Towards Lightweight Controllable Audio Synthesis with Conditional Implicit Neural Representations

Jan Zuiderveld
University of Amsterdam
Royal Conservatoire The Hague
janzuiderveld@gmail.com

Marco Federici
AMLab
University of Amsterdam
m.federici@uva.nl

Erik Bekkers
AMLab
University of Amsterdam
e.j.bekkers@uva.nl

1 Introduction

Controllable audio synthesis is a core element of creative sound design. Recent advancements in AI have made high-fidelity *neural audio synthesis* achievable. However, the high temporal resolution of audio and our perceptual sensitivity to small irregularities in waveforms make synthesizing at high sampling rates a complex and computationally intensive task, prohibiting real-time, controllable synthesis within many approaches. In this work we aim to shed light on the potential of Conditional Implicit Neural Representations (CINRs) as lightweight backbones in generative frameworks for audio synthesis.

Implicit neural representations (INRs) are neural networks used to approximate low-dimensional functions, trained to represent a single geometric object by mapping input coordinates to structural information at input locations. In contrast with other neural methods for representing geometric objects, the memory required to parameterize the object is independent of resolution, and only scales with its complexity. A corollary of this is that INRs have infinite resolution, as they can be sampled at arbitrary resolutions. Moreover, the fact that object samples are calculated independently allows sequential synthesis in memory- and/or computationally limited environments.

To apply the concept of INRs in the generative domain we frame generative modelling as learning a distribution of continuous functions [4]. This can be achieved by introducing conditioning methods to INRs, e.g. by concatenating latent codes to input coordinates [2] or modulating layer activations [1]. We refer to this family of architectures as CINRs.

The dominant architectures in INR literature are multilayer perceptrons (MLPs) with traditional nonlinearities. These networks are biased towards low-frequency functions, known as *spectral bias*[26]. Recently, several concurrent works have proposed to use periodic nonlinearities in INRs to facilitate learning finer, high-frequency details [22, 28, 30]. This increased expressiveness creates great potential for applying INRs in the domain of audio.

At the time of writing there is no published research applying CINRs to audio. To gauge the potential of CINRs for controllable audio synthesis we compare the ability of CINRs with periodic nonlinearities (PCINRs, based on π-GAN [1]) and transposed convolution neural networks (TCNNs, based on WaveGAN [3]) to reconstruct different medium-scale sets of NSYNTH [5] waveform samples.

Our experiments show that PCINRs learn faster and generally produce quantitatively better audio reconstructions than TCNNs with equal parameter counts. However, their performance is very sensitive to activation scaling hyperparameters. When learning to represent more uniform sets, PCINRs tend to introduce artificial high-frequency components in reconstructions. We validate this noise can be minimized by applying standard weight regularization during training or decreasing the compositional depth of PCINRs, and suggest directions for future research. [1]

[1] Sound examples and source code are publicly available at janzuiderveld.github.io/audio-PCINRs.

Workshop on Deep Generative Models and Downstream Applications at NeurIPS 2021, Vancouver, Canada.

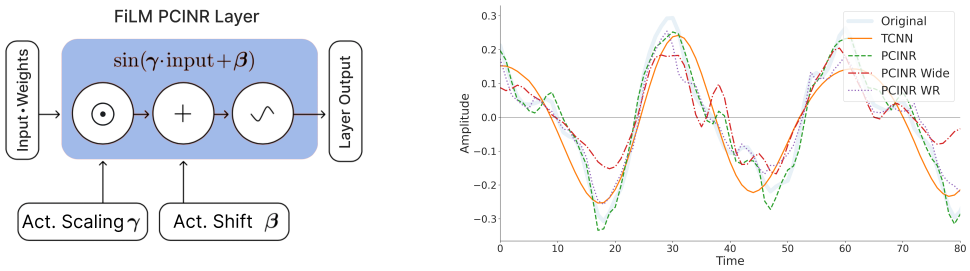

Figure 1: Left: Schematic overview of a FiLM PCINR Layer. Adapted from Chan et al. [1]. Right: 80 samples (0.0005s) of a waveform of *NSYNTH Diverse* and reconstructions of tested setups.

2 Overview of Methodology and Results

We investigate the ability of PCINRs to reconstruct two 1024 item sets of one second NSYNTH [5] waveform samples, *NSYNTH Keyboard* and *NSYNTH Diverse*. The performance of TCNNs with equal parameter counts is reported as a baseline. Tested PCINRs are conditioned using feature-wise linear modulation (FiLM [24]). In our FiLM implementation, latent codes modulate the activation scaling and shift before periodic nonlinearities are applied. Preliminary experiments showed high-frequency noise in the output of PCINRs. We test two regularization methods to minize this. *PCINR Wide*, an MLP with fewer layers and more hidden neurons. *PCINR WR*, the same MLP with weight regularization (WR) applied during training. See Figure 1. Details in Section 4 and 5.

Three quantitative evaluation metrics are reported: Contrastive learning-based multi-dimensional Deep Perceptual Audio similarity Metric (CDPAM [19]), Multi-resolution Short-Time Fourier Transform Mean Squared magnitude Error (Multi STFT MSE) and Mean Squared Error (MSE). Details in Section 6. Experimental results are reported in Table 1.

Table 1: Means of CDPAM, Multi STFT MSE and MSE for NSYNTH Diverse and NSYNTH Keyboard over all set items. Standard deviations are calculated over 3 training runs. For all metrics a lower score is better. Silence and noise ($\mu = 0, \sigma = 1$) scores are reported for reference.

	CDPAM: Overall quality		Multi STFT MSE: Noise		MSE: Absolute similarity	
Arch.	Diverse	Keyboard	Diverse	Keyboard	Diverse	Keyboard
TCNN	0.43 ± 0.21	0.43 ± 0.27	$\mathbf{0.06} \pm 0.05$	0.06 ± 0.04	4.05 ± 0.42	8.67 ± 0.83
PCINR	$\mathbf{0.35} \pm 0.24$	0.48 ± 0.18	$\mathbf{0.06} \pm 0.02$	0.14 ± 0.04	$\mathbf{1.32} \pm 0.3$	4.48 ± 1.02
- Wide	0.56 ± 0.28	$\mathbf{0.36} \pm 0.2$	0.11 ± 0.05	0.08 ± 0.02	6.55 ± 0.16	1.64 ± 0.03
- WR	1.11 ± 0.05	0.71 ± 0.05	0.07 ± 0.0	$\mathbf{0.04} \pm 0.0$	4.53 ± 0.22	$\mathbf{0.85} \pm 0.1$
Silence	1.65 ± 0.0	1.77 ± 0.0	0.16 ± 0.00	0.16 ± 0.00	46.19 ± 0.0	78.74 ± 0.0
Noise	1.06 ± 0.27	1.06 ± 0.32	2.27 ± 0.06	2.32 ± 0.04	$\sim 1 \times 10^3$	$\sim 1 \times 10^3$

3 Overview of Conclusions and Future work

PCINRs exhibit exceptional expressiveness making them well-suited for modelling sets of high-frequency one-dimensional continuous functions such as audio. They perform superior with TCNNs in quantitative metrics representing quality, noise and similarity. Qualitatively, PCINRs model more details, but induce a certain amount of high-frequency noise in the output signal. They offer direct control for balancing this trade-off between expressiveness and local smoothness. However, the optimal hyperparameters and performance of PCINRs are very sensitive to dataset characteristics such as note- and timbral diversity. It is unknown if these observations hold for large-scale PCINRs.

An idea which looks promising in light of our results, yet unexplored in INR literature, is splitting PCINRs fully-connected MLP in parallel subnetworks. In Section 7 we argue this would reduce the propagation of noise induced at the recurrent stationary points in periodic nonlinearities, and by doing so minimize noise in the learning signal during training and in the output signal during synthesis. This argumentation is supported by multiple research directions [7, 10, 17, 32].

4 Methodology

4.1 Problem formulation

We are interested in learning to represent a set of continuous audio waveforms covered in dataset D consisting of N discretely sampled waveforms. Waveforms in datasets D are represented by point sets X_i, consisting of M equally-spaced sampled amplitude values from the corresponding continuous amplitude functions $y_i : \mathbb{R}^1 \to \mathbb{R}^1$:

$$\mathcal{D} = \{X_i\}_{i=1}^{N}, \quad X_i = \{(t_j, a_j) : a_j = y_i(t_j)\}_{j=1}^{M}.$$

Where t_j are time coordinates, and a_j are the corresponding amplitudes or air pressure measurements at these time coordinates.

We represent audio waveforms by directly approximating amplitude functions y_i with continuous functions $f_i : \mathbb{R}^1 \to \mathbb{R}^1$, parameterized by MLPs ϕ_i with sets of (learnable) parameters $\theta_i \in \Theta$.

We assume parameter sets θ_i live in a low-dimensional manifold. We define:

- A function mapping latent representations $\mathbf{z}$ to intermediate latent representations $\mathbf{w} \in \mathcal{W}$, $\mathbf{w} = g(\mathbf{z})$ with $g : \mathcal{Z} \to \mathcal{W}$, parameterized by a neural network ψ with learnable parameters ξ, also known as the latent mapping network.
- A set of parameters α functioning as weights in ϕ_i shared between representations.
- A function defining how intermediate latent representations $\mathbf{w}$ influence shared parameters α to obtain θ_i, $\theta_i = c(\mathbf{w}_i, \alpha)$, also known as the conditioning method.

The shared parameters α, latent mapping network ψ and conditioning method c together parameterize the conditional distribution $p(\theta|\mathbf{z})$, allowing us to map latent representations to the parameter space $p(\theta) = E_z[p(\theta|\mathcal{Z} = \mathbf{z})]$. By learning $\mathbf{z}_i$ during training using an autodecoder setup, obtaining $\mathbf{w}_i$ by mapping $\mathbf{z}_i$ through $g(\mathbf{z}_i)$, conditioning shared parameters α using $\mathbf{w}_i$ and conditioning method $c(\mathbf{w}_i, \alpha)$, we obtain parameter set θ_i for ϕ_i. Then, we can evaluate $\phi_i|\theta_i$ at time coordinates t_j to obtain corresponding amplitude approximations $\hat{a_j}$:

$$\theta_i = c(g(\mathbf{z}_i), \alpha), \quad \hat{a_j} = \phi_i(t_j|\theta_i).$$

We optimize $\phi_i|\theta_i$ to represent functions y_i with absolute fidelity, MSE.

4.2 Architectural details

PCINR, based on π-GAN PCINR is adapted from the π-GAN [1] decoder, which is designed to produce implicit 3D radiance fields conditioned on $\mathbf{z}_i$. Ignoring the final parallel layer which integrates ray directions, π-GAN parameterizes ϕ_i as an 8 layer MLP with sine nonlinearities and a constant amount of hidden units (256). The conditional distribution $p(\theta|\mathbf{z})$ is parameterized as follows:

- Latent mapping network ψ consists of a 3 layer MLP with ReLU activations.
- The conditioning mechanism c falls under the category of FiLM. $\mathbf{w}^\gamma$ and $\mathbf{w}^\beta$ vectors are shared between layers, drastically decreasing the total size of $\mathbf{w}$.
- $\mathbf{w}^\gamma$ does not scale layer activations directly, but is summed element-wise with the predefined activation scaling factor ω_0 we know from SIRENs, resulting in the following conditioning method c in layer k:[2]

$$\theta_k^b, \theta_k^{\omega_0} = c(\mathbf{w}, (\boldsymbol{\alpha}_k^b, \omega_{0,k})) = \mathbf{w}_k^\beta + \boldsymbol{\alpha}_k^b, \mathbf{w}_k^\gamma + \omega_{0,k}.$$

- Shared parameters $\boldsymbol{\alpha}$ populate all weights native to ϕ_i.

See figure 2 for a schematic overview of the network structure as provided by the authors.

[2] Technical descriptions in the paper of π-GAN seem to indicate that $\mathbf{w}_k^\gamma$ directly replaces $\omega_{0,k}$. This resulted in instable behaviour in preliminary experiments. E. Chan shared parts of their implementation showing the discussed conditioning method in which $\mathbf{w}_k^\gamma$ and $\omega_{0,k}$ are summed.

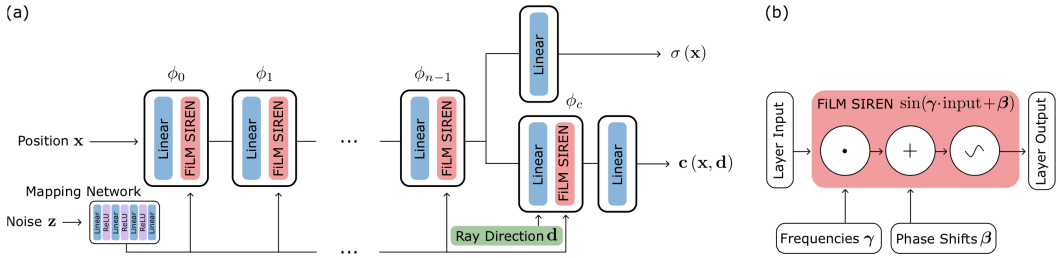

Figure 2:
$(\mathbf{a})$: Network structure of π-GAN.
$(\mathbf{b})$: Schematic overview of a FiLMed Sinusoidal Representation layer (SIREN [28]). Figure is taken from Chan et al. [1].

TCNN, based on WaveGAN TCNN is adapted from the WaveGAN decoder, which is based on of DCGAN [25]'s decoder, which popularized usage of GANs for image synthesis. This decoder uses transposed convolution to iteratively upsample low-resolution feature maps into a high-resolution image. In the WaveGAN decoder this is modified to work with audio by replacing its two-dimensional 5x5 filters with one-dimensional filters of length 25, and changing the stride factor for all convolutions from 2x2 to 4. These changes result in WaveGAN having the same number of parameters, numerical operations, and output dimensionality as DCGAN. Because DCGAN outputs 64x64 pixel images equivalent to just 4096 audio samples one additional layer is added to the model resulting in 16384 samples, slightly more than one second of audio at 16 kHz.

Latent code inference Autodecoders were introduced recently by Park et al. [23]. They have no encoder network. Latent embeddings $\mathbf{z_i}$ are instead treated as learnable parameters rather than inferred from observations at training. By storing and updating intermediate latent embeddings $\mathbf{z_i}$ with backpropagated training errors during training the decoder can function as an encoder, see Figure 3 for a schematic overview.

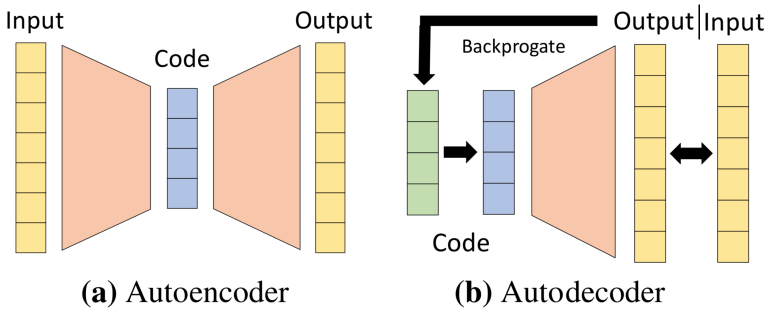

Figure 3: Schematic overview of autoencoder and autodecoder architectures. Figure adapted from Park et al. [23]

Due to the direct optimization of latent embeddings $\mathbf{z_i}$, autodecoders generally require less training iterations to reconstruct datasets faithfully. Autodecoders are more parameter efficient by not having an encoder, resulting in less operations per iteration and lower memory requirements during training. Finally, autodecoders can alleviate incompatibility issues between encoders and decoders. This is especially useful when dealing with implicit neural representations (INRs), as these are less trivial to be mirrored to use as an encoder.

Feature-wise linear modulation (FiLM) FiLM proposes to apply element-wise[3] scaling and shifting of intermediate layer activations based on latent representations $\mathbf{z}_i$. Framing the method in the conceptual framework described in section 4.1 we note the following specifics:

[3]In the case of convolutional networks, FiLM is applied feature map wise.

1. Intermediate latent representations $\mathbf{w}$ have the dimensiononality of twice the total amount of hidden units of conditioned layers in θ_i, one half for scaling ($\mathbf{w}^\gamma$) and one half for shifting ($\mathbf{w}^\beta$) layer activations.
2. The conditioning function c is applied as shown in the equation below. Note that parameters controlling bias θ_i^b are replaced by $\mathbf{w}_i^\beta$.
3. All other parameters θ in ϕ_i are parameterized by α, shared between representations.

$$y_k(\mathbf{x}_k) = act_k(\mathbf{w}_k^\gamma \cdot \mathbf{W_k}\mathbf{x}_k + \mathbf{w}_k^\beta).$$

FiLM is a unification of methods that modulate intermediate layer activations, as in normalization layers introduced originally in batch norm [13]. Other implementations include Dynamic Layer Norm for speech recognition [15], Conditional Instance Norm [8], Adaptive Instance Norm [12], and Conditional Batch Norm as used in Occupancy networks [21].

5 Experimental setup

Parameterizations of $\boldsymbol{\phi_i}$ & $\mathbf{p}(\boldsymbol{\theta}|\mathbf{z})$ The following parameterizations of ϕ_i & $p(\theta|\mathbf{z})$ are compared, exact parameter counts are in brackets:

1. Transposed Convolutional Neural Networks (TCNN [3]) [800k]
2. Periodic Conditional Implicit Neural Representations (PCINR [1], 256 hidden units, 8 layers) [790k]
3. Wide Periodic Conditional Implicit Neural Representations (PCINR Wide, 380 hidden units, 4 layers) [850k]
4. Periodic Conditional Implicit Neural Representations with Weight Regularization (PCINR WR, 256 hidden units, 8 layers) [790k]

For all parameterizations of ϕ_i & $p(\theta|\mathbf{z})$, the network architecture is changed such that parameter counts are in the order of 10^6. TCNN's parameter count is reduced by reducing the number of channels throughout the WaveGAN decoder network by a factor of eight. Models are trained with latent codes of size 256

5.1 Objective function

In our experiments we use MSE with addition of MSE between the derivative of target signals approximated with forward finite difference, and the derivative of reconstructions evaluated in the PCINRs (For TCNNs this is approximated by forward finite difference derivative of the reconstruction):

$$\mathcal{L} = \frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} \|\Phi(t_j, \mathbf{z_i}) - \mathrm{y}_i(t_j)\|^2 + \left\| \frac{\delta}{\delta t_j}[\Phi(t_j, \mathbf{z_i})] - \Delta t_j[\mathrm{y}_i(t_j)] \right\|^2 .$$

Where N is the batch size, M is the amount of sampled time coordinates per waveform and Δ is the forward finite difference. The addition of a MSE of derivatives of signals is justified by consistent training improvements observed in preliminary experiments.

All reported results in table 1 are after training until convergence ($\sim$ 5k epochs).

5.2 Weight regularization

We use weight regularization as in the following function:

$$\mathcal{L} = \frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M}\|\Phi(t_j, \mathbf{z_i})) - \mathrm{y}_i(t_j)\|^2$$
$$+ \left\|\frac{\delta}{\delta t_j}[\Phi(t_j, \mathbf{z_i})] - \Delta_{t_j}[\mathrm{y}_i(t_j)]\right\|^2 + \frac{\lambda}{2}\|\mathbf{W}_{\phi_i}\|^2.$$

Note that weight regularization is *only* applied to the weights of ϕ_i, not to those of ψ.

5.3 Datasets

We consider the following datasets:

1. 1 second, 1024 item, keyboard waveforms in MIDI notes [60, 64] subset of NSYNTH [5]. Balanced for note counts. Recorded at a sampling rate of 16kHz.
2. 1 second, 1024 item, keyboard, mallet and guitar waveforms in MIDI notes [24, 84] subset of NSYNTH. Balanced for note- and instrument counts. Recorded at a sampling rate of 16kHz.

Preliminary tests of baseline parameterizations of ϕ_i and $p(\theta|\mathbf{z})$ in the specific generative framework described in section 4.1 showed that ϕ_i and $p(\theta|\mathbf{z})$ with a reasonable amount of parameters were uncapable of reconstructing large datasets[4]. Thus, we set the dataset size for the main experiments to be challenging, but not infeasible based on preliminary experiments.

We selected a dataset subset with the smallest range of subsequent notes within one instrument family cumulating to 1024 items. This turned out to be the keyboard instrument family, MIDI note 60 (C4, 261.63Hz) to 64 (E4, 329.63Hz). The dataset was sampled balancing for note counts. We refer to this dataset as NSYNTH Keyboard.

We selected the other split of NSYNTH to contrast with NSYNTH Keyboard in pitch- and timbral diversity. Furthermore, we selected MIDI notes 24 (C1, 32.70Hz) to 84 (C6, 1046.50Hz). We selected three instruments families to introduce more timbral diversity: keyboard, mallet and guitar. The dataset was sampled balancing for note- and instrument family counts. We refer to this dataset as NSYNTH Diverse.

5.4 Optimizer

Modern neural networks are typically trained with first-order gradient methods, which can be broadly categorized into two branches: the accelerated stochastic gradient descent family such as SGD with momentum and the adaptive learning rate methods, such as Adam. SGD methods use a global learning rate for all parameters, while adaptive methods compute an individual learning rate for each parameter. Compared to the SGD family, adaptive methods typically converge fast in the early training phases, but have poor generalization performance. In our baseline decoder comparison we compare Adam [16] and Adabelief [33]. AdaBelief consists of the same algorithm as Adam, but also considers curvature information by scaling update directions by the change in gradient. Preliminary experiments showed consistent, favourable results for Adabelief in all setups.

5.5 Hyperparameter search

In all experiments we optimize PCINR hyperparameters activation scaling first and activation scaling hidden. For every combination of dataset D, architecture and latent embedding inference method. We optimize these hyperparameters by running respective experiments in a short training regime of 200 epochs, using Bayesian Search [29] with Hyperband early stopping [18] to guide the search. Preliminary experiments indicated that optimal activation scaling or coordinate multiplier values are very sensitive to many architectural- and data characteristics. Thus, these values are swept using the described procedure in all experiments.

[4]This depends on many factors including the generative framework, how its parameterized, latent embedding size and dataset uniformity

6 Evaluation

Capturing the perceptual fidelity of audio reconstructions in a metric is not straightforward. Many audio-to-audio distances exist, each with different sensitivities, indicating the complexity of the problem.

For this work, we decided to start with an extensive selection of metrics based on recent and established research in audio reconstruction and generation and implementation availability. Then, we select metrics for representing background noise presence and overall quality based on correlation with a small sample of human judgements of reconstructions.

Considered metrics We compared the following metrics:

1. time-domain MSE
2. time-domain derivative MSE
3. time-domain MSE + derivative MSE
4. Signal-to-Noise Ratio (SNR)
5. Segmented SNR (SegSNR)
6. discrete Fourier transform (DFT) magnitude MSE
7. DFT magnitude wasserstein distance
8. DFT angular phase MSE
9. DFT magnitude pos/neg difference
10. Log-spectral distance (LSD) [9]
11. Multi resolution short-time Fourier transform (STFT) MSE
12. Multi resolution STFT wasserstein distance
13. Multi resolution STFT pos/neg difference
14. CSIG, CBAK, COVL [11]
15. PESQ [27]
16. FAD [14]
17. CDPAM [19]

Chosen metrics We use Multi resolution STFT MSE for representing background noise presence and CDPAM for overall quality.

Multi resolution STFT MSE is calculated by averaging STFT magnitude MSE's as shown in the equation with hamming windowing for N=4 window sizes: {400, 800, 1600 3200}:

$$\frac{1}{N}\sum_{i}^{N} \||STFT_i(X_i)| - |STFT_i(\{\phi_i(t_j)\}_{j=1}^{M})|\|^2.$$

CDPAM is a contrastive learning based perceptual audio similarity metric parameterized as a deep neural network. It is trained on a dataset of audio similarity judgements based on triplet comparisons, asking subjects: Is A or B closer to reference C?.

7 Discussion and Future Work

In all evaluated setups latent representations $\mathbf{z}$ are optimized per audio sample. We do not enforce any distribution on the latent space, nor any other method to improve generative capabilities. Thus, the current implementations allow timbre interpolation, but these interpolations contain artifacts. It should also be noted that the current implementations are not built to do conditional generation (e.g. note- or text conditioned). Unseen samples can be encoded and synthesized by iteratively optimizing latent representations $\mathbf{z}$.

PCINR gains are largest for MSE, which is optimized directly by the loss function during training. However, this does not translate to strong gains in perceptual metrics, showing that PCINRs do not contain a strong oscillatory bias and suggesting they could benefit significantly from spectral or perceptual objective functions. We experimented with several of such, but none resulted in improvements.

We experimented with several spectral and perceptual objective functions, but none resulted in improvements. We argue this is caused by inconsistent gradients for near identical inputs and note three mutually reinforcing sources for this. Firstly, because the inputs of PCINRs are one-dimensional coordinates, they process relatively many close to identical inputs. Secondly, the loss landscapes of perceptual and spectral objective functions are locally inconsistent [31]. Finally, periodic nonlinearities with high scaling factors contain a high density of stationary points, which cause locally inconsistent signal propagation (which also stimulates noise in output signals).

Taking the inferior label noise robustness observed in PCINRs into account, methods for dealing with noisy loss signals could be another promising avenue. This would allow the usage of spectral and perceptual reconstruction losses, which correlate much better with perceptual qualities than MSE. Combined with the expressivity of sinusoidal representation networks conditioned using feature wise linear modulation this could greatly improve perceptual qualities of represented distributions. One might hope that gradient clipping can also aid in mitigating the detrimental effects of locally inconsistent loss functions on learning. However, it has been proven that in classification problems standard gradient clipping does not *in general* provide robustness to label noise [20].

The compositional nature of PCINRs makes it difficult to analyze representation characteristics in a signal processing context. Multiplicative filter networks [6] (MFNs) can be viewed as linear function approximators over an exponential number of Fourier or Gabor basis functions. This establishes a connection of the network architecture with the traditional Fourier and Gabor wavelet transforms, which are extensively studied in literature and widely used in many application domains, especially audio.

Comparisons with MFNs would be useful to indicate if the compositional depth of PCINRs form a substantial benefit. In the experiments reported by the authors, MFNs show similar performance as PCINRs in parameter equal experiments, and larger gains in performance when increasing network depth and width. We indeed found PCINRs to lack in terms of scaling gains. Combined with the intuition that Fourier-based MFNs contain a strong oscillatory bias, the architecture is a promising avenue for future research in the area of implicit neural audio synthesis.

8 Broader Impact

The development of neural audio synthesis technologies has the potential to negatively impact musicians who make a living as professional performers. To what extent this is already occurring is unclear, however, it is consistent with a trend of increased access to music production methods that had previously been reserved primarily for professional recording studios.

References

[1] Eric R. Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein (2020). *Pi-GAN: Periodic Implicit Generative Adversarial Networks for 3D-Aware Image Synthesis*. arXiv: 2012.00926 [cs]. URL: http://arxiv.org/abs/2012.00926 (cit. on pp. 1–5).

[2] Zhiqin Chen and Hao Zhang (2019). *Learning Implicit Fields for Generative Shape Modeling*. arXiv: 1812.02822 [cs]. URL: http://arxiv.org/abs/1812.02822 (cit. on p. 1).

[3] Chris Donahue, Julian McAuley, and Miller Puckette (2019). *Adversarial Audio Synthesis*. arXiv: 1802.04208 [cs]. URL: http://arxiv.org/abs/1802.04208 (cit. on pp. 1, 5).

[4] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet (2021). *Generative Models as Distributions of Functions*. arXiv: 2102.04776 [cs, stat]. URL: http://arxiv.org/abs/2102.04776 (cit. on p. 1).

[5] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi (2017). *Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders*. arXiv: 1704.01279 [cs]. URL: http://arxiv.org/abs/1704.01279 (cit. on pp. 1, 2, 6).

[6] Rizal Fathony, Devin Willmott, Anit Kumar Sahu, and J Zico Kolter (2021). "MULTIPLICATIVE FILTER NETWORKS". In: p. 11 (cit. on p. 8).

[7] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan (2020). *Deep Ensembles: A Loss Landscape Perspective*. arXiv: 1912.02757 [cs, stat]. URL: http://arxiv.org/abs/1912.02757 (cit. on p. 2).

[8] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens (2017). *Exploring the Structure of a Real-Time, Arbitrary Neural Artistic Stylization Network*. arXiv: 1705.06830 [cs]. URL: http://arxiv.org/abs/1705.06830 (cit. on p. 5).

[9] A. Gray and J. Markel (1976). "Distance Measures for Speech Processing". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.5, pp. 380–391 (cit. on p. 7).

[10] Benjamin D. Haeffele and Rene Vidal (2017). "Global Optimality in Neural Network Training". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, pp. 4390–4398 (cit. on p. 2).

[11] Yi Hu and Philipos Loizou (2008). "Evaluation of Objective Quality Measures for Speech Enhancement". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16, pp. 229–238 (cit. on p. 7).

[12] Xun Huang and Serge Belongie (2017). *Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization*. arXiv: 1703.06868 [cs]. URL: http://arxiv.org/abs/1703.06868 (cit. on p. 5).

[13] Sergey Ioffe and Christian Szegedy (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv: 1502.03167 [cs]. URL: http://arxiv.org/abs/1502.03167 (cit. on p. 5).

[14] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi (2019). *Fr\'echet Audio Distance: A Metric for Evaluating Music Enhancement Algorithms*. arXiv: 1812.08466 [cs, eess]. URL: http://arxiv.org/abs/1812.08466 (cit. on p. 7).

[15] Taesup Kim, Inchul Song, and Yoshua Bengio (2017). *Dynamic Layer Normalization for Adaptive Neural Acoustic Modeling in Speech Recognition*. arXiv: 1707.06065 [cs]. URL: http://arxiv.org/abs/1707.06065 (cit. on p. 5).

[16] Diederik P. Kingma and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs]. URL: http://arxiv.org/abs/1412.6980 (cit. on p. 6).

[17] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein (2018a). *Visualizing the Loss Landscape of Neural Nets*. arXiv: 1712.09913 [cs, stat]. URL: http://arxiv.org/abs/1712.09913 (cit. on p. 2).

[18] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar (2018b). *Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization*. arXiv: 1603.06560 [cs, stat]. URL: http://arxiv.org/abs/1603.06560 (cit. on p. 6).

[19] Pranay Manocha, Zeyu Jin, Richard Zhang, and Adam Finkelstein (2021). *CDPAM: Contrastive Learning for Perceptual Audio Similarity*. arXiv: 2102.05109 [cs, eess]. URL: http://arxiv.org/abs/2102.05109 (cit. on pp. 2, 7).

[20] Aditya Krishna Menon, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar (2020). "CAN GRADIENT CLIPPING MITIGATE LABEL NOISE?" In: p. 26 (cit. on p. 8).

[21] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger (2019). *Occupancy Networks: Learning 3D Reconstruction in Function Space*. arXiv: 1812.03828 [cs]. URL: http://arxiv.org/abs/1812.03828 (cit. on p. 5).

[22] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng (2020). *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. arXiv: 2003.08934 [cs]. URL: http://arxiv.org/abs/2003.08934 (cit. on p. 1).

[23] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove (2019). *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. arXiv: 1901.05103 [cs]. URL: http://arxiv.org/abs/1901.05103 (cit. on p. 4).

[24] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville (2017). *FiLM: Visual Reasoning with a General Conditioning Layer*. arXiv: 1709.07871 [cs, stat]. URL: http://arxiv.org/abs/1709.07871 (cit. on p. 2).

[25] Alec Radford, Luke Metz, and Soumith Chintala (2016). *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. arXiv: 1511.06434 [cs]. URL: http://arxiv.org/abs/1511.06434 (cit. on p. 4).

[26] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A Hamprecht, Yoshua Bengio, and Aaron Courville (n.d.). "On the Spectral Bias of Neural Networks". In: (), p. 10 (cit. on p. 1).

[27] A.W. Rix, J.G. Beerends, M.P. Hollier, and A.P. Hekstra (2001). "Perceptual Evaluation of Speech Quality (PESQ)-a New Method for Speech Quality Assessment of Telephone Networks and Codecs". In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221). Vol. 2, 749–752 vol.2 (cit. on p. 7).

[28] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein (2020). *Implicit Neural Representations with Periodic Activation Functions*. arXiv: 2006.09661 [cs, eess]. URL: http://arxiv.org/abs/2006.09661 (cit. on pp. 1, 4).

[29] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams (2012). *Practical Bayesian Optimization of Machine Learning Algorithms*. arXiv: 1206.2944 [cs, stat]. URL: http://arxiv.org/abs/1206.2944 (cit. on p. 6).

[30] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng (2020). *Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains*. arXiv: 2006.10739 [cs]. URL: http://arxiv.org/abs/2006.10739 (cit. on p. 1).

[31] Joseph Turian and Max Henry (2020). *I'm Sorry for Your Loss: Spectrally-Based Audio Distances Are Bad at Pitch*. arXiv: 2012.04572 [cs, eess]. URL: http://arxiv.org/abs/2012.04572 (cit. on p. 8).

[32] Hongyang Zhang, Junru Shao, and Ruslan Salakhutdinov (2018). *Deep Neural Networks with Multi-Branch Architectures Are Less Non-Convex*. arXiv: 1806.01845 [cs, stat]. URL: http://arxiv.org/abs/1806.01845 (cit. on p. 2).

[33] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James S. Duncan (2020). *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients*. arXiv: 2010.07468 [cs, stat]. URL: http://arxiv.org/abs/2010.07468 (cit. on p. 6).