AUDIO ENGINEER IN A BOX: USING DDSPS IN A COMPACT NEURAL NETWORK

Constantin von Estorff (471152), Alexander Krause (382022)

Technische Universität Berlin
Deep Learning for Audio Data (WiSe 2023/24)
Lecturers: Corvin Jaedicke, Fabian Seipel
Date: 31th of March 2024

ABSTRACT

This paper proposes a compact neural network for real-time live music enhancement on resource-constrained devices. For this purposes, we attempt to adapt a style-transfer model of music recordings to a live framework which which integrates differentiable digital signal processing (DDSP) modules. Unlike traditional methods using auto-encoders, music signals bypass the need for embedding and reconstruction, allowing for direct processing. This reduces model size and streamlines the inference pipeline. DDSP modules are essential, enabling gradient descent for network training. We train on a synthetic dataset replicating a live music venue's acoustics using impulse responses (IRs) from a club's PA system and smartphone recordings, simulating a concert experience. This approach ensures the network generalizes effectively to real-world scenarios with complex room acoustics and audience noise captured by smartphones. Our compact architecture with DDSP control offers a promising solution for real-time live music enhancement on embedded devices.

Index Terms— Music recording enhancement, Differentiable DSPs, Neural Network, Audio Processing

1. INTRODUCTION

To this date, real-time signal-enhancing, particularly of live music, on devices with limited computational resources presents a significant challenge [1] [2]. This paper introduces a compact neural network framework specifically designed for real-time live music enhancement on resource-constrained devices. Our approach diverges from traditional autoencoder-based methods by employing a combination of lightweight neural network control and differentiable audio effects. We have undertaken training using a synthetic dataset designed to replicate the acoustic conditions of live music venues. Hence, we aim to ensure that our neural network can generalize effectively to real-world scenarios, encompassing a wide range of room acoustics and audience noises typically captured by smartphones.

2. BACKGROUND

2.1. Inspiration

Our work draws on the *DeepAFx* framework, which integrates audio effect plugins into deep neural networks for flexible sound manipulation [3], [4]. The idea is thereby to integrate regularly implemented audio effect plugins into deep neural networks, enabling flexible, intelligable and quick manipulation of sound within the network architecture. Building on this, our project, "Audio Engineer in a Box," aims to refine this approach for real-time applications, focusing on reducing model size and computational overhead for seamless use in portable devices, thereby enhancing accessibility and usability for live music setups. However, while *DeepAFx* primarily targets producing setups, our approach endeavors to adapt and optimize this methodology for real-time application, particularly in live music environments.

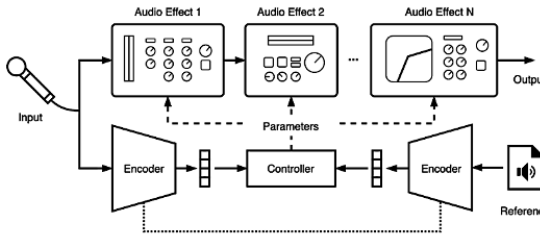

Fig. 1. Schematic overview over the DeepAFx approach, as depicted in [4]

Figure 1 gives an overview of the utilized approach. Fundamentally, it aims to feed a neural network with an input and a reference signal which are then being parallel sent to respective encoders. The aim is thereby to extract features from both signals which are then concatenated and being sent to a controller which in turn reduces the information to a previously defined count of parameters which are being then fed into DDSP processors. We used a parametric EQ and a dynamic range compressor which we implemented to fit into a Tensorflow framework. The input signal will then be pro-

cessed in order to match the 'style' of a reference input. Our contribution to the development of this model is to conceptualize the input signal as live-recordings which feature poor recording quality, bad mixing and gain-regulation as well as no accounting for acoustic flaws. The reference signal on the other hand represents clean, professionally produced live-recordings directly derived from mixers or stage recording set-ups. Similar to an auto-encoder, the loss is then calculated as the difference of the enhanced 'dirty' live-recording input and a ground-truth of its 'clean' version. Since the input data are self-generated and try to simulate live-environments, the ground truth and the reference can be generated from different parts of the same track.

2.2. Differentiable Digital Signal Processing

DDSP incorporates known signal models into neural networks, allowing the back-propagation of loss gradients through these DSP-parameters [5]. Advantages of DDSPs include their ability to model audio effects and processes more accurately and intuitively by directly manipulating sound's fundamental properties, thus overcoming limitations of traditional neural audio synthesis methods influenced by model biases [6]. By integrating DSP operations with neural networks, DDSPs allow for the development of highly interpretable, efficient, and accurate audio processing tools, setting a new standard for automated audio production and creative sound design. [3]

To integrate DSP operations within neural networks, three options have been established. First, automatic differentiation enables straightforward integration of differentiable mathematical operations within neural networks, although this approach demands expert knowledge and can introduce challenges like vanishing/exploding gradients [7]. Secondly, neural proxies emulate the signal processing behavior, including parameter control. This method is inherently differentiable and facilitates automatic effect control but demands accurate emulation and can be computationally intensive. Lastly, Simultaneous Perturbation Stochastic Approximation (SPSA) is a stochastic gradient approximation method used for efficient optimization of multivariate systems without requiring the gradient of the input signal. Compared to neural proxies or automatic differentiation, SPSA offers benefits in terms of computational efficiency and runtime performance, especially in real-time applications. Methods relying on neural proxies or automatic differentiation often incur higher computational costs due to the use of neural networks, both during inference and training. In contrast, SPSA, utilizes the DSP device solely during inference, resulting in lower computational overhead. Moreover, SPSA provides interpretable control without requiring explicit differentiation or re-implementation of the underlying system, making it suitable for our aim of real-time optimization tasks [3], [4] [8].

SPSA perturbs all parameters simultaneously, significantly reducing computational overhead. The SPSA gradient estimator approximates the partial derivative with respect to each parameter by evaluating the function at perturbed parameter values. Specifically, for parameter $\hat{\theta}_0)$, the ith element of ∇^{SPSA} is

$$\nabla^{SPSA} f(\hat{\theta}_0)_i = \frac{f(\hat{\theta}_0 + \epsilon\Delta^P) - f(\hat{\theta}_0 - \epsilon\Delta^P)}{2\epsilon\Delta_i^P}, \quad (1)$$

where ϵ represents a small, non-zero value, and Δ^P denotes a random vector drawn from a symmetric Bernoulli distribution [9][10].

While this method may require careful tuning of hyperparameters and learning rates to mitigate training instability [4], its computational efficiency and runtime performance may be an attractive choice for adapting to real-time optimized applications in audio signal processing.

3. SYNTHESIZED TRAINING DATA

3.1. Self-Supervised Learning

In this section, we describe our method of generating synthesized training data for our model, leveraging self-supervised learning (SSL) and a curated dataset. Our goal is to closely emulate the application's real-world use case, aiming for high specificity tailored to its requirements. SSL significantly reduces the need for labeled datasets by utilizing unlabeled data, offering a cost-effective strategy for learning useful data representations, thus decreasing reliance on extensive manual labeling and enhancing model robustness against adversarial examples and label corruption. This approach thus ultimately leads to a more compact network design [11] ahd potentially surpasses fully supervised methods in certain aspects [12, 13]. A primary challenge in SSL is the complexity of creating effective pretext tasks that ensure the learned representations are useful for downstream applications. There is also a risk of the model learning ineffective representations if the pretext tasks are not well-aligned with the ultimate objectives.

3.2. Datasets

Our model's development for enhancing live music recordings utilizes a dataset specifically designed to mirror the acoustic challenges present in live music environments. The dataset comprises audio from two primary sources, chosen for their distinct characteristics.

Jamendo [14]: With around 18,000 tracks of varied musical genres and styles, providing a wide spectrum for training despite variable production quality.

MUSDB18 [15]: This dataset, containing 150 high-quality tracks, is used to further increase the variance in our model's training data, enriching its learning environment.

Club Simulation IRs [16]: Initially, we aimed to synthesize data by combining frequency responses with IRs of

live music venues to replicate the full acoustic pathway of live settings, from loudspeaker outputs to venue ambiance, as captured by smartphone microphones. However, accessing suitable frequency response data proved difficult. Our search led us to the Club Simulation dataset, which resolved this issue by providing five IR files that encapsulate both the speaker frequency responses and the room's acoustic characteristics.

iPhone SE Recordings: In addition to the Club Simulation IRs, we recorded impulse responses with an iPhone SE to include real-world recording conditions in our dataset. These recordings were made in two environments: a bedroom and an anechoic chamber, producing ten IR files. This step was taken to better represent the capture of live music through common smartphones across different settings.

Together, these datasets form a solid foundation for training our model. However, it's important to note that with 1827 sampled songs for training and an additional 300 for testing, our coverage of scenarios is somewhat limited by the use of a single room environment in the Club Simulation IRs. Despite this limitation, our assembled datasets ensure the model is equipped to improve live music recordings across a variety of conditions.

3.3. Data Generation

For the creation of our training and testing datasets, we began by extracting 5-second segments from the Jamendo and MUSDB18 datasets, ensuring these segments were not silent to maintain content richness. This initial selection aimed to capture a wide array of musical expressions and audio qualities present in live music scenarios. Following the selection, these files underwent a stylization process to artificially introduce audio quality variations akin to those encountered in real-world live recording environments.

Audio production styles which were proposed by Steinmetz et al. [4] where then adopted to simulate various "dirty" sound characteristics often resulting from inadequate recording conditions. The transformation was achieved using a combination of Parametric EQ and compression to generate what we term "input data". Simultaneously, the original, unaltered segments—"reference data" — and the newly style-transformed segments were both convoluted with Impulse Responses (IRs) to further simulate live music venue acoustics. This convolution process ensures that both sets of data share common room-acoustical properties, differing solely in the artificially induced audio quality degradations.

4. MODEL ARCHITECTURE

Figure 2 gives an overview over our developed neural network. The network expects two audio samples as an input, of which both have 60.000 samples with a reduced sample rate of 24.000 *Hz*. It then propagates the input and reference signal through an encoder and then send them to a controller network yields parameters to ultimately apply to the DDSPs.

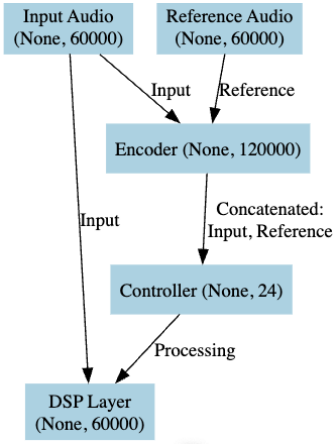

Fig. 2. Overall Structure of the network configuration

Encoder: The encoder is a small shared-weight convolutional neural network. It takes the 'dirty' input data and its 'clean' reference in order to transform them into mel-spectograms. Both spectograms are then being sent through multiple convolutional and pooling layer, and eventually batch-normalized before being concatenated and passed to the controller.

Controller: The controller was designed using a multi-layer perceptron architecture, utilizing the PReLU activation function [17] and incorporating dropout techniques to enhance its ability to generalize.

DDSP-Layer: The 'dirty' input signal is eventually being processed in the differentiable equalizer and the compressor, using the parameters which are yielded by the controller. The gradients are being calculated in a SPSA routine which we adopted from Ramirez et al. [3]. The output of the DDSP-Layer is then eventually an enhanced live-recording of the input-snippet.

Loss: The multi-resolution spectral amplitude distance [7] involves calculating the magnitude spectrogram for both the original and generated audio for a range of FFT sizes. The loss for a given FFT is calculated as the sum of the L1 norm of the difference between their logarithmic representations. The overall loss is the aggregation of these spectral losses across all considered FFT sizes. We transformed the FFTs into Mel-Spectograms before calculating loss. This ensures that the model learns to accurately reproduce coarse/global as well as fine/local features, i.e. it penalizes errors at multiple levels of detail, stabilizing the training process and improving gradient flow. Like [4], we also added a standard L1-loss.

4.1. Tuning and Training

We performed a hyper-parameter grid-search tuning with 25 trails and minimizing validation loss as an objective. We in-

cluded learning rate, FFT-Length for Mel-spectograms, the optimizer, L1-to-Multiresolutional Loss ratio and the number of hidden layers in the MLP as hyperparameter. Naturally, the tuning tended towards a bigger network. Thus, we trained a small (Total parameter: 82328, 321.59 KB, 256 hidden dimensions) and a larger (Trainable parameter: 4094936, 15.62 MB, 32 hidden dimensions) network.

We then trained two models for 150 epochs (small model) and 45 epochs (larger model) with a batch size of 64, and a dataset of ca. 1800 song snippets. We used a learning rate of 1.9*10-3 without decay in a SDG-optimizer. The training was performed parallel on a Apple M2 Pro 12-core CPU.

5. RESULTS

Assessing the quality of music is a complex task, especially when the original files are modified by room responses. To evaluate the performance of our model, we the metrics Mel Spectral Distance, RMS Energy Error, Spectral Centroid Error, and Loudness Error. These metrics allow us to quantify the differences between the predicted and target tensors, providing insights into the effectiveness of our approach.

Additionally, we benchmark our results against those reported by Steinmetz et al.[4] to gauge the model's performance. Furthermore, we compare the performance of our larger and smaller models and evaluate their computational efficiency to ensure practicality in real-world applications.

5.1. Evaluating Metrics

Metric	Small Model	Big Model	Deep-Afx
Mel Spectral Distance	38.08	37.03	2.15
RMS Energy Error	6.27	7.60	2.17
Spectral Centroid Error	202.4	243.8	144.7
Loudness Error	2.63	3.19	1.05

Table 1. Mean Metrics Distribution: Result overview of our small and big model, together with the synthetic production style-transfer test-results from [4], evaluation on the Jamendo dataset with Sr = 24 kHz, using the SPSA approach.

Evaluating our models against key audio quality metrics, the results are promising but highlight areas for improvement. As shown in Table 1 and Fig. 4 5, the Small Model achieves better Mel Spectral Distance and RMS Energy Error scores than the Big Model, indicating that it adapts the original audio's spectral properties and dynamic range more closely. However, the Spectral Centroid Error and Loudness Error are higher for both models compared to the Deep-Afx model, pointing to a need for better spectral and loudness consistency in future model iterations to focus on enhancing audio fidelity.

5.2. Quality of Predictions

Subjective auditory assessments indicate the Small Model surpasses the Big Model in audio quality, supported by quantitative evidence in Table 1. This alignment of perception with data demonstrates the robustness of our evaluation, affirming the Small Model's superior audio reproduction. However, neither model fully achieves the desired audio enhancement level, notably when DDSP modules are underutilized. This observation is further illustrated by the spectral closeness in the predicted versus reference audio, as seen in Figure 6.

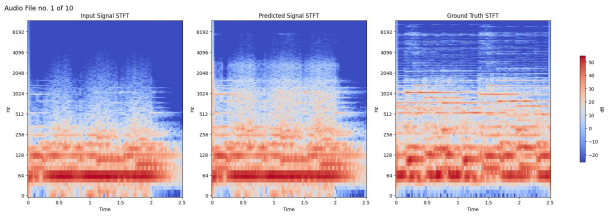

Fig. 3. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Small Model (Sample 1)

Looking at computational performance (see Table 2), both models operate within a similar time range. It's observed that the Small Model operates more efficiently when using a direct approach to control parameter application, which could have implications for real-time audio processing applications. In the appendix, Figures 6-17 additionally show more model comparisons on selected examples.

This progress suggests that our models are on the right track, with room for improvements that could potentially meet the requirements for enhanced real-time audio processing in future updates.

Prediction time [ms]	Small Model	Big Model
Main model	91.1	93.1
Analyzer model	89.7	92.5

Table 2. Prediction Times Overview: timing benchmarks run on 2014' Macbook Pro (8gb ram and i5-cpu).

6. DISCUSSION

6.1. Achievements

One of the primary accomplishments of this research is the establishment of a neural network framework that integrates DDSP modules to facilitate real-time audio processing on devices with limited computational capabilities. This represents a significant departure from traditional approaches, which often rely on autoencoder architectures. The empirical evidence supporting the superior audio quality reproduction of the Small Model over the Big Model, as delineated in

the metrics outlined in Table 1, affirms the efficacy of our proposed model architecture. Moreover, the discovery of an efficient processing route via the direct application of control parameters, particularly evident in the Small Model's performance, suggests a viable strategy for optimizing real-time computational efficiency.

6.2. Limitations

Despite the strides made, our study is not without its limitations. The absence of baseline testing against conventional models constrains our ability to contextualize the improvements fully. Additionally, the study does not include a detailed comparison of various differentiation methodologies, which may have unveiled more efficient or effective strategies for embedding DSP within neural networks. The limited application of data augmentation techniques may also curtail the model's generalizability across diverse live audio scenarios.

6.3. Future Directions

Looking ahead, the insights garnered from this study open up several intriguing avenues for future research. Enriching the model's capacity for style-transfer to autonomously adjust to different audio production styles could revolutionize personalized audio enhancement. This development would involve formulating a classifier capable of selecting the most fitting 'clean' reference based on the audio input's specific attributes. Addressing the aforementioned limitations by incorporating baseline comparisons, exploring a variety of differentiation methods, and employing advanced data augmentation strategies will further validate the model's robustness and extend its applicability. Continuous refinement of the model's utilization of DDSP modules is also anticipated to enhance the network's audio processing prowess, ensuring top-tier audio enhancement in real-world scenarios.

For an in-depth exploration of our findings and methodologies, interested readers are encouraged to consult the metrics notebook available in our repository and the additional materials included in the appendix section.

7. CONCLUSION

We introduced a compact neural network leveraging Differentiable Digital Signal Processing (DDSP) for enhancing live music in real-time on devices with limited resources. Our method diverges from traditional approaches by directly processing music signals, reducing model size and improving inference speed. Trained on a synthetic dataset that simulates live music venue acoustics, our models demonstrate adaptability to real-world acoustic scenarios.

Our research highlights the potential of integrating DDSP modules within neural networks for real-time audio processing. The Small Model, in particular, shows promising results in balancing audio quality enhancement and computational efficiency. However, future work is necessary to address current limitations and explore additional differentiation methods and data augmentation strategies. As this field progresses, we aim to further enhance the capability of live music enhancement technologies.

8. REFERENCES

[1] Jhonatan Geremias, A. Santin, E. Viegas, and A. Britto, "Towards real-time video content detection in resource constrained devices," *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2020.

[2] Ramyad Hadidi, Jiashen Cao, M. Woodward, M. Ryoo, and Hyesoon Kim, "Musical chair: Efficient real-time recognition using collaborative iot devices," *ArXiv*, vol. abs/1802.02138, 2018.

[3] Marco A. Martínez Ramírez, Oliver Wang, Paris Smaragdis, and Nicholas J. Bryan, "Differentiable signal processing with black-box audio effects," 2021.

[4] Christian J Steinmetz, Nicholas J Bryan, and Joshua D Reiss, "Style transfer of audio effects with differentiable signal processing," *arXiv preprint arXiv:2207.08759*, 2022.

[5] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts, "Ddsp: Differentiable digital signal processing," *arXiv preprint arXiv:2001.04643*, 2020.

[6] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon, "Differentiable rendering: A survey," *arXiv preprint arXiv:2006.12057*, 2020.

[7] Xin Wang, Shinji Takaki, and Junichi Yamagishi, "Neural source-filter waveform models for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 402–415, 2019.

[8] Ben Hayes, Jordie Shier, György Fazekas, Andrew McPherson, and Charalampos Saitis, "A review of differentiable digital signal processing for music & speech synthesis," *arXiv preprint arXiv:2308.15422*, 2023.

[9] James C Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE transactions on automatic control*, vol. 37, no. 3, pp. 332–341, 1992.

[10] James C Spall, "An overview of the simultaneous perturbation method for efficient optimization," *Johns Hopkins apl technical digest*, vol. 19, no. 4, pp. 482–492, 1998.

[11] Longlong Jing and Yingli Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 4037–4058, 2019.

[12] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang, "Self-supervised learning: Generative or contrastive," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 857–876, 2020.

[13] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *ArXiv*, vol. abs/1906.12340, 2019.

[14] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra, "The mtg-jamendo dataset for automatic music tagging," in *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States, 2019.

[15] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner, "The MUSDB18 corpus for music separation," Dec. 2017.

[16] interruptor.ch, "Club simulation - explanation and download page," year unknown, Accessed: 2024-03-22.

[17] Tongtong Jiang and Jinyong Cheng, "Target recognition based on cnn with leakyrelu and prelu activation functions," *2019 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC)*, pp. 718–722, 2019.

A. APPENDIX

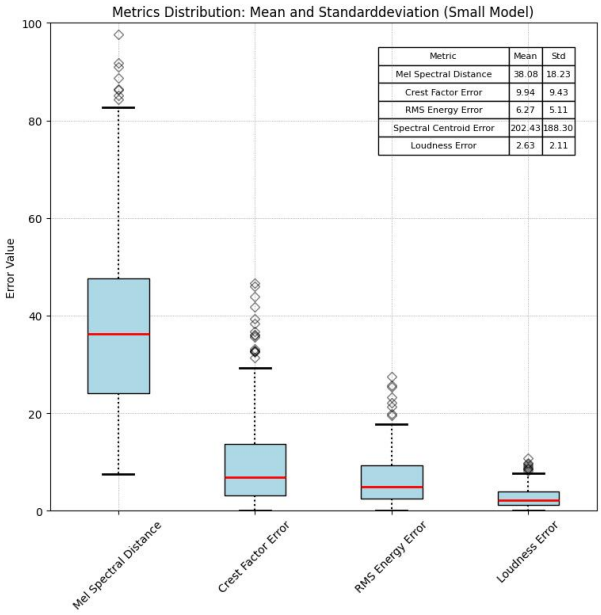

Fig. 4. Metrics Overview: Small Model

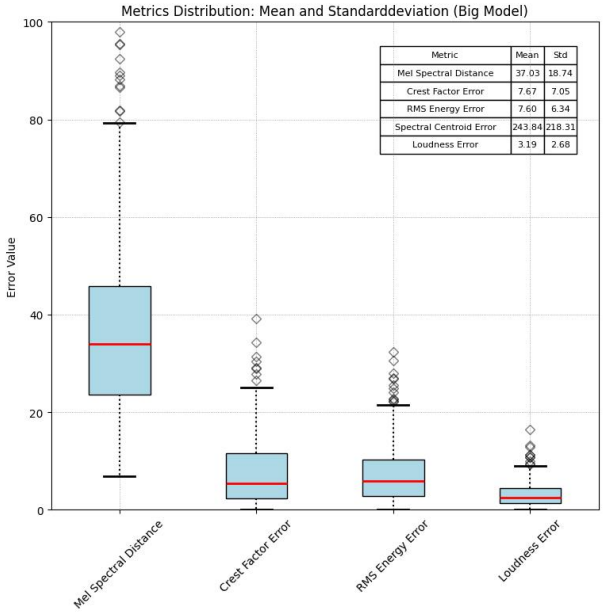

Fig. 5. Metrics Overview: Big Model

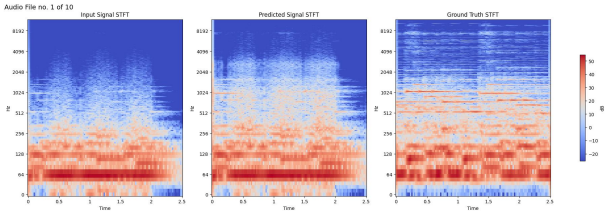

Fig. 6. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Small Model (Sample 1)

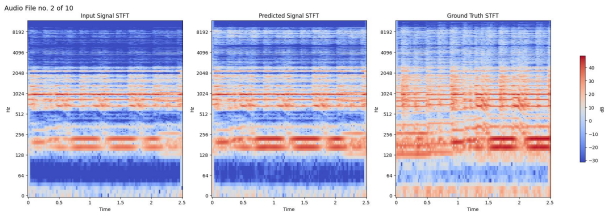

Fig. 7. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Small Model (Sample 2)

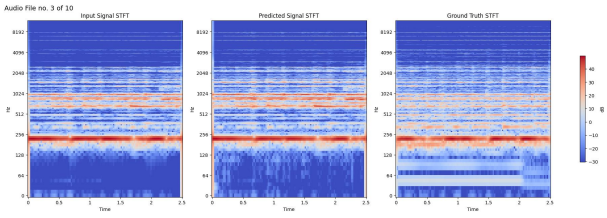

Fig. 8. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Small Model (Sample 3)

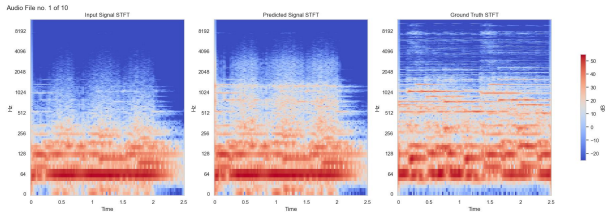

Fig. 9. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Big Model (Sample 1)

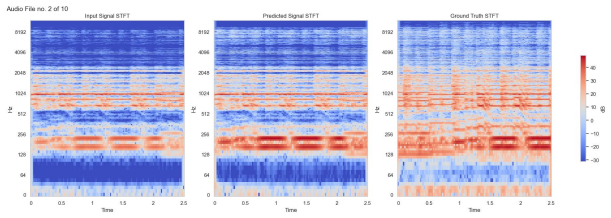

Fig. 10. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Big Model (Sample 2)

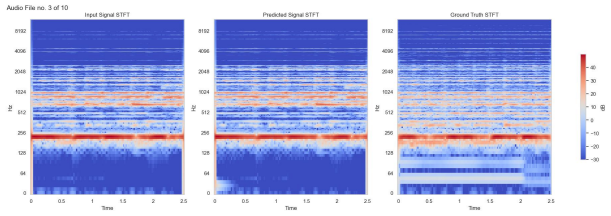

Fig. 11. STFT: 'Dirty' Samples (L), Predicted 'Clean' (M) and Ground Truth (L) for the Big Model (Sample 3)

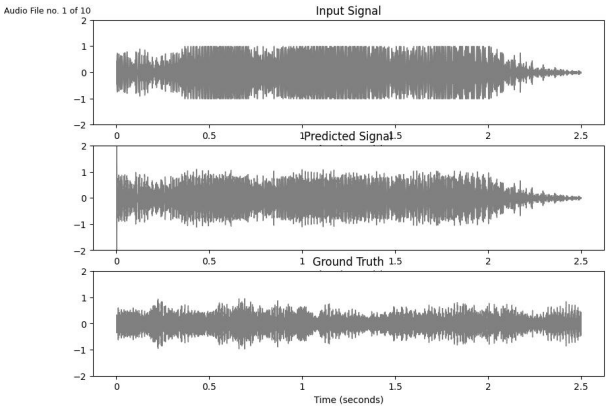

Fig. 12. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Samll Model (Sample 1)

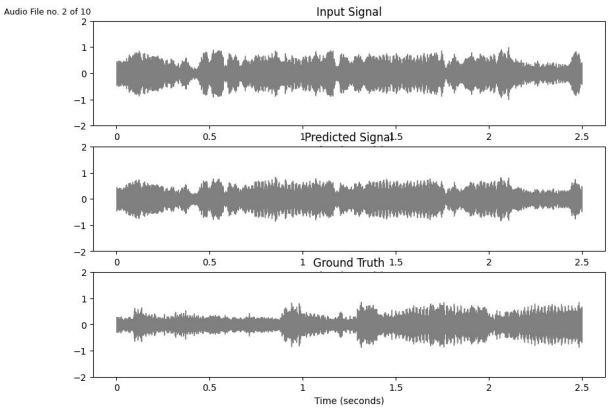

Fig. 13. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Small Model (Sample 2)

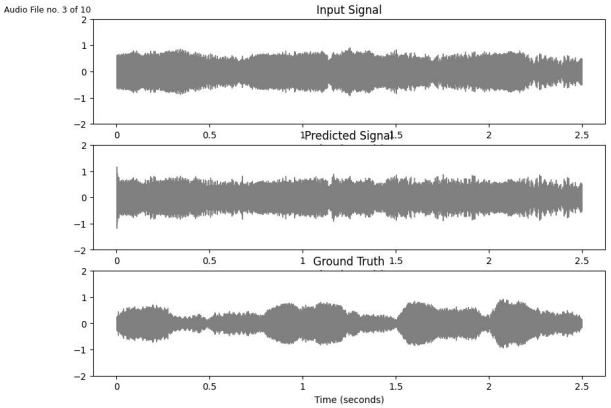

Fig. 14. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Small Model (Sample 3)

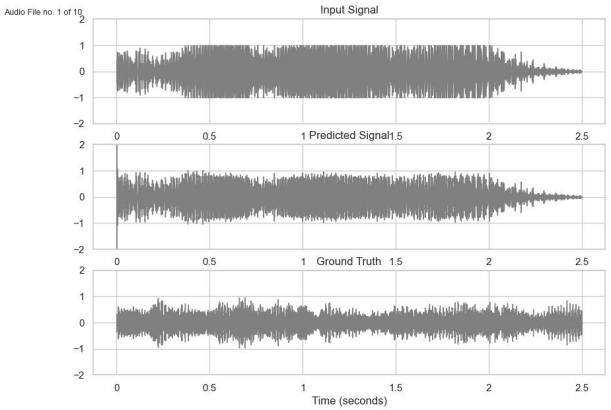

Fig. 15. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Big Model (Sample 1)

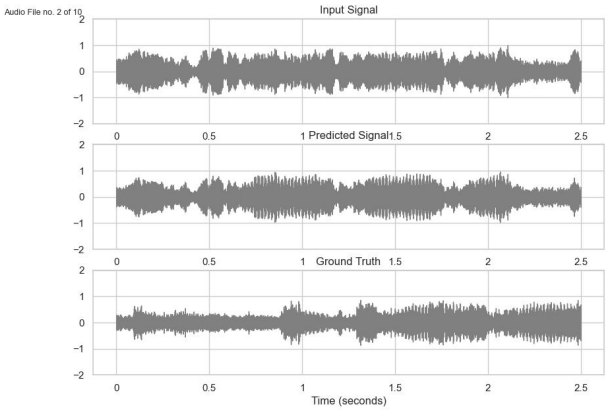

Fig. 16. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Big Model (Sample 2)

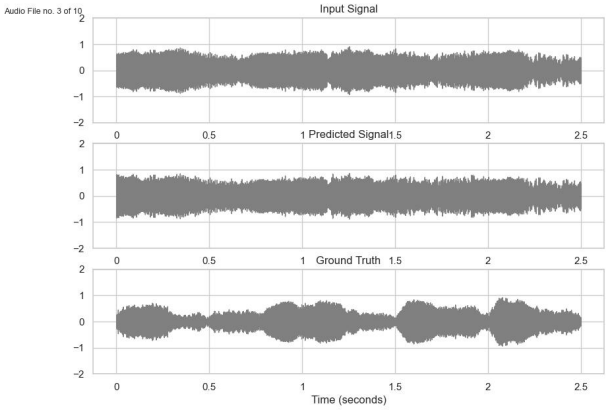

Fig. 17. Waveforms: 'Dirty' Samples, Predicted 'Clean', and Ground Truth for the Big Model (Sample 3)