ST-ITO: CONTROLLING AUDIO EFFECTS FOR STYLE TRANSFER WITH INFERENCE-TIME OPTIMIZATION

Christian J. Steinmetz **Shubhr Singh** **Marco Comunità** **Ilias Ibnyahya**
Shanxin Yuan **Emmanouil Benetos** **Joshua D. Reiss**
Centre for Digital Music, Queen Mary University of London, UK

ABSTRACT

Audio production style transfer is the task of processing an input to impart stylistic elements from a reference recording. Existing approaches often train a neural network to estimate control parameters for a set of audio effects. However, these approaches are limited in that they can only control a fixed set of effects, where the effects must be differentiable or otherwise employ specialized training techniques. In this work, we introduce **ST-ITO**, Style Transfer with Inference-Time Optimization, an approach that instead searches the parameter space of an audio effect chain at inference. This method enables control of arbitrary audio effect chains, including unseen and non-differentiable effects. Our approach employs a learned metric of audio production style, which we train through a simple and scalable self-supervised pretraining strategy, along with a gradient-free optimizer. Due to the limited existing evaluation methods for audio production style transfer, we introduce a multi-part benchmark to evaluate audio production style metrics and style transfer systems. This evaluation demonstrates that our audio representation better captures attributes related to audio production and enables expressive style transfer via control of arbitrary audio effects.

1. INTRODUCTION

Audio effects are signal processing devices used to transform or manipulate audio signals, such as adding reverberation, adjusting frequency balance with equalization, or adding edge with distortion. They play a central role in audio production, providing audio engineers with the ability to realize both practical and creative tasks with applications in music, film, broadcast, and video games [1]. Traditionally, operating these effects requires a significant amount of expertise, as audio engineers must combine a technical understanding with their artistic goals. As a result, the process of creating a high-quality audio production remains challenging, requiring a time-consuming process for professionals and a significant barrier for novices.

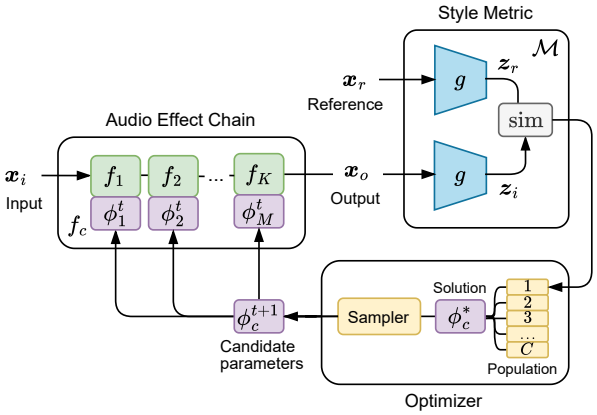

Figure 1. Style transfer with Inference-Time Optimization enables audio production style transfer through control of arbitrary audio effects. It employs a pretrained audio representation as a similarity metric, which is then optimized by searching the control parameter space of audio effects.

Intelligent music production aims to develop systems for automating aspects of audio engineering [2]. Early approaches employed rule-based systems, using hand-engineered rules based on best practices [3]. These system often generated outputs that satisfied certain assumptions or utilized well established conventions. However, the inability to construct sufficient sophisticated rule bases has motivated machine learning approaches, which instead learn from data without assuming a limited or fixed set of rules [4–7]. Nevertheless, these systems still lacked the ability to adapt based on user input, which is critical to the context-dependent nature of music production [8].

To address the context-dependent nature of this task and enable greater user control, *audio production style transfer* has been proposed [9, 10]. These systems rely on a reference recording and attempt to map elements of the audio production style from the reference onto the input. These systems either directly process the audio signal [11–13] or estimate parameters for audio effects [9, 10, 14, 15]. While direct transformation methods are powerful, they may introduce artifacts and lack grounding in traditional audio tools. Similarly, recent text-to-audio generation models also enable editing capabilities [16, 17], but suffer from the same limitations. On the other hand, parameter based methods enable efficient and controllable style transfer. However, current systems are limited to a fixed chain of effects, and require the use of differentiable signal processing [18], or inefficient alternative differentiation strategies such as gradient approximation [19] and neural proxies [7].

We propose a method to construct an audio production style transfer system that leverages inference-time optimization to facilitate real world applications. Instead of training a network to perform style transfer directly, we perform style transfer via an optimization process at inference, as shown in Figure 1. We iteratively search the parameter space of an effect chain with our proposed metric that measures the similarity in audio production style between the output recording and the reference. This approach enables the ability to control arbitrary audio effect chains, including non-differentiable effects, opening up the potential to control real-world audio effects. The contributions of our work are as follows:

- A simple and scalable pretraining strategy for constructing an audio production style similarity metric through audio effect estimation, named AFx-Rep.
- A system for audio production style transfer, ST-ITO, that optimizes the control parameters of arbitrary audio effects according to a similarity metric.
- An extension of the DeepAFx-ST system [14] with the addition of differentiable distortion and reverberation, which forms a strong baseline.
- A multi-task benchmark for evaluation of audio production style similarity metrics and audio production style transfer systems.

We provide audio examples, and open source our datasets, benchmark, and code to facilitate reproducibility [1] .

2. METHOD

In this work, we propose **ST-ITO**, Style Transfer with Inference-Time Optimization, a novel method for audio production style transfer that searches the parameter space of a set of audio effects to perform style transfer. As shown in Figure 1, our system features three main components: an audio effect chain that processes an input recording, an audio production style similarity metric, composed of pretrained encoder and a similarity measure, and an optimizer that is used to find control parameters. This enables style transfer by finding a configuration of the audio effects that produce an output with attributes of the reference style.

Our approach provides a number of benefits as compared to previous audio production style transfer systems. First, it enables the control of arbitrary audio effects, even those that have not been seen during training. Unlike existing systems that train with a set of fixed effects "in-the-loop", our approach enables adaptation to new effects at inference. Furthermore, our method removes requirements of previous systems. This includes removing the need for differentiable audio effects or alternative differentiation strategies, which can often be slow and difficult to train [14]. Finally, we provide further flexibility and control by enabling the addition or removal of audio effects at inference without re-training of the base model.

[1] https://github.com/csteinmetz1/st-ito

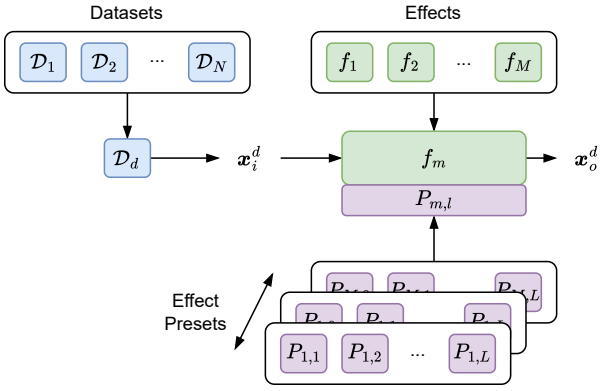

Figure 2. Self-supervised training for the pretext task where an audio signal $\boldsymbol{x}_i^d \sim \mathcal{D}_d$ is sampled randomly from one of N datasets and then processed by a randomly sampled audio effect E_m with an associated randomly sampled parameter preset $P_{m,l}$ to produce an output signal $\boldsymbol{x}_0^d$.

2.1 Audio production style metric

A central aspect of our approach is the development of an audio production style metric $\mathcal{M}(\boldsymbol{x}_a, \boldsymbol{x}_b)$. This metric measures the perceptual similarity in audio production between two recordings $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. As explained in Section 2.2, we optimize this metric by searching the parameter space of a set of audio effects to align the style of the processed recording with the reference. In general, production style relates to aspects of audio quality rather than the underlying content, including attributes such as dynamics, frequency balance, and the stereo field [20].

Pretrained audio representations. There is a growing body of work in general purpose representations of audio signals [21], popular approaches include CLAP [22] and BEATs [23]. These representations capture relevant attributes to facilitate downstream tasks such as detection and classification of sound sources and events. While it may be possible to directly adapt one of these representations for our task, evidence suggests they are not always sensitive to audio effect transformations [24]. We provide further evidence for this in Section 5. This motivates us to develop a method to produce our own audio representation that is more sensitive to audio effect transformations.

Self-supervised pretext task. We propose a simple and scalable self-supervised pretext task to construct an audio representation for our task without human annotated data. To encourage the encoder to extract features related to audio effects we employ an audio effect classification task composed of two parts. The model predicts both which effect has been applied and the associated preset.

As shown in Figure 2, we generate training examples using N audio datasets, a set of M audio effects, and L associated parameter presets for each effect. To generate a training example, a dataset $\mathcal{D}_d$ is selected at random from the set of datasets. Then one audio sample is selected from this dataset, which will form the input recording $\boldsymbol{x}_i^d$. Next, we sample an audio effect f_m and a random associated preset $P_{m,l}$ to configure the effect. Then we process the input with this effect to produce an output signal $\boldsymbol{x}_o^d$.

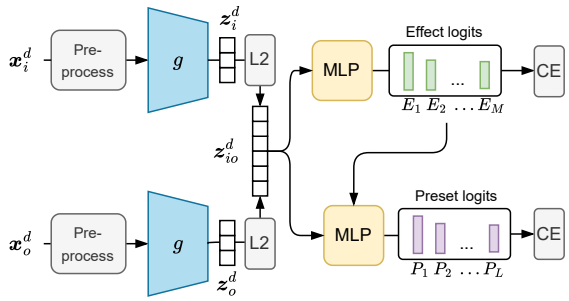

Figure 3. Representation learning via the pretext task where the input $\boldsymbol{x}_i^d$ and output $\boldsymbol{x}_o^d$ are processed by the encoder $g(\cdot)$ to produce embeddings. These embeddings are fed to a pair of MLP classifiers trained via cross-entropy that predict the effect class and preset class.

Training is shown in Figure 3 where the encoder $g(\cdot)$ extracts embeddings $\boldsymbol{z}_i$ and $\boldsymbol{z}_r$ from the input and output. These embeddings are concatenated and fed to a multilayer perceptron (MLP) that estimates the effect applied. A second MLP takes the effect logits as well as the embeddings to estimate the preset. After pretraining we discard the prediction heads and use the encoder $g(\cdot)$, which we refer to as AFx-Rep, to produce embeddings.

This multi-task formulation encourages the encoder to extract features not only about effects but also subtleties between different configurations of the same effect, which is important for style transfer. Our approach does not enforce invariance to the content, but can leverage any audio, not only unprocessed or effect normalized audio [25], required by previous work [13]. This allows us to further scale the training dataset size. In addition, since this audio may already contain other processing, our model is exposed to a wide range of effects beyond those we apply.

2.2 Inference-time optimization

To perform style transfer we begin with input $\boldsymbol{x}_i$ and reference $\boldsymbol{x}_r$ recordings. We assume the input has minimal processing, as our system does not remove effects [26,27]. Then, we require the user to provide an appropriate chain of audio effects to be controlled. This chain can be represented as the composition of K audio effects where each effect is represented by a function f_k parameterized by a control vector ϕ_k. The output $\boldsymbol{x}_o$ is obtained by sequentially applying these functions to the input, resulting in

$$\boldsymbol{x}_o = f_K(f_{K-1}(\dots f_2(f_1(\boldsymbol{x}_i;\phi_1);\phi_2)\dots;\phi_{K-1});\phi_K). \tag{1}$$

For convenience, we represent this chain as a single function $\boldsymbol{x}_o = f_c(\boldsymbol{x}_i;\phi_c)$, where $\phi_c = [\phi_1,\phi_2,\dots,\phi_K]$ concatenates all effect parameters into one vector. While our method supports arbitrarily complex effect chains, we consider only series connections.

In our setup, we perform style transfer through an optimization process via the maximization of a similarity between the output of our composite audio effect function and the reference signal given by

$$\max_{\phi_c} \ \text{sim}\Big(g(f_c(\boldsymbol{x}_i;\phi_c)), g(\boldsymbol{x}_r)\Big), \tag{2}$$

where $g(\cdot)$ denotes our audio representation, transforming audio signals into a feature space where audio production similarity is assessed. For the reference signal $\boldsymbol{x}_r$, the feature representation is $\boldsymbol{z}_r = g(\boldsymbol{x}_r)$. The optimization process initiates with a predefined set of control parameters, ϕ_0, and iteratively refines this estimate to enhance the similarity measure. At each step, candidate solutions are generated and evaluated based on their performance in mirroring the reference features, $\boldsymbol{z}_r$. This performance is quantified by the cosine similarity measure,

$$\text{sim}(\boldsymbol{z}_i,\boldsymbol{z}_r) = \frac{\boldsymbol{z}_i \cdot \boldsymbol{z}_r}{\max(\|\boldsymbol{z}_i\|\|\boldsymbol{z}_r\|,\epsilon)}, \tag{3}$$

where $\boldsymbol{z}_i = g(f_c(\boldsymbol{x}_i;\phi_c))$ is the feature vector of the processed input signal, $\cdot$ represents the dot product, $\|\cdot\|$ denotes the Euclidean norm, and ϵ is a small constant ensuring numerical stability, avoiding division by zero.

3. EXPERIMENTAL DETAILS

3.1 Pretraining

We employ the PANNs architecture [28] as a convolutional backbone. While initial testing indicated that more modern architectures such as HTS-AT [29] performed comparably, we found that PANNs was more efficient at inference. To enable the encoder to capture stereo information we produce separate embeddings for the mid and side signals, concatenating them into a single embedding, applying L2 normalization to each embedding before concatenation.

We train the encoder following the pretext task described in Section 2.1. We use seven publicly available audio datasets to cover a diverse range of audio content across music, speech, singing voice, and instruments. These datasets include MTG-Jamendo [30], ENST-Drums [31], URSing [32], FSD50k [33], Librispeech [34], Medley-solos-db [35], and GuitarSet [36]. To construct our set of audio effects we use 63 open source or freely available VST3 audio plugins compiled for Linux. These VSTs cover a wide range of effects including reverberation, dynamic range processing, equalization, distortion, modulation effects, and more. We use `pedalboard` [37] to load plugins and apply effects to audio signals.

We generate unique presets for each effect by randomly sampling 1000 parameter configurations and processing a random audio recording with each configuration. Then we extract MFCCs and perform K-means clustering ($K = 10$), with each cluster representing perceptually diverse parameter configurations. We then randomly select one configuration from each cluster to act as a preset.

While training with on-the-fly data generation is possible, we found running VSTs during training caused a significant bottleneck. We opted for offline data generation, where we generated 20,000 examples of length 524288 samples (≈ 11 sec at $f_s = 48$ kHz) from each dataset with randomly applied effects and presets. This corresponds to approximately 60 hours of audio content. We further increase diversity during training by taking different random crops of the pre-processed input and output segments, as well as applying random gain adjustments $[-32\,\text{dB}, 0\,\text{dB}]$.

We perform pretraining for 1 M steps with a batch size of 32 using the Adam optimizer. We use an initial learning rate of 1e-4, lowering the learning rate by a factor of 10 at 85% and 95% through training. We preprocess spectrogram inputs to the encoder by computing log-melspectrograms with window size of 2048 and hop size of 512. We clip the magnitudes between -80 and 40 dB and scale the final spectrogram between -1 and 1.

3.2 Inference-time optimization

To enable control of arbitrary effects, we employ a gradient-free optimizer as opposed to commonly used gradient-based solutions. While any gradient-free optimizer can be used in our system, we opt to use Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [38] since it has been shown to work well in spaces with similar dimensionality to the audio effect chain control parameter space (≈100) and is a relatively scalable method. After initial hyperparameter tuning, we use a population size of 64 and a maximum of 25 optimization steps. The σ hyperparameter is initialized to 0.3 and we use a fixed initialization of all parameters of 0.5 scaled in the range $[0, 1]$. We use early stopping, halting optimization after 10 steps of improvement less than 0.1.

Unless otherwise specified, we use AFx-Rep as the encoder in our similarity metric, and we consider control of two different audio effect chains. The first features five VST audio effects including distortion (TubeScreamer), parametric equalizer (ZamEQ2), dynamic range compressor (ZamCompX2), feedback delay (ZamDelay), and artificial reverberation (TAL-Reverb-4), resulting in a total of 73 parameters. The second chain features unseen audio effects internal to `pedalboard`, including distortion, dynamic range compression, parametric equalizer, delay, and artificial reverberation, resulting in 36 parameters.

4. BENCHMARK

4.1 Audio production style metrics

Zero-shot style classification. We adapt the style classification task from [14], which contains five different audio production styles using equalization and dynamic range compression: telephone (TL), bright (BR), warm (WM), broadcast (BC), neutral (NT). Training examples are generated by applying these style presets to speech from DAPS [39] and music from MUSDB18 [40]. To make the task more challenging and similar to the inference-time optimization use-case, we adapt the original task to the zero-shot case [41]. To do so, a query is constructed by sampling a random audio example to be classified as one of the five styles. Then other examples from each of the five styles are sampled randomly to form prototype classes. A representation of the query and each of the five prototypes is generated and a prediction is made by measuring the cosine similarity between the query and each of the prototypes. The class of the prototype with the highest similarity to the query forms the prediction.

	Styles					
Representation	TL	BR	WM	BC	NT	AVG
MFCCs	**1.00**	0.82	0.64	0.74	0.48	0.74
MIR Feats.	0.76	0.64	0.61	0.58	0.32	0.58
CLAP	0.72	0.60	0.51	0.57	0.41	0.56
Wav2Vec2	0.40	0.33	0.28	0.35	0.34	0.34
Wav2Clip	0.76	0.48	0.60	0.49	0.51	0.57
VGGish	0.47	0.58	0.43	0.61	0.43	0.50
BEATs	0.94	0.51	0.57	0.50	0.45	0.59
FX Encoder	0.96	0.94	0.29	0.70	0.54	0.69
DeepAFx-ST	**1.00**	0.93	0.67	0.78	0.42	0.76
DeepAFx-ST+	**1.00**	0.97	0.71	0.79	0.41	0.78
AFx-Rep (ours)	**1.00**	**1.00**	**0.88**	**0.85**	**0.59**	**0.86**

Table 1. Zero-shot style classification accuracy over 200 trials for music and speech across five unique styles.

Style retrieval. While the zero-shot style classification task evaluates the ability of a representation to differentiate among different styles, it considers only two basic effects and focuses on comparing significant differences. In order to more effectively evaluate the behavior of representations in a scenario similar to style transfer we designed a style retrieval task. In this task, a query style is produced by applying N effects with random parameters to an audio recording. A retrieval set is generated by processing M other recordings with differing random effect chains. One recording with differing content but the same effect chain as the query is included in the retrieval set. Similar to the zero-shot task, we measure the cosine similarity between the query and each of the items in the retrieval set. We can make the task more or less difficult by varying both the number of effects N in each style and the size of the retrieval set M. We source unseen audio examples for speech (DAPS [39]), guitar (IDMT-SMT-Guitar [42]), vocals (VocalSet [43]), and drums (IDMT-SMT-Drums [44]).

Baselines. We consider signal processing approaches, such as MFCCs and MIR features [45], as well as pretrained general purpose audio representations including VGGish [46], WAV2CLIP [47], wav2vec2.0 [48], CLAP [22], as well as BEATs [23]. We also compare against audio effect specific models including FX-Encoder [13] and the DeepAFx-ST encoder [14].

4.2 Audio production style transfer

Parameter estimation. To demonstrate the ability of our proposed approach to control a wide range of effects we design a parameter estimation task. We initialize an audio effect and set a target value for one parameter, processing a random audio signal to generate a reference. Then we sample another random recording to use as the input. We then run the optimization using each audio representation in our metric. To achieve accurate style transfer a system should estimate a control parameter with a similar, but not necessarily identical value as the reference. We report both the mean squared error (MSE) and the correlation coefficient ρ of estimated parameters. We consider six VST effects as well as six unseen effects from `pedalboard`.

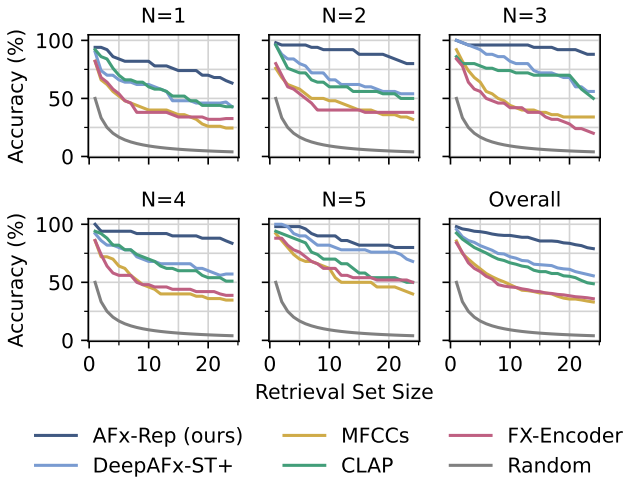

Figure 4. Accuracy for the style retrieval task using different audio representations across multiple source types with varying number of audio effects (N) and retrieval set size.

Real-world style transfer. While the parameter estimation task can demonstrate the ability of a style transfer system to control a singular audio effect, it does not capture the ability of the system to perform in a real-world scenario. In many cases multiple effects will be present at the same time, making the task more challenging. To evaluate this scenario we created six different audio production styles by constructing realistic audio effect chains of varying complexity in a digital audio workstation. These styles range from simple lowpass and highpass filtering to a complete channel strip featuring equalization, distortion, compression, delay, and reverberation. We then applied these styles to a range of audio content including speech, singing voice, and full music tracks. Each style transfer system is then tasked with transforming the unprocessed audio from one of these content types to the stylized version of a different recording containing the same kind of content.

Baselines. We compare our proposed style transfer system to both deep learning and signal processing solutions. We use a rule-based approach from previous work that includes an FIR matching equalization filter and a simple hill climbing-based dynamic range compressor [14]. We construct a strong deep learning baseline by extending DeepAFx-ST with differentiable reverberation [49] and distortion [50] effects, using `dasp-pytorch` [2]. We call this approach DeepAFx-ST+ and we train this model following the approach from the original work, but using the same datasets used to train our audio representation.

5. RESULTS

Zero-shot style classification. We evaluate the pretrained representations across ten trials for each of the five different styles. The class-wise and overall accuracy is reported in Table 1. First, we find MFCC based features perform better than expected, with high accuracy on the telephone (TL), bright (BR), and warm (WM) styles. However, performance is worse on broadcast (BC) and neutral (NT), likely because identifying these styles requires paying attention to dynamics. The MIR features do not achieve comparable performance. All of the general purpose audio representations perform worse than MFCCs on this task, with CLAP and BEATs appearing to perform best among them, but with an average accuracy 15 points lower. This confirms the hypothesis that general purpose representations fail to capture information about audio effects. FX-Encoder and DeepAFx-ST(+) perform better than the other pretrained models, with DeepAFX-ST variants outperforming MFCCs. Overall, we find that our proposed model, AFx-Rep, performs best in this task.

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

	MSE (↓)		ρ (↑)	
Effect (Parameter)	CLAP	AFx-Rep	CLAP	AFx-Rep
RoughRider (sensit)	0.183	**0.084**	0.300	**0.705**
DPlate (decay)	0.141	**0.025**	0.610	**0.945**
3BandEQ (high_)	0.033	**0.026**	0.876	**0.919**
MaGigaverb (size)	0.018	**0.012**	0.949	**0.969**
MetalTone (dist)	0.155	**0.040**	0.509	**0.862**
TAL-Chorus (wet)	0.097	**0.014**	0.654	**0.953**
*Chorus (mix)	**0.164**	0.172	0.300	**0.408**
*Reverb (room_)	0.048	**0.013**	0.822	**0.955**
*Delay (mix)	0.117	**0.052**	0.591	**0.815**
*Distortion (drive)	0.023	**0.005**	0.852	**0.944**
*Compressor (thresh)	0.134	**0.096**	0.518	**0.678**
*ParametricEQ (low_s)	0.110	**0.031**	0.727	**0.931**

Table 2. Parameter estimation with ST-ITO using CLAP and our proposed AFx-Rep. We report the mean squared error (MSE) and correlation coefficient (ρ) of the estimated parameters across 4 different settings and 3 trials per effect. Audio effects not seen during pretraining are denoted by $*$.

Style retrieval. We report the accuracy for a subset of methods in style retrieval as shown in Figure 4. We plot performance across differing number of effects N that constitute a style, as well as the size of the retrieval set, shown on the x-axis. As expected, for all scenarios, as the retrieval set grows the classification performance drops. While all methods are better than random guessing, we observe that MFCCs and FX-Encoder appear to perform worse. They are followed by CLAP and then the encoder from DeepAFx-ST+, which slightly outperforms CLAP. Finally, our proposed AFx-Rep model performs best across all scenarios, indicating its superior ability to capture elements related to audio production style.

Parameter estimation. In Table 2 we report the mean squared error (MSE) and correlation coefficient (ρ) in parameter estimation using ST-ITO with either CLAP or our proposed AFx-Rep model. In nearly all cases our AFx-Rep model functions as a superior similarity metric, achieving lower MSE and a higher correlation coefficient, with the exception of the MSE in Chorus, which appears to be challenging for both models. This demonstrates the ability of our approach to control a wide range of real-world audio effects, including effects not seen during retraining. These results reinforce the importance of an audio representation sensitive to audio effects, such as our proposed AFx-Rep.

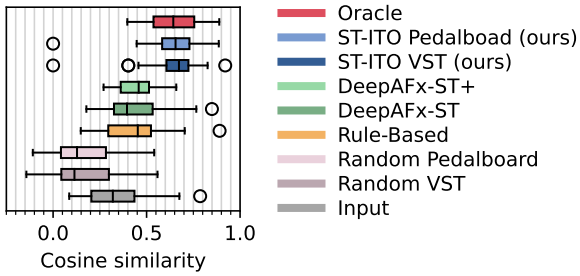

Figure 5. AFx-Rep similarity in real-world style transfer.

Real world style transfer. We report the similarity from our metric using AFx-Rep across 56 style transfer trials (Figure 5). The Input processed with an audio effect chain identical to the reference is also evaluated, which we refer to as Oracle. Note that the Oracle may not achieve effective style transfer as the starting point of the Input may require a different parameter configuration to match the reference. The random configuration of VSTs and `pedalboard` effect perform worse, and are followed by the Input, which features no processing. DeepAFx-ST, DeepAFx-ST+, and the Rule-Based system appear to perform similarly to each other, but better than Input. Variants of ST-ITO, one using VSTs and the other using unseen `pedalboard` effects, both outperform the rest, and are on par with the Oracle. This indicates the ability of our approach to optimize our metric, however, it is difficult to make conclusions about style transfer performance using this evaluation alone.

Subjective listening study. We recruited 23 participants with experience in audio engineering. They were tasked with evaluating style transfer systems on real-world test scenarios in a multiple stimulus listening study. Listeners were asked to provide a score from 0 to 100 for each stimulus to indicate its similarity to the reference, considering only the style and not the underlying content. In addition, we also included the unprocessed Input and the Oracle. Due to the subjectivity of this task, evaluators may not rate Input the lowest and Oracle the highest. We selected ten test cases across the real-world scenarios including vocals (V), music (M), and speech (S) as shown in Figure 6.

Overall, listeners found the Oracle most similar to the reference and the Input the least similar, as expected. However, there is variation in the score assigned to both, indicating some disagreement. For simple styles, such as V1 (lowpass) and M2 (highpass), we found the Rule-Based system worked well, even surpassing style transfer systems. However, in cases with multiple effects, the Rule-Based system does not work well, as in V2 (large space), V3 (small space), V4 (delay), S1 (small space), and S3 (distortion). Differences between ST-ITO and DeepAFx-ST+ are harder to discern. Our method outperforms in some cases, such as V2 (large space), V3 (small space), and V4 (delay), yet in in other cases there is no clear difference. We conclude that our approach is capable of style transfer at least on par with the enhanced DeepAFx-ST+, and does so controlling a chain of unseen VST audio effects, which is not possible with other approaches.

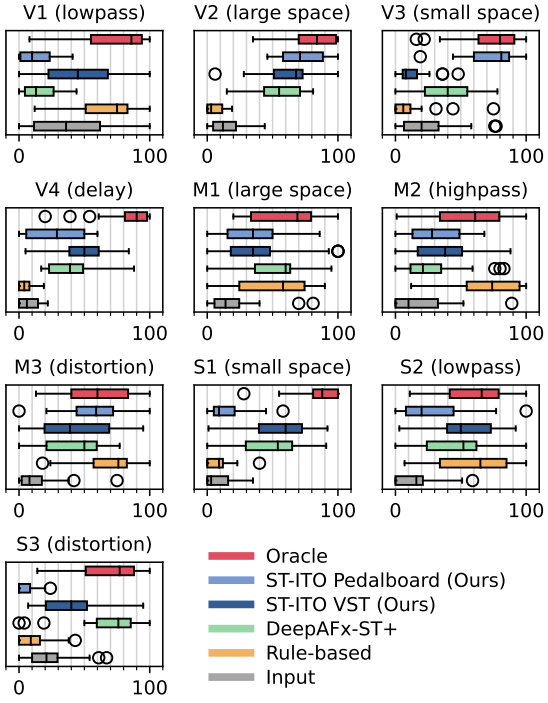

Figure 6. Subjective scores from $N = 23$ participants across vocals (V), music (M), and speech (S) examples.

6. DISCUSSION

While ST-ITO enables control of arbitrary audio effects and adapts to new effects at inference, it has some limitations. In the current formulation, our system requires an appropriate audio effect chain be provided. Future work could consider automatically constructing this audio processing graph as in blind estimation [51]. Furthermore, while our method does not require training "in-the-loop" with audio effects during representation learning, we must process many variants of the recording through the audio effect chain during style transfer. This leads to significantly longer inference times ($\approx$ 1 min) as compared to networks that estimate parameters directly ($\approx$ 1 sec). Future work could consider the design of more efficient optimizers through meta-learning by training an optimizer for a particular effect chain [52]. Finally, we have found that the current system does not work well for challenging style transfer applications, such as guitar tone matching.

7. CONCLUSION

In this work, we introduced ST-ITO, Style Transfer with Inference-Time Optimization. Unlike previous style transfer systems, ST-ITO searches the parameter space of any audio effect chain at inference, enabling control of arbitrary effect chains, including those with non-differentiable effects. Our methodology leverages a self-supervised audio production style metric and a gradient-free optimizer. We developed a set of benchmarks to evaluate both audio production style representations and style transfer systems. Results from this set of benchmarks indicate that our approach not only better captures details related to audio production style, but also provides enhanced flexibility and expressiveness in audio production style transfer.

8. ACKNOWLEDGMENTS

Supported by EPSRC UKRI CDT in AI+Music (Grant no. EP/S022694/1). EB is supported by RAEng/Leverhulme Trust research fellowship LTRF2223-19-106.

9. REFERENCES

[1] T. Wilmering, D. Moffat, A. Milo, and M. B. Sandler, "A history of audio effects," *Applied Sciences*, vol. 10, no. 3, p. 791, 2020.

[2] B. De Man, R. Stables, and J. D. Reiss, *Intelligent Music Production*. Focal Press, 2019.

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

[4] J. Scott, M. Prockup, E. M. Schmidt, and Y. E. Kim, "Automatic multi-track mixing using linear dynamical systems," in *Proceedings of the 8th Sound and Music Computing Conf., Padova, Italy*, 2011.

[5] D. Moffat and M. Sandler, "Machine learning multitrack gain mixing of drums," in *147th Convention of the Audio Engineering Society Convention*, 2019.

[6] M. Martinez Ramirez, D. Stoller, and D. Moffat, "A deep learning approach to intelligent drum mixing with the Wave-U-Net," *Journal of the Audio Engineering Society*, 2021.

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

[8] M. N. Lefford, G. Bromham, G. Fazekas, and D. Moffat, "Context aware intelligent mixing systems," *Journal of the Audio Engineering Socieity*, 2021.

[9] D. Sheng and G. Fazekas, "A feature learning siamese model for intelligent control of the dynamic range compressor," in *IEEE Intl. Joint Conf. on Neural Networks (IJCNN)*, 2019.

[10] S. I. Mimilakis, N. J. Bryan, and P. Smaragdis, "One-shot parametric audio production style transfer with application to frequency equalization," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[11] J. Koo, S. Paik, and K. Lee, "Reverb conversion of mixed vocal tracks using an end-to-end convolutional deep neural network," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[12] ——, "End-to-end music remastering system using self-supervised and adversarial training," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[13] J. Koo *et al.*, "Music mixing style transfer: A contrastive learning approach to disentangle audio effects," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[14] C. J. Steinmetz, N. J. Bryan, and J. D. Reiss, "Style transfer of audio effects with differentiable signal processing," *J. Audio Eng. Soc*, vol. 70, no. 9, 2022.

[15] C. Peladeau and G. Peeters, "Blind estimation of audio effects using an auto-encoder approach and differentiable digital signal processing," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.

[16] Y. Wang *et al.*, "Audit: Audio editing by following instructions with latent diffusion models," *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[17] B. Han *et al.*, "InstructME: An instruction guided music edit and remix framework with latent diffusion models," *arXiv preprint arXiv:2308.14360*, 2023.

[18] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *ICLR*, 2020.

[19] M. A. M. Ramírez, O. Wang, P. Smaragdis, and N. J. Bryan, "Differentiable signal processing with black-box audio effects," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[20] 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, 2024.

[21] J. Turian, J. Shier *et al.*, "Hear: Holistic evaluation of audio representations," in *NeurIPS 2021 Competitions and Demonstrations Track*. PMLR, 2022.

[22] Y. Wu, K. Chen, T. Zhang, Y. Hui, T. Berg-Kirkpatrick, and S. Dubnov, "Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[23] S. Chen *et al.*, "BEATs: Audio pre-training with acoustic tokenizers," in *ICML*, vol. 202. PMLR, 2023.

[24] S. H. Hawley and C. J. Steinmetz, "Leveraging neural representations for audio manipulation," in *154th Convention of the Audio Engineering Society*, 2023.

[25] M. A. Martínez-Ramírez *et al.*, "Automatic music mixing with deep learning and out-of-domain data," in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, December 2022.

[26] J. Imort, G. Fabbro, M. A. M. Ramírez, S. Uhlich, Y. Koyama, and Y. Mitsufuji, "Distortion audio effects: Learning how to recover the clean signal," in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2022.

[27] M. Rice, C. J. Steinmetz, G. Fazekas, and J. D. Reiss, "General purpose audio effect removal," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2023.

[28] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, 2020.

[29] K. Chen *et al.*, "HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[30] D. Bogdanov, M. Won, P. Tovstogan, A. Porter, and X. Serra, "The MTG-Jamendo dataset for automatic music tagging," in *ICML*, 2019.

[31] O. Gillet and G. Richard, "ENST-Drums: an extensive audio-visual database for drum signals processing," in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2006.

[32] B. Li *et al.*, "University of Rochester Audio-Visual Solo Singing Performance Dataset," 2022.

[33] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, 2021.

[34] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *IEEE Intl. Conf. on Acoustics, Speech and Signal processing (ICASSP)*, 2015.

[35] V. Lostanlen, C.-E. Cella, R. Bittner, and S. Essid, "Medley-solos-db: a crosscollection dataset for musical instrument recognition," *Zenodo*, 2018.

[36] Q. Xi, R. M. Bittner, J. Pauwels, X. Ye, and J. P. Bello, "Guitarset: A dataset for guitar transcription." in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2018.

[37] P. Sobot, "Pedalboard," Jul. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.7817838

[38] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *IEEE Intl. Conf. on Evolutionary Computation*, 1996.

[39] G. J. Mysore, "Can we automatically transform speech recorded on common consumer devices in real-world environments into professional production quality speech?—a dataset, insights, and challenges," *IEEE Signal Processing Letters*, vol. 22, no. 8, 2014.

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

[41] P. Wolters, C. Careaga, B. Hutchinson, and L. Phillips, "A study of few-shot audio classification," *arXiv preprint arXiv:2012.01573*, 2020.

[42] C. Kehling, J. Abeßer, C. Dittmar, and G. Schuller, "Automatic tablature transcription of electric guitar recordings by estimation of score-and instrument-related parameters." in *DAFx*, 2014.

[43] J. Wilkins, P. Seetharaman, A. Wahl, and B. Pardo, "Vocalset: A singing voice dataset." in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2018.

[44] C. Dittmar and D. Gärtner, "Real-time transcription and separation of drum recordings based on nmf decomposition." in *DAFx*, 2014, pp. 187–194.

[45] B. Man, B. Leonard, R. King, J. D. Reiss *et al.*, "An analysis and evaluation of audio features for multitrack music mixtures," in *Intl. Society for Music Information Retrieval Conf. (ISMIR)*, 2014.

[46] S. Hershey *et al.*, "Cnn architectures for large-scale audio classification," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[47] H.-H. Wu *et al.*, "Wav2clip: Learning robust audio representations from clip," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[48] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, 2020.

[49] C. J. Steinmetz, V. K. Ithapu, and P. Calamia, "Filtered noise shaping for time domain room impulse response estimation from reverberant speech," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2021.

[50] J. T. Colonel, M. Comunità, and J. Reiss, "Reverse engineering memoryless distortion effects with differentiable waveshapers," in *153rd Convention of the Audio Engineering Society*, 2022.

[51] S. Lee, J. Park, S. Paik, and K. Lee, "Blind estimation of audio processing graph," in *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[52] J. Casebeer, N. J. Bryan, and P. Smaragdis, "Meta-af: Meta-learning for adaptive filters," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, 2022.