal resolutions. Unique to U-Net like architectuers is the characteristic skip connections that carry information from the each level in the downsampling branch to the respective branch in the upsampling brach.

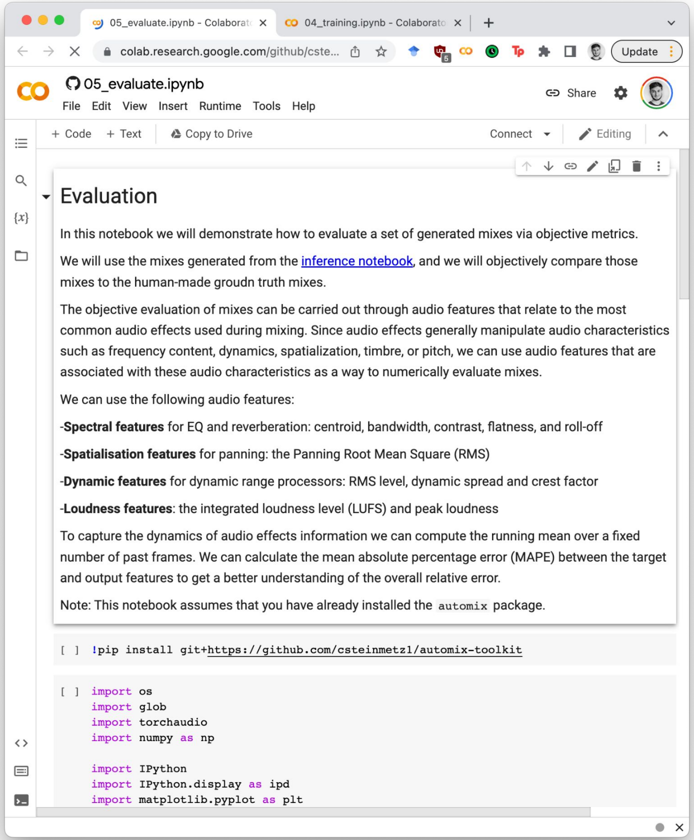
Models

[Link]



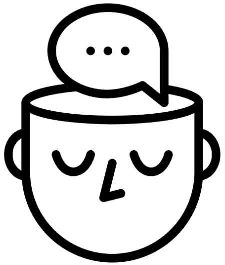
Training

[Link]()



Evaluation

Link



Part 6

Future Directions

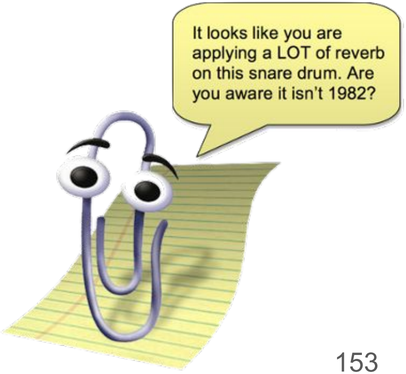
AI comes in many forms

“The Black Box”

“The Assistant”

“The Smart Interface”

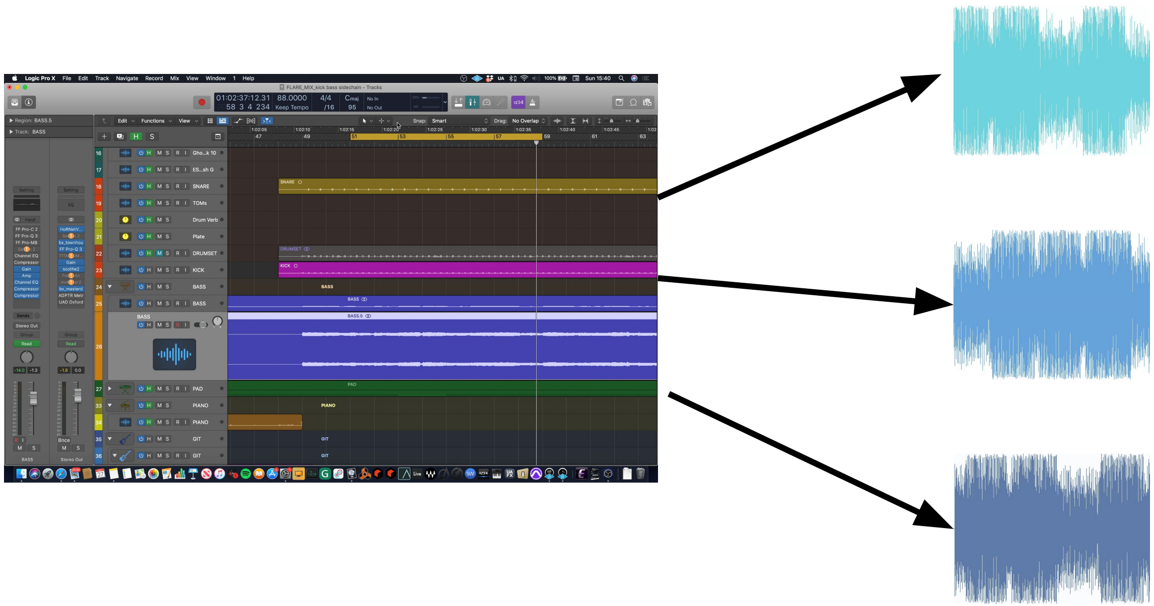
“The Diagnostician”



Generative models

The mixing task is a one to many mapping...

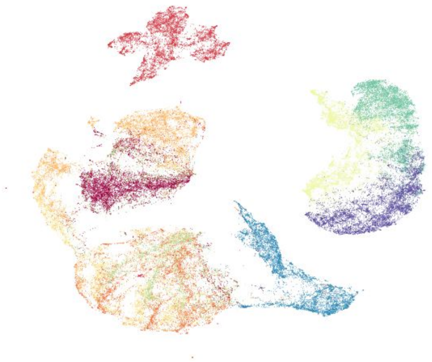
Mixes

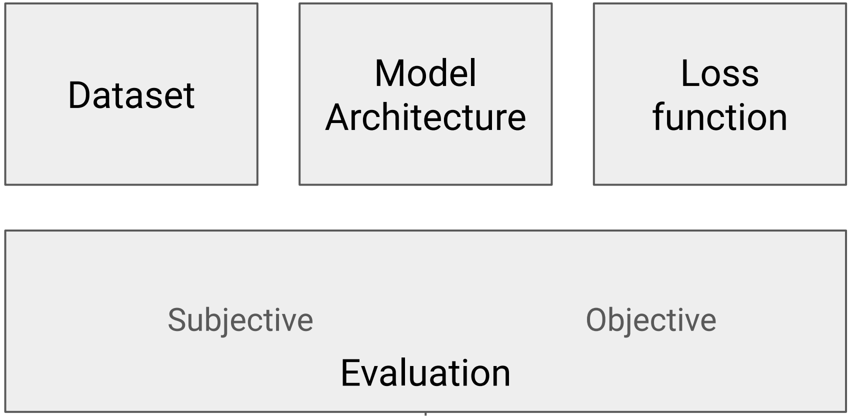


So we should treat it as such. Go beyond supervised learning?

Further Interests

- Learn a latent space of audio production representation
  - This will allow us to learn a global sound of the mix
  - Easily explore mixing space for quick iteration for user
- Better objective evaluation methods for models; what is a good mix afterall?
- A loss function that better captures mixing practices. Embedding loss?
- More ways to incorporate context.

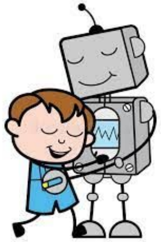




Last thoughts

- Static mixes and static chains -> learned chains and automation
- Black box - exploration of generative methods
- White box - more context, learned effect chains
- Audio quality closer to human engineers work
- Work with larger number of tracks - as in real world practice
- Apt evaluation techniques (objective and subjective)
- Systems learning long term coherence across more tracks and longer durations
- Mixing anomaly detection
- Expansion of mixing to film audio, broadcasting, game audio (principles for mixing varies)

Key Factors for Success of Smart Mixing tools



- Interaction models that facilitate trust
  - lack of interpretability and control - barrier to their adoption.

- High precision and quality of results generated
  - low-quality output not useful in professional workflows.

- Seamless integration into existing workflows
  - maximize efficiency and productivity.

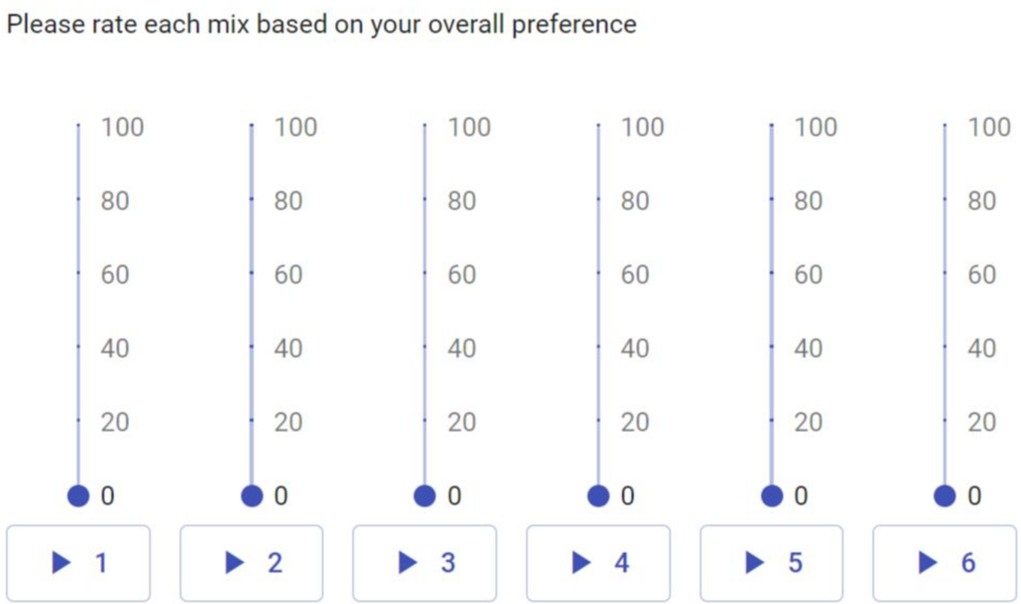
Towards a Human Centric Design Framework for AI Assisted Music Production
Tsiros, A. and Palladini, A. *NIME 2020*

Takeaways

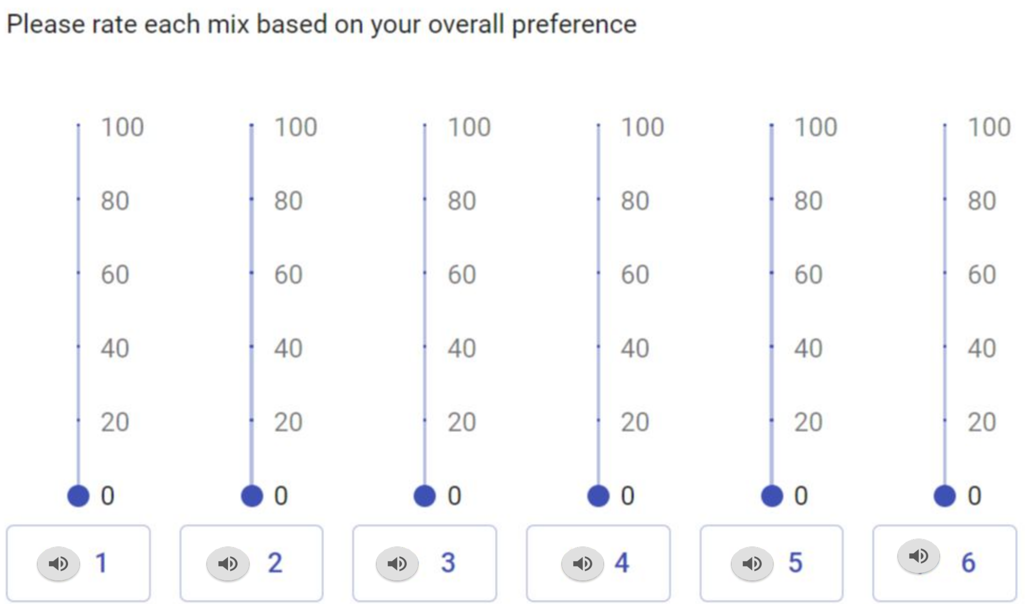
1. Mixing is a task that maps creative ideas and emotion to technical parameters
2. Approaches are often either *direct transformation* or *parameter estimation*
3. Evaluation remains challenging and we rely on well design listening tests
4. Many open questions and challenges with potentially fruitful outcomes

Demos

Mixes


Please rate each mix based on your overall preference
100
80
60
40
20
0
1
2
3
4
5
6

Mixes

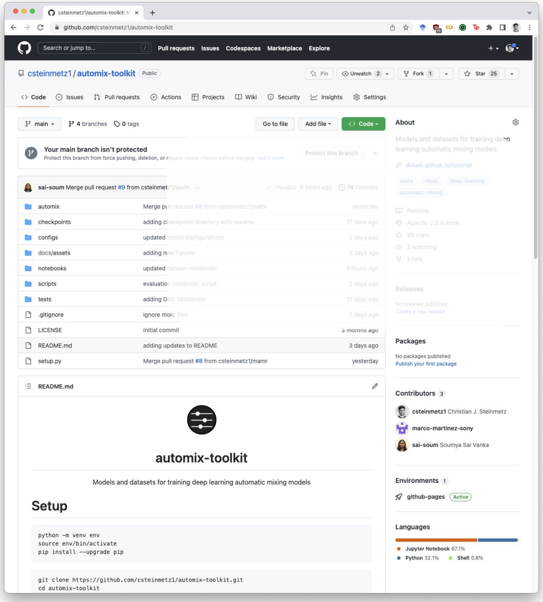

Please rate each mix based on your overall preference
100
80
60
40
20
0
1
2
3
4
5
6

Mixes

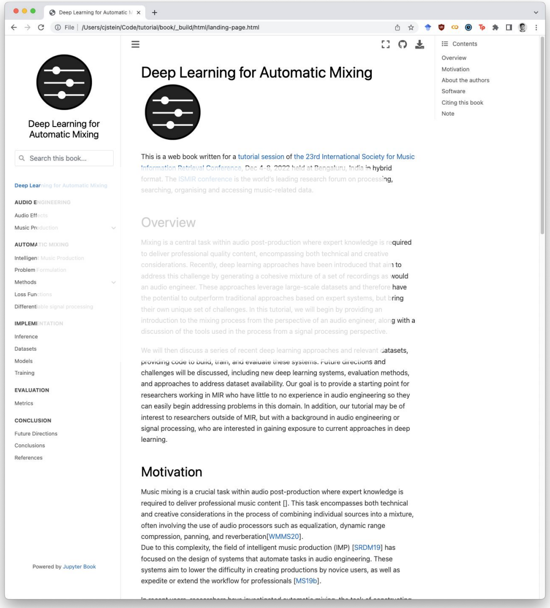
1. (Koo et al., 2022a) - Music Mixing Style Transfer with reference from MUSDB18 (same genre)
2. Mono mix
3. Gary Bromham - Professional audio engineer mix
4. (Steinmetz et al., 2021) - DMC mix trained with MedleyDB - Gain and Panning
5. (Martinez-Ramirez et al., 2022) - Fx Normalization
6. RoEx

Resources

Final Questions



GitHub



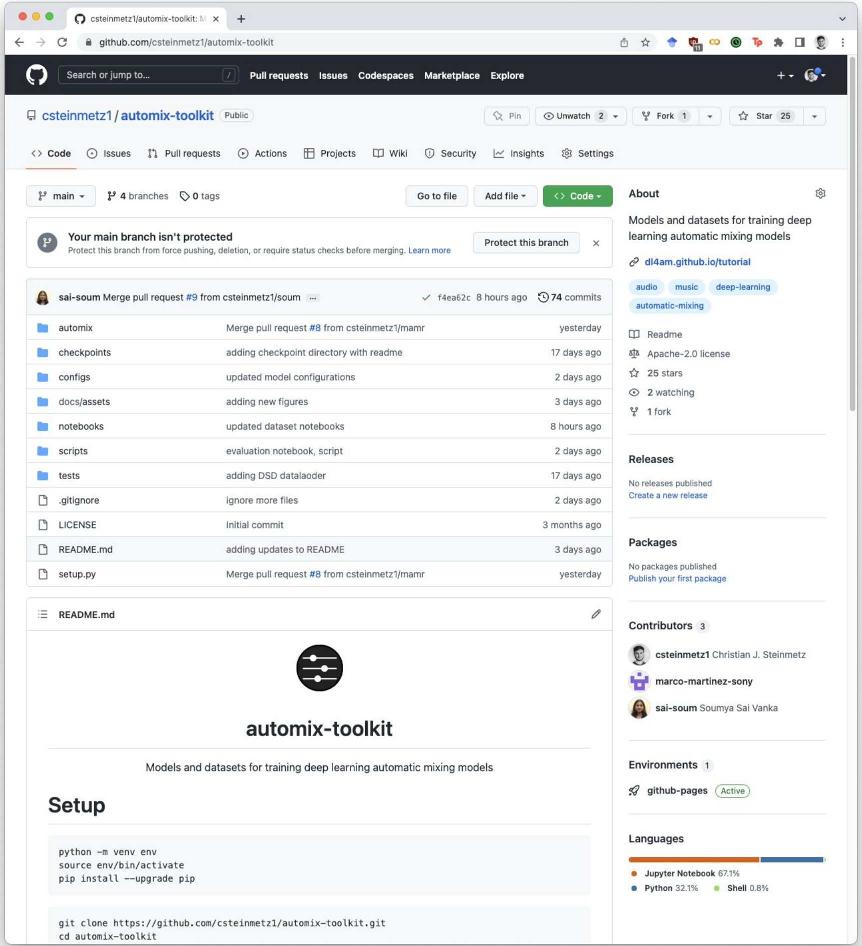
Book

automix-toolkit

https://github.com/csteinmetz1/automix-toolkit

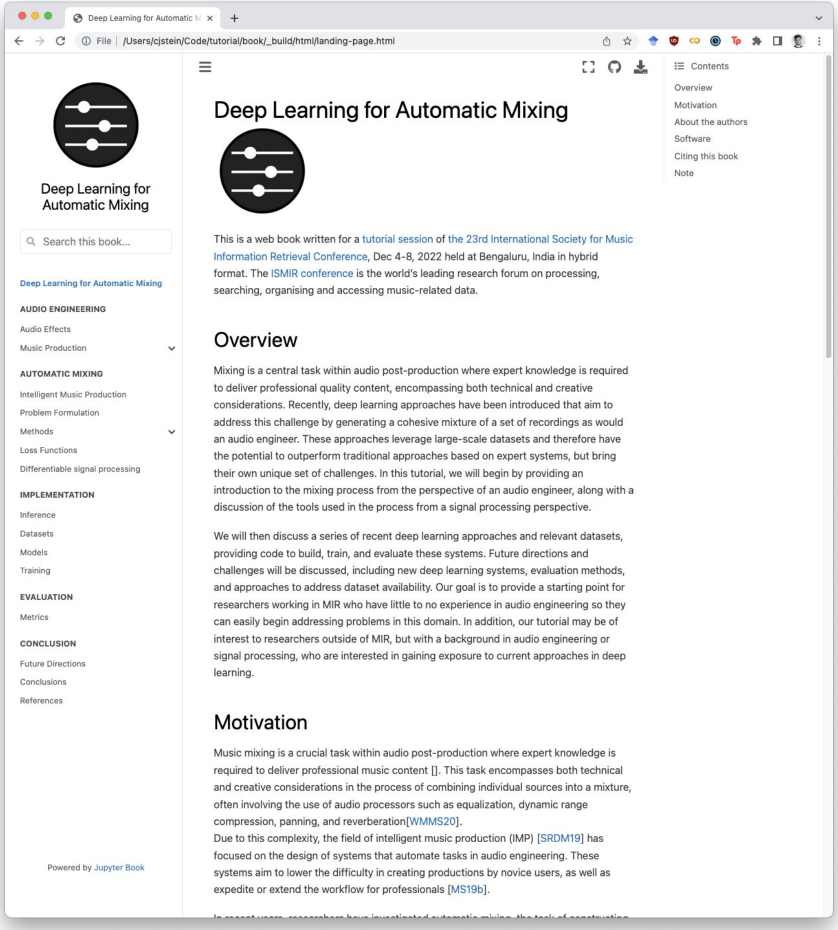
★ Star

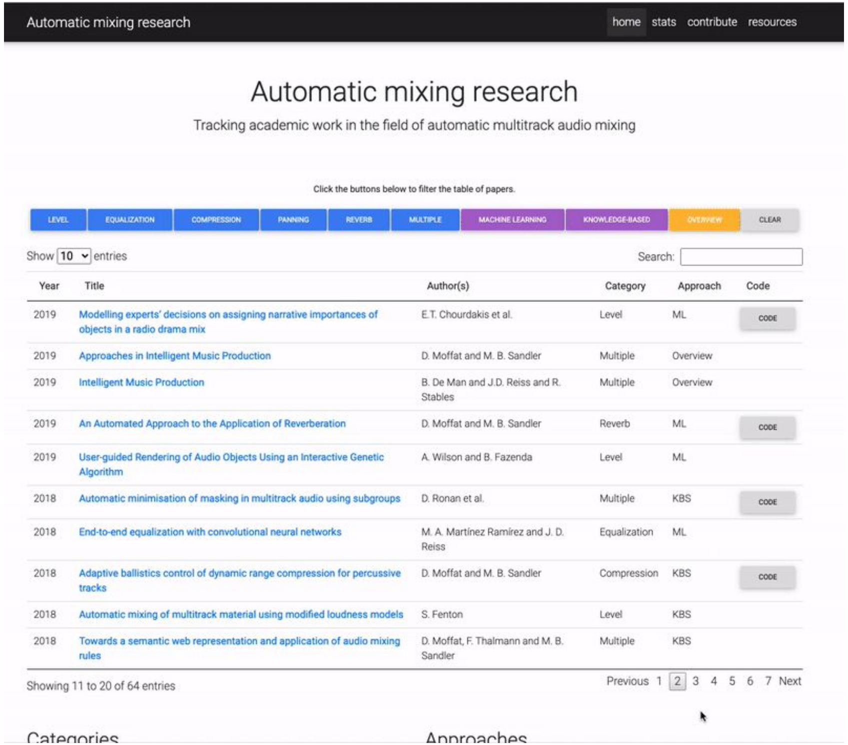
Star it on GitHub



Book

https://dl4am.github.io/tutorial




Automatic mixing research | home | stats | contribute | resources

Automatic mixing research

Tracking academic work in the field of automatic multitrack audio mixing

Click the buttons below to filter the table of papers.

LEVEL | EQUALIZATION | COMPRESSION | PANNING | REVERB | MULTIPLE | MACHINE LEARNING | KNOWLEDGE-BASED | OVERVIEW | CLEAR

Show 10 entries   Search:

Year	Title	Author(s)	Category	Approach	Code
2019	Modelling experts' decisions on assigning narrative importances of objects in a radio drama mix	E.T. Chourdakis et al.	Level	ML	CODE
2019	Approaches in Intelligent Music Production	D. Moffat and M. B. Sandler	Multiple	Overview	
2019	Intelligent Music Production	B. De Man and J.D. Reiss and R. Stables	Multiple	Overview	
2019	An Automated Approach to the Application of Reverberation	D. Moffat and M. B. Sandler	Reverb	ML	CODE
2019	User-guided Rendering of Audio Objects Using an Interactive Genetic Algorithm	A. Wilson and B. Fazenda	Level	ML	
2018	Automatic minimisation of masking in multitrack audio using subgroups	D. Ronan et al.	Multiple	KBS	CODE
2018	End-to-end equalization with convolutional neural networks	M. A. Martínez Ramírez and J. D. Reiss	Equalization	ML	
2018	Adaptive ballistics control of dynamic range compression for percussive tracks	D. Moffat and M. B. Sandler	Compression	KBS	CODE
2018	Automatic mixing of multitrack material using modified loudness models	S. Fenton	Level	KBS	
2018	Towards a semantic web representation and application of audio mixing rules	D. Moffat, F. Thalmann and M. B. Sandler	Multiple	KBS	

Showing 11 to 20 of 64 entries   Previous 1 2 3 4 5 6 7 Next

Categories   Approaches

More works on automatic mixing research

Searchable/filterable table of relevant papers and stats

https://csteinmetz1.github.io/AutomaticMixingPapers

Questions