HYBRID NEURAL AUDIO EFFECTS

Riccardo SIMIONATO (riccardo.simionato@imv.uio.no)[1] and
Stefano FASCIANI (stefano.fasciani@imv.uio.no)[1]

[1]*Department of Musicology*, **University of Oslo**, Norway

ABSTRACT

Machine learning techniques are commonly employed for modeling audio devices, particularly analog audio effects. Conditioned models have been proposed as well. The conditioning mechanism utilizes the control parameters of the modeled device to influence the sound alteration process. Neural networks are shown to be capable of interpolating between conditioning parameter values. This paper further investigates the interpolation ability of neural audio effects. In particular, we introduce additional conditioning parameters to instruct the neural network to learn and predict different audio effects using Feature-wise Linear Modulation and the Gated Linear Unit. The resulting model is a hybrid neural effect that can reproduce, depending on the conditioning values, the audio-altering process of a specific audio effect or interpolates between them. We created hybrid audio effects from a preamp circuit, an optical compressor, and a tape recorder. The designed models are able to learn the sound alteration processes for individual effects and their combinations without producing audible artifacts, and users can use the conditioning parameters to navigate in a continuous space where each point represents a different hybrid audio effect.

1. INTRODUCTION

Neural networks have been extensively used to model analog audio effects. Starting from a distortion pedal, using a multilayer forward network [1], and continuing with convolutional-based networks for vacuum-tube amplifiers and various distortion pedals [2, 3]. Recurrent Neural Networks (RNNs), such as Long-Short Time Memory (LSTM) and Gated recurrent unit (GRU) networks have been investigated as well for similar types of effects [4, 5]. Works including control parameters exploited gated activation function when using convolutional-based networks, while for the case of RNNs, the control parameter has been considered as an extra input to the network. Compressor audio effects have been explored as well. Temporal Convolution Network (TCN) has been used for an optical compressor [6] together with the Feature-wise Linear Modulation (FiLM) method [7] as a conditioning method. In this case, the network models also have two conditioning parameters of the target device. A sequence-to-sequence model, based on recurrent networks, was used for another optical compressor. In this case, the conditioning is incorporated by exploiting the LSTM internal states [8]. Starting with two control parameters, the work was later extended to a fully conditioned model, including all four control parameters of the target device [9]. Delay-based effects could present additional challenges due to the variable temporal misalignment between input and output. In [10], a tape-based delay is modeled using GRU networks. The delay trajectory is first analyzed and extracted using impulse train signals. This signal represents the variable delay in samples over time. It is used to demodulate the signal before training the models or to directly guide a delay line built using differentiable signal processing [11]. In this way, the network can learn the saturation of the magnetic tape and an arbitrary delay line can be added a second time. LFO delay-based effects have been considered as well. In [12], authors use as extra input the LFO signal representing the frequency response of a time-varying system over time. By doing so, the model can theoretically produce LFO delay-based effects with arbitrary modulations. Combining the interpolation capability of the neural networks and the possibility of creating hybrid analog audio effects with a non-existent equivalent in the real world, we conceived a multi-type effect combining three analog effect types: a vacuum tube-based preamplifier, an optical compressor, and a tape recorder. The conditioning process uses the FiLM method, while the model inference is based on RNNs. The design of a neural model that emulates various analog audio and interpolates among them has not yet been investigated in the literature. Here, we propose using neural networks to create a hybrid between different sound coloring audio effects. In particular, the proposed model emulates the saturation-based coloring of tape recorders, tube-based amplification, and optical-based compression. These devices are based on three different and distant physical mechanisms. The resulting models navigate a one- or multi-dimensional space between two or three sound colorings. The rest of the paper is organized as follows. Section 2 details the functioning of the effect considered in this work. Section 3. describes the methodology and dataset we collected for our studies. Section 4 reports and discusses the results, while Conclusions are included in Section 5.

2. SELECTED ANALOG EFFECTS

The first effect we selected for this experimentation is the Universal Audio vacuum tube preamplifier 610-B [1]. In this case, the device includes the Universal Audio 6176 Channel Strip, which also features a transistor-based limiter amplifier 1176LN [2].

The second selected effect is the TUBE-TECH CL1B compressor [3]. This device is tube-based and optical, where the audio signal feeds a lighting element that, in turn, illuminates a light-sensitive resistor in the compression circuit. The resistance affects the compression circuit, determining how much and how quickly it attenuates the incoming audio signal. The CL 1B presents an output tube-based push-pull amplifier with variable gain, which is used to add harmonic distortion after the compression stage and not to limit the dynamics.

Finally, the third selected effect is the Akai 4000D open-reel tape recorder [4]. Tape recorder devices are based on magnetic tapes, which store the signal [13]. These devices typically consist of a recording amplifier, a recording head, a moving tape medium, a playback head, and a playback amplifier. The input signal first passes into the amplifier circuit and into an eventual stage equalization to compensate for the sound coloring introduced by magnetic tape. The signal is then routed to the tape record head and to the output, which is summed with its delayed replicas. The recording head produces a spatial magnetic field determined by both the recording head's properties and the input current's magnitude. The playback head restores the signal to electrical form, which process depends on the tape speed. The resulting delay effect is given by the position of the recording head on the tape loop and the distance from the playback head. In the last stage, another amplification is applied to the signal before the output. The tape movement speed is not perfectly constant due to small fluctuations produced by imperfections in the tape transport mechanics. These imperfections include cyclical components produced by the moving parts in the transport mechanism and stochastic behavior. These inconsistencies in the movement can be heard as small fluctuations in pitch. Magnetic tapes also generate noise due to the recording and playback processes.

3. METHODOLOGY

In this paper, we explore the interpolation capability offered by ANNs for modeling a hybrid audio effect based on the sound-altering process of different devices. For this experimentation, we have selected three significantly different analog audio effects, such as a vacuum tube pre-amplifier, an optical compressor, and a tape recorder. When training the system, we use extra input parameters to inform the neural network which of the three effects has generated the current input-output raw audio pairs. These parameters are used to condition the model during inference, allowing for generating a mixture of processing characteristics learned from the different effects, even when including intermediate conditioning settings not encountered during training. This allows the user to navigate between the alteration characteristics of the three analog effects. To evaluate to what extent the neural network can learn in such a context, we utilize two or three different audio effects to train the hybrid. After training the network, we verify whether the model can accurately replicate the sound alteration process of individual effects using the same conditioning input used during training and evaluate if varying the conditioning parameters to new values generates audible artifacts or spurious responses.

3.1 Architecture

The model is based on an encoder-decoder LSTM-based architecture as in [8]. RNNs offer advantages due to their ability to capture time dependencies in the data by utilizing internal states rather than solely relying on the input to generate each output prediction. This aspect is crucial when the goal is to achieve low input-to-output audio latency, which is essential in live interactive applications.

The network architecture used in this work is shown in Figure 1. It consists of two LSTM layers, a linear fully connected (FC) output layer, and the conditioning block. The network utilizes the 64 most recent input audio samples to generate one output audio sample. The input is split into 63 past samples $[x_n, ..., x_{n-63}]$ to feed the first LSTM layer. The input sample x_n at the current time step is first sent to the second LSTM layer. The first LSTM layer acts as an encoder and processes the 63 samples to compute the internal states $[h, c]$, which are used by the second LSTM layer. The latter uses the internal states to infer the output sample. The output is sent to the conditioning block, which is manipulated based on a continuous control parameter indicating the perceptual of the type of effect to use. A linear FC layer with one unit computes the output sample. The LSTM layers present 8 hidden units each, while the FC layers inside the conditioning block have 16 each. The architecture presents a maximum of 856 trainable parameters, depending on the design of the conditioning strategy.

3.2 Conditioning

The conditioning block, shown in Figure 2, consists of a Feature-wise Linear Modulation (FiLM) layer followed by a Gated Linear Unit (GLU) [14] layer. The FiLM layer applies an affine transformation to the vector p_n representing the conditioning information:

$$\boldsymbol{\hat{o}_n} = \boldsymbol{\eta_a} o_n + \boldsymbol{\eta_b} \tag{1}$$

where η_a and η_b are two vectors obtained from splitting the output of a linear FC layer fed with p_n. A GLU layer follows the FiLM block to determine the amount of information that should be passed. The GLU layer consists of a linear FC layer that takes the FiLM output vector as input

[1] https://www.uaudio.com/hardware/mic-preamps/solo-610.html

[2] https://www.uaudio.com/hardware/6176.html

[3] http://www.tube-tech.com/cl-1b-opto-compressor/

[4] https://reel-reel.com/tape-recorder/akai-4000d/

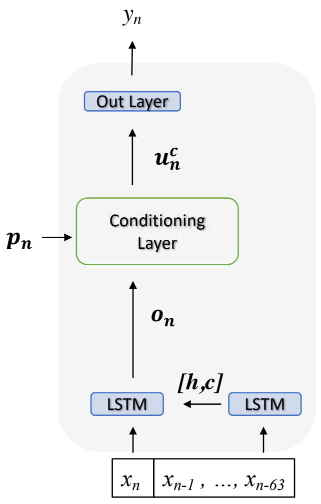

Figure 1. Architecture: the input is an array of the last 64 input audio samples $[x_n, ..., x_{n-63}]$, which is used to produce the output sample y_n. The input samples $[x_{n-1}, ..., x_{n-63}]$ are used as input of an LSTM layer, acting as an encoder. The input sample at the current time step x_n is first transformed by another LSTM layer acting as a decoder using internal states from the encoder. The output of the decoder LSTM layer is processed by the conditioning block followed by a linear FC layer with one unit, which produces the output audio sample y_n.

and computes a vector with twice its length. The resulting output is split equally into two vectors: q_1 and q_2. The softsign function is applied to q_2, and the resulting output is multiplied element-wise with q_1. Hence, the GLU block is described by the following:

$$\boldsymbol{o_n^c} = \boldsymbol{q_1} \otimes softsign(\boldsymbol{q_2}) \qquad (2)$$

The GLU layer determines the flow of information through the network, acting as a logical gate. The softsign activation function controls the extent to which the control parameters should influence the final output.

3.3 Datasets

The microphone tube preamplifier datatset [5] was recorded from a 6176 Vintage Channel Strip unit [15]. The preamp was overdriven, setting the gain to +10dB, the output level to 6, and both high and low boost/cut to 0 dB. This configuration resulted in high harmonic distortion.

The compressor dataset [6] originally included 5 equal-spaced values for the threshold, ratio, attack, and release parameters. For this work, we limited to only the recording with the threshold set to −10 dB, ratio to 4:1, attack to 0.5 ms, and release 0.05 seconds.

Lastly, the tape delay dataset is from [10]. In particular, we used the MAXELL, $\frac{7}{2}$ inches per second configuration. Since the audio files in the dataset present time delay, they were also time-aligned using the delay trajectory, which was included in the datasets before the training.

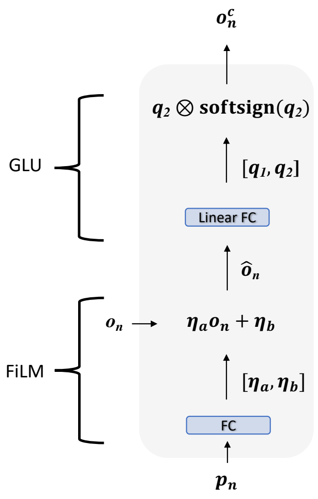

Figure 2. Conditioning Block: the conditioning information vector p_n fed a linear FC layer that produce an output with double its input length. This output vector is split into two vectors used to apply an affine transformation on the vector o_n, which represents the current input sample of the network. Subsequently, the transformed vector passes through another linear FC layer, producing another output vector double the length of its input. This output is also split into two halves. To one half is applied the softsign function, and the result is multiplied element-wise to produce the final output.

We limited the datasets to 3 minutes and 20 seconds, which is the length of the shortest dataset, specifically the CL1B compressor dataset. By doing so, we ensure an equal amount of recordings from the two different effects in the datasets used for this study. In addition, 3 minutes at 44.1 kHz is shown as sufficient data for training [4]. We experimented with training 4 different hybrid neural audio effects. We utilize the three different pairs of datasets with a single conditioning parameter and the three datasets with a triplet of individual conditioning parameters, as detailed in Table 1. We also train three separate models with only one effect dataset as a baseline for modeling accuracy.

3.4 Experimenting and Learning

The models are trained for 60 epochs and use the Adam [16] optimizer with a gradient norm scaling of 1 [17]. The training was stopped earlier in case of no reduction of validation loss for 10 epochs. We design a time-based schedule for the learning rate as follows:

$$lr = LR * 0.25^e. \qquad (3)$$

LR represents the initial learning rate, which is $3 \cdot 10^{-}4$, and e represents the current epoch's number. The loss function used is the mean square error (MSE) and is computed using the model's weights that minimize the validation loss throughout the training epochs. The input signal is split into segments of 600 samples (equivalent to 13.6

[5] https://zenodo.org/records/3562442

[6] https://www.kaggle.com/datasets/riccardosimionato/tubetech-cl-1b

Model	Dataset composition	p
P	preamplifier	-
C	compressor	-
T	tape recorder	-
TP	tape recorder preamplifier	0.0 1.0
CP	compressor preamplifier	0.0 1.0
CT	compressor tape recorder	0.0 1.0
CTP	compressor tape recorder preamplifier	$[1.0, 0.0, 0.0]$ $[0.0, 1.0, 0.0]$ $[0.0, 0.0, 1.0]$

Table 1. Models and datasets composition with the respective conditioning parameter values (p) used for the training.

ms) to be processed before updating the weights. The models are evaluated using the MSE, ESR, and multi-resolution STFT with [256, 512, 1024] as resolutions.

Finally, the dataset is split into 80% for the training set, 10% for the validation set, and 10% for the test set. The $80 - 10 - 10\%$ split was carried out at the individual dataset level, which ensures that both effects examples are included with an equal share in each subset and are equally used for training and evaluating the model. Minor manual adjustments are made to ensure that splitting points fall within segments of silence.

4. RESULTS

Table 2 details the errors for all the cases. Specifically, when considering models trained with a mixture of audio effects, the error refers only to the segment of the test set representing a specific effect. In this way, we assert how much the model accuracy drops when including more effects compared to a model that includes only one of them. We can see how, as expected, all the errors particularly increased in the case of CTP. In CTP, we can note a drastic drop in performance for the tape recorder test set. CTP represents the most complex case, where a single model is expected to emulate significantly different audio alteration processes. Considering the case with two effects, the emulation of the tape recording when the preamplifier dataset is included is less accurate. At the same time, a minor drop in accuracy happens if the compression is learned together with the tape delay. Even if there is significantly less performance drop, the same happens with the preamplifier. Learning the compression is more difficult when it needs to be learned in combination with the tape recorder. This suggests that it is particularly challenging for the model to add and remove time misalignments between input and output and, in turn, the time shifting during the process. The TP model resulted in more challenges during the learning process, likely due to their effect on the sound: the preamplifier generates a significant harmonic distortion in the signal, while the tape recorder introduces time fluctuations. The compressor, on the other hand, requires less alteration of the signal. Looking at the spectrograms, as can be seen in Figure 3, the prediction of the CTP model, referring to the compressor, is the least affected by the mixture of effects, while the preamplifier and the tape recorder present more mismatch in the high-frequency content respect to the singular cases represented by the T and C model. In accordance with the STFT errors, the preamplifier is the least affected among the cases considering two effects, while the compressor presents more mismatch CTP, although the CTP model presents lower errors. The tape delay presents the most mismatch in frequency content in CTP. This could be because tape delay time shifting is present in a smaller percentage, considering the total amount of data. The time shifting in the TP and CP cases represents half of the seen examples. Additionally, as shown in Figure 4, the T and CT models similarly emulate the time misalignment introduced by the tape recorder dataset, while the TP model, modeling also the preamplifier, is less accurate with this aspect. Also, the TP model underpredicts the amplitudes. The CTP model, even if it still mispredicts the amplitude, better models the time fluctuations. On the other hand, all models still capture the time shifting. Figure 5 shows the predictions of CP, CT, and TP models when the parameter is set at the middle of the navigable space, such as 0.5. In this scenario, the model generates a hybrid effect between the two analog effects, representing situations that are not reproducible in the real-world and, in turn, are not comparable with real recordings. The plots show how the model creates the hybrid effects. For example, looking at the time plot, the CP model amplifies a compressed version of the input signal. Additionally, Figure 6 shows the spectrograms of the output models when continuously changing the conditioning parameters by modulating it using sine waves at 0.05 Hz. From the spectrogram, we can see that no visible artifacts were introduced in the process. Similarly, Figure 7 shows the predictions in the case of the CTP model considering various parameter values, such as $[0.3, 0.3, 0.3]$, $[0.5, 0.25, 0.25]$,

Dataset	MSE	ESR	STFT
P	$4.34 \cdot 10^{-2}$	$1.55 \cdot 10^{-1}$	$3.94 \cdot 10^{-1}$
C	$1.52 \cdot 10^{-4}$	$1.07 \cdot 10^{-1}$	$3.10 \cdot 10^{-1}$
T	$3.46 \cdot 10^{-4}$	$1.44 \cdot 10^{-1}$	$2.51 \cdot 10^{-1}$
TP (T)	$1.36 \cdot 10^{-3}$	$5.66 \cdot 10^{-1}$	$8.11 \cdot 10^{-1}$
TP (P)	$5.05 \cdot 10^{-2}$	$1.81 \cdot 10^{-1}$	$3.83 \cdot 10^{-1}$
CP (C)	$4.11 \cdot 10^{-4}$	$2.91 \cdot 10^{-1}$	$5.42 \cdot 10^{-1}$
CP (P)	$5.00 \cdot 10^{-2}$	$1.79 \cdot 10^{-1}$	$3.78 \cdot 10^{-1}$
CT (C)	$1.03 \cdot 10^{-3}$	$7.31 \cdot 10^{-1}$	$9.63 \cdot 10^{-1}$
CT (T)	$4.52 \cdot 10^{-4}$	$1.88 \cdot 10^{-1}$	$3.02 \cdot 10^{-1}$
CTP (C)	$3.93 \cdot 10^{-4}$	$2.78 \cdot 10^{-1}$	$3.44 \cdot 10^{-1}$
CTP (T)	$2.16 \cdot 10^{-2}$	$9.02 \cdot 10^{-1}$	1.80
CTP (P)	$5.54 \cdot 10^{-2}$	$1.98 \cdot 10^{-1}$	$4.37 \cdot 10^{-1}$

Table 2. MSE, ESR, and STFT errors referring to the models trained detailed in Table 1. In this latter case, the errors also refer to the segment of the test set considering the conditioning cases separately. Specifically, T refers to the case of Tape, P to preamplifier, and C to compressor mode.

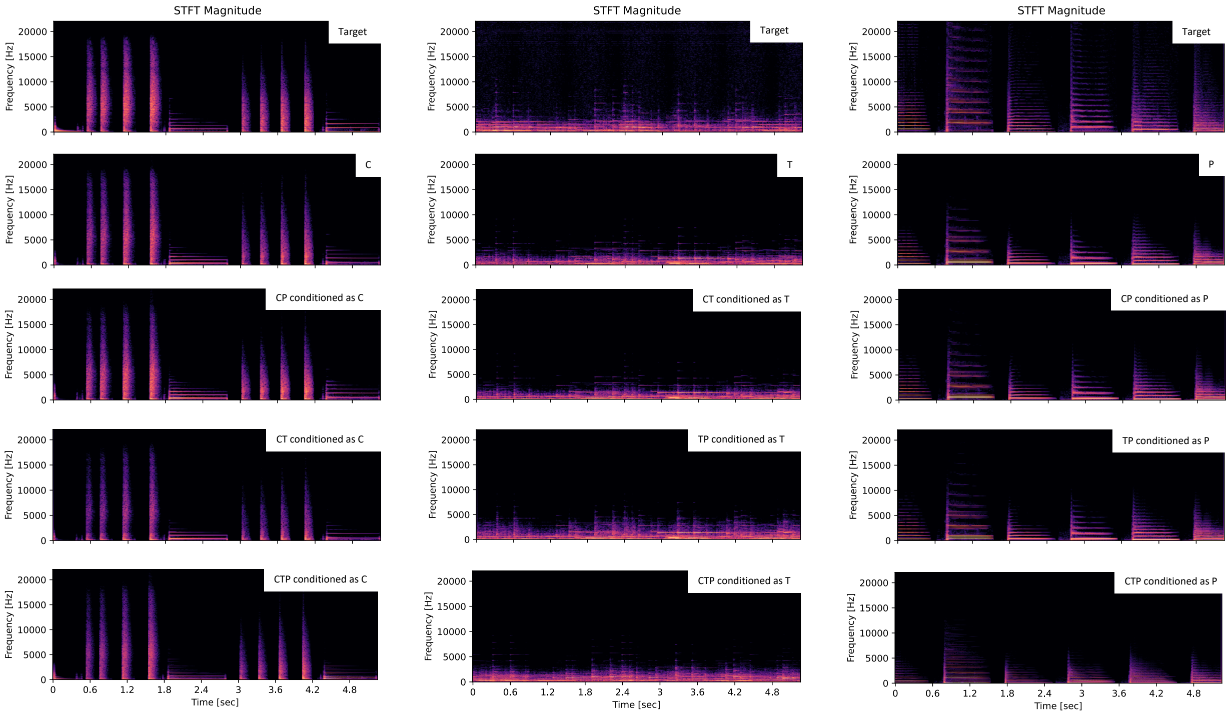

Figure 3. Normalized spectrograms of 5 seconds included in the test sets. Targets (top) against C (left), T (middle), and P (right) models, and against CT, TP, CP, and CTP models conditioned as C, as T, and as P, respectively.

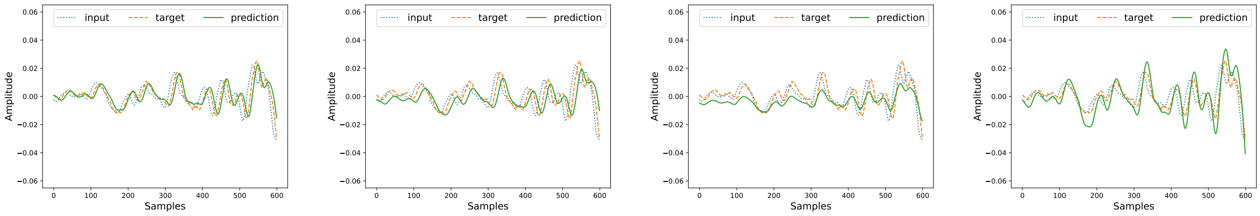

Figure 4. Input, target, and prediction referring to a segment of the tape recorder test set for the case of T, CT, TP, CTP, and TCP models. The plots show the time fluctuations present in the tape recorder dataset.

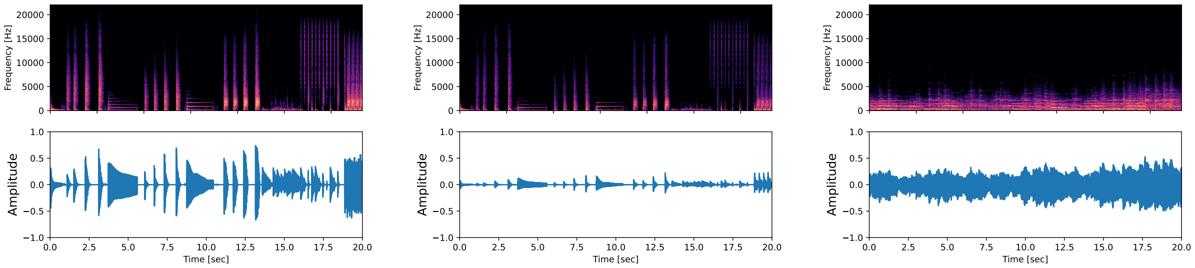

Figure 5. Waveforms and spectrograms (normalized) of 20 seconds included in the test sets, resulted from the CP (left), CT (middle), and TP (right) models when the parameter is 0.5.

$[0.25, 0.5, 0.25]$, and $[0.25, 0.25, 0.5]$. The conditioning vectors were chosen to have a norm not greater than $1.$. On the other hand, in the CP, TP, and CT models, where the parameters switch continuously between the effects, the conditioning is designed differently in this case. In particular, CTP has 3 different parameters governing the amount of the effects to add to the input signal. With these conditioning values, all the waveforms present visible compression, enhanced in the case with $[0.5, 0.25, 0.25]$ as the conditioning vector, as expected. Analogously, when the conditioning vector is $[0.25, 0.25, 0.5]$, the model amplifies more than the other cases. To further explore the model behavior, Figure 8 presents extreme conditioning cases, such as $[0.0, 0.0, 0.0]$ and $[1.0, 1.0, 1.0]$. When the val-

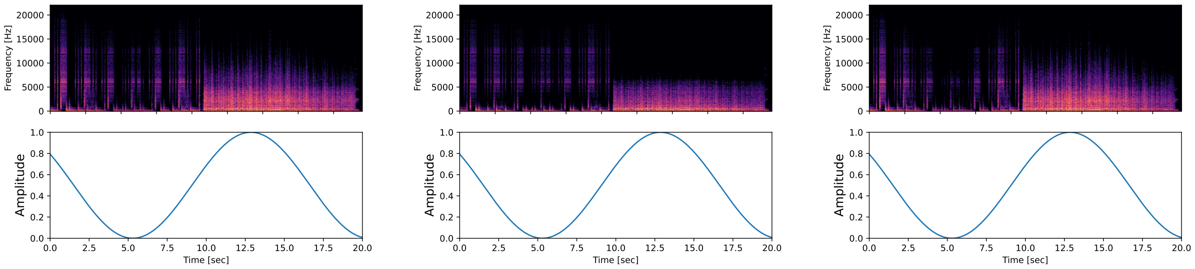

Figure 6. Normalized spectrograms of a 20 second signal resulted from the CP (left), CT (middle), and TP (right) models when the parameter is continuously changed by modulating it using a sine wave at 0.05 Hz.

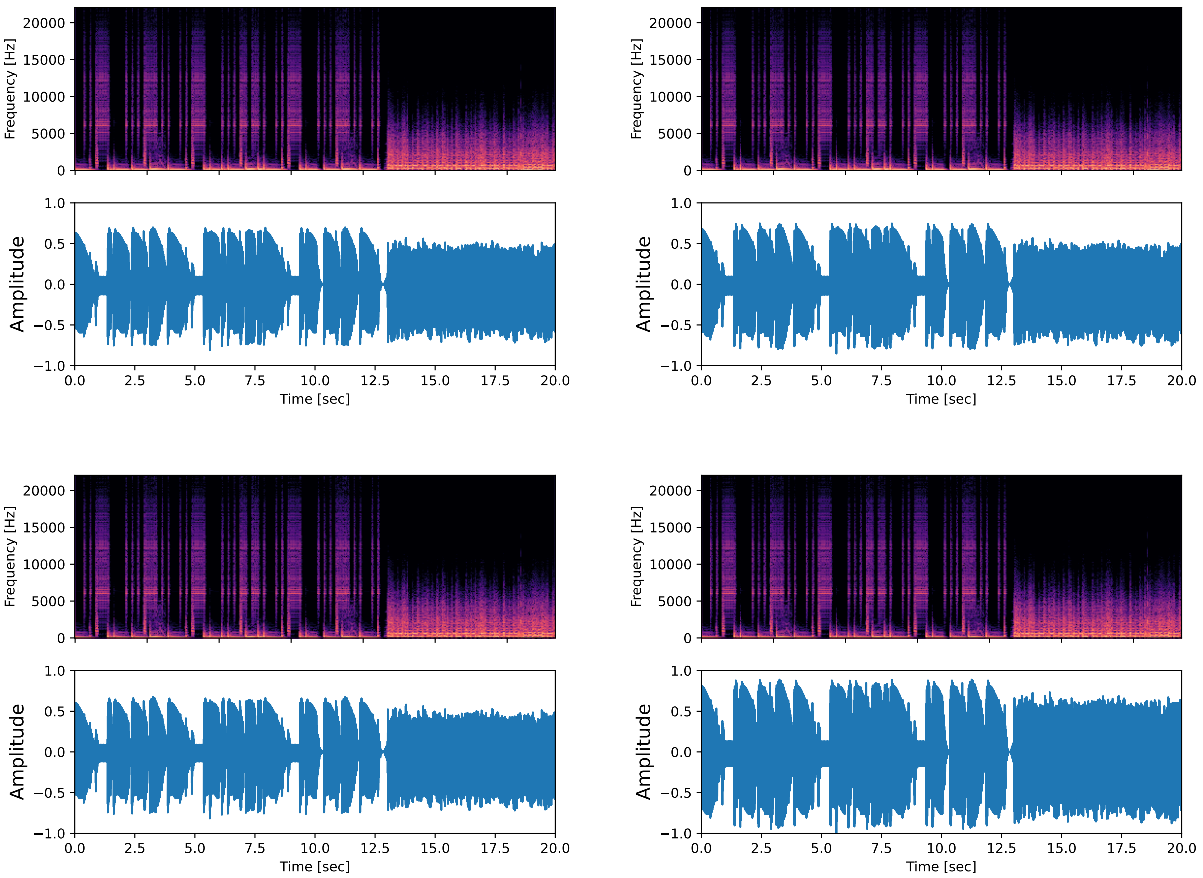

Figure 7. Waveforms and normalized spectrograms of a 20 second signal resulted from the CTP model when the parameter is $[0.3, 0.3, 0.3]$ (top-left), $[0.5, 0.25, 0.25]$ (top-right), $[0.25, 0.5, 0.25]$ (bottom-left), and $[0.25, 0.25, 0.5]$ (bottom-right).

ues are set to $[1.0, 1.0, 1.0]$, the model significantly amplifies the amplitude's input signal, as though three different predictions are added separately. Interestingly, when the parameter vector is set to $[0.0, 0.0, 0.0]$, the model significantly compresses the signal instead of replicating the input signal. This latter behavior suggests that the three parameters do not simply indicate the amount of the singular effects added to the input signal; instead, they describe a more complex transformation. Figure 8 shows two other cases with conditioning vectors having a norm greater than $1.$, such as $[0.5, 0.5, 0.5]$ and $[0.3, 0.8, 0.5]$. Also, in these cases, the signal is still significantly amplified and presents amplitudes greater than $1.$. Finally, Figure 9 shows the spectrogram of the CTP model when continuously changing the conditioning parameters by modulating it using three sine waves at 0.05, 0.07, and 0.09 Hz. The figure presents the case of musical and sawtooth waves. Also here, as in all of the presented examples, as can be seen, do not introduce visible artifacts, nor when listening to the audio sample that can be found in [7].

[7] https://github.com/RiccardoVib/Hybrid-Neural-Audio-Effects

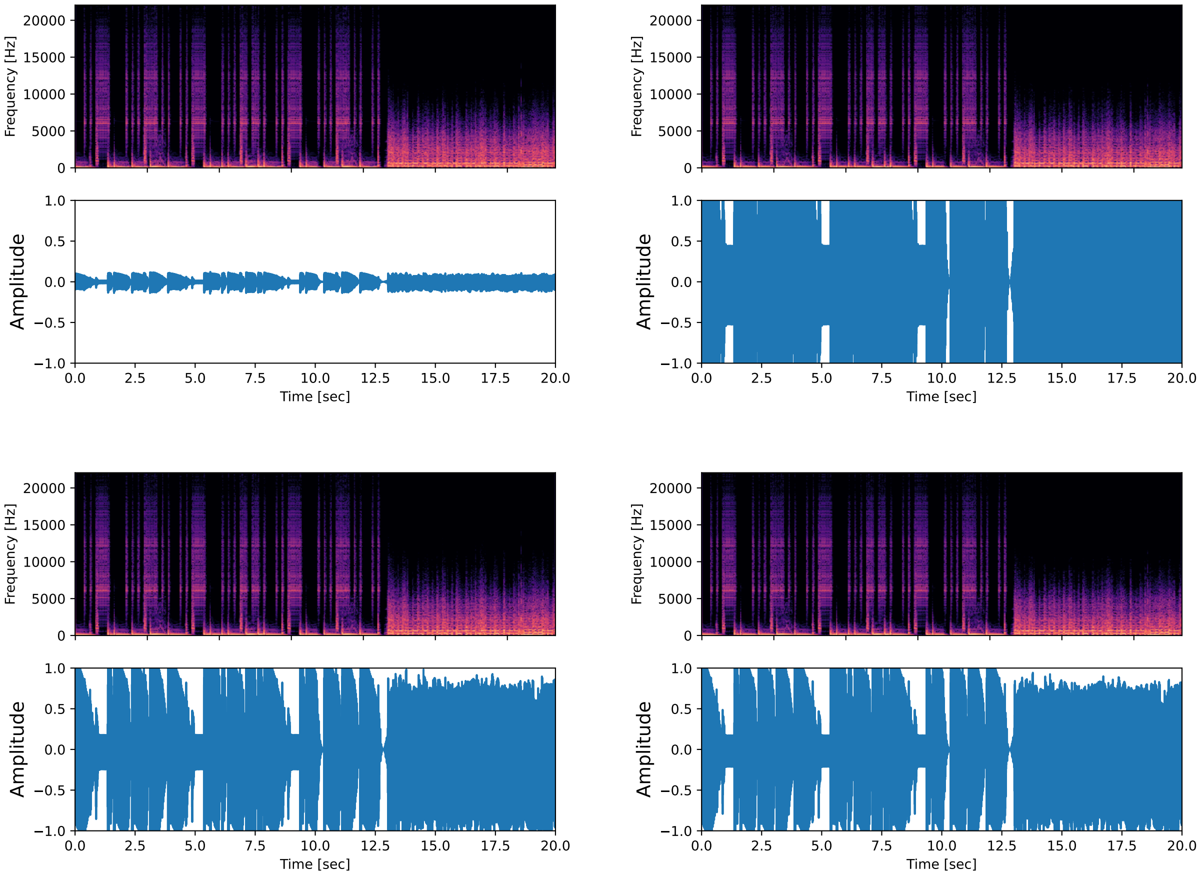

Figure 8. Waveforms and normalized spectrograms of a 20 second signal resulted from the CTP model when the parameter is [0.0,0.0,0.0] (top-left), [1.0,1.0,1.0] (top-right), [0.5,0.5,0.5] (bottom-left), and [0.3,0.8,0.5] (bottom-right).

5. CONCLUSION

Machine learning nowadays integrates digital signal processing to model physical systems, such as analog audio effects. In this context, the models must be conditioned based on the user control parameters to control how to alter the sound modification process. Neural networks are capable of interpolating between seen data. This paper investigates this interpolating ability to create hybrid models mixing different audio effects and navigating a space not directly existing in the physical world. With this goal, we defined continuous parameters with the range [0.0, 1.0] to condition the network and emulate different analog effects, such as a compressor, a preamplifier, and a tape recorder. We used three datasets from a real audio device and experimented with 4 different hybrid neural effects. Three were trained utilizing different pairs of datasets with a single conditioning parameter, and one was trained using all datasets conditioned with a three-dimensional vector. The conditioning layer is based on Feature-wise Linear Modulation and Gated Linear Unit. The final models can navigate between amplification, compression, and saturation types of effect, although with less accuracy than the model trained for a specific effect only. The hybrid models do not introduce visible or audible artifacts when changing across different sonic characteristics. The proximity in the 1D space of the preamplifier or the tape recorder, the first creating harmonic distortion and the latter introducing time fluctuations, interferes more with learning the other effect. We showed two conditioning cases: the first uses a parameter to navigate a 1D space and interpolate among the two learned effects; the second uses three parameters to determine the amount of the individual effects to add to the hybrid mixture of effects. The design of the parameters and how to associate them with the different effects influence the user's morphing control. Different designs offer different interaction affordances. We demonstrate that specific neural network architecture, even with a small number of parameters, can learn different nonlinear sound alteration characteristics, such as those of distortion and compression, and temporal profiles of such effects, such as the time-variant characteristics of the compression and time shifting of the tape delay. Moreover, the approach can be expanded to other effects, but the selected model must be able to learn each effect separately in a black-box manner. In addition, selecting effects with similar sound altering characteristics can result in more accurate hybrid neural effects, such as modeling multiple distortion devices.

In future work, the interoperability of the models will be further investigated by including control parameters of individual audio effects as additional conditioning parame-

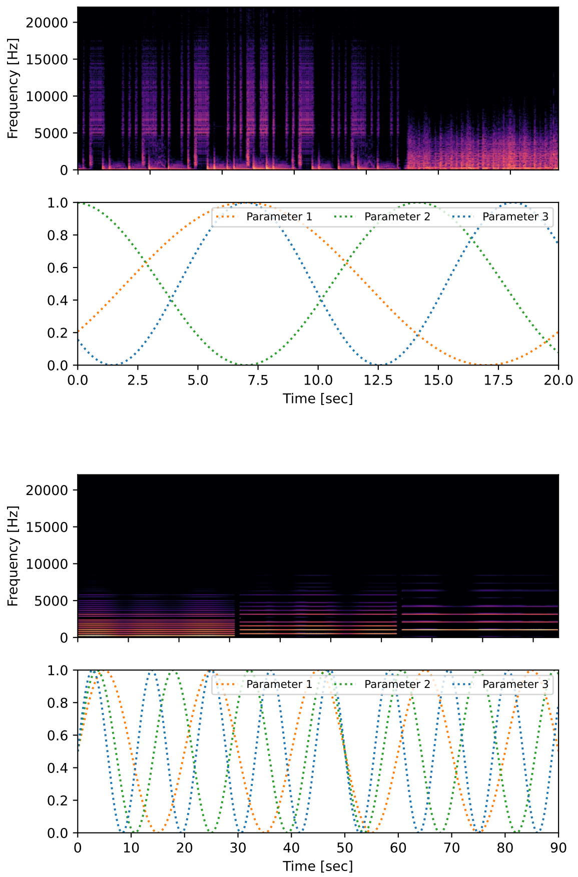

Figure 9. Normalized spectrogram of a 20 second music signal (top) and a 90 second sawtooth waves at 262.5, 525, and 1050 Hz (bottom), produced by the CTP model when the parameter is continuously changed by modulating it using a sine wave at 0.05, 0.07, and 0.09 Hz.

ters. This requires working with a homogeneous set of audio effects that present similar or compatible control parameters. Introducing user-control parameters to further condition the network can expand and enrich the sonic alteration capability of the hybrid audio effect.

6. REFERENCES

[1] D. S. Mendoza, *Emulating electric guitar effects with neural networks*, Barcelona, Spain, 2005.

[2] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, “Deep learning for tube amplifier emulation,” in *Proc. of 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK, 2019.

[3] J. Covert and D. L. Livingston, “A vacuum-tube guitar amplifier model using a recurrent neural network,” in *Proc. of 2013 IEEE Southeastcon*, Jacksonville, Florida, USA, 2013.

[4] E.-P. Damskägg, L. Juvela, V. Välimäki *et al.*, “Real-time modeling of audio distortion circuits with deep learning,” in *Proc. of 16th Int. Sound and Music Computing Conf. (SMC)*, Malaga, Spain, 2019.

[5] J. Chowdhury, “A comparison of virtual analog modelling techniques for desktop and embedded implementations,” *arXiv preprint arXiv:2009.02833*, 2020.

[6] C. J. Steinmetz and J. D. Reiss, “Efficient neural networks for real-time analog audio effect modeling,” in *152nd Audio Engineering Society Convention*, The Hague, Netherlands, 2022.

[7] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, “Film: Visual reasoning with a general conditioning layer,” in *Proc. of 2018 Conference on Artificial Intelligence (AAAI)*, New Orleans, Louisiana, USA, 2018.

[8] R. Simionato and S. Fasciani, “Deep learning conditioned modeling of optical compression,” in *Proc. of 25th Int. Conf. Digital Audio Effects (DAFx)*, Vienna, Austria, 2022.

[9] ——, “Fully conditioned and low-latency black-box modeling of analog compression,” in *Proc. of 26th Int. Conf. Digital Audio Effects (DAFx)*, Copenaghen, Denmark, 2023.

[10] A. Wright, V. Valimaki, O. Mikkonen, and E. Moliner, “Neural modeling of magnetic tape recorders,” in *Proc. of 26th Int. Conf. Digital Audio Effects (DAFx)*, Copenaghen, Denmark, 2023.

[11] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, “Ddsp: Differentiable digital signal processing,” in *Proc. of Int. Conf. Learning Representations (ICLR-2020)*, Brighton, UK, 2020.

[12] A. Wright and V. Valimaki, “Neural modeling of phaser and flanging effects,” *Journal of the Audio Engineering Society*, vol. 69, no. 7/8, pp. 517–529, 2021.

[13] M. Camras and D. W. Martin, “Magnetic recording handbook,” 1989.

[14] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, “Language modeling with gated convolutional networks,” in *International conference on machine learning*, Sydney, Australia, 2017.

[15] M. A. Martínez Ramírez, E. Benetos, and J. D. Reiss, “Deep learning for black-box modeling of audio effects,” *Applied Sciences*, vol. 10, no. 2, p. 638, 2020.

[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *Proc. of 2015 Int. Conf. Learning Representations (ICLR)*, 2015.

[17] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International conference on machine learning*, Atlanta, USA, 2013.