Audio Engineering Society

Convention Express Paper 41

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

Leveraging Neural Representations for Audio Manipulation

Scott H. Hawley[1,3] and Christian J. Steinmetz[2]

[1]*Belmont University, Nashville, TN, USA*
[2]*Centre for Digital Music, Queen Mary University of London, UK*
[3]*Harmonai, USA*

Correspondence should be addressed to Scott H. Hawley (scott.hawley@belmont.edu)

ABSTRACT

We investigate applying audio manipulations using pretrained neural network-based autoencoders as an alternative to traditional signal processing methods, since the former may provide greater semantic or perceptual organization. To establish the potential of this approach, we first establish if representations from these models encode information about manipulations. We carry out experiments and produce visualizations using representations from two different pretrained autoencoders. Our findings indicate that, while some information about audio manipulations is encoded, this information is both limited and encoded in a non-trivial way. This is supported by our attempts to visualize these representations, which demonstrated that trajectories of representations for common manipulations are typically nonlinear and content dependent, even for linear signal manipulations. As a result, it is not yet clear how these pretrained autoencoders can be used to manipulate audio signals, however, our results indicate this may be due to the lack of disentanglement with respect to common audio manipulations.

1 Introduction

Musical audio production workflows use a variety of parameterized transformations to perform the processing and re-synthesis of audio signals. Examples include the sliders on a multi-band equalizer, dynamic range changes made by adjusting gain or compression, spectral processing, and adjustments made by changing MIDI parameters. The development of increasingly "intelligent" music interfaces may be regarded as a pursuit to find transformations yielding *representations* that more closely match the perceptual or semantic content of interest to music producers, such as fewer knobs that control high-level aspects of the sound [1].

Neural network-based audio autoencoders have shown promise for many applications, including audio coding [2, 3] and the transfer of musical style features such as instrument type [4, 5] and audio production details [6]. The type of encoder chosen will produce encoded representations that are typically better suited for some tasks than others. Often such tasks take the form of classification and/or Music Information Retrieval [7, 8, 9]. We focus on the analysis and synthesis of audio signals by freezing the weights of pretrained autoencoders optimized for audio reconstruction. The latent representations arising in such autoencoders may encode semantic or stylistic information [9].

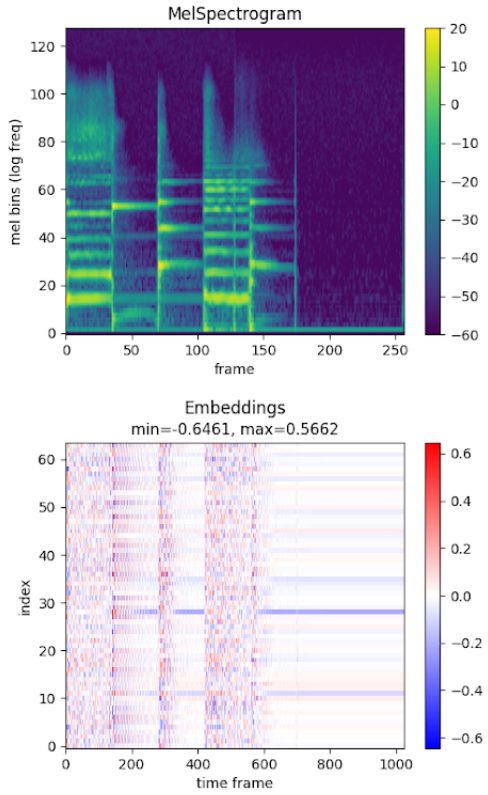

Fig. 1: Mel spectrogram (top) and latent representation "spectrogram" from the DiffAE diffusion autoencoder.

To place this work in context, one may consider spectrograms computed via short-time Fourier transforms (STFT), which can be viewed as columns of "vectors" in a multi-dimensional space. The representations produced by neural network systems can similarly be viewed either as a set of vectors or as columns in "neural [activation] spectrograms" or "feature maps". Illustrations of such spectrogram-like representations appear in Fig. 1.

Previous neural network methods for audio effect transformations [10, 11, 12] have been created in an end-to-end manner using supervised learning. In this paper, we consider transformations *only* upon the latent representations of pretrained auto-encoders. By leveraging the encoder and decoder portions for systems that have been pretrained on large datasets, one may be able to discover transformations of latent representations from relatively small datasets. In contrast to typical transfer learning approaches that further update the weights of a model, this study wishes to explore the potential of achieving useful signal manipulations by manipulating only the intermediate activations or representations of a frozen pretrained model.

This paper lays the groundwork for the development of few-shot or zero-shot musical audio transformations from a self-supervised training program. Our early attempts at zero-shot style transfer of audio production effects using vector algebra operations on representations from pretrained autoencoders [1] were largely unsuccessful, motivating the visualizations and classification tests of this paper: we aim for a better understanding of the features of such representation spaces.

We hypothesize that "good" embedding spaces would also provide for strong coupling to text-based control systems as is seen with text-to-image models [13, 14, 15]. Identifying perceptually meaningful transformations in the latent space could also enable the discovery and design of novel audio manipulations, paving the way for new methods of audio effect design and enabling users to more intuitively explore and discover novel sound manipulations [16].

A key challenge for working with latent space representations is to disentangle the different dimensions, e.g., so that user controls such as knobs and sliders have one primary (perceptual or semantic) effect [17, 18, 5]. Disentangling dimensions has also led to improvements in few-shot voice style transfer [19]. Applications of contrastive methods [20, 21] have shown that semantically populating the latent space and disentangling dimensions can be mutually achievable. We choose to work with existing autoencoders that have not necessarily been optimized for disentanglement of their latent dimensions, as a way to explore their level of disentanglement.

To investigate this we perform visualization of the representations projected into two and three dimensions using Principal Component Analysis and UMAP [22], as well as conduct classification experiments that aim to quantify both the degree of information encoded about manipulations and how this information is encoded. We hope that our investigations will lead to advances in musical audio production that make content-based and semantic operations easier to perform than are currently possible.

[1] Colab notebook: https://tinyurl.com/Destructo-ipynb, Oct. 2022

Table 1: Parameter settings for classification tests. All other settings were defaults except the Compressor used a "Ratio" value of 5.

Abbr	Effect Name	Parameter	Value
CLN	Clean	-	-
CHS	Chorus	Rate (Hz)	1
CMP	Compressor	Threshold (dB)	-50
DLY	Delay	Delay (s)	0.5
DIS	Distortion	Drive (dB)	25
HPF	Highpass Filter	Cutoff (Hz)	2000
LPF	Lowpass Filter	Cutoff (Hz)	70
PS	PitchShift	Semitones	4
RVB	Reverb	Room Size	0.8
TRV	Time Reverse	-	-

2 Methods

2.1 Models

The models we studied are two pretrained diffusion autoencoders developed internally by Harmonai [2]. The first model we refer to as "DiffAE," which uses a 64-dimensional latent representation space; the lower image in Fig. 1 is from the DiffAE model. The second model is a two-stage cascading latent diffusion model [23, 24] which we refer to as the "Stacked DiffAE" or simply "Stacked" model. Representations in both stages are 32 dimensional, but those of the second stage are more compact in time, containing a 16× coarser resolution in time than the first. Unless otherwise noted, we always use the smaller, more compact Stage 2 representations in this paper. The (larger) Stage 1 representations used in this paper are obtained on the *decoder* side by upsampling the (more compact) Stage 2 representations and performing additional diffusion. We have not used any Stage 1 representations from the encoder side. In addition to the architecture differences, the two models *were trained on different datasets*, sampled from an unreleased repository of a variety of music. In this study it is not our aim to determine which differences in representations are due to particular differences in the pretrained autoencoders, rather we use multiple models to temper any generalizations that might otherwise be drawn from considering only one autoencoder model.

[2]These models are not published but are similar to models under development at https://github.com/Harmonai-org

Table 2: Settings for parameter variation tests, using 32 increments from minimum to maximum, with HPF and LPF values varying logarithmically. All other settings were defaults.

Effect Name	Parameter	Min - Max
Distortion (DIS)	Drive (dB)	0 - 30
Reverb (RVB)	Room Size	0.01 - 0.99
Highpass Filter (HPF)	Cutoff (Hz)	50 - 10000
Lowpass Filter (LPF)	Cutoff (Hz)	50 - 10000

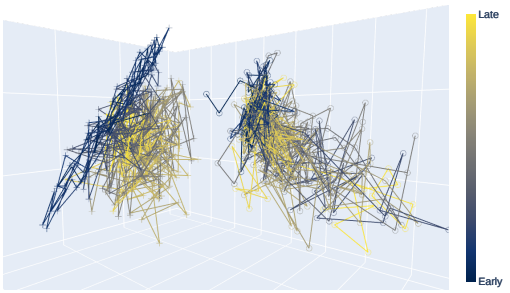

Fig. 2: PCA plot of time trajectories for one guitar sample (circles) and one piano sample (crosses), for the 32-dimensional Stacked DiffAE model.

2.2 Datasets

We constructed a dataset of 1024 audio samples at 48 kHz, each 5.4 seconds in length (2^{18} samples). We sourced 512 guitar sounds from GuitarSet [25], including performances of both solo notes and strumming chords. The remaining 512 samples were piano sounds sampled from the 2018 subset of the MAESTRO dataset [26]. We chose to use only two musical instruments, solo, so that the results of audio effect manipulations could stand out more clearly than they might if applied to a more widely-varied musical audio dataset. All input sounds were mono recordings that were 'doubled' to stereo (because the models expect stereo inputs), and loudness-normalized via `pyloudnorm`[3] [27] before and after passing through audio effects to remove level differences. We select nine different audio manipulations using fixed parameters to explore the clustering by effects and four effects for which we investigated the results of varying one key parameter. All audio effects, except a "clean" bypass and a time-reversal, were applied via `Pedalboard`[4] with settings shown in Table 1 for classification tests and Table 2 for parameter variation tests.

[3]https://github.com/csteinmetz1/pyloudnorm
[4]https://github.com/spotify/pedalboard

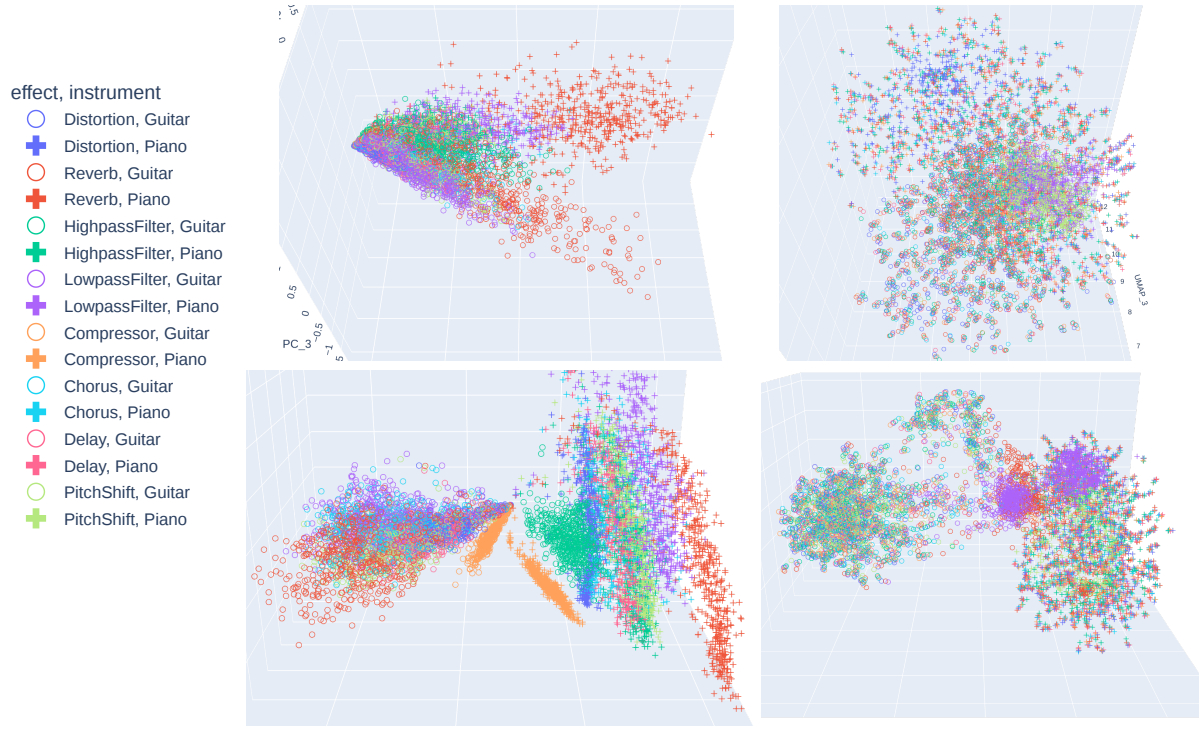

Fig. 3: 3D projections of flattened representations of the DiffAE autoencoder (top row) and stacked model (bottom row), showing PCA (left column) and UMAP (right column) mappings. In the UMAP plots, one sees that variations due to performance typically are more significant than variations due to audio effects. The two classes of instrument sounds map to largely disjoint areas of the latent space.

3 Results

3.1 Visualization

Fig. 2 shows the trajectories of representation vectors as a function of time, projected into three dimensions using Principal Component Analysis (PCA). The two trajectories shown are projected using the PCA transformation derived from the entire dataset. The "motion" of these representation vectors describes a complicated path. While methods such as RAVE [5] employ a "prior model" to learn to predict such trajectories autoregressively for music generation, in this study we simply note that the trajectories as seen in 3D PCA projections do not seem to follow any easily intuited pattern.

An alternative view results from flattening the representations for all dimensions and times into single vectors that inhabit a high-dimensional space. Fig. 3 shows flattened representations colored by audio effect class in PCA and UMAP [22] 3D projections. The PCA projections in the left column for both the DiffAE (top) and the Stacked (bottom) models show some separation by audio effects class with the latter showing stronger clustering by audio effects. However, the UMAP images in the right column of Fig. 3 show that except for one or two effects, the clustering of flattened representations is not class-determined: the variation due to musical performance (of the guitar or piano piece) plays a greater role in the data representations, with audio effects being a small perturbation.

To obtain a stronger "style" signal to investigate the separation by audio effect class, we time-average the representations and plot them in Fig. 4. The resulting PCA data (left column) appears similar to the previous Fig., however, the UMAP plots (right column) now show much stronger grouping by audio effects than in the previous figure. This suggests that time averages of the representations may serve as a useful proxy for

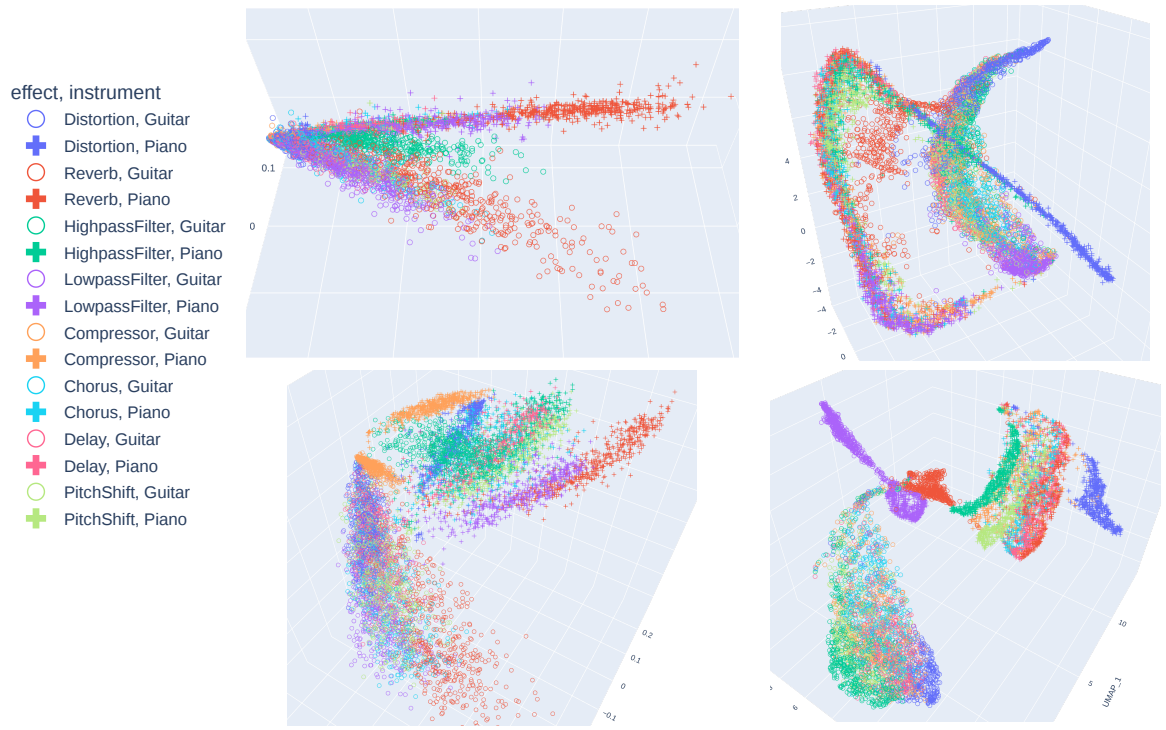

Fig. 4: 3D projections of time-averaged representations of the DiffAE autoencoder (top row) and stacked model (bottom row), showing PCA (left column) and UMAP (right column) mappings. One sees that the time-averaged representations typically offer better clustering and separation by audio effect than the flattened representations in Fig. 3. As in the previous figure, we observe large-scale clustering according to musical instrument class.

"style" information, and at the very least that the representations of these autoencoder models do encode some "semantic" structure with respect to audio effect class.

A final visualization study explores the variation in embeddings due to effect parameter values. Fig. 6 through 8 show the effects of varying key "knob" values for Distortion, Reverb, Highpass and Lowpass Filter effects. A primary observation is that varying an affect parameter typically results in a nonlinear path through the space, even for "linear" effects such as the filters. This helps explain the unsuccessful results of earlier attempts at few-shot neural audio style transfer based on algebraic operations on representation vectors. Fig. 9 combines the display of multiple effects parameters applied to a few audio samples. Again, one sees curved paths even for "linear" effects such as HPF, whereas perhaps unexpectedly the Distortion paths tend to appear comparatively unidirectional.

3.2 Manipulation classification

To quantitatively measure the level of information about signal manipulations captured by our autoencoders we use manipulation classification as a proxy task. Similar to the evaluation methodology employed in self-supervised representation learning, we use a linear probe on top of the embeddings from frozen pre-trained encoders to perform classification [28]. In our setup, we train linear probes on the 10-way classification task, predicting which of the 9 different audio manipulations was applied to the source audio or if no manipulation is present. Linear probes consist of a single layer that maps from the embedding space to the number of classes. This results in a parameter count proportional to the embedding dimensionality.

Since our encoders produce a sequence of representations (in time), we adopt two different approaches to feeding the extracted embeddings to the linear probe.

Encoder	Dim	Probe	Class-wise Accuracy (%)										
			CHS	CLN	CMP	DLY	DST	HPF	LPF	PS	RVB	TRV	AVG
VGGish-T	128	1.3 k	98.0	80.6	90.1	81.3	91.4	94.5	96.9	86.7	90.4	94.5	92.9
VGGish-F	2176	21.8 k	98.0	76.5	85.2	76.8	92.1	90.1	96.9	83.3	88.7	95.1	90.3
Stacked2-T	32	0.3 k	24.0	54.9	28.1	34.9	86.6	78.5	95.0	66.7	79.3	32.0	57.7
Stacked2-F	16384	163.9 k	45.2	42.2	30.2	65.1	84.7	80.9	94.4	68.2	71.3	39.2	63.1
Stacked1-T	32	0.3 k	29.9	59.2	27.1	26.5	88.6	71.4	96.9	54.2	74.8	36.8	57.4
Stacked1-F	262144	2621.5 k	14.6	37.6	9.4	29.2	41.9	59.1	88.7	24.5	58.5	48.6	41.1
DiffAE-T	64	0.7 k	11.2	32.5	3.6	9.3	92.9	46.1	86.9	31.0	63.5	27.1	40.3
DiffAE-F	131072	1310.7 k	4.4	32.7	47.7	16.3	73.6	17.0	27.0	19.3	18.6	25.6	28.9

Table 3: Effect classification accuracy when training a linear probe on top of normalized representations from pretrained encoders. T denotes time-averaged embeddings and F denotes flattened embeddings. Best-performing models are denoted in boldface.

First, we consider the time average of all embeddings, which has the effect of reducing variance across time frames. However, time averaging may remove information critical for detecting audio effects if the effect exhibits behavior over larger time scales, for example, reverberation. To account for this case, we also consider the complete flattened sequence of embeddings.

In our evaluation, we compare representations from the Stacked autoencoder (Stage 1 and Stage 2), as well as the DiffAE. As non-autoencoding baseline we also use embeddings extracted from the pretrained VGGish model[5] [29]. Similar to the autoencoders, the VGGish was not trained to explicitly capture information about audio manipulations. However, unlike the autoencoders, it does not enable reconstruction of the original signal given an encoded sequence.

All probes are trained using a batch size of 32 with the AdamW optimizer and a learning rate of $3 \cdot 10^{-4}$ for a maximum of 500 epochs. We use early stopping with a patience of 50 epochs monitoring the validation accuracy. Results from these classification tests are shown in Table 3. While all features enable classification accuracy higher than random guessing, we find that the VGGish features significantly outperform all of the autoencoder representations. This indicates that all representations encode some information that is predictive of audio manipulations, however, there is comparatively less information captured in the autoencoder representations when used both in the time-averaged and flattened configuration.

[5]https://github.com/hearbenchmark/hear-baseline

3.3 Dimension masking

While the multi-class classification experiment provides some insight into the level of information about audio manipulations, it does not provide any insight into *how* this information is encoded. Developing an understanding of how information is encoded is critical for leveraging the embedding space for signal manipulation. To approach this question, we designed a second classification experiment.

In this experiment, we consider a binary classification task where a linear model is trained to classify if a given effect is present or if the signal is clean. To understand how information is encoded within the pretrained representation, we mask one dimension of the representation and then train a classifier. We then repeat this process for each dimension of the representation and for each of the 9 effects, measuring any decrease in classification accuracy. If masking one dimension significantly decreases performance, it indicates that information predictive of the manipulation in question is encoded in this dimension.

Due to its superior performance in the previous experiment, we consider the Stacked autoencoder model, which features a 32-dimensional representation. Therefore we train 32 linear classifiers for each of the 9 effects, masking one dimension of the representation each time to produce a total of 228 models. Similar to the previous experiments, we use a batch size of 32 along with the AdamW optimizer and a learning rate of $3 \cdot 10^{-4}$. All models are trained for 100 epochs and

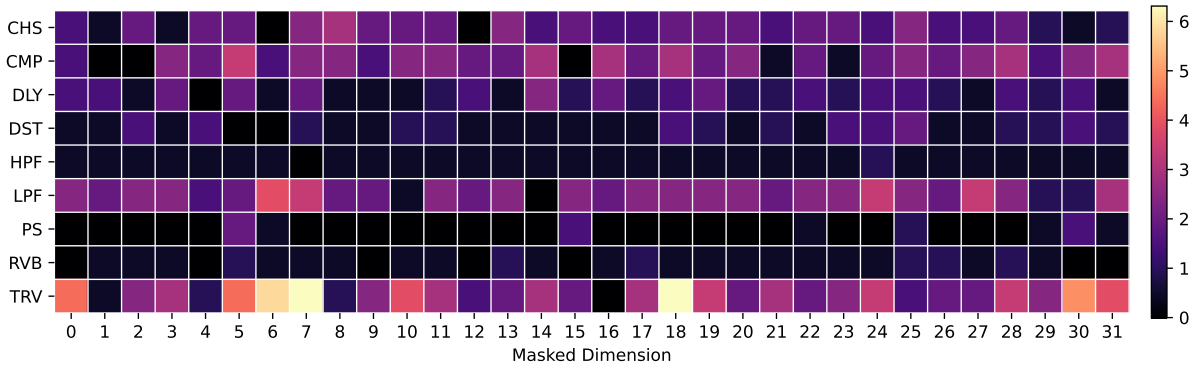

Fig. 5: Evaluation of the relative importance of each dimension of the Stacked autoencoder (Stage 2) ($D = 32$) using time-averaged embeddings when detecting the presence of each audio effect. The colormap indicates the decrease in the number of percentage points for the binary classification accuracy when masking one dimension of the representation as compared to the best-performing configuration for each effect type.

we select the best model for evaluation based on the validation accuracy.

We report the results in Fig. 5, where the color map indicates the number of percentage points lost by masking the associated dimension as compared to the best-performing model for that effect. We note that for all dimensions and effects, it appears that no single dimension is particularly predictive. In the worst case, masking dimension 7 and 18 decreased the classification accuracy for time reversal by ≈ 6 percentage points. However, it seems that masking any dimension of the representation caused a larger decrease in accuracy for certain effects, such as time reverse, lowpass filter, chorus, and compressor, while effects like highpass filter, pitch shift, and reverberation saw little variation. From these results we conclude that disentanglement of the representation with respect to the manipulations studied is likely low.

4 Summary

We explored the latent spaces of two pretrained audio autoencoders by manipulating musical audio samples via multiple classes of audio effects. The inner representations of these models show some clustering according to audio effects when one considers the time average of the effects, whereas the full representations in flattened form tend to cluster more strongly based on each individual musical performance. In all cases, the space tended to be divided by musical instrument type (e.g. guitar samples on one side, piano on another).

To provide a quantitative measurement of the separation by audio effect, we applied classification tests using a linear probe, comparing against a VGGish model as a baseline. We found that the frozen autoencoder representations tended to perform significantly worse than the VGGish model for classification by audio effects, with the time-averaged representations performing better than the flattened ones – in agreement with our observations of the visualizations.

By varying the parameters of audio effects, we observe that the resulting path of time-averaged representations through latent space tends to be nonlinear, even for "linear" effects such as Highpass and Lowpass filters. Thus representation-based methods for audio manipulation must take this inherent nonlinearity into account.

The findings of this paper may inform future efforts to develop efficient, sophisticated methods for musical audio production which can take advantage of the tendency for neural network autoencoders to encode semantic content. We speculate that additional work in disentangling the dimension of the latent spaces may yield improved results for audio production workflows.

5 Acknowledgments

This research was supported by Stability AI via Harmonai. We wish to thank Zach Evans for use of his pretrained autoencoders. C.S. is funded in part by UKRI and EPSRC as part of the "UKRI CDT in Artificial Intelligence and Music," under grant EP/S022694/1. S.H. acknowledges Max Ortner for helpful discussions.

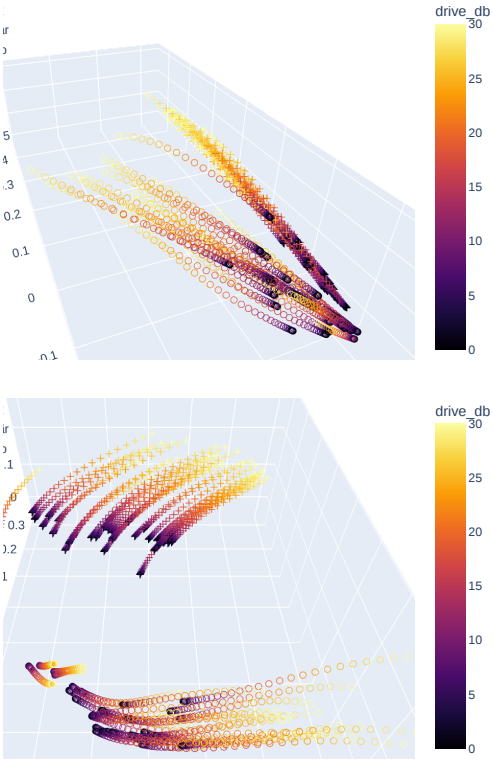

Fig. 6: Varying the Distortion effect on 16 guitar and 16 piano samples, for the DiffAE (top) and Stacked DiffAE (bottom) encoders. Here we show PCA plots of time-averaged embeddings.

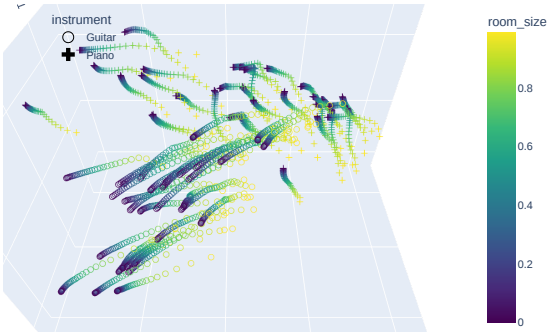

Fig. 7: PCA plot of parameter adjustment paths for Reverb effect using time-averaged representations from the Stacked model, for guitar (circles) and piano (crosses).

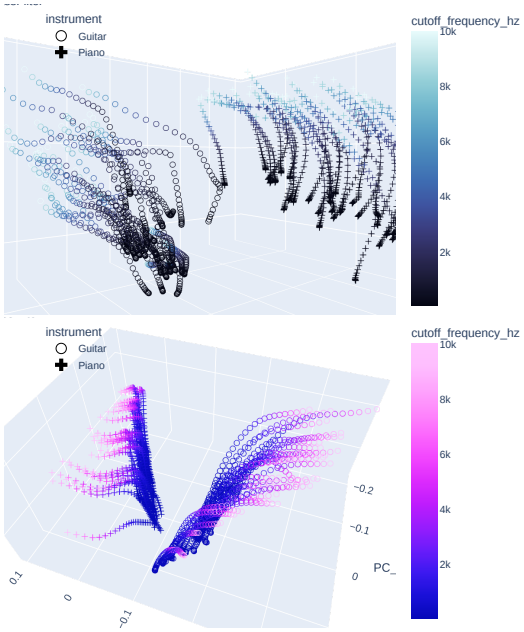

Fig. 8: PCA plots of time-averaged representations in the Stacked model, adjusting Highpass (top) and Lowpass (bottom) filters for guitar (circles) and piano (crosses)

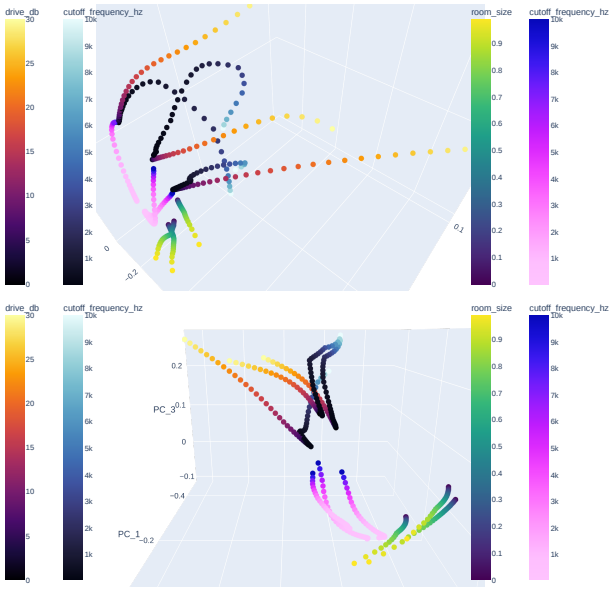

Fig. 9: Multi-effects trajectories for DiffAE (top) and Stacked DiffAE (bottom) models.

References

[1] Stables, R., Enderby, S., De Man, B., Fazekas, G., and Reiss, J. D., "SAFE: A system for extraction and retrieval of semantic audio descriptors," 2014.

[2] Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M., "SoundStream: An End-to-End Neural Audio Codec," *CoRR*, abs/2107.03312, 2021.

[3] Défossez, A., Copet, J., Synnaeve, G., and Adi, Y., "High Fidelity Neural Audio Compression," *arXiv preprint arXiv:2210.13438*, 2022.

[4] Engel, J., Hantrakul, L., Gu, C., and Roberts, A., "DDSP: Differentiable Digital Signal Processing," 2020, doi:10.48550/ARXIV.2001.04643.

[5] Caillon, A. and Esling, P., "RAVE: A variational autoencoder for fast and high-quality neural audio synthesis," 2021, doi:10.48550/ARXIV.2111.05011.

[6] Steinmetz, C. J., Bryan, N. J., and Reiss, J. D., "Style Transfer of Audio Effects with Differentiable Signal Processing," *Journal of the Audio Engineering Society*, 2022.

[7] Turian, J., Shier, J., Khan, H. R., Raj, B., Schuller, B. W., Steinmetz, C. J., Malloy, C., Tzanetakis, G., Velarde, G., McNally, K., Henry, M., Pinto, N., Noufi, C., Clough, C., Herremans, D., Fonseca, E., Engel, J., Salamon, J., Esling, P., Manocha, P., Watanabe, S., Jin, Z., and Bisk, Y., "HEAR: Holistic Evaluation of Audio Representations," 2022, doi:10.48550/ARXIV.2203.03022.

[8] Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I., "Jukebox: A Generative Model for Music," 2020, doi:10.48550/ARXIV.2005.00341.

[9] Castellon, R., Donahue, C., and Liang, P., "Codified audio language modeling learns useful representations for music information retrieval," 2021, doi:10.48550/ARXIV.2107.05677.

[10] Martinez Ramirez, M., Benetos, E., and Reiss, J., "Deep Learning for Black-Box Modeling of Audio Effects," *Applied Sciences*, 10, p. 638, 2020, doi:10.3390/app10020638.

[11] Hawley, S. H., Colburn, B., and Mimilakis, S. I., "Profiling Audio Compressors with Deep Neural Networks," *Journal of the Audio Engineering Society*, 2019.

[12] Steinmetz, C. J. and Reiss, J. D., "Efficient neural networks for real-time modeling of analog dynamic range compression," in *152nd AES Convention*, 2022.

[13] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I., "Zero-Shot Text-to-Image Generation," *CoRR*, abs/2102.12092, 2021.

[14] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B., "High-Resolution Image Synthesis with Latent Diffusion Models," *CoRR*, abs/2112.10752, 2021.

[15] Brooks, T., Holynski, A., and Efros, A. A., "InstructPix2Pix: Learning to Follow Image Editing Instructions," 2022.

[16] Steinmetz, C. J. and Reiss, J. D., "Steerable discovery of neural audio effects," in *5th Workshop on Creativity and Design at NeurIPS*, 2021.

[17] Härkönen, E., Hertzmann, A., Lehtinen, J., and Paris, S., "GANSpace: Discovering Interpretable GAN Controls," 2020, doi:10.48550/ARXIV.2004.02546.

[18] Härkönen, E., Hertzmann, A., Lehtinen, J., and Paris, S., "GANSpace: Discovering Interpretable GAN Controls," *CoRR*, abs/2004.02546, 2020.

[19] Yuan, S., Cheng, P., Zhang, R., Hao, W., Gan, Z., and Carin, L., "Improving Zero-Shot Voice Style Transfer via Disentangled Representation Learning," in *International Conference on Learning Representations*, 2021, doi:10.48550/arXiv.2103.09420.

[20] Fonseca, E., Ortego, D., McGuinness, K., O'Connor, N. E., and Serra, X., "Unsupervised Contrastive Learning of Sound Event Representations," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 371–375, 2021, doi:10.1109/ICASSP39728.2021.9415009.

[21] Koo, J., Martinez-Ramirez, M. A., Liao, W.-H., Uhlich, S., Lee, K., and Mitsufuji, Y., "Music Mixing Style Transfer: A Contrastive Learning Approach to Disentangle Audio Effects," 2022, doi:10.48550/ARXIV.2211.02247.

[22] McInnes, L., Healy, J., Saul, N., and Grossberger, L., "UMAP: Uniform Manifold Approximation and Projection," *The Journal of Open Source Software*, 3(29), p. 861, 2018.

[23] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding," 2022.

[24] Evans, Z., Hawley, S. H., and Crowson, K., "Musical audio samples generated from joint text embeddings," *The Journal of the Acoustical Society of America*, 152(4), pp. A178–A178, 2022, doi: 10.1121/10.0015956.

[25] Xi, Q., Bittner, R. M., Pauwels, J., Ye, X., and Bello, J. P., "GuitarSet: A Dataset for Guitar Transcription." in *ISMIR*, pp. 453–460, 2018.

[26] Hawthorne, C., Stasyuk, A., Roberts, A., Simon, I., Huang, C.-Z. A., Dieleman, S., Elsen, E., Engel, J., and Eck, D., "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset," in *International Conference on Learning Representations*, 2019.

[27] Steinmetz, C. J. and Reiss, J. D., "pyloudnorm: A simple yet flexible loudness meter in Python," in *150th Convention of the Audio Engineering Society*, Audio Engineering Society, 2021.

[28] Chen, X., Xie, S., and He, K., "An empirical study of training self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9640–9649, 2021.

[29] Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., et al., "CNN architectures for large-scale audio classification," in *ICASSP*, pp. 131–135, IEEE, 2017.