Emulating Diode Circuits with Differentiable Wave Digital Filters

Jatin Chowdhury
Unaffiliated
jatin@ccrma.stanford.edu

Christopher Johann Clarke
Singapore University of Technology and Design
christopher_clarke@mymail.sutd.edu.sg

ABSTRACT

Wave Digital Filters and neural networks are two popular solutions for circuit modelling. This paper describes the development of a Differentiable Wave Digital Filters library. Diode clipper circuits were constructed. A dataset was collected from the circuits and, with the library, was used to train a real-time deployable model. The trained model has higher accuracy and similar computation time when compared to traditional white-box models.

1. INTRODUCTION

Virtual analog (VA) modelling is often divided into two non-distinct types of approaches. "White-box" modelling involves developing a circuit model based on the physical interactions of the circuit elements, while "black-box" modelling involves taking measurements from the circuit, and creating a digital system that replicates perceptually relevant aspects of the circuit's behaviour [1]. VA modelling approaches that combine elements of both white-box and black-box methods are typically referred to as "grey-box" approaches.

Wave Digital Filters (WDFs) are a white-box method that works by modelling individual circuit elements in the wave domain, and modelling the interactions of those elements with wave domain "adaptors" [2, 3]. WDFs are a powerful tool due to their modular and flexible nature; however, as with many other white-box approaches, they may provide inaccurate results when modelling circuits containing components that behave in a non-ideal manner.

In recent years, there has been significant research on developing black-box models using neural networks [4–6]. While this approach can achieve high levels of accuracy when comparing the model to the reference circuit, one drawback is that it can be difficult for neural network-based models to include circuit control parameters, particularly continuous controls such as potentiometers. Wright et al. suggest training neural networks using control values as additional inputs [6], however this approach often requires training a larger network, which requires more computing resources to run in real-time.

One potential solution to the respective issues of the WDF and neural network modelling techniques is to combine them via Differentiable Digital Signal Processing (DDSP) [7]. The fundamental idea behind DDSP is to implement basic signal processing building blocks within a framework of automatic differentiation. Then, gradient descent may be used to optimize various parameters of the signal processing algorithm for a given set of input and target data. In recent years, DDSP has been applied to IIR filter design [8] and parameter discovery for white-box circuit models [9].

This paper proposes a grey-box modelling technique using Differentiable Wave Digital Filters (DWDFs). The DWDF technique involves constructing a WDF model of a reference circuit, and then training neural networks to replace one or more of the circuit elements in the WDF model. With this technique, non-ideal components can be modelled with a high degree of accuracy, since the neural networks may be trained with data collected from the actual circuit. Further, including the circuit's control parameters in the model is trivial, so long as the control parameters are connected to circuit elements being modelled with traditional WDF elements.

The structure of this paper is as follows: Section 2 discusses the development of a DWDF library, and the use of that library for solving simple parameter discovery tasks. Section 3 presents a process for training neural networks to emulate the behaviour of anti-parallel diodes in the wave domain. Section 4 considers the implementation of WDF models with neural network components for real-time use.

2. DIFFERENTIABLE WAVE DIGITAL FILTERS

While several WDF libraries exist [10, 11], they are primarily focused on implementing real-time circuit models, and are not well-suited for differentiation. With that in mind, a new WDF library was implemented in Python, using the TensorFlow framework for automatic differentiation [12]. The source code for the DWDF library is available on GitHub [13].

2.1 Library Implementation

Wave Digital Filters operate on wave variables, rather than the Kirchoff variables typically used for analyzing circuits (voltage v, and current i). The wave domain variables are defined generically as,

$$\begin{aligned} a &= R_0^{\rho-1} v + R_0^{\rho} i \\ b &= R_0^{\rho-1} v - R_0^{\rho} i \end{aligned} \tag{1}$$

where a is defined as the incident wave for a given circuit port, b is the reflected wave, and R_0 is the port impedance. ρ is a wave definition parameter: "voltage waves" are defined for $\rho = 1$, while "current waves" are given when $\rho = 0$. In this writing, only voltage waves will be used.

2.1.1 DWDF 1-Ports

Most simple circuit elements, such as resistors and capacitors may be implemented as 1-port elements. A resistor is defined by the voltage wave relationship,

$$b = 0 \tag{2}$$

while a capacitor maybe similarly characterized as,

$$b = a z^{-1} \tag{3}$$

where z^{-1} is defined as a 1-sample delay. Both of these circuit elements may be trivially implemented with TensorFlow, and may therefore be differentiated automatically. For cases where the resistance or capacitance of a given circuit element may not be known, the library also allows the component value to be initialised as a "trainable" variable.

2.1.2 DWDF Adaptors

Simple WDF adaptors such as series and parallel adaptors may be implemented generally as N-port elements. These adaptors are typically implemented as 3-port adaptors, since any N-port adaptor can be made up of a chain of 3-port adaptors. A series adaptor is defined by the voltage wave relationship,

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0 & -1 & -1 \\ -R_1 & \frac{R_2}{R_1+R_2} & \frac{R_1}{R_1+R_2} \\ -R_2 & -\frac{R_2}{R_1+R_2} & \frac{R_1}{R_1+R_2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{4}$$

where a_n, b_n, and R_n are the incident wave, reflected wave, and port impedance at a given port. A parallel adaptor is similarly defined by the voltage wave relationship.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{R_2}{R_1+R_2} & \frac{R_1}{R_1+R_2} \\ 1 & -\frac{R_1}{R_1+R_2} & \frac{R_1}{R_1+R_2} \\ 1 & \frac{R_2}{R_1+R_2} & -\frac{R_2}{R_1+R_2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \tag{5}$$

The DWDF library implements these adaptors using the one-multiply form described in [2]. When training a DWDF structure, TensorFlow will automatically propagate gradients through the relevant adapters so that a quantity anywhere in the structure maybe optimized via gradient descent.

2.2 Parameter Discovery with DWDFs

As a test of the DWDF models constructed with the library, two simple parameter discovery tasks were attempted, similar to those outlined in [9].

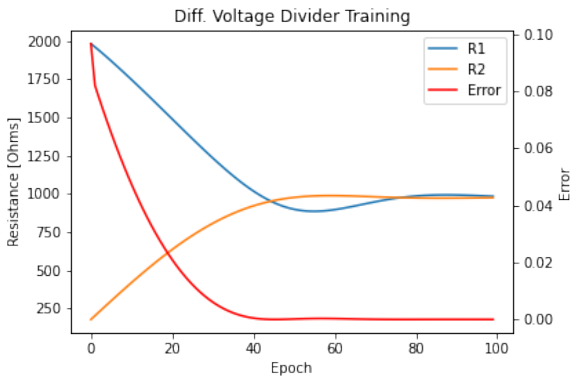

Figure 1: Training the voltage divider WDF model.

2.2.1 Voltage Divider

For the first task, synthetic data was generated for a simple voltage divider circuit made up of two equivalent resistors, corresponding to a gain of $G = 0.5$. A corresponding WDF model was constructed, using resistors with starting values of 2 kΩ and 100 Ω, both initialised as trainable variables. The WDF model was trained on the synthesized data for 100 epochs, using an Adam optimizer [14],

$$\theta_t \leftarrow \theta_{t-1} - \alpha \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \cdot \frac{m_t}{\sqrt{v_t} + \hat{\epsilon}}, \tag{6}$$

where θ is the quantity being optimized, α is the initial learning rate, m_t is the exponential moving average of the gradient, and v_t is the squared gradient. Hyperparameters β_1 and β_2 represent the exponential decay rates of the first- and second-order moment estimates respectively, with default values $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the voltage divider circuit, the Adam optimizer was given an initial learning of $\alpha = 25\ \Omega$. The model was trained using a mean-squared error (MSE) loss function,

$$L_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} (y_t(i) - y_p(i))^2 \tag{7}$$

where y_t represents the "target" data, y_p represents the "predicted" signal output by the model, and N represents the length of the signal in samples. Table 1 shows the start and end values for each circuit element, as well as the final error. Fig. 1 plots the component values and error over the course of the training process. The component values after training correspond to a gain of $G = 0.502$.

2.2.2 RC Lowpass Filter

For the second task, synthetic data was generated for a first-order RC lowpass filter circuit, with a cutoff frequency of $f_c = 720$ Hz. A corresponding WDF model was constructed, with an initial cutoff frequency of 159 Hz ($R = 1$ kΩ, $C = 1\ \mu$F), with both component values initialised as trainable variables. The model was again trained with an MSE loss function for 100 epochs. The resistor was trained with an Adam optimizer with an initial learning rate of $\alpha = 25\ \Omega$, while the capacitor was trained with a separate Adam optimizer with an initial learning rate of

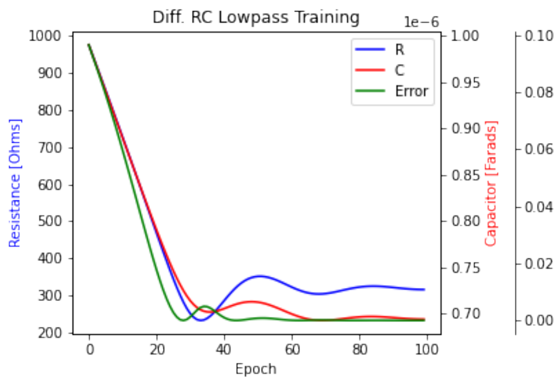

Figure 2: Training the RC lowpass WDF model.

Circuit	Element			Error [V]
Voltage Divider	R_1	Start	2 kΩ	3.4e−6
		End	984.75 Ω	
	R_2	Start	100 Ω	
		End	975.65 Ω	
RC Lowpass	R	Start	1 kΩ	2.58e−5
		End	315.99 Ω	
	C	Start	1 μF	
		End	693.8 nF	

Table 1: Training statistics for voltage divider and RC lowpass parameter discovery tasks.

$\alpha = 10$ nF. Table 1 shows the start and end values for each circuit element, as well as the final error. Fig. 2 plots the component values and error over the course of the training process. The component values after training correspond to a cutoff frequency of $f_c = 726$ Hz.

3. NEURAL WDF DIODE MODELS

While traditional wave domain diode models can achieve high accuracy when compared to the expected behaviour of ideal diodes, they typically require the evaluation of the Lambert $\mathcal{W}$ function at least once per-sample, which can limit the performance of real-time implementations [15]. As a result, real-time implementations often use lookup tables or approximations, which offer a trade-off between accuracy and performance [16]. Further, manufacturing inconsistencies and other real-world factors may cause diodes to behave non-ideally, thereby decreasing the accuracy of traditional wave domain models.

Neural networks offer a potential solution to these limitations, by leveraging data measured from a physical circuit to help train a more accurate model. As an example, this section will present a WDF model of an RC diode clipper (similar to one found in many guitar distortion pedals), in which the diodes are modelled using a neural network.

3.1 Diode Circuit Data

Data for the neural networks was prepared by constructing the diode clipper circuit shown in Fig. 3. This circuit varied with the amount of diodes on the “upward-facing” side

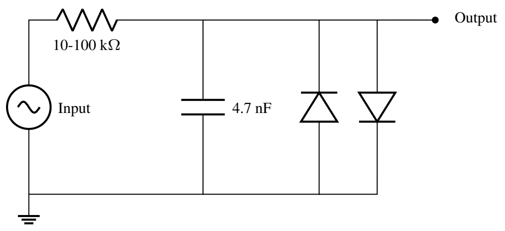

Figure 3: Diode Clipper with anti-parallel diode configuration. In this schematic, there are only 1 upward-facing and 1 downward-facing diode.

Diode Notation	Upward Diodes	Downwards Diodes
1U1D	1	1
1U2D	1	2
1U3D	1	3
2U2D	2	2
2U3D	2	3
3U3D	3	3

Table 2: This table shows the notation used for the labelling of the various diode clipper schematics, wherein 1U1D refers to a diode clipper with 1 upward-facing diode and 1 downward-facing diode.

and “downward-facing” side. Table 2 shows the different anti-parallel diode configurations sampled.

Following the construction of the 6 diode clipper circuits, a dataset was prepared. The input data for this dataset was taken from the IDMT-SMT-Guitar Dataset [17]. About 14 seconds of audio was used as input to the diode clipper circuit. The audio was output from a Universal Audio Apollo-Twin to the diode clipper circuit, and was sampled with a Digilent Analog Discovery 2 USB Oscilloscope, as shown in Fig. 4. The oscilloscope offers a range of sampling rates from 50 mHz to 100 MHz; measurements were made at 50 kHz since it was the closest option to a standard audio sampling rate. Each diode clipper circuit was sampled at 5 different resistor values (10 kΩ, 25 kΩ, 45 kΩ, 75 kΩ, 100 kΩ). The capacitor value remained unchanged.

Although 14 seconds seems like a small amount of data, one must note that model performance is being evaluated at the sample level, meaning that there will be about 4×10^6 samples to be used for training. 80% of the data was used for training the model, and 20% was used to validate the model’s accuracy.

3.2 Diode Network Architecture

A network architecture was chosen as a sequence of fully-connected layers similar to the network presented in [18], with two inputs (the incident wave and port impedance) and one output (the reflected wave). Since diodes typically exhibit nonlinear behaviour, a $\tanh$ activation function is used in between each neural network layer. To improve training speed, the port impedance was replaced with the $\log$ of the port impedance, and the reflected wave was replaced with the negation of the reflected wave. The model

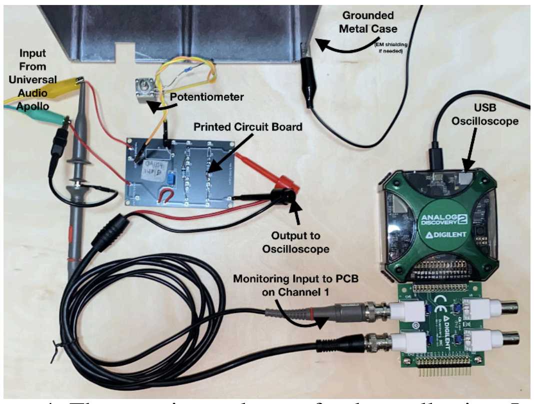

Figure 4: The experimental setup for data collection. Input comes from a Universal Audio Apollo-Twin audio interface, and data is captured and logged by a Digilent Analog Discovery USB Oscilloscope.

hyperparameters consist of the number of "hidden" layers to use in between the model inputs and outputs, as well as the size of the hidden layers. A visualization of an example "2x4" network with 2 hidden layers, each with 4 fully-connected units can be seen in Fig. 5. When trained, the diode network represents a memoryless mapping between the inputs and outputs.

$$b = -f\left(\begin{bmatrix} a \\ \log(R) \end{bmatrix}\right) \tag{8}$$

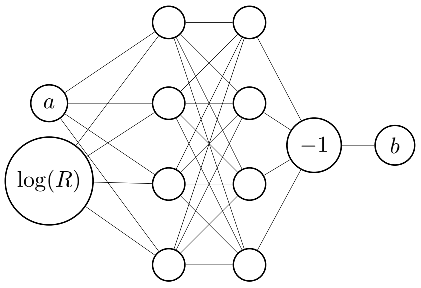

Figure 5: An example "2x4" diode network, with 2 hidden layers, each with 4 fully-connected units.

3.3 Diode Network Pre-Training

While it is expected that the diode network could be trained entirely within the DWDF model, training would be much slower than training the network outside of the DWDF model (due to the overhead introduced by the other WDF elements). With that in mind, each diode network was "pre-trained" against synthetic data, generated using wave domain diode equations derived from the Shockley diode law [15]. The training signal consists of a linear ramp of incident voltage waves ranging from $[-2.5, 2.5]$ V, repeated for exponentially increasing port impedances in the range $[10, 10^9]$ Ω (see Fig. 6).

Diode networks were pre-trained for 2000 epochs, using an Adam optimizer with a starting learning rate of $2e{-}5$. The networks were trained with a combined loss function

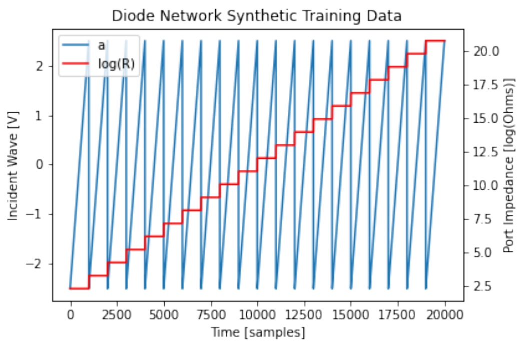

Figure 6: Synthetic data used for pre-training diode networks.

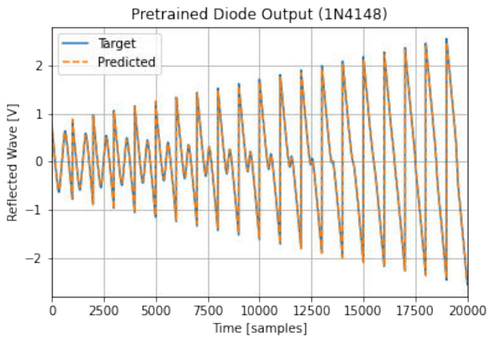

Figure 7: Pre-trained network results for 1-up/1-down 1N4148 diodes.

of mean-squared error, plus normalized error-to-signal ratio (ESR), defined as,

$$L_{\text{ESR}} = \sqrt{\frac{1}{N}\frac{\sum_{i=1}^{N}(y_t(i) - y_p(i))^2}{\sum_{i=1}^{N} y_t(i)^2}} \tag{9}$$

where y_t and y_p represent the "target" and "predicted" signal respectively, and N represents the length of the signal in samples. Then, from Equation (7).

$$L_{\text{TOT}} = L_{\text{MSE}} + L_{\text{ESR}} \tag{10}$$

Fig. 7 shows the results of pre-training a 2x8 network to model a set of 1-up/1-down 1N4148 diodes.

3.4 RC Diode Clipper Training

A DWDF model of the diode clipper circuit was constructed, with the diode set in the WDF model replaced by the neural network architecture shown above.

3.4.1 *Training Hyperparameter Search*

The DWDF models were trained using an Adam optimizer, as described in Equation (6) In order to determine the ideal training parameters for the differentiable diode clipper models, the training process was run for 100 epochs using the 1-up/1-down dataset with a 2x16 network, using a set of different training parameters for each run, as shown in Fig. 10. After completing the hyperparameter search, the set of hyperparameters resulting in the lowest loss values were: $\alpha = 1.0$e-4, $\beta_1 = 0.5$, and $\beta_2 = 0.999$.

3.4.2 *Determining Ideal Network Size*

Next, the training process was run for 500 epochs using the 1-up/1-down dataset with a variety of network sizes, as shown in Table 3. From the results, it can be seen that using a "wider" network with larger hidden layers can improve the network accuracy more so than using a "deeper" network with more hidden layers. It was determined that the 2x16 network size should be used for training future networks, since it was able to achieve the highest accuracy.

Model	Pre-Training	Epoch 0	Epoch 100	Epoch 500
2x4	2.57e-3	3.22e-2	1.15e-2	9.31e-3
2x8	8.55e-4	3.01e-2	7.68e-3	5.96e-3
2x16	1.03e-4	1.14e-2	6.90e-3	4.42e-3
4x4	1.49e-3	2.13e-2	9.67e-3	7.88e-3
4x8	7.11e-4	2.11e-2	8.28e-3	4.92e-3

Table 3: Training results for 1-up/1-down models with different network sizes. "Pre-Training" shows the final loss values after pre-training. The final three columns show the validation loss values after epochs 0, 100, and 500.

3.4.3 *Training Results*

DWDF models of the diode clipper circuit containing 2x16 networks in place of the wave domain diode element were trained for 500 epochs for each diode configuration, using the training hyperparameters determined above. Table 4 shows the results of these training runs. The pre-training loss shown in the second column of Table 4 is the network error after the pre-training step described above. Note that the loss at Epoch 0 (third column) can be interpreted as the error between the ideal diode equations and the measured data. The neural network models were able to improve upon the initial error by more than a factor of two in almost all cases.

Finally, an additional 2x16 model was trained for 2000 epochs for the 1-up/1-down dataset, results in a final validation loss of 3.78e-3. Plots of the training and validation output signals before and after the training run can be seen in Fig. 9. From visual inspection, it appears that the largest discrepancies between the two signals occur during the extreme peaks in the signal. It is expected that adjustments to the loss function used to train the networks could improve the network performance for these parts of the signal.

Config	Pre-Training	Epoch 0	Epoch 100	Epoch 500
1U-1D	1.03e-4	1.14e-2	6.90e-3	4.42e-3
1U-2D	1.70e-4	2.36e-2	7.79e-3	6.12e-3
1U-3D	1.25e-4	2.77e-2	6.30e-3	5.05e-3
2U-2D	1.71e-4	1.38e-2	8.12e-3	7.45e-3
2U-3D	1.01e-4	1.89e-2	8.66e-3	7.04e-3
3U-3D	3.07e-4	1.25e-2	8.25e-3	6.04e-3

Table 4: Training results for 2x16 networks with different diode configurations. "Pre-Training" shows the final loss values after pre-training. The final three columns show the validation loss values after epochs 0, 100, and 500.

Another useful comparison is to examine the transconductance of the neural network diode model compared to the transconductance characteristic predicted by the Shockley diode law [19], shown in Fig. 8. From the asymmetry in the transconductance of the neural model, it is likely that the upward- and downward-facing diodes used in the circuit did not have identical characteristics, as the ideal model assumes. Further, the loss in current at higher voltages indicates that the diodes may have exhibited some internal resistance that is not accounted for by the ideal model.

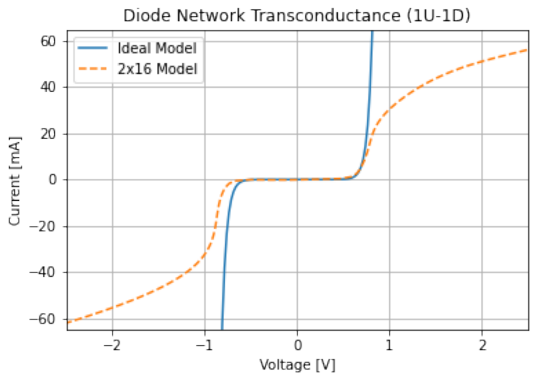

Figure 8: Comparing the transconductance characteristic between the ideal Shockley diode law, and the 2x16 neural network for the 1-up/1-down diode configuration.

4. REAL-TIME CONSIDERATIONS

VA models are often implemented as part of a real-time system that may be used for sound mixing/mastering, or musical performance. With that end in mind, an audio plugin was developed containing an implementation of the 1-up/1-down WDF diode clipper model using several different methods for modelling the wave domain diodes. The first implementation uses the wave domain diode equations presented in [15], along with a high-precision C++ implementation of the Wright Omega function in order to evaluate the Lambert $\mathcal{W}$ function [20]. The second implementation uses the same diode equations, this time using an approximate implementation of the Wright Omega function [16]. The remaining implementations use the trained diode models developed in the previous section, implemented using the RTNeural library for performing neural network inferencing in real-time [21]. Source code for the plugin is available on GitHub [13].

4.1 Performance Comparison

In order to compare the performance of the different diode models, a performance benchmark was developed using the Google Benchmark library. [1] . The benchmark initialises the diode clipper WDF with the given diode model, and processes 100 milliseconds of audio at a sample rate of 96 kHz. This process will repeat until the benchmarks time

[1] `https://github.com/google/benchmark`

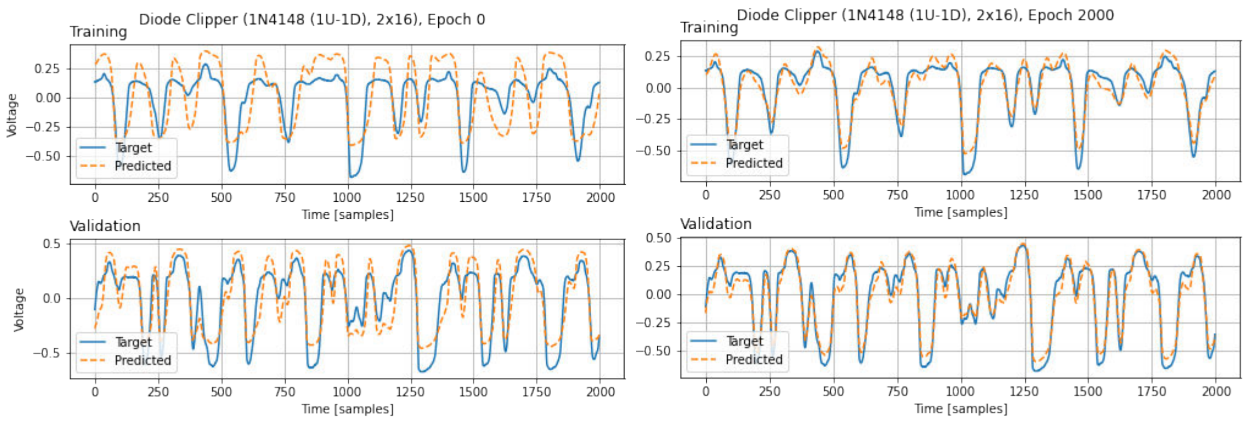

Figure 9: Before and after comparison of the training process for the 2x16 model.

Model	# Iterations	x Ideal
Ideal Model	1816	–
Approx. Model	20779	11.44
2x4 Model	7006	3.86
2x8 Model	4401	2.42
2x16 Model	2302	1.27
4x2 Model	3531	1.94
4x8 Model	2903	1.60

Table 5: Results of the diode clipper performance benchmarks. "Ideal/Approx. Model" refers to the diode model implemented with a high-precision/approximate Wright Omega function. The second column shows how many iterations the benchmark was able to perform within 5 seconds. The third column shows how many times faster each model is when compared to the ideal model.

out after 5 seconds. The number of iterations completed within 5 seconds can then be used as a "score" to compare the run-time performance between the models. The benchmarks were run on a 2018 Mac Mini, with a 3.2 GHz Intel Core i7 CPU. Table 5 shows the results of the performance benchmarks, including the number of iterations completed within 5 seconds, as well as the number of iterations compared against the ideal model score. The results of the benchmark show that the neural network models can out-perform the high-precision implementation, although all the neural network models are clearly out-performed by the approximate model. For the purposes of practical implementations of diode clipper circuit models, the implementer should choose between the speed of using a model based on mathematical approximations of lookup tables relative to the improved accuracy given by the neural network model.

5. CONCLUSION

This paper has outlined the development of a Differentiable Wave Digital Filter library, that can be used to train neural network models of circuit components via gradient descent. The DWDF library has been used to train neural network models of anti-parallel diodes, which may be used in models of audio circuits. From a "white-box" perspective, DWDFs offer the ability to augment wave digital circuit models with data measured from physical circuits. From a "black-box" perspective, DWDFs offer a method to construct neural network circuit models that is modular and utilizes prior knowledge about the circuit. The WDF models constructed with trained neural networks can be implemented for real-time use with comparable performance to a model constructed with traditional WDF elements.

Future research in this area will focus on extending the scope of DWDF models, to include models of circuits with more complex topologies, such as circuits with multi-port elements including tubes, transistors, and op-amps. In particular, training differentiable models of $\mathcal{R}$-type adaptors [3] could offer many possibilities for developing data-driven models of more complicated circuits. In particular, WDF models of circuits containing multi-port nonlinearities often require large multi-dimensional lookup tables or computationally expensive iterative solvers. Replacing these multi-port nonlinearities with neural networks has the potential to improve the real-time performance of these circuit models.

DWDFs also make it possible for neural models of circuit components that were originally trained in one circuit to be used in a WDF model of a completely separate circuit. Exploring this possibility is an interesting topic for future research.

Another potentially interesting line of study is the use of machine learning to generate a WDF topology for an unknown circuit. While the DWDF strategy presented here would be useful for optimizing the generated topologies, the topology generation itself would require an approach that is not based on gradient descent, such as genetic algorithms or heuristics.

Acknowledgments

The authors would like to thank the Center for Computer Research in Music and Acoustics (CCRMA) at Stanford University for providing computing resources necessary for this research. Thanks also to Camille Noufi, Dirk Roosenburg, and Champ Darabundit for useful conversations about neural network training strategies.

6. REFERENCES

[1] F. Germain, "Non-Oversampled Physical Modeling for Virtual Analog Simulations," Ph.D. dissertation, Stanford University, June 2019. [Online]. Available: https://searchworks.stanford.edu/view/13250111

[2] A. Fettweis, "Wave Digital Filters: Theory and Practice," *Proceedings of the IEEE*, vol. 74, no. 2, pp. 270–327, Feb. 1986.

[3] K. J. Werner, "Virtual Analog Modeling of Audio Circuitry Using Wave Digital Filters," Ph.D. dissertation, Stanford Univeristy, June 2016. [Online]. Available: https://searchworks.stanford.edu/view/11891203

[4] M. A. Martínez Ramirez and J. D. Reiss, "Modeling of Nonlinear Audio Effects with End-to-End Deep Neural Networks," *arXiv e-prints*, p. arXiv:1810.06603, Oct. 2018.

[5] E. Damskägg, L. Juvela, and V. Välimäki, "Real-Time Modeling of Audio Distortion Circuits with Deep Learning," in *Proc. of the 16th Sound and Music Computing Conference (SMC-2019)*, May 2019.

[6] A. Wright, E. Damskägg, and V. Välimäki, "Real-Time Black-Box Modelling with Recurrent Neural Networks," in *Proc. of the 22nd Int. Conference on Digital Audio Effects (DAFx-19)*, Sept. 2019.

[7] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=B1x1ma4tDr

[8] B. Kusnetsov, J. D. Parker, and F. Esqueda, "Differentiable IIR Filters for Machine Learning Applications," in *Proc. of the 23rd Int. Conference on Digital Audio Effects (DAFx-20)*, Sept. 2020.

[9] F. Esqueda, B. Kusnetsov, and J. D. Parker, "Differentiable White-Box Virtual Analog Modelling," in *Proc. of the 24th Int. Conference on Digital Audio Effects (DAFx-21)*, Sept. 2021.

[10] D. Roosenburg, E. Stine, R. Michon, and J. Chowdhury, "A Wave-Digital Modeling Library for the Faust Programming Language," in *18th Sound and Music Computing Conference (SMC-2021)*, June 2021.

[11] M. Rest, R. Dunkel, K. J. Werner, and J. Smith, "RT-WDF—A Modular Wave Digital Filter Library with Support for Arbitrary Topologies and Multiple Nonlinearities," in *Proc. of the 19th Int. Conference on Digital Audio Effects (DAFx-16)*, Sept. 2016, p. 287–294.

[12] M. Abadi *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[13] J. Chowdhury and C. J. Clarke, "Differentiable WDFs," https://github.com/jatinchowdhury18/differentiable-wdfs, 2022.

[14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2015.

[15] K. J. Werner, V. Nangia, A. Bernardini, J. Smith, and A. Sarti, "An Improved and Generalized Diode Clipper Model for Wave Digital Filters," *Journal of the Audio Engineering Society*, Oct. 2015.

[16] S. D'Angelo, L. Gabrielli, and L. Turchet, "Fast Approximation of the Lambert W Function for Virtual Analog Modelling," in *Proc. of the 22nd Int. Conference on Digital Audio Effects (DAFx-19)*, Sept. 2019.

[17] C. Kehling, J. Abeßer, C. Dittmar, and G. Schuller, "Automatic Tablature Transcription of Electric Guitar Recordings by Estimation of Score- and Instrument-Related Parameters." in *DAFx*, 2014, pp. 219–226.

[18] J. D. Parker, F. Esqueda, and A. Bergner, "Modelling of nonlinear state-space systems using a deep neural network," in *Proc. of the 22nd Int. Conference on Digital Audio Effects (DAFx-19)*, Sept. 2019.

[19] W. Shockley, "The Theory of P-N Junctions in Semiconductors and P-N Junction Transistors," *The Bell System Technical Journal*, vol. 28, no. 3, pp. 435–489, 1949.

[20] P. W. Lawrence, R. M. Corless, and D. J. Jeffrey, "Algorithm 917: Complex Double-Precision Evaluation of the Wright ω Function," *ACM Trans. Math. Softw.*, vol. 38, no. 3, Apr. 2012. [Online]. Available: https://doi.org/10.1145/2168773.2168779

[21] J. Chowdhury, "RTNeural: Fast Neural Inferencing for Real-Time Systems," 2021. [Online]. Available: https://arxiv.org/pdf/2106.03037.pdf

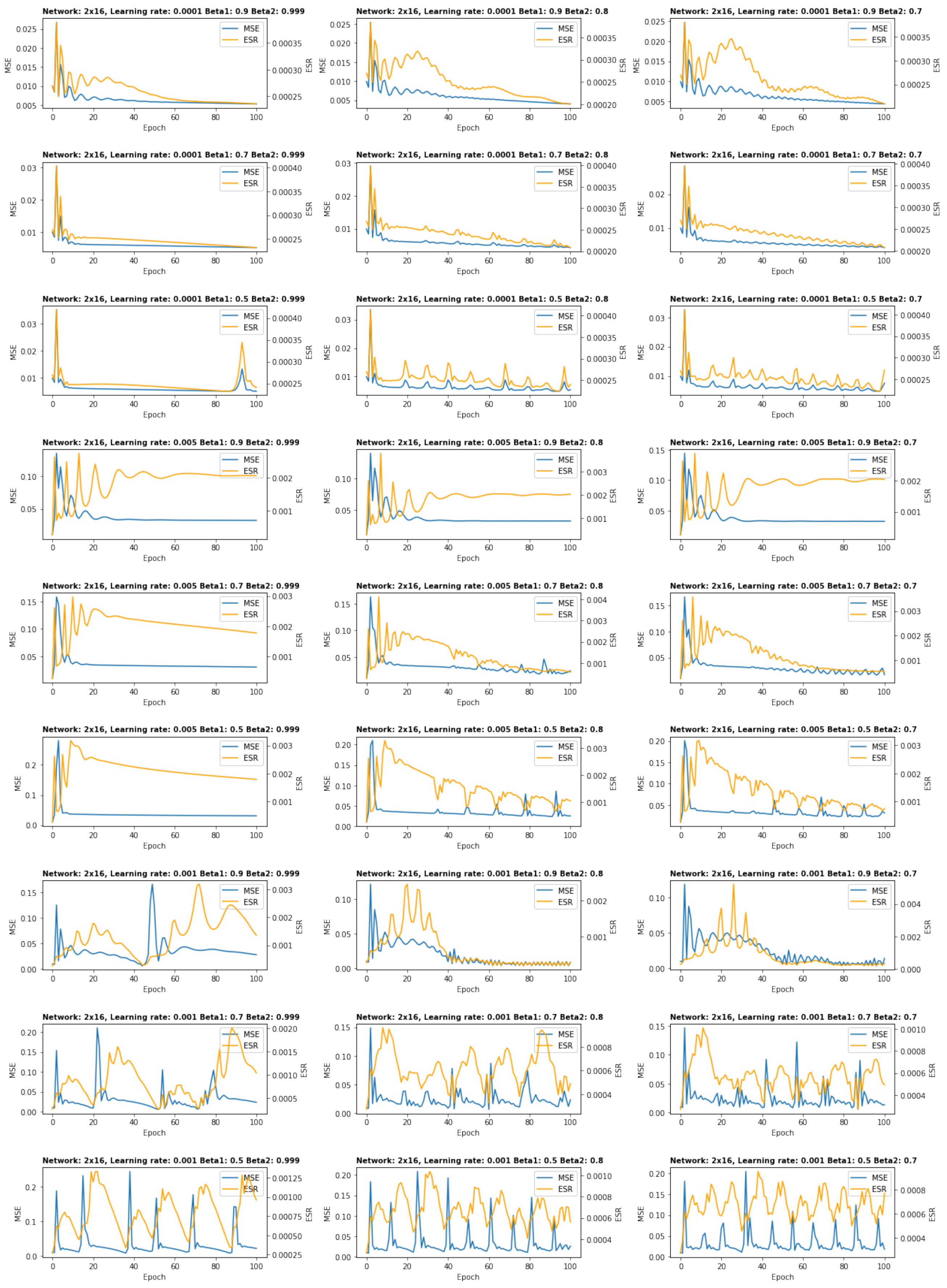

Figure 10: Results of network hyperparameter search.