

Bachelor's Programme in Electrical Engineering

Neural Network Approaches to Guitar Amplifier and Effect Modeling: A Review on Control Parameter Integration Methods

Eetu Kurkinen

© 2025

This work is licensed under a [Creative Commons](https://creativecommons.org/licenses/by-nc-sa/4.0/) “Attribution-NonCommercial-ShareAlike 4.0 International” license.



Author Eetu Kurkinen

Title Neural Network Approaches to Guitar Amplifier and Effect Modeling: A
Review on Control Parameter Integration Methods

Degree programme Electrical Engineering

Major Information Technology

Teacher in charge D.Sc. Pasi Lassila

Advisor D.Sc. Lauri Juvela

Date 27 November 2025

Number of pages 27

Language English

Abstract

Digital modeling of guitar amplifiers presents significant challenges due to the complex non-linear behavior of analog circuits and the need to capture parameter-dependent characteristics. This bachelor's thesis is conducted as a literature review and explores neural network-based modeling approaches for guitar amplifiers and audio effects, with a particular emphasis on methods for integrating controllable parameters, such as gain, volume, and tone controls. Three main conditioning techniques are examined: direct conditioning, Feature-wise Linear Modulation (FiLM), and hyper networks. Direct conditioning concatenates the parameter information with the input signal, FiLM scales and shifts the intermediate network features via an affine transformation, and hyper networks utilize a separate neural network to generate parameter values for the main processing network. Direct conditioning offers simplicity and efficiency but may lack in modeling accuracy, while FiLM and hyper networks provide more expressive control at the cost of increased complexity. Results indicate that recurrent architectures can achieve high parameter efficiency and convincing accuracy with minimal computational resources if implemented correctly. Convolutional approaches can achieve real-time performance through parallel computation and by making modifications to the architecture. The review demonstrates that neural network-based controllable amplifier models can achieve high subjective quality, while addressing the scalability issue of traditional static models. However, challenges remain in achieving simultaneous real-time performance, sufficient accuracy, and data efficiency.

Keywords Amplifier modeling, neural network, control parameter

Tekijä Eetu Kurkinen

Työn nimi Kitaravahvistimien ja -efektien mallinnus neuroverkoilla: katsaus kontrolliparametrien integroimismenetelmiin

Koulutusohjelma Sähkötekniikan kandidaattiohjelma

Pääaine Informaatioteknologia

Vastuopettaja TkT Pasi Lassila

Työn ohjaaja TkT Lauri Juvela

Päivämäärä 27.11.2025

Sivumäärä 27

Kieli englanti

Tiivistelmä

Kitaravahvistimien digitaalinen mallintaminen on keskeinen tutkimusalue musiikkiteknologiassa. Vahvistimien epälineaaristen komponenttien monimutkainen käyttäytyminen ja tämän vaikutukset syntyvään ääneen tekevät niiden tarkasta emuloinnista erittäin haastavaa. Tässä kandidaatintutkielmassa tarkastellaan kirjallisuudessa esitettyjä neuroverkkoihin perustuvia vahvistinmalleja ja erityisesti säädettävien kontrolliparametrien toteuttamistapoja kyseisiin malleihin. Työ rajautuu tarkastelemaan mustan laatikon mallinnusmenetelmiä sekä kolmea erilaista ehdollistamistekniikkaa.

Työ toteutetaan kirjallisuustutkimuksena, jossa vertaillaan erilaisia neuroverkoarkkitehtuureita, kuten takaisinkytkettyjä verkkoja, konvoluutioverkkoja sekä näiden variaatioita. Kontrolliparametriohjauksen toteuttamiseen tarkastellaan kolmea menetelmää: suoraa ehdollistamista, Feature-wise Linear Modulation -tekniikkaa (FiLM) ja hyperverkkoja. Suorassa ehdollistamisessa parametriarvot syötetään neuroverkkoon sisääntulosignaalin yhteydessä, FiLM-tekniikassa moduloidaan verkon aktivaatioita affinikuvauksilla ja hyperverkkoja hyödynnettäessä erillinen neuroverkko tuottaa parametriarvoja pääverkolle.

Tutkimustulosten perusteella neuroverkkopohjaiset parametriojattavat kitaravahvistinmallit voivat saavuttaa korkean subjektiivisen tason. Takaisinkytketyistä verkoista Long Short-Term Memory (LSTM)- ja Gated Recurrent Unit (GRU) -mallit osoittautuivat parametritehokkaiksi, kun taas konvoluutiopohjaiset Temporal Convolutional Networks (TCN) -arkkitehtuurit saavuttivat helpommin reaaliaikaisen suorituksen rinnakkaislaskennan ansiosta. Suoran ehdollistamisen eduiksi osoittautuivat helppo toteutus ja parametritehokkuus, kuitenkin mallinnustarkkuutta rajoittaen. FiLM-tekniikka ja hyperverkot puolestaan mahdollistavat hyvinkin tarkan ehdollistamisen, mutta parametrien lukumäärä voi nousta äkillisesti.

Neuroverkot tarjoavat joustavan ja tehokkaan lähestymistavan kitaravahvistimien mallintamiseen. Parametriohjauksen sisällyttäminen malleihin ratkaisee perinteisen lähestymistavan skaalautuvuusongelman, jossa jokainen säätöyhdistelmä vaatisi erillisen mallin. Parametriojattavat mallit ovat kehittyneitä, mutta haasteena on saavuttaa samanaikaisesti reaaliaikainen suorituskyky, riittävä tarkkuus, sekä tarpeeksi matala mallin opetus aika ja tarvittava opetusdatan määrä. Jatkotutkimuksissa tulisi keskittyä mallien yleistämiskykyyn ja reaaliaikaiseen suorituskykyyn.

Avainsanat Vahvistinmallinnus, neuroverkko, kontrolliparametri

Contents

Abstract	3
Abstract (in Finnish)	4
Contents	6
Operators and abbreviations	7
1 Introduction	8
2 Background	9
2.1 Guitar Amplifiers and Effects	9
2.2 Amplifier Modeling Approaches	9
2.2.1 White-Box Modeling	10
2.2.2 Black-Box Modeling	10
2.2.3 Gray-Box Modeling	11
2.3 Neural Network Architectures for Amplifier Modeling	11
2.3.1 Recurrent Neural Networks	12
2.3.2 Convolutional Neural Networks	13
2.3.3 Network Training and Optimization	15
3 Neural Amplifier Models with Parameter Controls	16
3.1 Direct Conditioning	16
3.2 Feature-wise Linear Modulation	18
3.3 Hyper Networks	21
4 Summary	24
References	25

Operators and abbreviations

Operators

- * Convolution
- \odot Hadamard product
- $\sum_{i=m}^n a_i$ Sum of a_i , from $i = m$ to n

Abbreviations

- CNN Convolutional Neural Network
- ESR Error-to-Signal Ratio
- GRU Gated Recurrent Unit
- FiLM Feature-wise Linear Modulation
- LSTM Long Short-Term Memory
- MAE Mean Absolute Error
- MLP Multilayer Perceptron
- MSE Mean Squared Error
- NN Neural Network
- RNN Recurrent Neural Network
- STFT Short-time Fourier Transform
- TCN Temporal Convolutional Network
- VA Virtual Analog
- WDF Wave Digital Filter

1 Introduction

Digital guitar amplifier modeling aims to recreate the behavior of analog audio circuitry computationally. Traditional analog tube amplifiers have long been favored by musicians for their tonal qualities and subjective warmth [1], but they present practical challenges, including high cost, physical size, and maintenance requirements.

The fundamental challenge in digital amplifier modeling is to capture the complex non-linear behaviors of analog circuitry. These non-linearities are essential to the characteristic sound of tube amplifiers [2].

Recent advances in neural networks have opened new possibilities for black-box modeling approaches, where the amplifier behavior is learned directly from input-output data without requiring detailed circuit knowledge. However, a critical limitation of early black-box models was their inability to incorporate control parameters. Traditional approaches required training of separate models for each combination of control settings, making it impractical to create truly controllable digital amplifiers given the exponential growth of parameter space.

This thesis reviews recent developments in neural network-based amplifier modeling with a specific focus on methods to include control parameters in the models. Three models with three different conditioning techniques are examined: direct conditioning [3], where control parameters are concatenated with the input signal, Feature-wise Linear Modulation (FiLM) [4], which modulates the intermediate activations at each layer via feature-wise affine transformation, and hyper networks [5], where a separate neural network generates weights for the main processing network based on control settings.

The thesis is structured as follows. Section 2 provides background on guitar amplifiers, traditional modeling approaches, and neural network architectures commonly used for audio processing.

2 Background

2.1 Guitar Amplifiers and Effects

The main function of a guitar amplifier is to strengthen the signal from an electric guitar pickup. Traditional analog amplifiers can also shape the tone by non-linear distortion and frequency-based filtering [6].

Amplifier circuits are typically based on vacuum tubes or solid-state electronics. The invention of transistors has replaced vacuum tubes in almost every application, but guitar amplifiers have been one of the few to still rely on tube-based designs [6]. Although they are expensive, bulky, and inefficient, several guitarists still choose to play with a vacuum tube amplifier over a solid-state counterpart [1]. The electric guitar is still largely recorded in a traditional way, with a microphone capturing a speaker cabinet connected to a tube amplifier. Many guitarists perceive tube amplifiers as producing a warmer, more desirable tone compared to solid-state amplifiers, and despite advances in transistor technology, tube amplifiers remain the preferred choice for many professional musicians [1, 2].

In addition to amplifiers, a variety of effect pedals are typically used to shape the tone of an electric guitar. Different effects can be generated by signal processing methods, for example, filtering, modulation, and operations in the frequency or time domain [7]. Some common types of effects are wah-wah, chorus, flanger, pitch-shifter, reverberation, delay and overdrive, among others [8]. Guitarists commonly construct signal chains by arranging multiple effects in series, where each effect's output feeds into the next effect. The resulting tone not only depends on the individual effects and their parameter settings but also critically on the order of arrangement. Through experimentation, musicians develop personalized signal chains with desired sonic characteristics that constitute their signature sound [1, 2, 8].

The physical limitations of analog equipment, such as size, weight, cost, maintenance requirements, and the need for high volume levels to achieve the desired tonal characteristics, have motivated research into digital solutions [2]. A second important factor is the shift from analog to digital audio technologies in both recording and archiving audio [9].

Digital solutions offer advantages beyond cost-effectiveness, including improved workflow, portability, and the ability to generate tones that cannot be achieved with analog circuits alone. However, accurately capturing the complex non-linear behavior, dynamic response, and the subjective warmth of analog circuitry while maintaining real-time computational efficiency remains a significant challenge [2].

2.2 Amplifier Modeling Approaches

Virtual Analog (VA) modeling aims to recreate the behavior of analog audio circuits through digital signal processing [7]. Digital modeling approaches are typically categorized into three paradigms: white-box methods explicitly model individual circuit components and their interactions, black-box methods learn input-output mappings from data, and gray-box methods combine partial circuit knowledge with

data-driven techniques [10].

Each paradigm presents distinct advantages and limitations [11]. White-box methods offer great physical accuracy, but require detailed circuit knowledge and are often computationally too heavy for real-time use [12, 3]. Black-box methods are flexible and do not require circuit information, but depend on extensive training data and may lack interpretability [11]. Gray-box methods balance these concerns, but still require some knowledge about the circuit [13, 14].

While early black-box and gray-box approaches relied on techniques such as Volterra series [15] and Wiener-Hammerstein models [16], the emergence of deep learning has elevated this field.

Neural networks (NN), particularly the recurrent neural network (RNN) and convolutional neural network (CNN) architectures, have become the dominant black-box modeling approach due to their ability to capture complex non-linearities and temporal dependencies without requiring detailed circuit knowledge [12, 17, 18]. Modern neural approaches can achieve modeling accuracy comparable to that of white-box methods while maintaining real-time computational efficiency and requiring no circuit schematics.

2.2.1 White-Box Modeling

White-box modeling, also known as physics-based or circuit-level modeling, recreates amplifier behavior by explicitly modeling individual electronic components and their interactions based on circuit analysis [10]. These approaches use techniques such as SPICE simulation, Wave Digital Filters (WDF) [19], and nodal analysis [20] to represent resistors, capacitors, inductors, tubes, and transistors according to their physical equations [3].

The primary advantage of white-box modeling is the physical accuracy and interpretability. Since the model is based on circuit topology and component values, the parameters have a direct physical meaning, and modifications can be made by adjusting specific components, for example changing a capacitor value in a tone stack or changing tubes [20].

However, white-box methods face significant practical limitations. First, they require complete knowledge of circuit schematics and component values, which may not be available for all equipment. Secondly, the computational cost is often restrictive for real-time applications. SPICE simulations of complex tube amplifiers can take hours to render only seconds of audio, making them unsuitable for live performance or music production [3]. Although simplified white-box models can achieve real-time performance, they often sacrifice the accuracy that motivates the white-box approach in the first place.

2.2.2 Black-Box Modeling

Black-box modeling methods treat the amplifier as an unknown system and aim to learn its input-output behavior from data, without prior knowledge of the circuit structure

and component physics. These methods have gained popularity in guitar amplifier modeling due to their flexibility and the reduced engineering efforts required [18].

Among neural network architectures, recurrent structures such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU) have proven effective [21]. Other approaches include WaveNet-style architectures [17] and modifications of temporal convolutional networks (TCNs) [4].

The main advantage of black-box modeling is the ability to model any amplifier without circuit knowledge or component specifications. In addition, modern neural network frameworks enable relatively straightforward model development.

Nevertheless, black-box methods have some limitations. They typically require a large amount of high-quality training data that covers the whole operating range of an amplifier [3]. An accurate model should replicate the behavior of an amplifier in all control combinations, which causes challenges in modeling and validation. Since the combinatorial space of different control settings grows exponentially with the number of control parameters, it is difficult to cover the entire control space in a reasonable amount of time. As a result, black-box models may suffer from poor generalization when presented signals outside the training distribution.

2.2.3 Gray-Box Modeling

Gray-box modeling approaches combine partial circuit knowledge with data-driven methods [14]. A common gray-box approach is the use of block-oriented models to simulate a distortion system. For example, two RNN blocks can be used to emulate the non-linear preamplifier and power amplifier, while a linear white-box model is responsible for the equalization section of the amplifier [13].

The primary advantage of gray-box modeling is the balance between accuracy, data efficiency, and interpretability. By including structural knowledge, the models typically require less training data and generalize better to unknown conditions than pure black-box models [13].

Gray-box methods face the same challenges as the white- and black-box methods. Circuit topology might not be available for all the equipment, and on the other hand, the models might struggle to simulate the behavior and effects caused by component interactions within the whole system, rather than just the individual stages [13]. It is also crucial to choose the architecture correctly, since an inappropriate one might fail to capture important details regardless of parameter optimization quality.

2.3 Neural Network Architectures for Amplifier Modeling

Neural networks have emerged as powerful tools for guitar amplifier modeling, offering the ability to capture complex non-linear dynamics directly from input-output data. This data driven approach makes them well suited for amplifier modeling, where the interactions between non-linear components and reactive elements would be difficult and computationally heavy to solve analytically [22].

The fundamental principle behind neural network-based modeling is supervised learning. The network is given paired training examples consisting of input audio

signals (clean guitar) and corresponding output signals (processed through the actual target amplifier), and its goal is to minimize the difference between its predictions and the actual output of the amplifier [22].

2.3.1 Recurrent Neural Networks

Recurrent neural networks are designed to process sequential data by maintaining internal hidden states that evolve over time [23]. The presence of recurrent connections is a key feature of RNNs, allowing information to persist across time steps, acting as a form of memory for the network.

A standard Elman RNN processes an input sequence x_t at each time step t by updating its hidden state h_t [24]:

$$h_t = \tanh(W_{ih}x_t + b_{ih} + W_{hh}h_{t-1} + b_{hh}), \quad (1)$$

$$y_t = h_t, \quad (2)$$

where W_{ih} and W_{hh} are weight matrices, b_{ih} and b_{hh} are bias vectors, and \tanh acts as the non-linear activation function. The current hidden state h_t depends both on the current input x_t and the previous hidden state h_{t-1} , creating the recurrent connection.

However, “vanilla” RNNs suffer from the vanishing gradient problem due to backpropagation through time. This limitation motivated the development of more sophisticated RNN architectures with gating mechanisms, such as LSTM [25] and GRU [26].

LSTM networks address the vanishing gradient problem through a cell structure with multiple gates. An LSTM cell at time step t computes the following operations:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}), \quad (3)$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}), \quad (4)$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}), \quad (5)$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}), \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t, \quad (7)$$

$$h_t = o_t \odot \tanh(c_t), \quad (8)$$

where \odot denotes the Hadamard product (element-wise product). The input gate i_t determines what new information should be added to the cell state, the forget gate f_t determines what information should be discarded from the previous cell state, the candidate cell g_t creates new candidate information that could be added to the cell state, c_t updates the cell state, and the output gate o_t determines what parts of the cell state should be output as the hidden state h_t .

GRUs offer an alternative to LSTMs by simplifying the gating mechanism. The GRU architecture merges the forget and input gates of LSTM into a single update gate and eliminates the separate cell state [26] and computes the following operations at time step t :

$$r_t = \sigma(W_{ir}x_t + b_{ir} + W_{hr}h_{t-1} + b_{hr}), \quad (9)$$

$$n_t = \tanh(W_{in}x_t + b_{in} + r_t \odot (W_{hn}h_{t-1} + b_{hn})), \quad (10)$$

$$z_t = \sigma(W_{iz}x_t + b_{iz} + W_{hz}h_{t-1} + b_{hz}), \quad (11)$$

$$h_t = (1 - z_t) \odot n_t + z_t \odot h_{t-1}, \quad (12)$$

where the reset gate r_t determines how much of the previous hidden state should be forgotten, the candidate hidden state, n_t , represents the new candidate information that could be added to the hidden state, the update gate z_t determines how much of the previous hidden state information to keep versus how much of the new candidate to use, and h_t is the final hidden state update.

2.3.2 Convolutional Neural Networks

Convolutional neural networks have traditionally been used for image classification [27], but have been successfully adapted for sequential data using 1D convolutions along the time axis [28]. A standard 1D convolution applies a learnable filter (kernel) across the input sequence x :

$$y_t = \sum_{k=0}^{K-1} w_k \cdot x_{t-k} + b, \quad (13)$$

where K is the kernel size, w_k are the filter weights, and b is a bias term. For real-time amplifier modeling, causality must be strictly maintained, since the output can only depend on current and previous inputs, never the future ones. Dilated convolutions are also often used to grow the receptive field of a network, allowing it to capture long-range dependencies. A dilated convolution with a dilation factor d is defined as:

$$y_t = \sum_{k=0}^{K-1} w_k \cdot x_{t-k \cdot d} + b. \quad (14)$$

Thus, dilation introduces fixed gaps between the consecutive elements of an input [28]. When $d = 1$, the dilated convolution reduces to a regular convolution (Eq. 13).

One of the most widely used CNN-based architectures for amplifier modeling is WaveNet [29] and its variants [12, 17, 18]. The WaveNet architecture utilizes layers of dilated causal convolutions, illustrated in Fig. 1, allowing the network have a large receptive field [29].

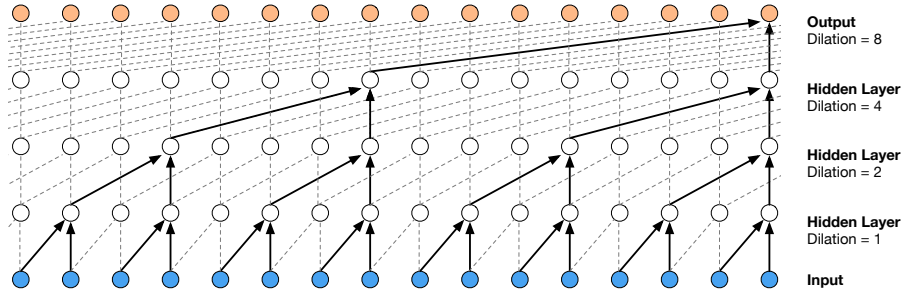


Figure 1: A stack of dilated convolution layers [29, Fig. 3]

For activation, WaveNet uses gated units that compute the element-wise product of a tanh activation and a sigmoid activation:

$$z = \tanh(W_{f,k} * x) \odot \sigma(W_{g,k} * x), \quad (15)$$

where $*$ denotes convolution, k is the layer index, f and g denote the filter and gate, respectively, and W is a learnable convolution filter [29]. The architecture consists of layers of residual blocks with both residual and skip connections, illustrated in Fig. 2.

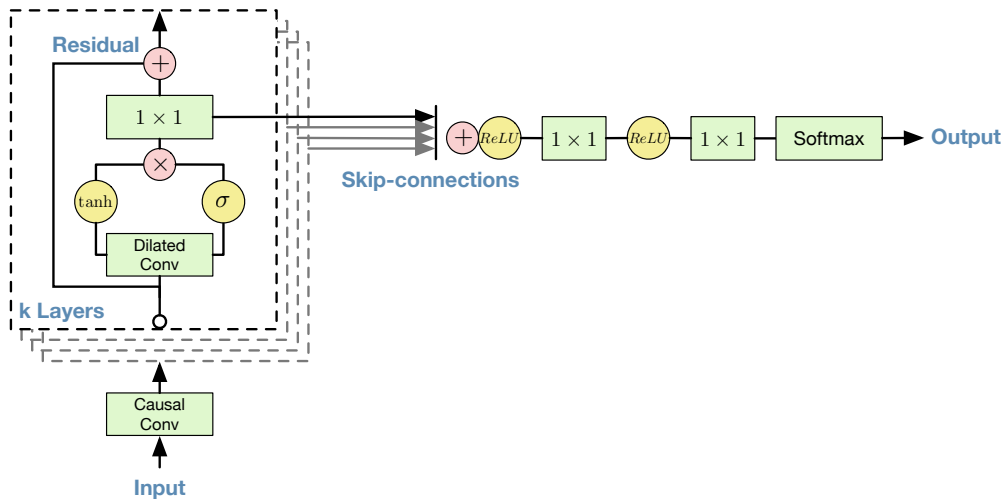


Figure 2: Overview of the WaveNet architecture [29, Fig. 4]

The term “Temporal Convolutional Networks”, TCNs, can be used to describe a family of architectures that use causal convolutions and output a sequence of the same length as their input [28]. The main advantages of TCNs compared to RNNs are parallelism, stable gradients, and the flexibility to modify the receptive field size with dilation. However, TCNs may perform poorly if the parameters are not chosen correctly for the domain [28].

2.3.3 Network Training and Optimization

The fundamental goal of any neural network model is to predict an output as close to the actual output as possible. An optimization algorithm aims to speed up the training process of a model by finding the optimal parameters [30]. It is also necessary to define a loss function that evaluates the performance of the model. In practice, the optimizer learns the parameters that minimize the average loss over a training set. The objective is to develop a model that can successfully generalize to inputs outside the training data.

One of the most popular optimization algorithms is the gradient descent and its multiple variants. The vanilla gradient descent calculates the gradient of the loss function with respect to the parameters and updates them according to the learning rate and the gradient direction [31]:

$$\theta_{k+1} = \theta_k - \eta \cdot \nabla \mathcal{L}(\theta_k), \quad (16)$$

where θ_{k+1} is the updated parameter vector at the $(k + 1)^{th}$ iteration, θ_k is the current parameter vector at the k th iteration, η is the learning rate, and $\nabla \mathcal{L}(\theta_k)$ is the loss function gradient \mathcal{L} with respect to the parameters θ_k .

As for the loss function, a common starting point could be the mean-squared error (MSE):

$$\mathcal{L}_{MSE}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{NT} \sum_{n=1}^N \sum_{t=1}^T (\mathbf{y}_t^{(n)} - \hat{\mathbf{y}}_t^{(n)})^2, \quad (17)$$

where N is the number of training examples, T is the sequence length, \mathbf{y}_t is the target value, and $\hat{\mathbf{y}}_t$ is the predicted value. The MSE can be extended to the error-to-signal ratio (ESR) by normalizing it by the target signal energy

$$\mathcal{L}_{ESR}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\sum_{n=1}^N \sum_{t=1}^T (\mathbf{y}_t^{(n)} - \hat{\mathbf{y}}_t^{(n)})^2}{\sum_{n=1}^N \sum_{t=1}^T (\mathbf{y}_t^{(n)})^2}. \quad (18)$$

In the context of amplifier modeling, it may be beneficial to apply a pre-emphasis filter to the network output and target signal before the calculation of loss [32]. This is due to the fact that the human ear perceives different frequencies at different loudness levels.

3 Neural Amplifier Models with Parameter Controls

A key challenge in neural amplifier modeling is the ability to control the behavior of the model through parameter settings such as gain, volume, and tone controls. Traditional black-box models typically learn a static mapping for a specific configuration of amplifier parameter settings. This approach is extremely impractical when designing a model with parametric controls, since a typical guitar amplifier may have five or even more control knobs, creating an extensive parameter space that would require thousands of individual models to cover sufficiently.

To address this limitation, researchers have studied different ways to enable a single model to emulate an amplifier over its entire range of control settings. Several studies have been conducted experimenting with different neural network architectures and ways of conditioning them with parameter controls.

3.1 Direct Conditioning

Direct conditioning represents the most straightforward approach to introduce parametric controls to neural guitar amplifier models. In this method, control parameter settings are concatenated directly with the audio input signal.

Juvela et al. [3] proposed a model, illustrated in Fig. 3, that receives parameter setting values \mathbf{c} normalized to the range $[0, 1]$ as conditioning, and maps an input signal \mathbf{x} to match a target output \mathbf{y} . A loss function \mathcal{L} scores the similarity of the model output $\hat{\mathbf{y}}$ and the target output \mathbf{y} , and adjusts the network parameters θ accordingly. In practice, a training dataset \mathcal{D} is formed consisting of N training examples:

$$\mathcal{D} = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}, \mathbf{c}^{(1)}), \dots, (\mathbf{x}^{(i)}, \mathbf{y}^{(i)}, \mathbf{c}^{(i)})\}_{i=1}^N. \quad (19)$$

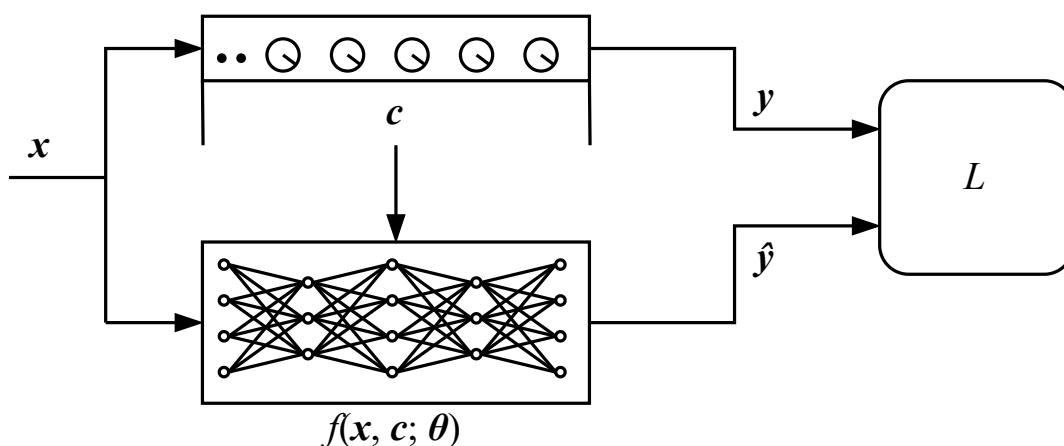


Figure 3: The neural amplifier model training scheme [3, Fig. 1].

The study [3] modeled the Matchless DC-30TM amplifier, which has five control potentiometers for volume, treble, bass, tone cut, and master volume. Each training example consisted of a pair of second long input-target audio segments, drawn randomly

from a collection of guitar and bass recordings, and sampled at 48 kHz. Furthermore, the dataset was randomly split into 15000 training and 1000 validation examples.

For the neural network architecture, a single layer LSTM [25] model with 32 cells was used. This model was trained for 1M iterations using the ESR loss (Eq. 18), and the Adam optimizer [33]. A white-box SPICE model was also developed for listening tests and comparison.

The model [3] was evaluated by conducting a remote Difference Mean Opinion Score (DMOS) listening test. The test comprised five evaluation cases, each of which was rated by 30 experts. A single case consisted of a reference and a test sample, and the listeners were asked to rate how well the test sample resembled the reference on a scale from 1 (bad) to 5 (excellent). Ratings were collected for the LSTM model, the SPICE model, a hidden reference, and a low anchor, which was produced by applying a 3.5 kHz low-pass filter on the reference samples. As seen in Fig. 4, both the SPICE model and the LSTM neural network model achieved very high subjective quality with no significant difference between them.

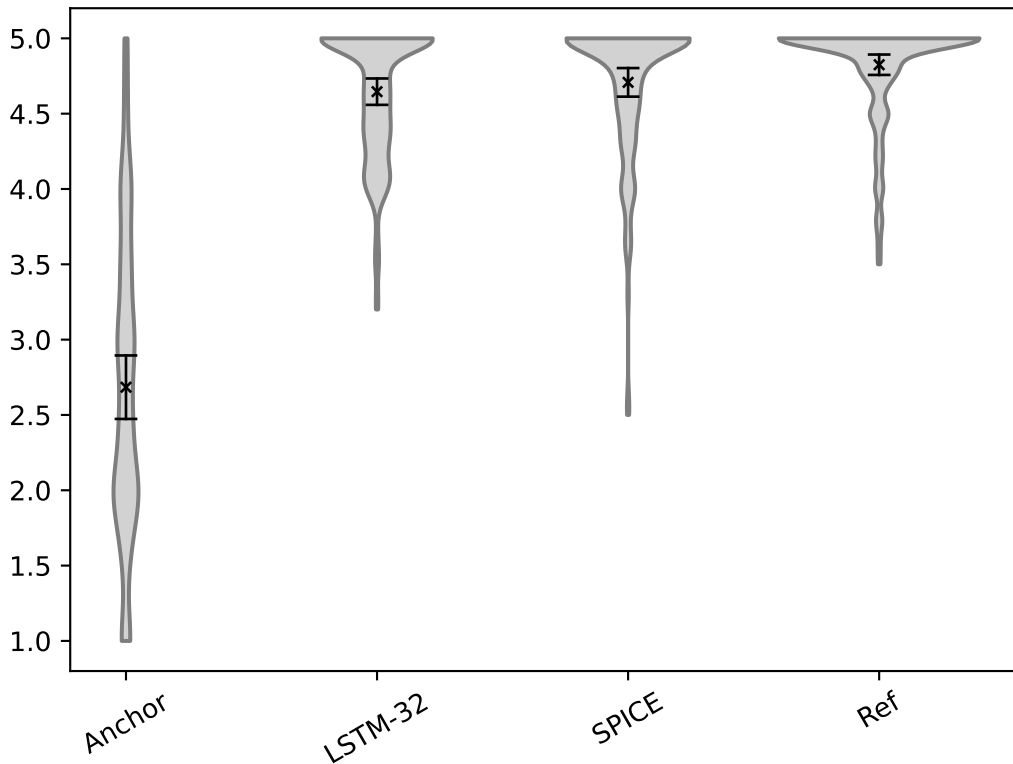


Figure 4: The listening test DMOS rating distributions [3, Fig. 6].

In addition to the proposed amplifier model [3], the authors introduced a robot, illustrated in Fig. 5, for automated data collection. The robot uses electric motors attached to each control knob of the physical amplifier, enabling precise positioning at any desired combination of settings. Moreover, the robot is integrated with an audio interface that plays input samples and records the output, while keeping track of the

knob positions. To address the exponential scaling problem inherent in controllable modeling, each control position is randomly sampled from a uniform distribution, resulting in unbiased coverage of the control space while keeping the dataset size manageable.

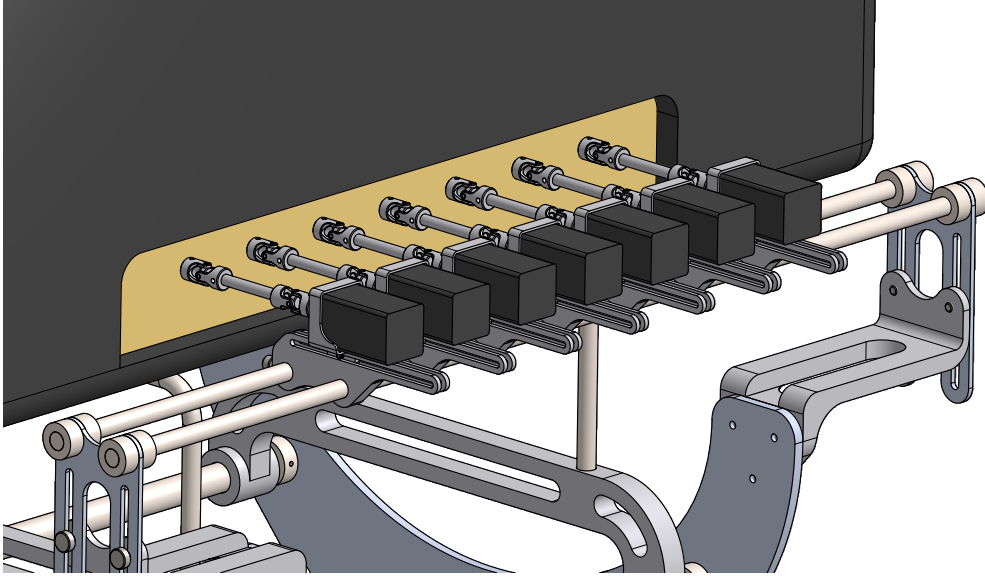


Figure 5: Data collection robot attached to an amplifier [3, Fig. 2].

3.2 Feature-wise Linear Modulation

Feature-wise Linear Modulation (FiLM) provides a method to adaptively influence the output of a neural network by applying affine transformations to the intermediate features of a network, based on some input [34]. The scaling and bias parameters $\gamma_{i,c}$ and $\beta_{i,c}$ are produced by the learned functions $f(\mathbf{x}_i)$ and $h(\mathbf{x}_i)$ that map the conditioning input to the layer-specific modulation coefficients. More precisely, $\gamma_{i,c}$ and $\beta_{i,c}$ modulate the network’s activations $\mathbf{F}_{i,c}$ by an affine transformation:

$$\text{FiLM}(\mathbf{F}_{i,c}|\gamma_{i,c},\beta_{i,c}) = \gamma_{i,c}\mathbf{F}_{i,c} + \beta_{i,c}, \quad (20)$$

where the subscripts refer to the i^{th} input’s c^{th} feature or feature-map.

Steinmetz and Reiss [4] applied FiLM-conditioned TCNs to model the analog LA-2A dynamic range compressor. Their architecture, illustrated in Fig. 6, consists of residual blocks containing one-dimensional dilated convolutions, followed by batch normalization, conditional FiLM, and a parametric rectified linear unit (PReLU) nonlinearity. Their conditioning pathway uses a multilayer perceptron (MLP) to project the device control parameters $\phi \in \mathbb{R}^P$ into an embedding $z \in \mathbb{R}^d$. A linear layer at each TCN block transforms this embedding to produce $2C_n$ values, representing the scaling and bias parameters for all C_n channels at layer n .

The receptive field of a TCN with N layers follows the recursion

$$r_n = r_{n-1} + (K - 1) \cdot d_n, \quad (21)$$

where K is the kernel size and d_n is the dilation factor at layer n . Typical TCN designs employ base-2 dilation growth $d_n = 2^n$ [29], requiring $N = 10$ layers with kernel size $K = 13$ to achieve a receptive field of approximately 300 ms at a sampling rate of 44.1 kHz. Steinmetz and Reiss [4] proposed accelerated dilation growth $d_n = 10^n$, achieving a comparable receptive field with only $N = 4$ layers. This reduction in network size improves computational efficiency since, while computations across the temporal dimensions can be parallelized, computations through the depth of the network remain sequential.

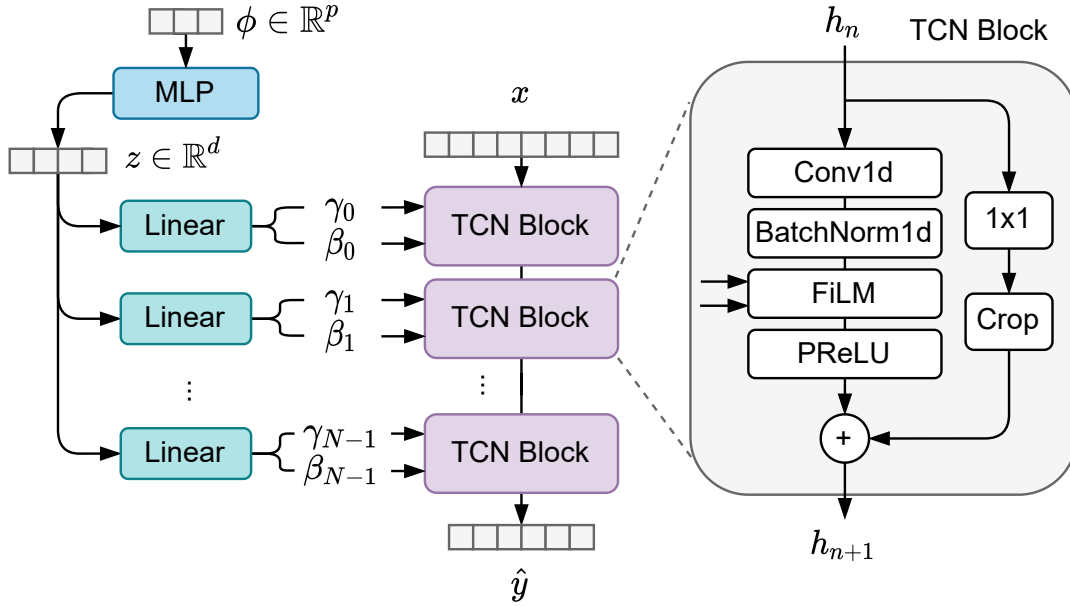


Figure 6: Series of TCN blocks along with a conditioning module (MLP) [4, Fig. 1].

The study [4] compared multiple architectures: a single-layer LSTM model with 32 hidden layers (LSTM-32) [21], a noncausal TCN with base-2 dilation growth and a receptive field size of 324 ms at $f_s = 44.1$ kHz (TCN-324-N) [35], and variants of the proposed base-10 dilation growth TCN with 32 channels in both causal and noncausal configurations with receptive field sizes of 101 ms (TCN-100-N/C), 302 ms (TCN-300-N/C), and 1008 ms (TCN-1000-N/C). The suffixes in the model names indicate causality (-C) and noncausality (-N).

The models were trained using the SignalTrain dataset [36], which contains approximately 20 hours of input-output recordings at $f_s = 44.1$ kHz. The LA-2A dynamic range compressor has a binary switch for compress/limit mode selection, and a continuous peak reduction parameter. The dataset covers 40 different parameter configurations. The training combined loss functions in both time and frequency domains. The mean absolute error (MAE) was computed for the time domain component $\mathcal{L}_{\text{time}}$, while the multi-resolution short-time Fourier Transform (STFT) error [37] was used for the frequency domain component $\mathcal{L}_{\text{freq}}$. The overall loss was given as the sum of these components $\mathcal{L}_{\text{overall}} = \mathcal{L}_{\text{time}} + \alpha \cdot \mathcal{L}_{\text{freq}}$, where $\alpha = 1$ was

used in every experiment.

The objective evaluation suggested that the causal TCNs were able to achieve performance comparable to their noncausal variants. Among TCNs, the models with the receptive field size around 300 ms achieved the best performance. The efficient causal model TCN-300-C with 51k parameters achieved real-time operation on CPU with a real-time factor $RT = 2.2\times$ at a frame size of 2048 samples, compared to $RT = 0.5\times$ of the baseline noncausal model TCN-324-N with 162k parameters. The real time factor (RT) was defined as:

$$RT := \frac{S}{T f_s}, \quad (22)$$

where S is the number of samples processed at a sample rate of f_s , and T is the time in seconds to process those samples. Additionally, TCN-300-C achieved a MAE of 1.44×10^{-2} and a multi-resolution STFT error of 0.603, compared to 1.70×10^{-2} and 0.587 of the baseline model TCN-324-N. The objective evaluation also showed that the LSTM-32 model performed particularly well, while using $32\times$ less parameters than the TCN-324-N model. However, while parametrically efficient, the LSTM-32 model could not achieve real-time performance and required over 8 times longer to train (108 hours) compared to the TCN-300-C model (13 hours). It is important to note that the models were implemented with PyTorch, and optimized C++ versions may achieve speedups in computation.

The performance of the model was further evaluated with an online listening test using five different stimuli. The stimuli were processed using the baseline model [36] of the SignalTrain dataset, the LSTM-32 model, and the proposed TCN-300-C model trained with only 1% of the dataset. Both the LSTM-32 model and the TCN-300-C model performed slightly below the reference, while participants noticed noise-like artifacts in the SignalTrain model. As seen in Fig. 7, both the LSTM-32 model and the TCN-300-N model captured the LA-2A characteristics well, but differed in cases of strong gain reduction.

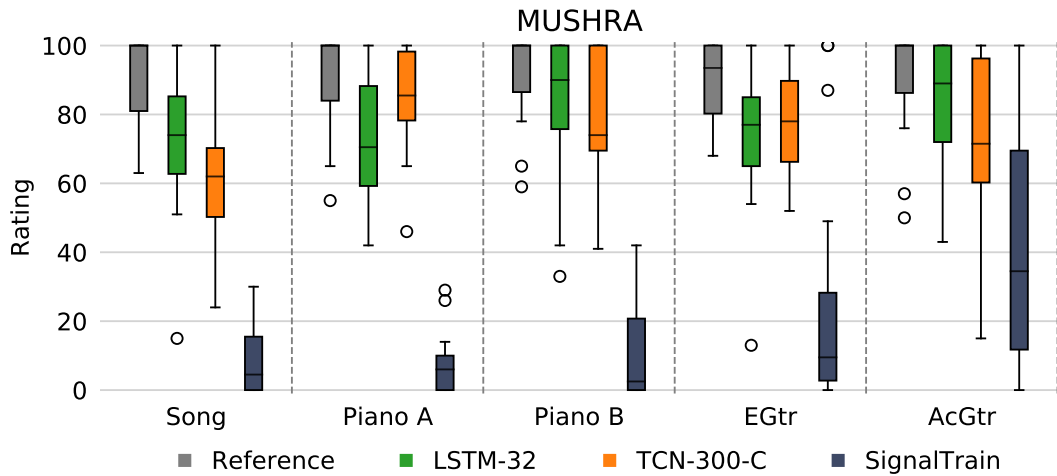


Figure 7: Model ratings of the listening test with 18 participants [4, Fig. 4].

3.3 Hyper Networks

Hyper networks offer an alternative conditioning approach to neural amplifier modeling, where a separate neural network generates the weights of the main processing network based on control parameters. This may potentially enable more expressive parameter-dependent behavior compared to direct conditioning.

Yeh et al. [5] proposed a hyper network-based conditioning method for RNNs in VA modeling. Their approach addresses the limitations of the direct conditioning method. Direct conditioning is parameter efficient and easy to implement, but it may fail to model complicated input/output relationships.

The study [5] introduced two hyper network variants: StaticHyper-RNN and DynamicHyper-RNN. In StaticHyper-RNN, illustrated in Fig. 8, the hyper network generates fixed weight matrices W based on the conditioning input. The weights are generated once, and kept fixed across the entire sequence. More formally, considering a standard RNN (Eq. 1) with shared bias, and given the conditioning vector ϕ , the hyper network attempts to learn the functions f_{ih} , f_{hh} , and f_b to generate the weight matrices W_{ih} , W_{hh} , and the bias vector b :

$$f_{ih}(\phi) = W_{ih}, \quad f_{hh}(\phi) = W_{hh}, \quad f_b(\phi) = b. \quad (23)$$

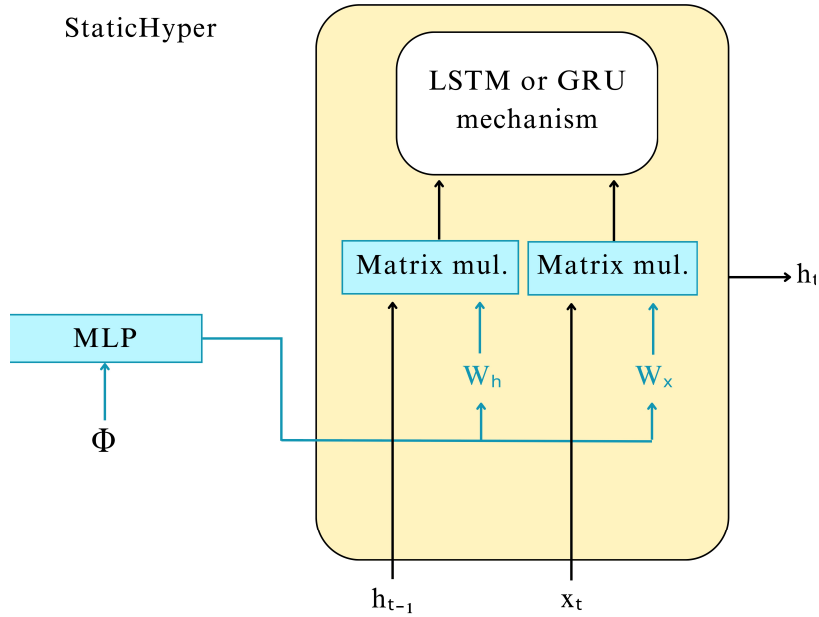


Figure 8: StaticHyper-RNN architecture. The MLP aims to generate the weight matrices according to conditioning Φ [5, Fig. 2].

DynamicHyper-RNN, illustrated in Fig. 9, extends this concept by generating weights at each time step using another RNN. The hyper network, denoted hyperRNN, receives a concatenation of the conditioning vector ϕ and the previous hidden state h_{t-1} of the main network, mainRNN, as its input x_t^p :

$$x_t^p = \begin{pmatrix} h_{t-1}^m \\ \phi \end{pmatrix}, \quad (24)$$

where the superscripts m and p refer to mainRNN and hyperRNN, respectively. The hyperRNN then processes this input:

$$h_t^p = \tanh(W_{hh}^p h_{t-1}^p + W_{ih}^p x_t^p). \quad (25)$$

The feature vectors z_h and z_x are extracted from h_t^p through learned transformation functions:

$$\mathbf{f}_h(h_t^p) = z_h, \quad \mathbf{f}_x(h_t^p) = z_x. \quad (26)$$

The mainRNN uses these feature vectors to modulate its weight matrices W_{hh}^m and W_{ih}^m :

$$h_t^m = \tanh(d_h(z_h) \odot W_{hh}^m h_{t-1}^m + d_z(z_x) \odot W_{ih}^m x_t^m), \quad (27)$$

where d_h and d_z are additional learnable transformation functions.

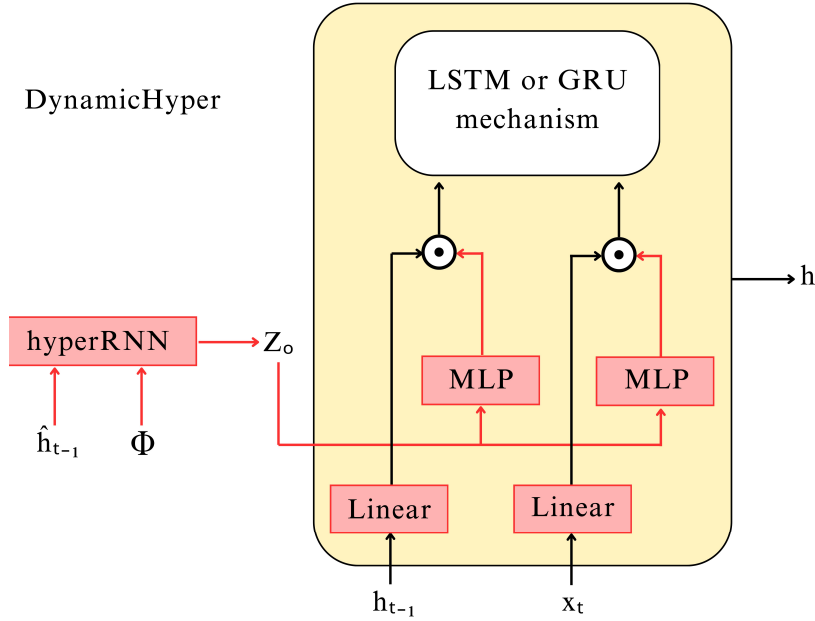


Figure 9: The DynamicHyper-RNN architecture [5, Fig. 3]

In addition to the hyper network variants, a FiLM-based RNN was proposed. These three models were compared with various models using various architectures and conditioning mechanisms. For CNN-based models, micro-TCN [4] and GCN [38] were chosen as baselines, using both direct and FiLM conditioning methods. The chosen RNN-based baselines, LSTM and GRU [21], only use direct conditioning.

The models were evaluated using two different datasets: the SignalTrain dataset [36] for the LA-2A dynamic range compressor and a newly collected dataset for

the Boss OD-3 overdrive pedal [5]. The Boss OD-3 has three continuous control parameters for level, tone, and gain. The level control was not considered dynamic in the dataset, but the tone and gain controls were segmented into five equal intervals, providing control values ranging from 0 to 4.

The results demonstrated that all proposed hyper network-based conditioning methods outperformed the direct conditioning method. The DynamicHyper-RNN achieved the best objective score, though at higher computational cost compared to FiLM-RNN and StaticHyper-RNN. In terms of architectures, the GRU models surpass the LSTM-based models in both modeling cases. Among CNN-based models, GCN outperforms TCN for the Boss OD-3 overdrive pedal, while TCN achieves superior results for the LA-2A compressor.

4 Summary

This thesis has reviewed neural network-based approaches to guitar amplifier modeling with controllable parameters, examining three distinct conditioning methods: direct conditioning, Feature-wise Linear Modulation (FiLM), and hyper networks. The review demonstrates that black-box neural network models can achieve high-quality emulation results, while addressing the scalability issue of traditional static models.

Among the neural network architectures examined, the recurrent networks LSTM [3] and GRU [5] have proven to be highly parameter efficient. The gating mechanisms in these architectures enable them to capture long-term temporal dependencies, which is essential to model amplifier dynamics and memory effects. However, these models may have difficulties performing real-time, and the LSTM-models particularly require significantly longer training times compared to convolutional alternatives.

Convolutional approaches, especially TCNs, offer advantages through parallel computation capabilities. The use of dilated convolutions allows TCNs to achieve large receptive fields with relatively shallow networks, enabling real-time performance on standard consumer-grade hardware [4].

Regarding conditioning methods, direct conditioning offers the simplest implementation and parameter efficiency, but may lack in modeling accuracy. FiLM and hyper networks can greatly improve the model's expressivity, but at a cost of increased complexity and computational demand [5].

The review highlights trade-offs between methods and architectures. It is crucial to choose the architecture and conditioning method for the target device, based on its features. The available data, hardware, and the demand for real-time performance have to be taken into consideration as well.

References

- [1] J.-P. Herbst, “Empirical explorations of guitar players’ attitudes towards their equipment and the role of distortion in rock music,” *Current Musicology*, no. 105, 2020.
- [2] J.-P. Herbst and A. P. Vallejo, *Rock guitar virtuosos : advances in electric guitar playing, technology, and culture*. Cambridge elements. Elements in popular music, Cambridge: Cambridge University Press, 1st ed., 2023.
- [3] L. Juvela, E.-P. Damskäg, A. Peussa, J. Mäkinen, T. W. Sherson, S. I. Mimilakis, K. Rauhanen, and A. Gotsopoulos, “End-to-end amp modeling: from data to controllable guitar amplifier models,” *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023.
- [4] C. J. Steinmetz and J. D. Reiss, “Efficient neural networks for real-time modeling of analog dynamic range compression,” *152nd Convention of the Audio Engineering Society*, May 2022.
- [5] Y.-T. Yeh, W.-Y. Hsiao, and Y.-H. Yang, “Hyper recurrent neural network: Condition mechanisms for black-box audio effect modeling,” in *Proceedings of the 27th International Conference on Digital Audio Effects (DAFx)*, (Guildford, UK), September 2024.
- [6] D. J. Dailey, *Electronics for guitarists*. New York: Springer, 2nd ed., 2013.
- [7] U. Zölzer, *DAFX : digital audio effects*. Chichester, West Sussex: Wiley, 2nd ed., 2011.
- [8] J.-M. Reveillac, *Musical sound effects : analog and digital sound processing*. Waves Series, London, England ;: ISTE, 1st ed., 2018.
- [9] P. Theberge, *Any sound you can imagine : making music, consuming technology*. Hanover: Wesleyan U.P, 1997.
- [10] J. Pakarinen and D. T. Yeh, “A review of digital techniques for modeling vacuum-tube guitar amplifiers,” *Computer Music Journal*, vol. 33, no. 2, pp. 85–100, 2009.
- [11] Y.-T. Yeh, Y.-H. Chen, Y.-C. Cheng, J.-T. Wu, J.-J. Fu, Y.-F. Yeh, and Y.-H. Yang, “DDSP guitar amp: Interpretable guitar amplifier modeling,” in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025.
- [12] A. Wright, E.-P. Damskäg, L. Juvela, and V. Välimäki, “Real-time guitar amplifier emulation with deep learning,” *Applied Sciences*, vol. 10, no. 3, 2020.

- [13] S. Miklanek, A. Wright, V. Välimäki, and J. Schimmel, “Neural grey-box guitar amplifier modelling with limited data,” *International Conference on Digital Audio Effects (DAFx23)*, pp. 151 – 158, September 2023.
- [14] F. Eichas and U. Zölzer, “Gray-box modeling of guitar amplifiers,” *Journal of the Audio Engineering Society*, vol. 66, pp. 1006–1015, December 2018.
- [15] S. Orcioni, A. Terenzi, S. Cecchi, F. Piazza, and A. Carini, “Identification of Volterra models of tube audio devices using multiple-variance method,” *Journal of the Audio Engineering Society*, vol. 66, pp. 823–838, October 2018.
- [16] F. Eichas and U. Zölzer, “Virtual analog modeling of guitar amplifiers with Wiener-Hammerstein models,” in *Proceedings of the 44th Annual Convention on Acoustics (DAGA 2018)*, (Munich, Germany), March 2018.
- [17] E.-P. Damskäg, L. Juvela, E. Thuillier, and V. Välimäki, “Deep learning for tube amplifier emulation,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 471–475, 2019.
- [18] 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, 2020.
- [19] M. Karjalainen and J. Pakarinen, “Wave digital simulation of a vacuum-tube amplifier,” in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5, pp. V–V, 2006.
- [20] D. T. Yeh, J. S. Abel, and J. O. Smith, “Automated physical modeling of nonlinear audio circuits for real-time audio effects—part i: Theoretical development,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 728–737, 2010.
- [21] A. Wright, E.-P. Damskäg, and V. Välimäki, “Real-time black-box modelling with recurrent neural networks,” *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx-19)*, September 2019.
- [22] Z. Zhang, E. Olbrych, J. Bruchalski, T. J. McCormick, and D. L. Livingston, “A vacuum-tube guitar amplifier model using long/short-term memory networks,” in *SoutheastCon 2018*, pp. 1–5, 2018.
- [23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [24] J. L. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.
- [25] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

- [26] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder–decoder approaches,” in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (D. Wu, M. Carpuat, X. Carreras, and E. M. Vecchi, eds.), (Doha, Qatar), pp. 103–111, Association for Computational Linguistics, Oct. 2014.
- [27] L. Massaron, *Deep learning. For dummies*, Hoboken, N.J: J. Wiley, 1st edition ed., 2019.
- [28] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” 2018.
- [29] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “Wavenet: A generative model for raw audio,” 2016.
- [30] F. Mehmood, S. Ahmad, and T. K. Whangbo, “An efficient optimization technique for training deep neural networks,” *Mathematics*, vol. 11, no. 6, 2023.
- [31] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.04747, 2016.
- [32] A. Wright and V. Välimäki, “Perceptual loss function for neural modeling of audio systems,” *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 251–255, 2019.
- [33] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014.
- [34] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, “Film: Visual reasoning with a general conditioning layer,” 2017.
- [35] C. Steinmetz, J. Pons, S. Pascual, and J. Serrà, “Automatic multitrack mixing with a differentiable mixing console of neural audio effects,” *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 71–75, 2020.
- [36] S. H. Hawley, B. Colburn, and S. I. Mimilakis, “SignalTrain: Profiling audio compressors with deep neural networks,” *ArXiv*, vol. abs/1905.11928, 2019.
- [37] R. Yamamoto, E. Song, and J.-M. Kim, “Probability density distillation with generative adversarial networks for high-quality parallel waveform generation,” in *Interspeech*, 2019.
- [38] M. Comunità, C. Steinmetz, H. Phan, and J. D. Reiss, “Modelling black-box audio effects with time-varying feature modulation,” *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2022.