بازنویسی مقاله

نام و نام خانوادگی : ستاره همایون مقدم

شماره دانشجویی : 980120254

پارافیز مقاله درس DSP

عنوان مقاله : AUTO-DSP: LEARNING TO OPTIMIZE ACOUSTIC ECHO CANCELLERS

دانشکده برق دانشگاه علوم و تحقیقات

A brief of this paper :

The paper introduces a novel approach called "Auto-DSP" that aims to automatically learn optimal update rules for adaptive filtering algorithms, particularly in the context of acoustic echo cancellation.

Rather than relying on hand-derived, expert-designed update rules, the Auto-DSP method frames the adaptive filtering process as a differentiable operation that can be optimized using a learned optimizer. This learned optimizer is trained directly on data, without the need for external labels or manual tuning.

The key idea is to train a neural network-based optimizer that can output gradient-based update rules for the adaptive filter parameters. By using backpropagation through time, the optimizer can be meta-learned to produce high-performing update rules for a variety of adaptive filtering architectures, including both linear and nonlinear models.

The authors demonstrate the effectiveness of this approach on acoustic echo cancellation tasks, showing that the learned optimizers can outperform traditional hand-tuned baselines in terms of convergence speed, robustness to nonlinearities, and adaptation to unseen acoustic environments. This suggests that the Auto-DSP method has the potential to revolutionize how adaptive filters are designed and optimized in the future.

Introduction

Acoustic echo cancellation (AEC) is a critical component in modern communication systems, especially in teleconferencing, hands-free communication, and speakerphones. AEC systems are designed to remove the acoustic echoes that occur when a sound is played through a loudspeaker and picked up by a microphone, leading to annoying feedback and degraded audio quality. Traditionally, AEC algorithms have been designed using handcrafted signal processing techniques, which often require extensive tuning and customization for different acoustic environments. However, with the recent advancements in machine learning and digital signal processing (DSP), there is a growing interest in developing automated approaches to optimize AEC systems.

In this article, we will explore the concept of using machine learning to optimize AEC systems, specifically focusing on the application of Auto-DSP (Automatic Digital Signal Processing) techniques. We will discuss the challenges of traditional AEC design, the potential benefits of using machine learning for optimization, and the current state of research in this area.

-what is a AEC system?

In the context of **Digital Signal Processors (DSPs)**, **Acoustic Echo Cancellation (AEC)** is an essential audio processing effect. Let me break it down for you:

1. **Scenario**:
   * Imagine an audio conference call where there’s a room with microphones and loudspeakers on the “near side” (where you are) and someone at the “far side.”
   * The “near side” refers to the room where you’re physically located, and the “far side” is where the other participants are.
2. **Purpose of AEC**:
   * AEC systems in DSPs aim to prevent acoustic echo from being transmitted back to the far-end.
   * When you speak, your voice comes out of the loudspeaker in the near-end room and follows various acoustic paths to reach the near-end microphone.
   * Without AEC, your voice would bounce off walls, windows, tables, and other surfaces, creating an annoying echo that interferes with communication.
3. **How AEC Works**:
   * The AEC component estimates the echo and subtracts it from the microphone signal.
   * It ensures that you don’t hear your own voice delayed (which can be distracting) while preserving the sound of other near-end talkers.
   * [Additional sub-systems within AEC include Adaptive Filter, Adaptive Algorithm, Double-Talk Detection (DTD), Non-Linear Processing (NLP), Noise Reduction (NR), and Comfort Noise (CN)](https://q-syshelp.qsc.com/q-sys_5.1/Content/Appendix/Acoustic%20Echo%20Cancellation%20White%20Paper.htm).

Challenges of Traditional AEC Design

Traditional AEC systems are typically designed using a combination of adaptive filtering techniques, such as least mean squares (LMS) or normalized least mean squares (NLMS) algorithms, along with various signal processing blocks for echo suppression and residual echo suppression. These systems require careful tuning of parameters and filters to achieve optimal performance, and often struggle to adapt to changing acoustic environments or non-stationary echo paths. Furthermore, the performance of traditional AEC systems can be limited by the complexity of the acoustic environment, the presence of non-linearities in the echo path, and the need for real-time operation.

The Need for Automated Optimization

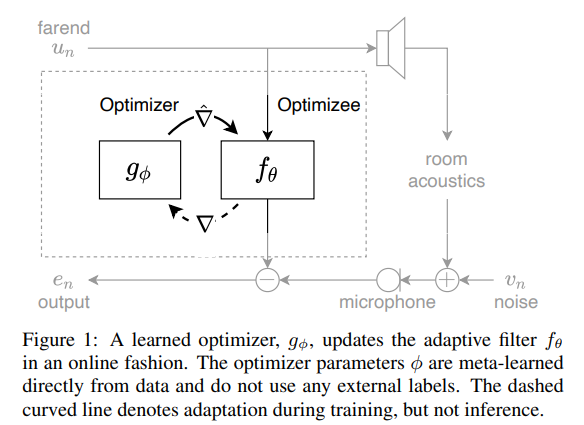
Automated optimization techniques, such as machine learning algorithms, offer the potential to address many of the challenges faced by traditional AEC design. By leveraging large datasets of acoustic impulse responses and echo signals, machine learning algorithms can learn to identify optimal filter coefficients and signal processing parameters that minimize the residual echo while preserving the quality of the desired speech signal. Additionally, machine learning models can adapt to changes in the acoustic environment and non-linearities in the echo path, leading to improved robustness and performance.

Current State of Research

The application of machine learning to optimize AEC systems is an active area of research, with several promising approaches being explored. One approach involves using deep learning techniques, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), to learn complex mappings from acoustic impulse responses to optimal filter coefficients. Another approach focuses on reinforcement learning algorithms, which can learn to optimize AEC systems through interaction with the acoustic environment. Additionally, researchers are exploring the use of transfer learning techniques to adapt pre-trained machine learning models to new acoustic environments with minimal data requirement

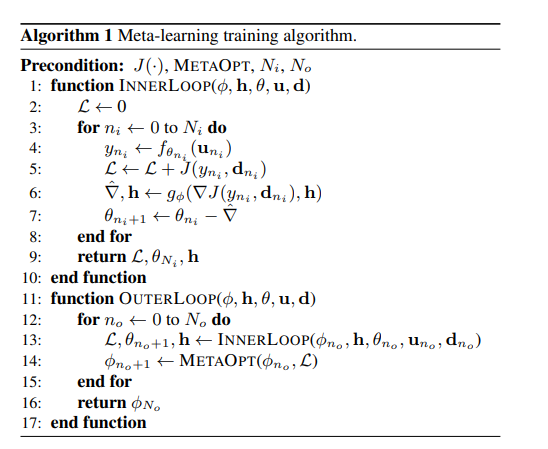
Auto-DSP: Learning to Optimize AEC

Auto-DSP refers to the use of machine learning algorithms to automatically optimize digital signal processing techniques for specific applications, such as AEC. In the context of AEC optimization, Auto-DSP involves training machine learning models on large datasets of acoustic impulse responses and echo signals, with the goal of learning to predict optimal filter coefficients and signal processing parameters for different acoustic environments. The trained models can then be used to automatically optimize AEC systems in real-time, without the need for manual parameter tuning or customization.



Algorithm

To learn an adaptive filter update rule from data, we first define a learned optimizer, g φ (•), as a function doing the optimizing and an optimize, f θ (•), as a differentiable adaptive filter to be optimized, and J(•) as the optimize loss. Second, we set the optimizer to be a neural network that accepts as input raw optimize gradients ∇f θn (•) and a state vector h and outputs a learned gradient descent update rule,



Meta-learning in ML

refers to learning algorithms that learn from other learning algorithms. Let me break it down for you:

1. **Ensemble Learning**:
   * Most commonly, meta-learning involves using machine learning algorithms that learn how to best combine the predictions from other machine learning algorithms. This falls under the field of ensemble learning.
   * For example, techniques like **stacking** learn how to combine predictions from multiple models to improve overall performance.
2. **Model Selection and Tuning**:
   * Meta-learning can also refer to the manual process of model selection and algorithm tuning performed by practitioners on a machine learning project.
   * Modern **AutoML** (Automated Machine Learning) algorithms aim to automate this process, making it more efficient.
3. **Learning Across Multiple Tasks**:
   * Another aspect of meta-learning is learning across multiple related predictive modeling tasks, known as **multi-task learning**.
   * Meta-learning algorithms learn how to learn from these related tasks, which can accelerate learning on new tasks.

In summary, meta-learning is about **learning to learn**—whether it’s combining predictions, automating model selection, or leveraging knowledge from related tasks.

Optimize architecture & loss

The optimize, or adaptive filtering being optimized, provides the architecture used for filtering signals.

It is defined by filter parameters θ, a filtering architecture fθ(·), and an optimize loss function. For illustrative purposes, we can consider a basic time-domain adaptive filter optimize.

In this case, the optimize parameters θ correspond to transversal finite impulse response (FIR) filter coefficients θ = {wˆ n ∈ R N }, the optimize architecture corresponds to the inner product between an input vector un ∈ R N and the filter coefficients

fθ(un) = yn = wˆ H nun

, and the optimize loss corresponds to a mean squared error objective,

J(yn, dn) = 1 N PN n |yn − dn| 2 , where dn ∈ R is the desired

, known response. In this case, we can reduce the optimizee update

(1) to wˆ n+1 = wˆ n − gφ(un · (wˆ H nun − dn) ∗ )

(2) where ∗ denotes complex conjugation and H denotes Hermitian transposition

Optimize configuration

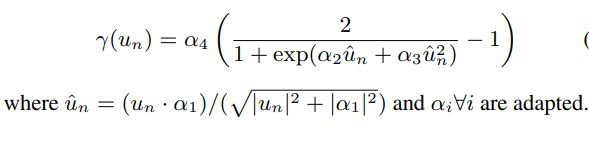
To demonstrate our approach, they consider the adaptive filtering task of acoustic echo cancellation or interference cancellation.

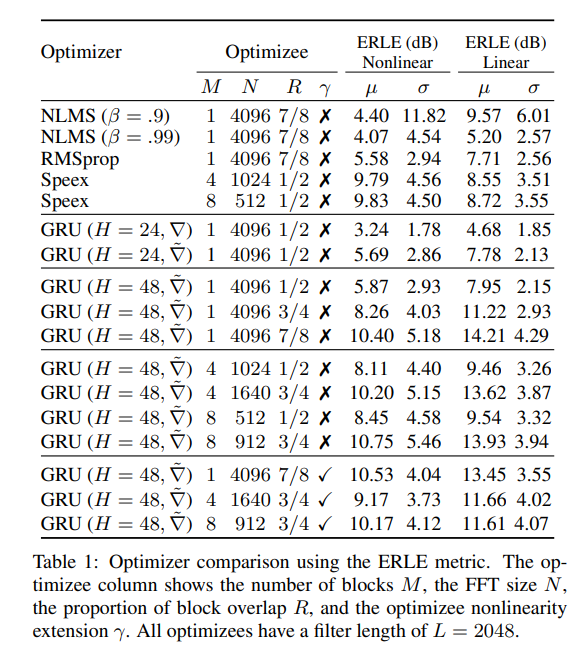
The optimize parameters θ include frequency domain filter coefficients and a small set of nonlinear coefficients.

The filter coefficients are partitioned into multiple delayed blocks and used within the framework of overlap-save short-time Fourier transform processing [37]. MDF filters are commonly used for AEC and leverage the benefits of both frequency-domain adaptation and low latency.

For our optimize loss, which implicitly defines the optimizer loss, we use the mean squared error. In more detail, our MDF filter consists of frequency domain filter coefficients W ∈ C M×N , where M is the number of delayed blocks, N is the fast Fourier transform (FFT) size, P = M · N/2 is the number of filter parameters, and L is the filter length in samples.

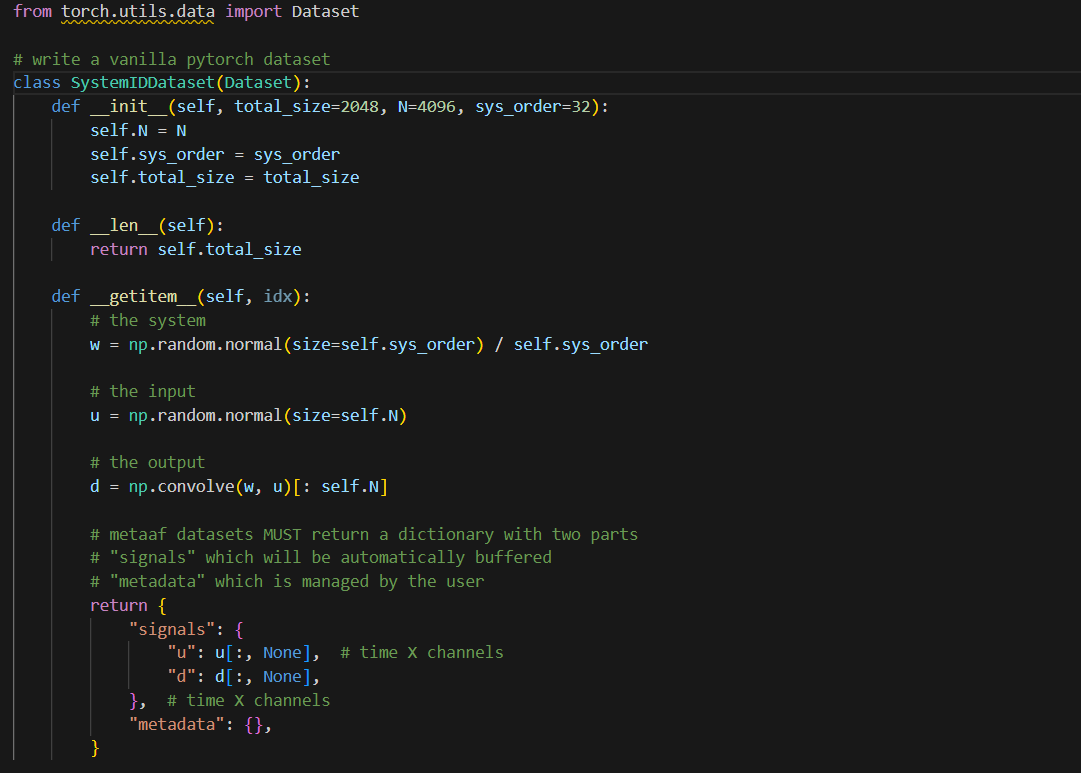
The filter matrix is applied to the delayed frequency domain near-end inputs U ∈ C M×N to yield a filtered output via yn = last N/2 terms of{FFT−1 ((W U) >1N )}, where > is a matrix transpose, is the hadamard product, and 1N is an N × 1 matrix of ones. To construct U, we buffer the time-domain nearend signal to length N with time overlap R, forming un˜ ∈ R N , shift Um = Um+1 for m = 1, 2, · · · , M − 1, and assign UM = FFT(un˜ ). Finally, we antialias W after each update so that each block has N/2 nonzero time-domain parameters. For our nonlinearity extension, we preprocess each element un of the farend reference signal through a parametric sigmoid



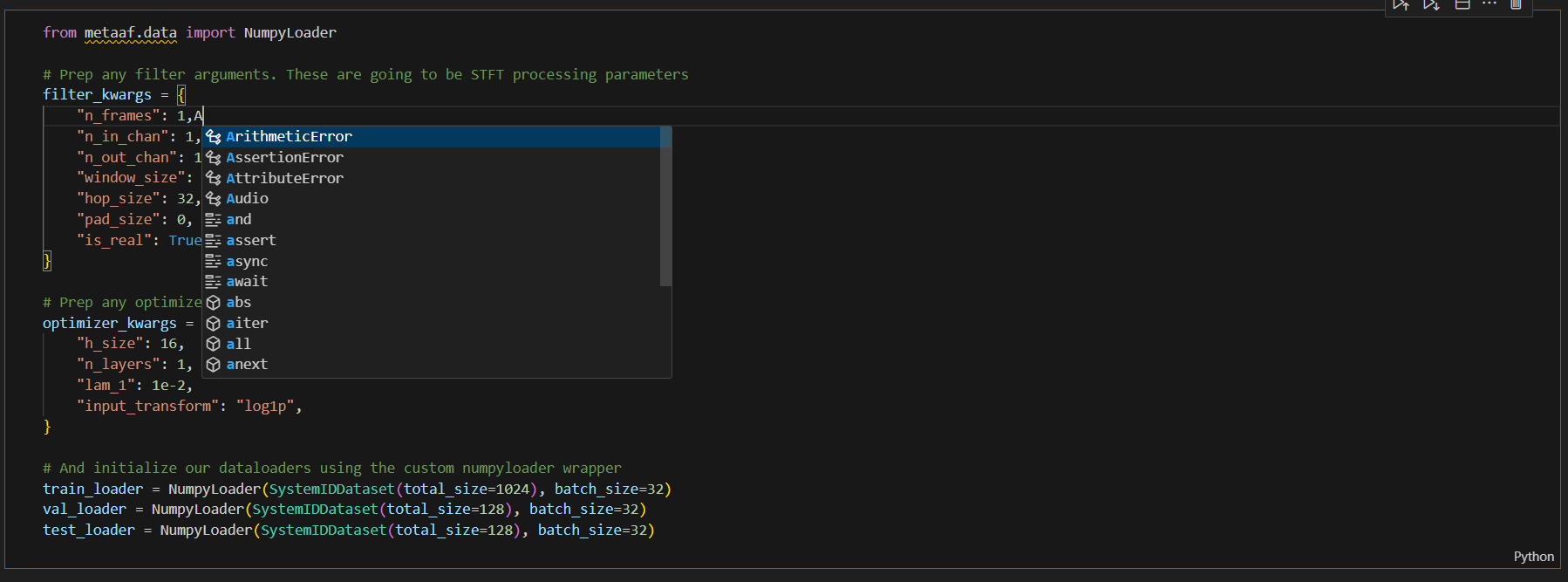


Making dataset in meta

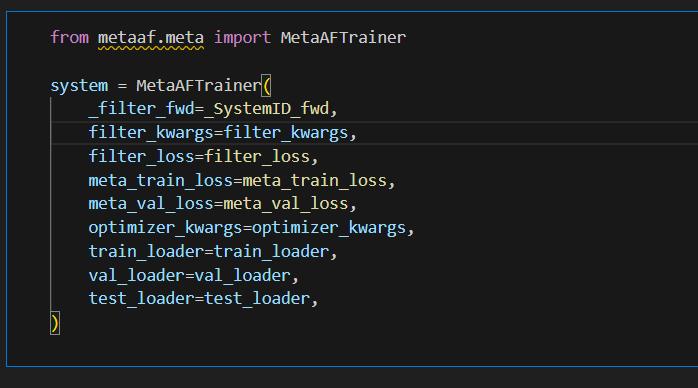
When you make a metaaf dataset, you write a vanilla pytorch dataset. The dataset should not use jax and must return a dictionary with the keys "signals" and "metadata". We enforce this format since the "signals" are automatically segmented and buffered. In this example, we make a simple system identification dataset which returns signals but not metadata.

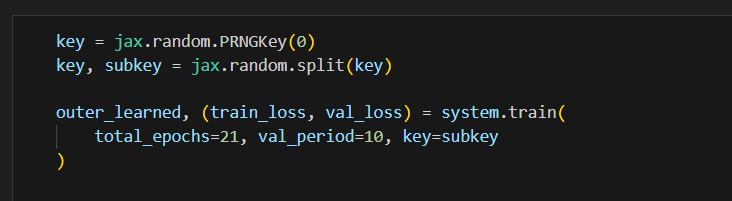


Next, we setup all filter keyword arguments. These will be used by the OLS baseclass to correctly buffer inputs and run the online STFT processing. These can also be done via argparse, since all metaaf modules have argparse utilities. Here, we use dictionaries for simplicity.

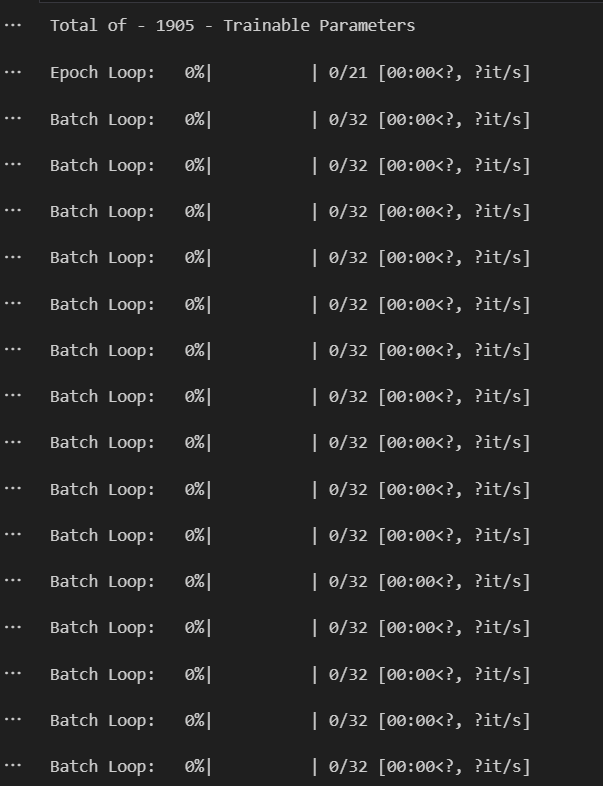


Now, we create a metaaf system. This system manages the training and will later provide inference utilities. We need to pass it the forward functions, losses, keyword arguments as well as the dataloaders. We'll set some optimizer options. For more advanced functionality, we can write our own forward passes, overridde other options, and even pass in training callbacks. These could do things like save checkpoints, log outputs, and more.



Train

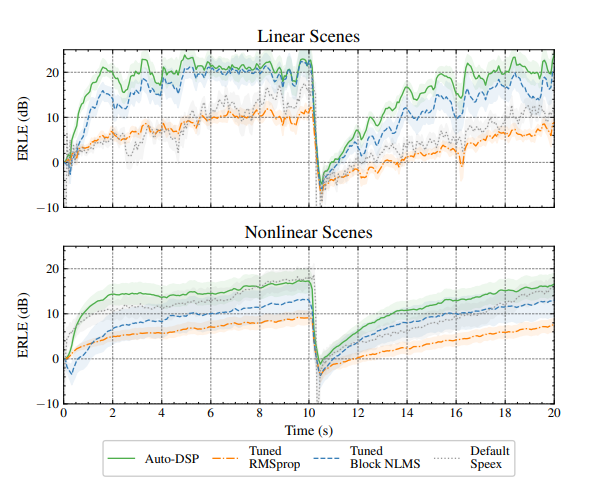
And the parameters are :



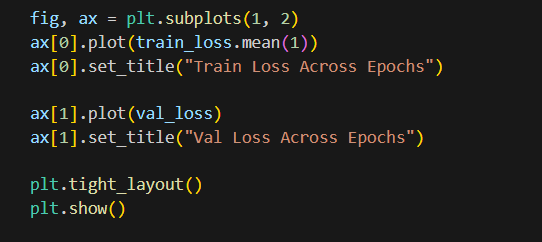
RESULTS

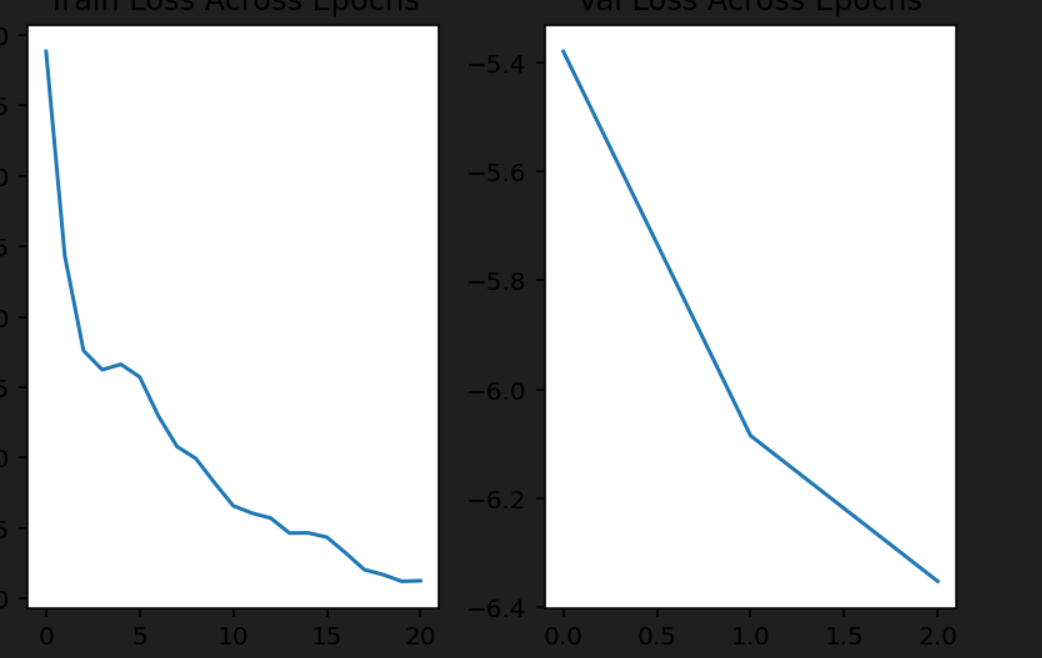
We evaluate our learned optimizers across multiple optimizer configurations and optimize configurations as well as linear and nonlinear scenes and compare against standard hand-derived update rules. Our baselines consist of step-size tuned block frequencydomain NLMS optimizer with smoothing constant (β = .9, .99), a step-size tuned frequency-domain RMSprop optimizer, and the Speex AEC.

While Speex is representative of a well-engineered hand-tuned optimizer it was not optimized for this dataset whereas the other optimizers are. Baseline results are shown in the first section of Table 1. We denote the learned optimizer hidden size by H, the number of filter parameters as P, the FFT size as N, the number of MDF blocks as M, the overlap between blocks as R, whether the optimize has a nonlinear component with γ, and provide both the average µ and standard deviation σ ERLE. All baseline and learned optimizes have an effective filter length of L = 2048 taps



RESULTS in META





Conclusion

The optimization of AEC systems using machine learning techniques holds great promise for improving the performance and robustness of AEC algorithms. By leveraging large datasets and advanced machine learning algorithms, it is possible to automate the optimization process and achieve superior performance in a wide range of acoustic environments. As research in this area continues to advance, we can expect to see the integration of Auto-DSP techniques into commercial AEC products, leading to enhanced audio quality and user experience in telecommunication and audio conferencing applications

.