DIFF-MST: DIFFERENTIABLE MIXING STYLE TRANSFER

First Author
Affiliation1
author1@ismir.edu

Second Author
Retain these fake authors in submission to preserve the formatting

Third Author
Affiliation3
author3@ismir.edu

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 differentiable mixing console, a transformer controller, and an audio production style loss function. By inputting raw tracks and a reference song, our model estimates control parameters for audio effects within a differentiable mixing console, producing high-quality mixes and enabling post-hoc adjustments. Moreover, our architecture supports an arbitrary number of input tracks without source labelling, enabling real-world applications. We evaluate our model's performance against robust baselines and showcase the effectiveness of our approach, architectural design, tailored audio production style loss, and innovative training methodology for the given task. We provide code, pre-trained models, and listening examples online.

1. INTRODUCTION

Music mixing involves technical and creative decisions that shape the emotive and sonic identity of a song [1]. The process involves creating a cohesive mix of the given tracks using audio effects to achieve balance, panorama, and aesthetic value [2]. Given the complexity of the task, mastering the task of mixing often requires many years of practice. To address this, several solutions have been proposed to provide assistance or automation [3,4]. Automatic mixing systems have been designed using knowledge engineering [5, 6], machine learning, and more recently deep learning methods [7–11]. Automatic mixing systems can be further subdivided into direct transformation systems and parameter estimation systems, as shown in Fig. 2. Direct transformation systems operate on tracks and predict a mix directly, in an end-to-end fashion, with the loss calculated between the ground truth mix and the predicted mix.

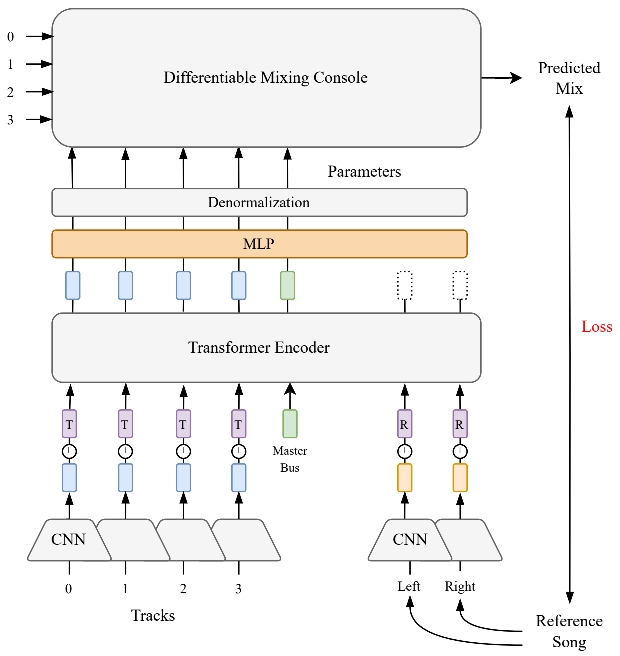

Figure 1. Diff-MST, a differentiable mixing style transfer framework featuring a differentiable multitrack mixing console, a transformer-based controller that estimates control parameters for this mixing console, and an audio production style loss function that measures the similarity between the estimated mix and reference mixes.

On the other hand, parameter estimation systems take input tracks and predict control parameters for a dedicated mixing console. In such systems, the loss can either be calculated on the predicted control parameters (parameter loss) based on the availability of ground truth, or on the predicted audio against the ground truth mix (audio loss). Parameter loss, calculated on the parameters, may not be optimal for multiparameter signal processing blocks since various combinations of parameters could potentially produce similar outcomes. [7, 11] utilizes a deep learning-based direct transformation system for mixing, while [8] employs a parameter estimation-based deep learning approach. However, many of these systems are constrained to a small number of input tracks or struggle to generalize effectively to real-world mixing scenarios. Furthermore, most of these approaches generate a mix without accounting for the desired sound and emotion. Due to the subjective nature of the task, an end-to-end approach without user control is less desirable in professional practice [12].

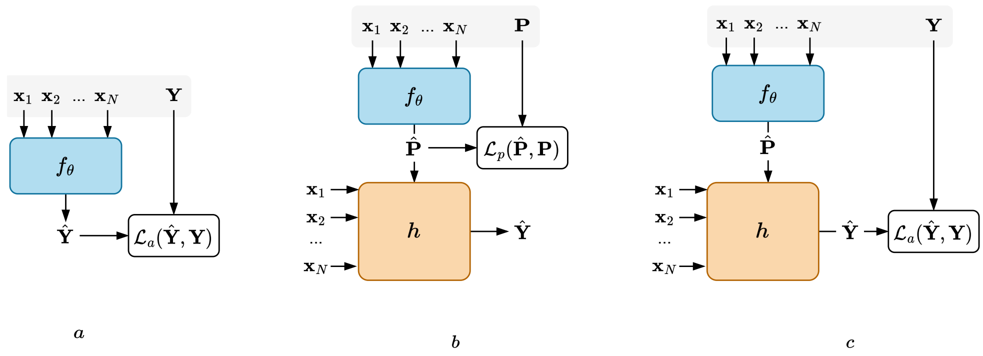

Figure 2. Formulations for deep learning-based automatic mixing systems. (a) Direct transformation (b) Parameter estimation on parameter loss (c) Parameter estimation on audio loss. Here, x_i for $i \in [1, N]$ are the N input tracks, f_θ is the transformation, Y and $\hat{Y}$ are the ground truth and predicted mix, P and $\hat{P}$ are the ground truth and predicted control parameters and L_a and L_p are the audio and parameter loss respectively.

1.1 Mixing Style Transfer

In professional practice, the audio engineer often uses reference songs and guidelines provided by the client to make mixing decisions [13]. This encourages the development of automatic mixing systems that are aware of the intention of the mix engineer. In our context, mixing style transfer refers to mixing in the style of given reference songs [14]. This pertains to capturing the global sound, dynamics and spatialisation of the reference song. Recently, deep learning systems have been proposed for audio production style transfer. While some approaches have considered estimating the control parameters for audio effects [15], they are so far limited to controlling only a single or small set of effects with a singular input. Systems for multitrack mixing [16] consider multiple tracks or a sum of tracks and operate in an end-to-end fashion, but this limits interpretability and controllability. In this work, we introduce a novel deep learning-based approach to mixing multitrack audio material using a reference song, which utilises a differentiable mixing console to predict parameter values for gain, pan, 4-band equalization, compressor, and a master bus. Our proposed system is differentiable, interpretable and controllable, and can learn the mixing style from the given reference song. The contributions of this work can be summarised as follows:

1. A framework for mixing style transfer that enables control of audio effects mapping the production style from a reference onto a set of input tracks.
2. A differentiable multitrack mixing console consisting of gain, parametric equalisation, dynamic range compression, stereo panning, and master bus processing, which enables end-to-end training.
3. Evaluation of our approach compared to strong baselines with objective metrics.
4. We demonstrate the benefits of our system, including generalisation to an arbitrary number of input tracks, no requirement for labelling of inputs or enforcement of specific taxonomies, high-fidelity processing without artifacts, and greater efficiency.

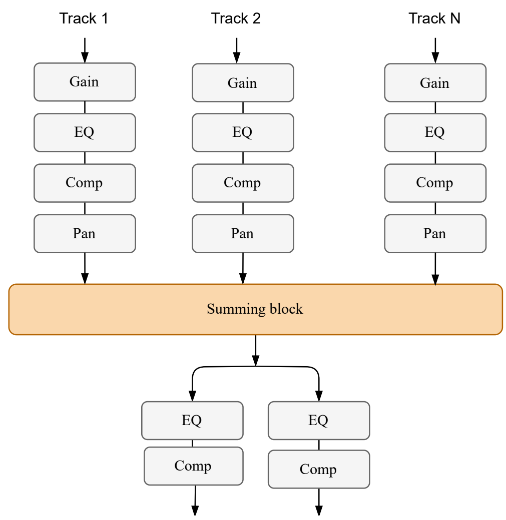

Figure 3. Differentiable Mixing console

2. METHOD

2.1 Problem Formulation

We can formulate the mixing style transfer task as follows. Let T be a matrix of N mono input raw tracks $\{t_1, t_2, t_3, \ldots, t_N\}$ and M_r be the matrix of stereo reference mix containing two channels. A shared weight encoder $f_{\theta r}$ and $f_{\theta t}$ are employed to extract information from the reference and input tracks respectively. This information is then aggregated and fed into a transformer controller network comprising a transformer encoder and a multi-layer perception (MLP) g_ϕ. The primary task of this network is to estimate the parameter matrix P, which consists of N parameter vectors p, each responsible for configuring the chain of audio effects for a respective track in T. Subsequently, the differentiable mixing console $h(T, P)$ processes the input tracks T using the parameters P to generate a predicted mix M_p that mirrors the style of the reference mix M_r.

$$P = g_\phi(f_{\theta t}(T), f_{\theta r}(M_r)) \tag{1}$$

$$M_p = h(T, P) \tag{2}$$

2.2 Differentiable Mixing Style Transfer System

We propose a differentiable mixing style transfer system (Diff-MST) that takes raw tracks and a reference mix as input and predicts mixing console parameters and a mix as output. As shown in Figure 1, our system employs two encoders, one to capture a representation of the input tracks and another to capture elements of the mixing style from the reference. A transformer-based controller network analyses representations from both encoders to predict the differentiable mixing console (DMC) parameters. The DMC generates a mix for the input tracks using the predicted parameters in the style of the given reference song. Given that our system oversees the operations of the DMC rather than directly predicting the mixed audio, we circumvent potential artefacts that may arise from neural audio generation techniques [17, 18]. This also creates an opportunity for further fine-tuning and control by the user.

2.3 Differentiable Mixing Console (DMC)

The process of multitrack mixing involves applying a chain of audio effects, also known as a channel strip, on each channel of a mixing console. The audio engineer may use these devices to reduce masking, ensure a balance between the sources, and address noise or bleed. Incorporating this prior knowledge of signal processing in the design of our mixing system, we propose an interpretable and controllable differentiable mixing console (DMC). Our console applies a chain of audio effects comprising gain, parametric equaliser (EQ), dynamic range compressor (DRC), and panning to each of the tracks to produce wet tracks. The sum of wet tracks is then sent to a master bus on which we insert stereo EQ and a DRC. This produces a mastered mix of the given tracks. We incorporate a master bus in our console as it is usual to use a mastered song as a reference in workflows. Therefore, having a master bus in the mixing console chain allows for easier optimisation of the system. To enable gradient descent and training in a deep learning framework, we require the mixing console to be differentiable. To achieve this, we use differentiable effects from the `dasp-pytorch`[1]. The pipeline of the DMC is presented in Figure 3.

2.4 Spectrogram Encoder

The encoder comprises a magnitude spectrum-based convolutional network. The encoder computes spectrograms using short-time Fourier transform with a Hann window of size N = 2048 and a hop size of H = 512. The magnitude spectrogram passed through a convolutional network. The convolutional encodings are passed through a linear layer and return an embedding of size 512. The model features a separate shared-weight mix encoder $f_\theta r$ and track encoder $f_\theta t$ for each of the reference mix and the input tracks respectively. We treat each channel of a stereo audio as a separate track. Therefore, we load the stereo mix and any other stereo input track into separate tracks. We then pass T and M_r through the encoder and compute embeddings.

[1] https://github.com/csteinmetz1/dasp-pytorch/

2.5 Transformer Controller

The controller features a transformer encoder and a shared-weight MLP. The transformer encoder generates style-aware embeddings using self-attention across the output of the spectrogram encoder$f_\theta r$ and $f_\theta t$ and a master bus embedding which is learned during training. The MLP predicts the control parameters corresponding to the channel strip for each track, and the master bus embeddings are used to predict the master bus control parameters. A shared weight MLP is used to predict channel strip parameters for each channel. We generate the predicted mix M_p by passing the control parameters through the DMC along with the tracks. This architecture enables our system to be invariant to the number of input tracks as shown in Figure 1.

2.6 Audio Production Style Loss

The style of a mix can be broadly captured using features that describe its dynamics, spatialisation and spectral attributes [13]. We propose two different losses to train and optimise our models.

Audio Feature (AF) loss: This loss is composed of traditional MIR audio feature transforms [19]. These T features include the root mean square (RMS) and crest factor (CF), stereo width (SW) and stereo imbalance (SI) and bark-spectrum (BS) corresponding to the dynamics, spatialisation and spectral attributes respectively. We optimise our system by calculating the weighted average of the mean squared error on the audio features that minimises the distance between M_p and M_r. We compute the audio feature transforms T along with the weights w as follows:

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

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

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

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

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

where N represents the sequence length, x is the input tensor, FB is the filterbank matrix, STFT(x) represents the short-time Fourier transform of x, and ϵ is a small constant of value 10^{-8} added for numerical stability. The net loss is computed as follows:

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

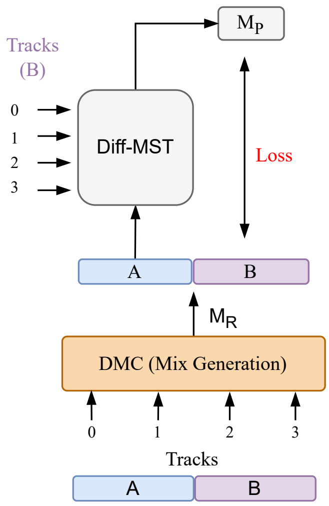

Figure 4. First training strategy from Section 3.

where w_j is the weight associated with j^{th} transform T_j and MSE corresponds to mean squared error.

MRSTFT loss: The multi-resolution short-time Fourier transform loss [20, 21] is the sum of L_1 distance between STFT of ground truth and estimated waveforms measured in both log and linear domains at multiple resolutions, with window sizes $W \in [512, 2048, 8192]$ and hop sizes $H = W/2$. This is a full-reference metric meaning that the two input signals must contain the same content.

3. EXPERIMENT DESIGN

The task requires a dataset with multitrack audio, style reference, and the ground truth mix of the multitrack in the style of the reference for training. However, due to the lack of suitable datasets, we deploy a self-supervised training strategy to enable learning of the control of audio effects without labelled or paired training data. We achieve this by training our model under two different regimes which mainly vary in data generation and loss function.

Method 1: We extend the data generation technique used in [15] to a multi-track scenario as shown in Figure 4. We first randomly sample a $t = 10$ sec segment from input tracks and generate a random mix of these input tracks by using random DMC parameters. We then split the segment of the randomly mixed audio and the input tracks into two halves, namely, M_{rA} and M_{rB} and T_A and T_B of $t/2$ secs each, respectively. The model is input with T_B as input tracks and M_{rA} as the reference song. The predicted mix M_p is compared against M_{rB} as the ground truth for backpropagation and updating of weights. Using different sections of the same song for input tracks and reference song encourages the model to focus on the mixing style while being content-invariant. This method allows the use of MRSTFT loss for optimisation as we have the ground truth available. The predicted mix is loudness normalised to -16.0 dBFS before computing the loss.

Method 2: We sample a random number of input tracks between 4-16 for song A from a multitrack dataset and use a pre-mixed real-world mix of song B from a dataset consisting of full songs as the reference. We train the model using AF loss mentioned in Section 2.6 computed between M_p and M_r. This method also allows us to train the model without the availability of a ground truth. Unlike Method 1, this approach exposes the system to training examples more similar to real-world scenarios where the input tracks and the reference song come from a different song. However, due to the sampling, some input track and reference song combinations may not be realistic.

3.1 Datasets

Multitrack: For both training methods, we utilise multitrack from MedleyDB [22, 23] and Cambridge.mt [2] which contains a total of 196 and 535 songs respectively, sampled at f_s = 44100 Hz. For both datasets, we generate a train/test/validation split of 80/20/20. During training, songs are picked at random from the training split of both datasets. Thereafter, we randomly sample a section of the song as input tracks. We find a random offset for sampling multitrack by finding a section of the mix $x[i]$ that has mean energy above the threshold, $\frac{1}{N}\sum_{i=1}^{N}|x[i]|^2 \geq 0.001$. During training, each channel corresponding to a stereo raw track is treated as a separate mono track. We check the mean energy of each track to avoid loading silent tracks. All input tracks are loudness normalised to -48.0 dBFS.

Reference Songs: For Method 1 we generate a random mix using random parameters and input tracks as mentioned in Section 3 and loudness normalise the random mix to -16 dBFS. For Method 2, we use real-world songs from MTG-Jamendo which contains more than 55k songs songs in MP3 format [24]. We pick a random segment $y[i]$ of a random song from the dataset as a reference and check for mean energy above the threshold, $\frac{1}{N}\sum_{i=1}^{N}|xy[i]|^2 \geq 0.001$. We loudness normalise the reference to -16 dbFS and load stereo information on separate channels.

3.2 Training Details

Our model contains 190 M trainable parameters, 76.5M corresponding to the track and mix encoder, and 37.9 M for the transformer controller. We train five variations of our model differing in the number of tracks, methodology and loss function used. To remedy the bottleneck of reading multitrack audio data from disk, we load data into RAM every epoch from both the training and validation sets respectively. The number of training steps per epoch is comprised of passing over these examples 20 times for training and 4 times for validation, sampling random examples at each step. This provides a tradeoff between training speed and data diversity. We train all our models with a batch size of 2 a learning rate of 10^{-5} with the *Adam* optimiser. We accumulate gradients over 4 batches and use `pytorch` for training.

[2] https://cambridge-mt.com/

Diff-MST-MRSTFT: We generate data using the method 1 described in Section 3 and calculate MRSTFT loss for weight update and backpropogation. We train two variations of the model with a maximum of 8 tracks and 16 tracks as input, each for 1.16 M steps.

Diff-MST-MRSTFT+AF: We fine-tune both versions of the pre-trained Diff-MST-MRSTFT using the synthetically generated data of method 1 in Section 3 with AF loss described in Section 2.6 for 20k steps.

Diff-MST-AF: This method uses real-world songs as a reference for training. We train this model for 1.16 M steps using the AF loss described in Section 2.6. We train with a varying number of tracks with an upper limit of 16.

3.3 Baselines

We compare the performance of our model against three baselines: the equal loudness mix, the mix generated using the pre-trained mixing style transfer model by [16], and two human mixes. We picked three songs from the Cambridge online multitrack repository belonging to the genres of electronic, pop, and metal for our main evaluation. Each of the songs contains between 12 and 22 input tracks. We selected references from popular songs.

Equal Loudness: We loudness normalise the tracks to -48.0 dBFS and taje the mean among the tracks to generate the mix which is then normalised. This generates a loudness-normalised sum of input tracks. We consider this system to be the lowest anchor as it does not consider any style information or mixing transformations.

MST [16]: The method uses a pre-trained source separation model to generate stems from input and reference mix and perform stem-to-stem style transfer using a contrastive learning-based pre-trained audio effect encoder. The stems are mixed using a TCN-based model conditioned on style embeddings. Since the model performs mix-to-mix transformation, we make use of the equal loudness mix of input tracks as the input to be transformed by the model. This allows us to extend the system to perform mixing style transfer for any number of input tracks. This puts the system at a disadvantage as it is trained to work for mix-to-mix scenarios where good-quality mixes are used as input, leading to better-quality extracted stems.

Human Mixes: We asked two audio engineers with professional practice to mix the three songs using the corresponding references. Each of them mixed all three songs until the end of the first chorus.

4. OBJECTIVE EVALUATION

We evaluate the performance of our model against three baselines listed in Section 3.3. For the first evaluation, we compare the mixes generated by all five of our systems described in Section 3.2 and the baselines for three

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	7.85e-02	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 1. 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.

songs belonging to the genres of pop, electronic and metal. We manually picked the songs for the input tracks and the references for each of these cases. A 10-second section ranging between the middle of the first verse to the middle of the first chorus was used for evaluation in Table 2. We loudness normalise the reference mix to -16 dBFS and the predicted mix to -22 dBFS before predicting the metrics. We report the average AF loss and individual weighted audio feature transforms from Section 2.6 for all three songs. Our Diff-MST system trained on real-world songs as reference using AF loss performs the best, closely followed by the MST [16], human engineer mix, and the mix from our Diff-MST-MRSTFT+AF-16 system.

For the second evaluation, we compute average metrics across 100 randomly sampled examples with multitrack taken from the unseen set of Cambridge multitrack and reference songs from MUSEDB18 [25]. We compare the performance of our systems and the baselines MST [16] and the equal loudness system as shown in Table 1. We report individual weighted audio features from the AF loss along with average loss and Frèchet Audio distance (FAD) [26]. The FAD metric is employed to gauge the efficacy of music enhancement approaches or models by comparing the statistical properties of embeddings generated by their output to those of embeddings generated from a substantial collection of clean music. In this context, we analyze the distributions of real-world songs against the mixes generated by various systems using the VGGish model. Again, Diff-MST-MRSTFT+AF-16 outperforms other approaches at capturing the dynamics, spatialisation and spectral attributes of the reference songs.

5. DISCUSSION

Overall, the results indicate the effectiveness of our approach, architecture choice, custom audio production style loss, and novel training regime for the task. The reported metrics for both evaluations show improved performance when trained on a larger number of tracks. Furthermore, we also see that the systems trained or fine-tuned using AF loss generally perform better than those trained with MRSTFT loss, specifically in improving the spatialisation and dynamics of the mixes, thus showing the efficacy of our hand-crafted audio feature-based loss function.

The significant difference in the Bark spectrum values between the equal loudness and our system's mixes suggests

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**	0.185
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	0.186
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	0.180

Table 2. 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, SI in e-02. We have provided audio examples as supplementary material.

that mixes generated using our system have undergone significant spectral processing, resulting in an increased spectral similarity between the reference song and the predicted mix. The metrics indicate inferior performance for the Diff-MST-MRSTFT-8/16 model compared to all our proposed models. This may be attributed to the training data, which is generated using random mixing console parameters, often resulting in mixes that sound unrealistic. However, fine-tuning with AF loss during the last steps notably enhances performance. This improvement could be attributed to AF loss compelling the model to enhance dynamics and spatialization, as evidenced by the reported metrics. We observe a notable enhancement in performance through training on real-world songs, underscoring the significance of high-quality real-world data.

Although the system demonstrates promising outcomes, it is not without its limitations. While we note higher metric values for certain features on the human mixes, this can be explained by the fact that human engineers often strive to capture the overall essence of the reference song. However, they may also incorporate creative elements leading to spatialization and dynamics that diverge significantly from the reference. Our metrics serve to quantify the similarity between the reference song and the predicted mix, which is suitable for the task at hand but may fall short in assessing the creative or unconventional decisions made by human engineers during the mixing process. Additionally, while FAD indicates the predicted audio quality, it mat not capture the intricate nuances involved in the mixing process, such as frequency masking and achieving balance and spatialization.

Moreover, we noticed a decline in the system's mixing capabilities as the number of input tracks increased beyond what it was trained on. Additionally, our mixing console lacks a crucial reverb module essential for comprehensive mixing tasks. Determining the optimal method for processing the entire song poses a challenge, as inferring over the entire song length may result in overly sparse embeddings. Our current system also falls short in modelling mixing context in all possible senses as discussed in [27]. However, we address this challenge by incorporating a reference input, typically selected by the mixing engineer or client. The reference song serves as a proxy for some of the contextual information that engineers typically rely on when making mixing decisions. Lastly, while real-world mixing often entails dynamic adjustments to effect parameters over the course of a song, our system is presently constrained to static mixing configurations.

6. CONCLUSION

In this work, we proposed a framework for mixing style transfer for multitrack music using a differentiable mixing console. Our system is rooted in strong inductive bias, taking inspiration from real-world mixing consoles and channel strips and predicts control parameters for these signal processing blocks allowing interpretability and controllability. Our system supports inputting any number of raw tracks, without source labelling. Furthermore, we circumvent possibilities for audio degradation and artifacts with our design choice for a parameter estimation-based system. Objective evaluations demonstrate that our Diff-MST-MRSTFT+AF-16 system surpasses all baseline methods. The reported metrics give us an insight into the impact of architectural and training design choices. We show that training on a larger number of input tracks improves the performance substantially while running inference on real-world examples that generally contain a larger number of input tracks. We also demonstrate the benefits of training on real-world quality audio examples.

While our research has produced promising results based on objective metrics, it is important to acknowledge our evaluation's constraints, as we have not conducted subjective assessments via listening tests. While objective metrics offer valuable insights into the model's performance, integrating subjective evaluations would provide a more comprehensive understanding of its efficacy in practical applications. Future work includes conducting an extensive subjective evaluation alongside assessing the usability of a prototype of the system that is integrated into the real-world workflow in the digital audio workstation (DAW). Further, work towards developing a robust understanding and objective metrics for mix similarity and mixing style is imperative for enhancing these systems.

7. REFERENCES

[1] M. Miller, *Mixing Music*. London, UK: Dorling Kindersley Ltd, 2016.

[2] R. Izhaki, *Mixing Audio: Concepts, Practices, and Tools*. New York, USA: Routledge, 2017.

[3] B. De Man, J. D. Reiss, and R. Stables, "Ten years of automatic mixing," in *3rd Workshop on Intelligent Music Production*, September 2017.

[4] C. J. Steinmetz, S. S. Vanka, M. A. Martínez Ramírez, and G. Bromham, *Deep Learning for Automatic Mixing*. Bengaluru, India: Proc. of the 23rd Int. Society for Music Information Retrieval Conf. (ISMIR), December 2022. [Online]. Available: https://dl4am.github.io/tutorial

[5] A. Tom, J. D. Reiss, and P. Depalle, "An automatic mixing system for multitrack spatialization for stereo based on unmasking and best panning practices," in *146th Audio Engineering Society Convention*. Dublin, Ireland, UK: Audio Engineering Society, 2019.

[6] D. Moffat and M. Sandler, "Machine learning multitrack gain mixing of drums," in *147th Audio Engineering Society Convention*. Dublin, Ireland, UK: Audio Engineering Society, 2019.

[7] M. A. Martínez-Ramírez, D. Stoller, and D. Moffat, "A deep learning approach to intelligent drum mixing with the Wave-U-Net," *Journal of the Audio Engineering Society*, vol. 69, pp. 142–151, March 2021.

[8] C. J. Steinmetz, J. Pons, S. Pascual, and J. Serrà, "Automatic multitrack mixing with a differentiable mixing console of neural audio effects," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

[9] J. T. Colonel and J. Reiss, "Reverse engineering of a recording mix with differentiable digital signal processing," *The Journal of the Acoustical Society of America*, vol. 150, no. 1, pp. 608–619, 2021.

[10] C. J. Steinmetz, "Learning to mix with neural audio effects in the waveform domain," Master's thesis, Universitat Pompeu Fabra, September 2020.

[11] M. A. Martínez-Ramírez, W.-H. Liao, G. Fabbro, S. Uhlich, C. Nagashima, and Y. Mitsufuji, "Automatic music mixing with deep learning and out-of-domain data," in *Proc. of the 23rd Int. Society for Music Information Retrieval Conf. (ISMIR)*, Bengaluru, India, 2022.

[12] S. Vanka, M. Safi, J.-B. Rolland, and G. Fazekas, "Adoption of AI technology in music mixing workflow: An investigation," in *154th Audio Engineering Society Convention*. Audio Engineering Society, 2023.

[13] 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," *Journal of the Audio Engineering Society*, vol. 72, no. 1/2, pp. 5–15, 2024.

[14] S. Vanka, J.-B. Rolland, and G. Fazekas, "Intelligent music production: Music production style transfer and analysis of mix similarity," in *16th Digital Music Research Network (DMRN+ 16) Workshop*, London, UK, 2021.

[15] C. J. Steinmetz, N. J. Bryan, and J. D. Reiss, "Style transfer of audio effects with differentiable signal processing," *Journal of the Audio Engineering Society*, vol. 70, no. 9, pp. 708–721, September 2022.

[16] J. Koo *et al.*, "Music mixing style transfer: A contrastive learning approach to disentangle audio effects," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Rhodes Island, Greece: IEEE, April 2023, pp. 1–5.

[17] J. Pons, S. Pascual, G. Cengarle, and J. Serrà, "Upsampling artifacts in neural audio synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3005–3009.

[18] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *International Conference on Learning Representations*, 2020.

[19] B. Man, B. Leonard, R. King, J. D. Reiss *et al.*, "An analysis and evaluation of audio features for multitrack music mixtures," in *Proc. of the 15th Int. Society for Music Information Retrieval Conf. (ISMIR)*, Taipei, Taiwan, 2014.

[20] X. Wang, S. Takaki, and J. Yamagishi, "Neural source-filter waveform models for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 402–415, 2019.

[21] C. J. Steinmetz and J. D. Reiss, "auraloss: Audio focused loss functions in pytorch," in *Digital music research network one-day workshop (DMRN+ 15)*, London, UK, December 2020.

[22] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "MedleyDB: A multitrack dataset for annotation-intensive mir research." in *Proc. of the 15th Int. Society for Music Information Retrieval Conf. (ISMIR)*, vol. 14, Taipei, Taiwan, 2014, pp. 155–160.

[23] R. M. Bittner *et al.*, "MedleyDB 2.0: New data and a system for sustainable data collection," in *Proc. of the 17th Int. Society for Music Information Retrieval Conf. (ISMIR)*, New York, USA, 2016.

[24] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, and X. Serra, "The MTG-Jamendo dataset for automatic music tagging," in *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning*, Long Beach, CA, United States, 2019.

[25] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," December 2017.

[26] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet Audio Distance: A Reference-Free Metric for Evaluating Music Enhancement Algorithms," in *Proc. Interspeech 2019*, 2019, pp. 2350–2354.

[27] M. N. Lefford, G. Bromham, G. Fazekas, and D. Moffat, "Context-aware intelligent mixing systems," *Journal of the Audio Engineering Society*, vol. 69, pp. 128–141, March 2021.