AUTOMATIC EQUALIZATION FOR INDIVIDUAL INSTRUMENT TRACKS USING CONVOLUTIONAL NEURAL NETWORKS

Florian Mockenhaupt[1], *Joscha Simon Rieber*[1], *and Shahan Nercessian*[2]

Native Instruments
[1]Langenfeld, Germany [2]Boston, MA, USA
{florian.mockenhaupt|joscha.rieber|shahan.nercessian}@native-instruments.com

ABSTRACT

We propose a novel approach for the automatic equalization of individual musical instrument tracks. Our method begins by identifying the instrument present within a source recording in order to choose its corresponding ideal spectrum as a target. Next, the spectral difference between the recording and the target is calculated, and accordingly, an equalizer matching model is used to predict settings for a parametric equalizer. To this end, we build upon a differentiable parametric equalizer matching neural network, demonstrating improvements relative to previously established state-of-the-art. Unlike past approaches, we show how our system naturally allows real-world audio data to be leveraged during the training of our matching model, effectively generating suitably produced training targets in an automated manner mirroring conditions at inference time. Consequently, we illustrate how fine-tuning our matching model on such examples considerably improves parametric equalizer matching performance in real-world scenarios, decreasing mean absolute error by 24% relative to methods relying solely on random parameter sampling techniques as a self-supervised learning strategy. We perform listening tests, and demonstrate that our proposed automatic equalization solution subjectively enhances the tonal characteristics for recordings of common instrument types.

1. INTRODUCTION

Equalizers (EQs) form an important class of audio processors capable of directly increasing or decreasing the loudness of specific frequencies within an audio signal [1]. The EQ is a fundamental tool for shaping the tone of sound, and apart from leveling and panning, EQ processing is essential to ensuring the harmonious blending of multiple elements within a musical mix [2]. One of the most commonly found forms of EQs is the parametric EQ, owing to its high level of user control, low-latency operation, and lack of pre-ringing artifacts [3]. A parametric EQ is characterized by multiple controllable EQ bands, each specified by their type (shelving, peaking, high/low-pass filters, etc.), frequency, gain, and quality factor (Q) [4].

Automatic EQ algorithms determine suitable tonal treatments to be applied to a given audio recording, with the presence or even in lieu of a reference recording characterizing the tonal style of interest [5]. Such treatments can be achieved by predicting settings for some underlying parametric EQ processor [6], or by means of an end-to-end model which directly applies equalization (and potentially other effects [7]) to the audio in a non-parametric manner [8]. Here, we focus our attention on the former, in order to promote predictable and artifact-free processing, computational efficiency, and seamless integration into existing parametric EQs, offering interpretable control and automation functionalities to end users of varying skill levels. Under this paradigm, a basic automatic EQ objective may be one that aims to balance all frequencies evenly, as in [9]. As demonstrated in [10], it is evident that different instrument types have their own characteristic spectral properties and therefore, targeting a linear frequency response (e.g. a pink noise target) across all forms of instrument tracks is unlikely to be suitable. Accordingly, more targeted criteria can be derived, potentially by data-driven means [11]. Specifically, the development of an effective automatic EQ system then becomes intimately related to realizing a suitable parametric EQ matching solution, whereby EQ parameters are inferred so that their resulting frequency response most closely resembles some idealized target treatment.

Signal processing-based EQ matching algorithms have been proposed for both graphic [12] and fully parametric [4] EQs. With the widespread adoption of neural networks (NN) across intelligent music production systems, several deep learning approaches have also been considered. Earlier deep learning-based methods attempted to minimize rudimentary losses defined over predicted and target EQ parameters [13, 14, 15]. In our own past work [3], we demonstrated that a multi-layer perceptron (MLP) network optimized using differentiable biquadratic filters (biquads) significantly improved matching performance relative to such methods, enabling objective functions to be defined directly in terms of target spectra and the spectra derived from predicted EQ parameters. More recently, DeepAFX-ST [5] generalized upon this concept even further, proposing a novel method to predict parameters for a processing chain consisting of both a parametric EQ and a compressor in order to mimic the production style of a reference audio clip. This approach leveraged differentiable implementations of both processors in order to optimize an end-to-end training objective defined over time-domain waveforms.

Regardless of the audio processors which they automate, or the respective objective functions which are optimized during their training, a common deficiency to parametric matching algorithms based on deep learning continue to be the self-supervised learning strategy that they employ in order to generate training data, whereby training targets are ultimately created by ascribing random configurations to audio effects. While this minimizes data engineering efforts, it inevitably imposes the need to determine suitable distributions from which to sample random parameters, which more often than not are merely chosen by intuition. Meanwhile, previous deep learning-based filter design studies have outlined the impact of parameter distribution selection on model performance, and accordingly, how they may potentially fail in mismatched environments [16]. Therefore, it is quite likely that resulting systems could struggle to generalize in real-world scenarios, whereby desired target treatments may deviate considerably from those seen during training, reducing their adoption and confidence in their overall effectiveness when deployed in an automated music production context.

In this paper, we propose a system to automatically equalize individual instrument tracks. At inference time, our algorithm identifies the instrument present in an input audio recording using a suitably trained convolutional neural network (CNN), and uses the resulting instrument class prediction to select a spectral target from a bank of instrument-specific spectral distributions that we have aggregated over an annotated set of produced audio samples. Consequently, it is instructive to note that our automatic EQ does not rely on reference audio examples, unlike several existing approaches [5, 15]. Next, we calculate the log-magnitude spectrum of the input audio and subtract it from this ideal spectrum. The resulting difference spectrum describes how frequencies need to be cut or boosted by an underlying parametric EQ in order to match the desired spectral target. A parametric EQ matching model subsequently tries to predict EQ parameters whose resulting frequency response most closely matches the spectral difference curve taken as its input. Lastly, an appropriately configured parametric EQ set to parameter values inferred from the matching model serves as a back end which processes the input audio signal, resulting in the automatically treated audio.

At the core of our proposed system is its parametric EQ matching model. To this end, we illustrate several improvements relative to established state-of-the-art matching techniques, building upon our own previous work which enabled direct optimization of loss functions defined in the spectral domain [3]. Accordingly, we introduce an enhanced CNN-based model architecture, reworking its activations and objectives for its training. Most notably however, we suggest a two-stage training procedure, whereby we directly leverage real-world audio data as training examples during a fine-tuning phase, which more closely resemble demonstrative inputs that would be encountered during practical use of our system. In its most basic form, we can derive realistic training targets for our model in an automated fashion by computing spectral differences between labeled training samples and their corresponding ideal instrument target spectra. In doing so, we effectively reduce the domain mismatch observed between training and real-world inference scenarios, whereby our matching model is intrinsically and more effectively optimized for the latter, as compared to existing deep-learning based audio processor automation methods that rely on synthetic data generation techniques to construct their training data.

The remainder of this paper is organized as follows. Section 2 gives a brief overview of our proposed automatic EQ system, Section 3 provides an in-depth explanation of our enhanced parametric EQ matching model and its improved training procedure, Section 4 describes the datasets used for training parametric EQ matching models in this work, Section 5 evaluates methods by both quantitative and qualitative means, and Section 6 draws conclusions based on these findings.

2. AUTOMATIC EQUALIZATION SYSTEM

Our automatic EQ method is depicted in Figure 1, consisting of multiple stages in order to predict EQ parameters from a given input audio recording that ultimately drive an underlying parametric EQ processor. We first identify the instrument contained in the input audio recording using a CNN-based instrument classifier model. According to its predicted class, an instrument-specific spectral curve is selected as an ideal target. Next, we compute the spectrum of the input audio and subtract it from the target curve. The resulting curve is then processed by a CNN-based EQ matching model in order to predict suitable EQ parameters. These settings can then be used by an appropriately configured parametric EQ in order to create the automatically equalized audio. In this work, the various analysis blocks within our system assume a nominal audio sample rate $f_s = 44.1$ kHz, while the parametric EQ processor itself naturally adapts to arbitrary sample rates.

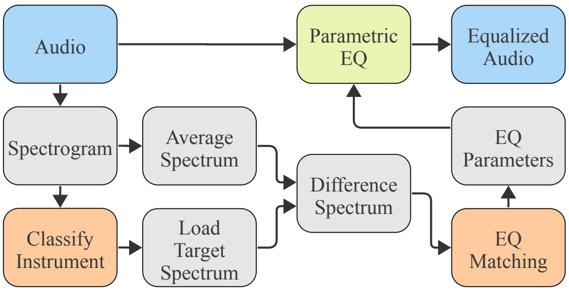

Figure 1: Proposed automatic EQ system.

2.1. Spectrogram Computation

Spectrograms serve as a common input for various components within our system. We compute a spectrogram by first segmenting the signal using Hann windows of size 2048 with an overlap of 50%. Each window is then processed by computing its discrete Fourier transform and transforming the magnitude spectrum onto the decibel (dB) scale. We then derive a vector $\mathbf{f}$ consisting of 256 logarithmically spaced frequency bins ranging from 20 Hz to 22 kHz, and linearly interpolate the spectrogram onto this grid.

2.2. Instrument Classifier

In order to classify the instrument type within a source recording, we trained a ResNet-like CNN [17] with approximately 2M parameters and 35 output instrument classes, including several common instrument types such as electric guitar, piano, vocal, drums and/or various percussive elements. Our internal training dataset covered roughly 45k instrument-annotated audio samples with a sample rate of 44.1 kHz. To prepare our training data for model ingestion, we split audio samples into 6 second segments. We compute the spectrogram for each segment as described in Section 2.1, resulting in a 256×256 matrix as input for the CNN. We evaluated the classifier on 10k additional test samples that were equally distributed across classes, and achieved a classification accuracy of 79%. As the majority of incorrect predictions occur between similar classes, e.g. ride cymbals get frequently misclassified as hi-hats, and these instruments tend to have similar spectral properties, such classification errors were not found to be significant for our use case.

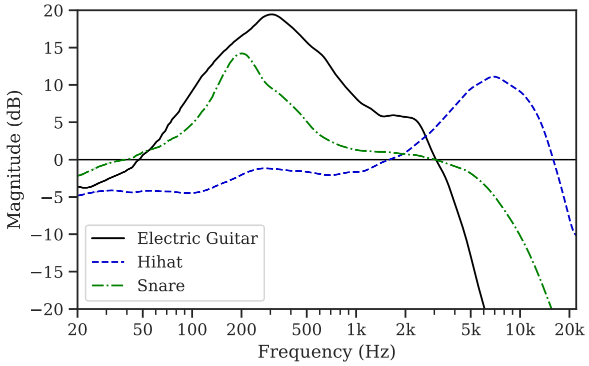

Figure 2: Examples of instrument-specific target spectra derived from our training set.

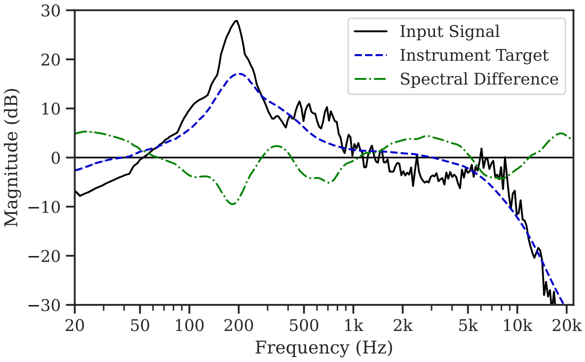

Figure 3: Spectral difference computation for a snare drum input signal with its corresponding ideal instrument target spectrum.

2.3. Instrument-specific Ideal Spectra

We derive ideal instrument-specific spectral targets, aggregating spectra across a curated subset of our training dataset as a function of instrument class. We calculate spectra according to the procedure described in Section 2.1, averaging the result along the time axis. Specifically, we calculate the spectrum for each sample, and compute the average spectrum corresponding to each instrument class accordingly. Lastly, we normalize all resulting curves to be zero-mean on the dB scale. Some select spectral target curves derived according to this procedure are illustrated in Figure 2.

2.4. Spectral Difference Calculation

We can determine the parts of the frequency spectrum whose loudness should ultimately be boosted or cut by the parametric EQ by performing a spectral difference on the dB scale between the input audio and its corresponding ideal instrument target. We first subtract the measured spectrum of the input audio from the target spectrum. Next, we apply Gaussian smoothing with a kernel standard deviation $\sigma = 3$ in order to filter out local peaks and disregard noisy ripples in the spectral difference. Accordingly, we focus on matching global spectral properties instead of compensating for local resonances. After smoothing, we normalize all resulting curves to be zero-mean on the dB scale, and limit their maximum absolute value to 12 dB by means of scaling. Figure 3 illustrates our described procedure for a select input signal/instrument target pair.

2.5. Parametric EQ Matching Model

We use a parametric EQ matching model to determine a set of EQ parameters whose resulting frequency response closely matches a given input spectral difference curve, as calculated in the previous stage. As such, the output parameters predicted by our model are designed to correspond exactly to the configuration of an underlying parametric EQ processor within our overall system. Since the parametric EQ matching model is such a pivotal component to our overarching method, we dedicate the entirety of Section 3 to outline its architecture and training procedure in further detail.

2.6. Parametric EQ Design

Parametric EQs are most often implemented as a cascade of biquads, i.e. second-order infinite impulse response filters (IIR) [18]. Each biquad implements the difference equation

$$\begin{aligned} y[n] = b_0x[n] + b_1x[n-1] + b_2x[n-2] \\ -a_1y[n-1] - a_2y[n-2] \end{aligned} \tag{1}$$

and has the transfer function

$$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}, \tag{2}$$

where the roots of the numerator and denominator polynomials dictate the nature of its filtering [19]. The resulting frequency response of a biquad can be evaluated over a vector of digital frequencies ω by evaluating equation (2) with $e^{j\omega}$ as its argument.

While biquads can model arbitrary second-order digital filters, there exist explicit formulae for defining parametric peaking and shelving filters with specified center/cutoff frequency f in Hz, gain g in dB, and Q/slope Q at a given sampling rate f_s [20]. To this end, it is common to use the bilinear transform in order to derive digital filters from their analog prototypes. A downside of this is that warping of the response is introduced for center/cutoff frequencies near the Nyquist frequency, as described in [21]. Thus, we use compensated second-order peak and shelving filters as proposed in [22] and [23]. These filter designs provide a closer approximation of the analog frequency response, while being more efficient than using oversampling.

A K-band parametric EQ can be created by cascading K biquads in series, where each biquad corresponds to an EQ band controlled by the user [24]. As in previous works [3, 5, 15], we consider a parametric EQ with $K \geq 2$ bands, consisting of exactly one low shelf (band 1), one high shelf (band K), and $K-2$ peaking filters (bands 2 through $K-1$). The composite transfer function of the parametric EQ is then given by

$$H_{eq}(z) = \prod_{k=1}^{K} H_k(z) \tag{3}$$

and the magnitude frequency response of the parametric EQ can be computed as

$$\left|H_{eq}(e^{j\omega})\right| = \left|\prod_{k=1}^{K} H_k(e^{j\omega})\right|. \tag{4}$$

Table 1: *Parametric EQ parameter ranges.*

	Low Shelf	Peak 1	Peak 2	High Shelf
f_{min}	30 Hz	200 Hz	600 Hz	1.5 kHz
f_{max}	450 Hz	2.5 kHz	7 kHz	16 kHz
g_{min}	-12 dB	-12 dB	-12 dB	-12 dB
g_{max}	+12 dB	+12 dB	+12 dB	+12 dB
q_{min}	0.75^*	0.1	0.1	0.75^*
q_{max}	0.75^*	3.0	3.0	0.75^*

* Fixed/not inferred by our parametric EQ matching model.

Here, we evaluate magnitude frequency responses at $\omega = (\pi/f_s)\mathbf{f}$ and express them on the dB scale, effectively matching the logarithmic frequency grid and output scale used in our various spectral computations.

In this work, we demonstrate our approach with $K = 4$, resulting in a parametric EQ comprised of a low shelf, 2 peak filters, and a high shelf. The bands of our parametric EQ have fixed individual parameter ranges, as detailed in Table 1. Gain parameters are to be within the range of ± 12 dB range across all bands. Frequency ranges are band-specific, with boundary values selected based on their common use in audio mixing contexts. The Qs for shelving filters are set to a fixed value of 0.75, noting that a variety of parametric EQs leverage shelving filters having fixed or default Qs that are at or near this value. The range of Qs for peaking filters spans from 0.1 to 3.0, as in [3]. Ultimately, this means that our automatic EQ algorithm is responsible for determining a total of 10 novel EQ parameters, corresponding to 4 frequencies, 4 gains, and 2 Qs. Processing of the input audio signal through a parametric EQ with these resulting settings yields the automatically equalized audio output of our system.

3. NEURAL PARAMETRIC EQUALIZER MATCHING

As described in [3], deep learning approaches for parametric EQ matching are capable of outperforming classical approaches [4] under appropriate training conditions. Accordingly, we consider two different NN architectures in this work, as shown in Figure 4. Moreover, we consider a multi-stage training procedure leveraging various combinations of loss terms and data sources, as indicated in Figure 5. Base models are first trained on synthetic data using a parameter loss. These base models are then fine-tuned using a spectral loss function, and trained on either synthetically generated or real-world data for comparison.

3.1. Model Architectures

The first model we consider is an MLP similar to the one described in [3], serving as a baseline for this work. Its architecture consists of 3 linear layers with 256 neurons and ReLU activation functions in each layer, which process spectral difference curves as input to the model. The final output linear layer has 256 input and 10 output neurons, corresponding to the variable parameters of our 4-band parametric EQ.

We also propose a second, enhanced model in the form of a CNN, which combines the baseline architecture with an additional convolutional front end. The rationale for such a front end is to learn useful local patterns from desired frequency response curves seen during training. Architecturally, we exchange the first linear layer with 3 convolutional layers with 16, 32 and 32 channels, each followed by a ReLU activation function. All convolutional layers have a kernel size of 5, a stride of 1 and no padding. The number of neurons in the first linear layer is increased to 7808 according to the length of the flattened tensor after the convolutional layers. Subsequent hidden and output layers are identical to the MLP.

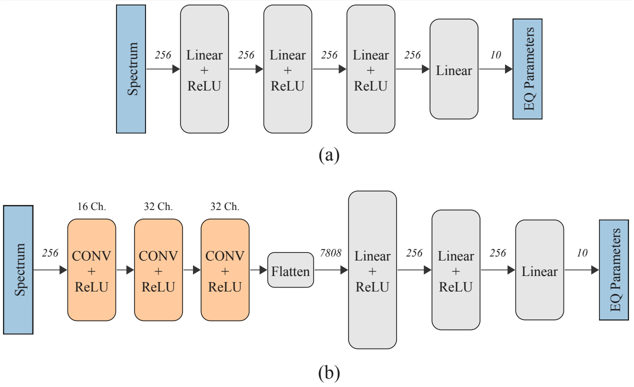

Figure 4: Model architectures for parametric EQ matching: (a) baseline MLP model as in [3] and (b) our enhanced CNN model with added convolutional front end.

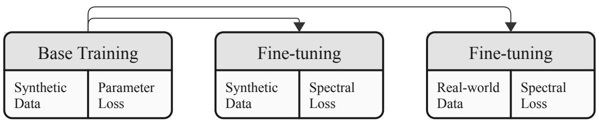

Figure 5: Base training and fine-tuning stages of neural parametric EQ matching models.

Another upgrade that we consider in this work is a deliberate decision to avoid activations at the output of the model entirely, unlike our own past works and other related follow-ons [3, 5, 19]. We noted that applying such activations in this context can make models susceptible to vanishing gradients during training, ultimately limiting performance. Naturally, we desire to constrain the values that EQ parameters can take on to some desired range, but we empirically find that we can achieve this more effectively using the combination of techniques outlined in Sections 3.2-3.4. Considering our use of fixed slopes for shelving filters, it turns out that our inferred filter parameterization is guaranteed to yield stable filters without the need for any additional parameter-specific activation functions. As an illustrative example, we note that inferred frequencies will effectively alias to a frequency within the Nyquist band as a natural consequence of our filter design equations.

3.2. EQ Parameter Normalization

In order to neutralize the effect that differing value ranges for different EQ parameter types (i.e. frequency, Q, and gain) can have on model training, we normalize all EQ parameters to values between 0 and 1 via

$$f_{\text{norm}} = \frac{\log\left(\frac{f}{f_{\text{min}}}\right)}{\log\left(\frac{f_{\text{max}}}{f_{\text{min}}}\right)}, \tag{5}$$

$$g_{\text{norm}} = \frac{g - g_{\text{min}}}{g_{\text{max}} - g_{\text{min}}}, \tag{6}$$

and

$$Q_{\text{norm}} = \frac{Q - Q_{\text{min}}}{Q_{\text{max}} - Q_{\text{min}}}. \tag{7}$$

where we use the minimum/maximum parameter boundaries as defined on a band-specific basis for the various EQ parameter types in Table 1. Underlying EQ parameters can be computed from normalized model outputs by applying their respective inverse normalization functions.

3.3. Base Training Using a Parameter Loss Function

In a first step, we perform regression with an EQ parameter loss function using synthetically generated spectral difference curves derived from randomized EQ parameter settings (whose sampling strategy will be described in further detail in Section 4.1). This helps the model to predict EQ parameters in the desired normalized range, and provides a starting point for further fine-tuning in a second stage. In this case, the inner workings of the parametric EQ is assumed to be unknown to the model, and as such, there is no direct means of performing backpropogation through the audio processor itself. The parameter loss is defined as

$$\mathcal{L}_{\text{v}}(\mathbf{v}, \hat{\mathbf{v}}) = ||\mathbf{v} - \hat{\mathbf{v}}||_1, \tag{8}$$

with $\hat{\mathbf{v}}$ and $\mathbf{v}$ as predicted and target EQ parameters, respectively. We posit that model parameters based on this training are certainly better calibrated to the matching task than a random initialization. As noted in [3], optimizing a parameter loss in this manner has been previously considered in related works [13, 14, 15].

3.4. Model Fine-tuning Using a Spectral Loss Function

Motivated by the use of differentiable audio processors to enable end-to-end training [25], our previous work in [3] already demonstrated that EQ matching performance can be improved by using differentiable biquads to calculate the magnitude frequency response from predicted EQ parameters. The key insight expressed there was that parametric EQ filter design formulae and equation (4) (with corresponding ω) could readily be implemented using differentiable operations within deep learning libraries, allowing for backpropogation through a loss defined directly in terms of output magnitude frequency responses and model input spectral difference curves. Accordingly, we can define a spectral loss as

$$\mathcal{L}_{\text{x}}(\mathbf{x}, \hat{\mathbf{x}}) = ||\mathbf{x} - \hat{\mathbf{x}}||_1 \tag{9}$$

with $\hat{\mathbf{x}}$ as the resulting frequency response (on the dB scale) calculated from predicted EQ parameters $\hat{\mathbf{v}}$. The spectral difference curve $\mathbf{x}$ naturally serves as both the input to the model as well as a training target (as is the case with an auto-encoder).

As previously mentioned, the bands of our parametric EQ have fixed individual parameter ranges. Accordingly, we introduce an additional loss term to our training objective during fine-tuning to ensure that the predicted EQ parameters are within these defined ranges, foregoing the need for output activation functions which may contribute to vanishing gradients. This penalty term is shown in Figure 6, and is defined as

$$\mathcal{L}_{\text{p}}(\hat{\mathbf{v}}) = ReLU(-\hat{\mathbf{v}}) + ReLU(\hat{\mathbf{v}} - 1). \tag{10}$$

The net effect of this added term is if a predicted parameter is out of its desired range during training, the absolute difference between the predicted parameter and the exceeded boundary is added to the loss, bearing resemblance to penalty methods used in some constrained optimization algorithms [26]. Finally, the composite objective function used during model fine-tuning is

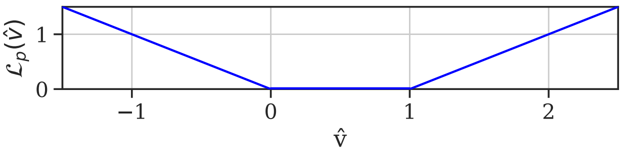

Figure 6: Penalty loss term for fine-tuning.

$$\mathcal{L}_{\text{f}}(\mathbf{x}, \hat{\mathbf{x}}, \hat{\mathbf{v}}) = \mathcal{L}_{\text{x}}(\mathbf{x}, \hat{\mathbf{x}}, \hat{\mathbf{v}}) + \lambda \mathcal{L}_{\text{p}}(\hat{\mathbf{v}}), \tag{11}$$

where we set $\lambda = 1$ in this work for demonstrative purposes. We can fine-tune models according to this objective using either synthetically generated data or using spectral difference curves derived from real-world data, and compare both approaches in this work.

4. DATASETS

In this section, we describe the construction of datasets used to train our parametric EQ matching models, as well as to assess both parametric EQ matching performance and the fidelity of equalized audio products from our composite automatic EQ algorithm. In addition to synthetic data generation mechanisms using random parameter sampling techniques as in several previous works [3, 5, 13, 15], we demonstrate how real-world data can also be utilized during parametric EQ matching model training due to the system design of our automatic EQ method.

4.1. Synthetic Data Via Random EQ Parameter Sampling

We leverage synthetically generated data for the base training of the EQ matching model, and as an alternate data source during fine-tuning for comparative purposes. In this case, we generate input spectral difference curves to serve as model input by randomly sampling EQ parameters and compute their corresponding magnitude response. The normalized representation of these EQ parameters can be utilized as training targets in the base training case using a parameter loss function. Their corresponding frequency responses are used as model inputs, and alternatively as training targets in the fine-tuning case making use of a spectral loss function. Random frequencies are determined for each EQ band by

$$f_{\text{rand}} = f_{\text{min}} \cdot e^{x_{\text{rand}} \cdot \log\left(\frac{f_{\text{max}}}{f_{\text{min}}}\right)} \tag{12}$$

with $x_{\text{rand}} \sim \text{Uniform}(0, 1)$. This guarantees the same probability for all frequencies on a logarithmic scale. Random EQ band gains are calculated as

$$g_{\text{rand}} = x_{\text{rand}}^3 \cdot g_{\text{max}} \cdot x_{\text{sign}} \tag{13}$$

with $x_{\text{rand}} \sim \text{Uniform}(0, 1)$, as well as a random multiplicative sign term $x_{\text{sign}} \sim 2 \cdot \text{Bernoulli}(0.5) - 1$ which leads to negative gains with a probability of 50%. The x_{rand}^3 term results in a higher probability for selecting small absolute gain values, as we expect lower gains to occur more frequently in real-world applications. Shelving filters have a fixed Q of 0.75, and for peak filters, we generate random Q values $Q_{\text{rand}} \sim \text{Uniform}(q_{\text{min}}, q_{\text{max}})$. Lastly, in order to achieve more natural results for our generated spectra, we perform augmentation by adding random noise to each amplitude value. Using this procedure, we generate a training dataset containing 156,730 synthetically generated curves with their associated EQ parameters.

4.2. Real-world Data

While synthetically generated training data can serve as a decent proxy for high-fidelity training examples in closed experimental settings, we would ideally like to train our parametric EQ matching model using real-world data examples resembling those that we are likely to encounter during inference. In order to create such a training dataset, we processed a 31,346 sample subset of our internal dataset, and applied a form of nearest neighbor data augmentation similar to [27]. Specifically, we measured the spectrum of each audio example, and compute corresponding spectral difference curves, using not only the respective ground truth instrument target spectrum, but those of 4 other closely related target spectra derived from similar instrument classes for each example. The use of "nearby" instrument target curves in this manner creates additional input/target pairs for training that could still conceivably occur in practical applications. Using this augmentation technique, we create 5 spectral difference curves per audio example, resulting in a total number of 156,730 training input spectra. Additionally, we curate a held-out test dataset consisting of 8k samples and their respective spectral difference curves, which we use to evaluate the various models trained in this work.

At inference time, we leverage our instrument classifier to determine the instrument type of input recordings to our system (and therefore their corresponding spectral target) in the absence of a corresponding ground truth label. As long as our instrument classifier is adequately accurate, we can make the claim that in this instance, training examples based on real-world audio data are more likely to resemble those seen during inference time, as compared to training examples based solely on synthetically generated curves via random parameter sampling. Therefore, one would intuit that fine-tuning on real-world audio data in this manner should improve parametric EQ matching performance as it pertains to practical use cases within our automatic EQ system, since we have explicitly sought out to bridge this domain mismatch gap. Moreover, training examples based on real-world audio data are more likely to be indicative of suitably produced tonal treatments of individual instrument tracks, which we assert has good implications with regards to the treated audio that would ultimately be created by our automatic EQ method.

5. EXPERIMENTAL RESULTS

We perform a number of experiments in order to demonstrate the effectiveness of both our enhanced parametric EQ matching capabilities as well as our proposed automatic EQ system taken as a whole. Our quantitative evaluation investigates the impact of architecture, loss function, and data type on parametric EQ matching model performance, while our subjective evaluation is used to validate whether or not our automatic EQ system can indeed improve tonal characteristics of input audio recordings in a fully automated fashion. All evaluations are performed using our held-out real-world audio data test set, reflecting realistic use cases for our methods.

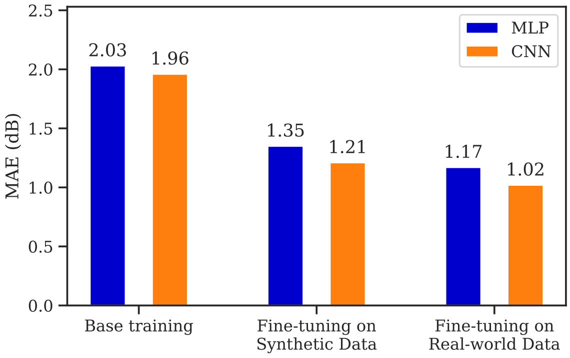

Figure 7: Quantitative evaluation of the neural parametric EQ matching models trained in this work, as measured on our curated test set.

5.1. Objective Evaluation of Parametric EQ Matching Models

We trained a total of 6 models in order to compare parametric EQ matching accuracy across several factors. First, we trained base models using both MLP and CNN architectures. Next, we performed separate fine-tunings on synthetic and real-world data for each respective base model, as described in Section 3. All training stages across all models used the Adam optimizer with a learning rate of 10^{-4} and a batch size of 128. We trained models for 3 epochs during each stage while reducing the learning rate after every epoch by a factor of 0.1. Our objective evaluation methodology involved computing the average spectral loss, as assessed over all 8k examples in our real-world test set. Given its definition and domain on which it operates, we simply refer to this as the mean absolute error (MAE) of a given model.

Our quantitative evaluation is summarized in Figure 7, revealing several key observations within the different training configurations we tested. While base training with a parameter loss notionally aligns models towards our given task, we observe a significant decrease in MAE when directly optimizing a spectral loss function, even when training on synthetically generated data derived via random EQ parameter settings. Specifically, the reduction amounts to 0.68 dB for the MLP and 0.75 dB for the CNN, aligning with prior findings documented in [3]. We note a further decrease in MAE when utilizing real-world training data. The reduction for the MLP and CNN is 0.18 dB and 0.19 dB, respectively. Moreover, the superiority of the CNN architecture over the MLP is evident, with consistent MAE reductions observed across all trainings. Using the CNN, the base training yields an MAE reduction of 0.07 dB against the MLP. When fine-tuning on synthetic or real-world data, the use of the CNN in place of the baseline MLP results in MAE reductions of 0.14 dB and 0.15 dB, respectively.

Fine-tuning of the MLP with synthetic data mirrors the approach outlined in [3], recording an MAE of 1.35 dB. Meanwhile, our enhanced CNN architecture yields an MAE of 1.02 dB when fine-tuned on real-world data. Therefore, we exhibit a net decrease of 0.33 dB (a 24% improvement) in comparison to our previous state-of-the art approach. Figure 8 illustrates sample outputs for the various models we trained in this work, where we can visually note the improved matching capability of our best performing model over previously published baselines.

5.2. Subjective Evaluation of our Automatic EQ System

We demonstrate the effect of our automatic EQ system on real-world individual instrument recordings by conducting a listening test. This subjective evaluation consisted of 35 test trials from audio examples within our held-out test set, including individual tracks reflecting a number of common instrument types, consisting of a full acoustic drum set, individual drum elements, electric guitar, bass, brass, synths, organ, and vocals. We considered the untreated audio and audio processed with our proposed system (according to predicted EQ parameters from our best performing EQ matching model) as stimuli for each test case. To prevent participants to be biased by different loudness, we normalized all examples to -23 LUFS (loudness units relative to full scale [28]). In each trial, participants were tasked to decide whether they prefer the original audio or the automatically equalized one. We encouraged participants to make these decisions based on the following criteria: 1) the individual instrument sounds like it commonly "should," 2) problematic frequencies are suitably suppressed, and 3) the individual instrument track is more likely to fit within a full mix context. Accordingly, we constructed our evaluation as a blind A/B test, whereby participants do not know which audio example is the original one and which one is treated across all test cases, whilst providing a third neutral option to indicate no preference. This is based on the recent findings presented in [29], where the authors compared A/B and mean opinion score listening test formats exemplary of those used to evaluate text-to-speech systems, and showed that A/B tests are more reliable and less influenced by the number of listeners.

The results of our listening tests are summarized in Figure 9. In total, 26 participants within our organization (with proficient listening skills and varied musical experience) took part in the evaluation. It is observed that participants preferred the equalized treatments produced by our proposed automatic EQ technique nearly 2 to 1 over the original audio, and were undecided in 14% of cases.

6. CONCLUSIONS

In this paper, we proposed an automatic EQ system that predicted settings for an underlying parametric EQ, using a combination of CNNs for parametric EQ matching and instrument classification. We quantitatively showed how our proposed enhancements to neural parametric EQ matching led to fidelity gains in accuracy relative to our previous state-of-the-art approach in [3]. This was accomplished by utilizing a CNN architecture, revamping model activations and learning objectives, and lastly, by leveraging real-world audio samples as part of a two-stage training process. To this end, the use of our instrument classifier within the proposed system allowed us to significantly reduce domain mismatch gaps between training and real-world inference scenarios relative to previous methods [5, 15], increasing the overall effectiveness of our composite system. Accordingly, we conducted listening tests and showed that our proposed automatic EQ system was capable of subjectively enhancing the sound quality for common instrument and vocal recordings, all without the need for providing reference audio at inference time. Future work will be dedicated to additional refinements to our system, and investigating further how our parametric EQ matching paradigm scales to various EQ configurations. We are also interested in devising similar automation systems for other forms of audio processors, such as the direct dynamics processor in [30].

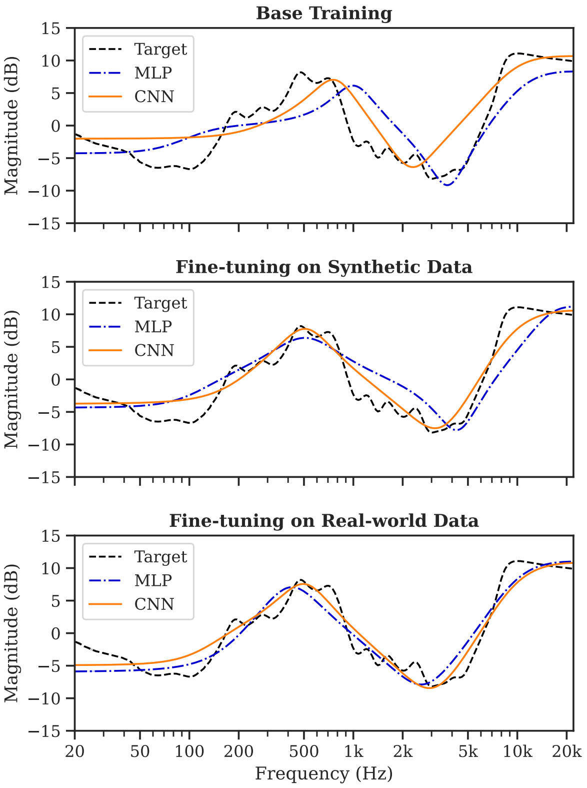

Figure 8: Parametric EQ matching comparisons of the various models trained in this work, demonstrated using a piano recording as input.

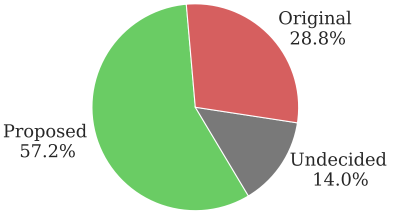

Figure 9: High-level summary of our subjective evaluation.

7. REFERENCES

[1] V. Välimäki and J. D. Reiss, "All about audio equalization: Solutions and frontiers," *Applied Sciences*, vol. 6, no. 129/5, pp. 1–46, May 2016.

[2] U Zölzer, *Digital Audio Signal Processing*, Springer, Hamburg, Germany, second edition, 2007.

[3] S. Nercessian, "Neural parametric equalizer matching using differentiable biquads," in *Proceedings of the International Conference on Digital Audio Effects*, Sept. 2020, pp. 265–272.

[4] J. S. Abel and D. P. Berners, "Filter design using second order peaking and shelving sections," in *Proceedings of the International Computer Music Conferences*, Miami, FL, USA, Nov. 2004.

[5] C. J. Steinmetz, N. J. Bryan, and J. D. Reiss, "Style transfer of audio effects with differentiable signal processing," *Journal of the Audio Engineering Society*, vol. 70, no. 9, pp. 708–721, 2022.

[6] J. Iriz, M.A. Patricio, A. Berlanga, and et al., "CONEqNet: Convolutional music equalizer network," *Multimedia Tools and Applications*, vol. 82, pp. 3911–3930, Jan. 2023.

[7] M. A. Martínez Ramírez, W. H. Liao, G. Fabbro, S. Uhlich, C. Nagashima, and Y. Misufuji, "Automatic music mixing with deep learning and out-of-domain data," in *Proceedings of the 23rd International Society on Music Information Retrieval Conference (ISMIR)*, Dec. 2022.

[8] M. A. Martínez Ramírez and J.D. Reiss, "End-to-end equalization with convolutional neural networks," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Sept. 2018.

[9] E. Perez-Gonzalez and J. Reiss, "Automatic equalization of multichannel audio using cross-adaptive methods," in *Proceedings of the 127th Convention of the Audio Engineering Society (AES)*, Oct. 2009.

[10] G. Davis and R. Jones, *The Sound Reinforcement Handbook*, Hal Leonard Corporation, Milwaukee, WI, 2002.

[11] B. De Man, J. D. Reiss, and R. Stables, "Ten years of automatic mixing," in *Proceedings of the 3rd Workshop on Intelligent Music Production*, Salford, UK, Sept. 2017.

[12] J. Liski and V. Välimäki, "The quest for the best graphic equalizer," in *Proceedings of the International Conference on Digital Audio Effects*, Edinburgh, UK, Sept. 2017, pp. 95–102.

[13] V. Välimäki and J. Rämö, "Neurally controlled graphic equalizer," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, Dec. 2019.

[14] J. Rämö and V. Välimäki, "Neural third-octave graphic equalizer," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Birmingham, UK, Sept. 2019, pp. 1–6.

[15] S. I. Mimilakis, N. J. Bryan, and P. Smaragdis, "One-shot parametric audio production style transfer with application to frequency equalization," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 256–260.

[16] J. T. Colonel, C. J. Steinmetz, M. Michelen, and J. D. Reiss, "Direct design of biquad filter cascades with deep learning by sampling random polynomials," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2022.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition Conference (CVPR)*, June 2016, pp. 770–778.

[18] K. Williston, *Digital Signal Processing: World Class Designs*, World Class Designs. Elsevier Science, 2009.

[19] S. Nercessian, A. Sarroff, and K.J. Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 890–894.

[20] R. Bristow-Johnson, "RBJ Audio-EQ-Cookbook," Available at https://www.musicdsp.org/en/latest/Filters/197-rbj-audio-eq-cookbook.html, accessed March 17, 2024.

[21] M. Massberg, "Digital low-pass filter design with analog-matched magnitude response," in *Proceedings of the 131st Convention of the Audio Engineering Society (AES)*, Oct 2011.

[22] S. J. Orfanidis, "Digital parametric equalizer design with prescribed nyquist-frequency gain," *J. Audio Eng. Soc*, vol. 45, no. 6, pp. 444–455, 1997.

[23] J. S. Abel and D. P. Berners, "Discrete-time shelf filter design for analog modeling," in *Proceedings of the 115th Convention of the Audio Engineering Society (AES)*, New York, NY, October 2003, Audio Engineering Society.

[24] K.B. Christensen, "A generalization of the biquadratic parametric equalizer," in *Proceedings of the 115th Convention of the Audio Engineering Society (AES)*, Oct. 2003.

[25] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Proceedings of the International Conference on Learning Representations*, Addis Ababa, Ethiopia, Apr. 2020, pp. 26–30.

[26] A.C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, Jan. 2002.

[27] G. Narita, J. Shimizu, and T. Akama, "GANStrument: Adversarial instrument sound synthesis with pitch-invariant instance conditioning," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, June 2023.

[28] ITU, *Rec. ITU-R BS.1770-1, Algorithms to measure audio programme loudness and true- peak audio level.*, International Telecommunications Union, 2007.

[29] J. Camp, T. Kenter, L. Finkelstein, and R. Clark, "MOS vs. AB: Evaluating text-to-speech systems reliably using clustered standard errors," in *Proceedings of Interspeech*, 2023.

[30] S. Nercessian, R. McClellan, and A. Lukin, "A direct microdynamics adjusting processor with matching paradigm and differentiable implementation," in *Proceedings of the International Conference on Digital Audio Effects*, Sept. 2022, pp. 248–255.