社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

ファインチューニングを利用した歪みエフェクタの高速モデリング

少路　春希† 　吉本　健人† 　阪　大樹† 　黒田　大貴† 　北原　大地†

田中　賢一郎† 　平林　晃†

† 立命館大学大学院情報理工学研究科　〒 525–8577 滋賀県草津市野路東 1–1–1
E-mail: † is0413sx@ed.ritsumei.ac.jp, is0383ip@ed.ritsumei.ac.jp, is0410rr@ed.ritsumei.ac.jp,
kuroda-h@fc.ritsumei.ac.jp, d-kita@fc.ritsumei.ac.jp, ken-t@fc.ritsumei.ac.jp, akirahrb@media.ritsumei.ac.jp

あらまし　深層学習に基づく歪みエフェクタのモデリングを高速に行う手法を提案する．エフェクタの個体や設定を変えて何回もモデリングすることを考えると 1 回当たりの学習時間の短縮が求められるが，単に学習データ量を削減するとモデリング精度が低下してしまう．本研究では，目的の歪みエフェクタを少量データからモデリングする際に，異なる歪みエフェクタに対して十分訓練済みのネットワークパラメータを初期値としてファインチューニングを行う．数値実験で，提案手法が少量データからでも精度を維持したまま歪みエフェクタを高速にモデリングすることを示す．
キーワード　歪みエフェクタ，深層学習，WaveNet，転移学習，ファインチューニング

Fast Distortion Pedal Modeling with Fine-Tuning

Haruki SHOJI†, Kento YOSHIMOTO†, Daiki SAKA†, Hiroki KURODA†, Daichi KITAHARA†,
Kenichiro TANAKA†, and Akira HIRABAYASHI†

† Graduate School of Information Science and Engineering, Ritsumeikan University
1–1–1 Nojihigashi, Kusatsu, Shiga, 525–8577 Japan
E-mail: † is0413sx@ed.ritsumei.ac.jp, is0383ip@ed.ritsumei.ac.jp, is0410rr@ed.ritsumei.ac.jp,
kuroda-h@fc.ritsumei.ac.jp, d-kita@fc.ritsumei.ac.jp, ken-t@fc.ritsumei.ac.jp, akirahrb@media.ritsumei.ac.jp

Abstract　We propose a fast modeling method for distortion pedals based on deep learning. For modeling many times with different pedals and settings, it is desired to shorten the training time per one model, but simply reducing the amount of training data decreases the modeling accuracy. In this paper, when modeling a target distortion pedal from a small amount of data, we propose to apply fine-tuning where network parameters well-trained for another distortion pedal are used as initial values. Numerical experiments show that the proposed method trains the model of the target distortion pedal very quickly from a small amount of data while maintaining the modeling accuracy.
Key words　Distortion Pedal, Deep Learning, WaveNet, Transfer Learning, Fine-Tuning

1. はじめに

エレキギターなどの電気信号を出力する楽器を演奏する際，演奏音に豊かな表現力を持たせるために様々なエフェクタが使用される．特に，ギターアンプへ大音量で音を入力した場合と似た歪み効果をもたらすエフェクタは広く用いられており，アーティストは好みの音色を作るために複数の個体を繊細に使い分ける．その中でも，“ヴィンテージ” と称される 1970〜1980 年代に生産された製品は人気が高いが，生産が既に終了しており入手しづらい．こうした製品の保存や利便性向上を目的とし，アナログエフェクタの出力音をデジタルデバイスで忠実に再現するデジタルモデリング技術が研究されている [1–9].

デジタルモデリング技術は 2 種に大別できる．1 つは，機器の電子回路の各素子を数理的にモデル化する手法である [1, 2]. この手法は，高精度なモデリングが可能であるが，エフェクタの回路図だけでなく，トランジスタ，ダイオード，真空管などの素子の非線形特性も必要とするため，作業コストが極めて高い．もう 1 つは，機械学習やシステム同定によって入出力音の関係から全体の特性をモデリングする手法 [3–9] である．特に近年は，エフェクタの回路の情報を一切必要としないブラックボックスモデリングと呼ばれる手法が盛んに研究されている [6–9]. ブラックボックスモデリングでは，入力音とそれに対応するエフェクタの出力音さえ取得しておけばよいため，回路ベースのモデリング手法と比較して作業コストを大幅に低減できる．

ブラックボックスモデリングとして，WaveNet [10] を用いた手法が Damskägg らにより提案されている [7,8]．WaveNet では，畳み込み層の工夫によって，計算量を抑制しながら広範囲の入力音を用いて出力音を高精度に再現することが可能となる．Damskägg らの手法で高精度なモデリングを行うには計 300 秒程度の入出力音データを収録する必要があり，更に WaveNet の学習にも 4 時間程度を要する．エフェクタモデリングでは，ユーザごとの好みに応じた多種多様なモデリングが求められるため，1 回のモデリング時間をより短くすることが重要である．

モデリングに用いる入出力音データを計 30 秒程度にすれば収録時間と学習時間の両方を短縮できるが，この程度の少量データからの学習では出力音の再現が不十分となってしまう．そこで本研究では，精度を維持しつつモデリングを高速化するために，転移学習を利用することを提案する．転移学習とは，類似タスクに対して十分に訓練されたニューラルネットワークのパラメータを何らかの形で目的タスクに流用することである．一般に，ネットワークの低層部分では類似タスクとも共通する普遍的特徴が学習される．したがって，低層部分のパラメータを類似タスクの訓練済みモデルのもので代用することで，目的タスクの学習データが比較的少ない場合でも高品質なモデルが形成可能になる．転移学習の中でも，訓練済みモデル全体を転移した後に目的タスクの学習データを用いてパラメータを微調整する手法はファインチューニングと呼ばれる．この手法ではネットワークの高層部分にも初期値を与えるため，高層部分をランダム初期値とする場合と比べて学習が更に高速化される．

本研究では，目的の歪みエフェクタをモデリングする際に，異なる歪みエフェクタに対して訓練済みの WaveNet を予め用意しておき，ファインチューニングを適用することで計 30 秒程度の少量データから高速高精度なモデリングを実現する．また，ファインチューニングを適用する際に WaveNet の低層部分のパラメータを完全に固定することで，更新パラメータ数を減少させ，WaveNet の学習時間をより一層短縮する．歪みの強いエフェクタである Ibanez 社製 SD9 と歪みの弱いエフェクタである Ibanez 社製 TS MINI を用いた数値実験によって提案する高速モデリング手法の有効性を示す．

2. 深層学習を用いた歪みエフェクタモデリング

2.1 WaveNet によるモデリング手法

本論文では，文献 [7–9] に基づいた WaveNet によってモデリングを行う．WaveNet は元々，直前の過去 R 個の信号から次の信号を予測することを繰り返して音声や音楽などを生成する自己回帰型ネットワークであった [10]．Damskägg らはこれを入力音を歪み音に変換する図 1 のような順伝播型モデルに修正した [7,8]．以下では，アナログエフェクタの入力音と出力音を同期してサンプリングした離散信号をそれぞれ $x[n]$ と $y[n]$ とし，WaveNet に $x[n-R+1], x[n-R+2], \ldots, x[n]$ を入力した際に出力されるモデリング音を $\hat{y}[n]$ $(n = 1, 2, \ldots, N)$ とする．

最初に，Pre-Processing Layer (第 0 層) が 1 チャネルの入力信号 $x[n]$ から

$$\boldsymbol{x}_0[n] = \boldsymbol{w}_0 x[n] + \boldsymbol{b}_0 \tag{1}$$

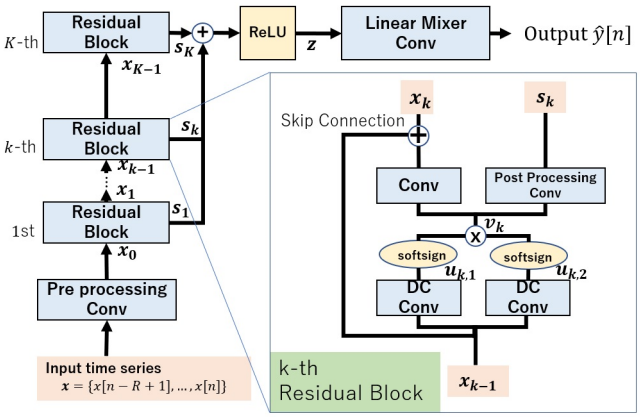


図 1　順伝播型 WaveNet のネットワーク構造の概要

のように L チャネルの信号 $\boldsymbol{x}_0[n] \in \mathbb{R}^L$ を出力する．ここで，$\boldsymbol{w}_0 \in \mathbb{R}^L$ はサイズ 1 の畳み込みフィルタ，$\boldsymbol{b}_0 \in \mathbb{R}^L$ はバイアス項である．式 (1) のパラメータをまとめて直積 $\theta_0 = (\boldsymbol{w}_0, \boldsymbol{b}_0)$ で表せば，Pre-Processing Layer のパラメータ数は $2L$ 個である．

次に，L チャネルの信号 $\boldsymbol{x}_0[n]$ は K 個の残差ブロックを通る．$k = 1, 2, \ldots, K$ とすると，第 k 層の残差ブロックの 2 つの出力 $\boldsymbol{x}_k[n] \in \mathbb{R}^L$ と $\boldsymbol{s}_k[n] \in \mathbb{R}^L$ は入力信号 $\boldsymbol{x}_{k-1}[n] \in \mathbb{R}^L$ から以下のように計算される．まず，入力 $\boldsymbol{x}_{k-1}[n]$ に対して，dilated causal (DC) convolution と呼ばれる演算を 2 通り実行する．

$$\begin{cases} \boldsymbol{u}_{k,1}[n] = \displaystyle\sum_{m=0}^{M} W_{k,1}[m] \boldsymbol{x}_{k-1}[n - m d_k] + \boldsymbol{b}_{k,1} \\ \boldsymbol{u}_{k,2}[n] = \displaystyle\sum_{m=0}^{M} W_{k,2}[m] \boldsymbol{x}_{k-1}[n - m d_k] + \boldsymbol{b}_{k,2} \end{cases} \tag{2}$$

ここで，$W_{k,1}[m], W_{k,2}[m] \in \mathbb{R}^{L \times L}$ $(m = 0, 1, \ldots, M)$ はサイズ $M+1$ の畳み込みフィルタ，$\boldsymbol{b}_{k,1}, \boldsymbol{b}_{k,2} \in \mathbb{R}^L$ はバイアス項，d_k は dilation 係数であり，本研究では $d_k = 2^{(k-1) \bmod 9}$ のように設定する．ただし，mod は剰余演算を表す．直前の過去 R 個の入力から通常の畳み込み演算を行う causal convolution では，受容野 R を大きくする際に多くの層数を必要とするため，計算時間と学習時間がどちらも長くなってしまう．入力音をダウンサンプリングして計算量を削減することも考えられるが，この場合はエイリアシングの問題が新たに生じてしまう [11]．一方，DC convolution では，式 (2) のように一定間隔 d_k で間引いた特徴のみから畳み込み演算を行う．出力計算に用いる受容野の大きさは $R = M(\sum_{k=1}^{K} d_k) + 1$ となるため，層数 K に対して受容野 R が効率的に拡大される．結果として，計算時間と学習時間の大幅な抑制と出力値の高精度な制御の両立が可能になる．

DC convolution の 2 つの結果 $\boldsymbol{u}_{k,1}[n], \boldsymbol{u}_{k,2}[n] \in \mathbb{R}^L$ から，gated activation unit が以下のように $\boldsymbol{v}_k[n] \in \mathbb{R}^L$ を計算する．

$$\boldsymbol{v}_k[n] = g\left(\boldsymbol{u}_{k,1}[n]\right) \odot g\left(\boldsymbol{u}_{k,2}[n]\right) \tag{3}$$

ここで，$\odot$ はアダマール積，g は活性化関数で文献 [8,9] に倣い $g(x) = \mathrm{softsign}(x) = \frac{x}{1+|x|}$ とする．第 k 層の出力 $\boldsymbol{x}_k[n]$ は

$$\boldsymbol{x}_k[n] = W_{k,3} \boldsymbol{v}_k[n] + \boldsymbol{b}_{k,3} + \boldsymbol{x}_{k-1}[n] \tag{4}$$

のように計算され，もう一方の出力 $\boldsymbol{s}_k[n]$ は

$$\boldsymbol{s}_k[n] = W_{k,4}\boldsymbol{v}_k[n] + \boldsymbol{b}_{k,4} \tag{5}$$

のように計算される．ここで，$W_{k,3}, W_{k,4} \in \mathbb{R}^{L\times L}$ はサイズ 1 の畳み込みフィルタ，$\boldsymbol{b}_{k,3}, \boldsymbol{b}_{k,4} \in \mathbb{R}^{L}$ はバイアス項である．式 (2), (3), (4), (5) に含まれるパラメータをまとめて直積 $\theta_k = (W_{k,1}[m], W_{k,2}[m], W_{k,3}, W_{k,4}, \boldsymbol{b}_{k,1}, \boldsymbol{b}_{k,2}, \boldsymbol{b}_{k,3}, \boldsymbol{b}_{k,4})$ で表せば，各残差ブロックのパラメータ数は $2(L^2(M+2)+2L)$ 個である．ただし，第 K 層の残差ブロックは $\boldsymbol{s}_K[n]$ のみを出力し，$\boldsymbol{x}_K[n]$ は出力されないため，パラメータ数が L^2+L 個少なくなる．

第 1〜K 層の出力 $\boldsymbol{s}_1[n], \boldsymbol{s}_2[n], \ldots, \boldsymbol{s}_K[n]$ は skip connection により結合され，その後 rectified linear unit (ReLU) を通過し，

$$\boldsymbol{z}[n] = \mathrm{ReLU}\left(\sum_{k=1}^{K} \boldsymbol{s}_k[n]\right) \tag{6}$$

のように信号 $\boldsymbol{z}[n] \in \mathbb{R}^{L}$ となる．ここで，$\mathrm{ReLU}(s) = \max\{0, s\}$ である．そして，最終層の Linear Mixer によって L チャネルの信号 $\boldsymbol{z}[n]$ から 1 チャネルのモデリング音 $\hat{y}[n]$ が出力される．

$$\hat{y}[n] = \boldsymbol{w}_{K+1}^{\top}\boldsymbol{z}[n] \tag{7}$$

ここで，$\boldsymbol{w}_{K+1} \in \mathbb{R}^{L}$ はサイズ 1 の畳み込みフィルタである．最終層のパラメータ数は L 個，つまり $\theta_{K+1} = \boldsymbol{w}_{K+1}$ である．

以上，WaveNet に含まれる全てのパラメータをまとめて直積 $\Theta = (\theta_0, \theta_1 \ldots, \theta_{K+1})$ で表せば，WaveNet のパラメータ総数は $2K(L^2(M+2)+2L) - L^2 + 2L$ となる．また，$\lambda = 0, 1, \ldots, K$ とし，第 λ 層までの低層部分とそれ以降の高層部分をそれぞれ

$$\begin{cases} \Theta[0:\lambda] = (\theta_0, \theta_1, \ldots, \theta_\lambda) \\ \Theta[\lambda+1:K+1] = (\theta_{\lambda+1}, \theta_{\lambda+2}, \ldots, \theta_{K+1}) \end{cases}$$

と記すこととする．式 (1) から式 (7) の計算をまとめた関数を f_Θ で表せば，WaveNet の入出力関係は以下のように表される．

$$\hat{y}[n] = f_\Theta(x[n-R+1], x[n-R+2], \ldots, x[n]) \tag{8}$$

最後に，パラメータ Θ の学習方法を説明する．学習データを $\mathcal{D} = \{(x[n], y[n])\}_{n=1}^{N}$ で表す．真の出力音 $y[n]$ とモデリング音 $\hat{y}[n]$ それぞれに対して，高周波成分の再現性を向上させるため，伝達関数が $H(z) = 1 - 0.95z^{-1}$ である高域強調フィルタを適用して $y_\mathrm{f}[n]$ と $\hat{y}_\mathrm{f}[n]$ を得る．損失関数として error-to-signal ratio (ESR) と呼ばれる $y_\mathrm{f}[n]$ と $\hat{y}_\mathrm{f}[n]$ の正規化平均二乗誤差を用い，これを最小化するように何らかの最適化手法で各パラメータの更新を繰り返す．一連の過程を写像 $\mathcal{A}$ として以下のように表す．

$$\mathcal{A}(\mathcal{D}, 0:K+1 \,|\, \Theta_\mathrm{ini}) \approx \operatorname*{argmin}_{\Theta} \frac{\sum_{n=1}^{N} |y_\mathrm{f}[n] - \hat{y}_\mathrm{f}[n]|^2}{\sum_{n=1}^{N} |y_\mathrm{f}[n]|^2} \tag{9}$$

ここで，$0:K+1 \,|\, \Theta_\mathrm{ini}$ は第 0 層から第 $K+1$ 層までの全てのパラメータを初期値 Θ_ini から更新することを意味し，$\approx \mathrm{argmin}$ は損失関数の値が下がり切った時点でのパラメータを意味する．通常は，ランダム初期値 Θ_rand から Θ を以下のように求める．

$$\Theta = \mathcal{A}(\mathcal{D}, 0:K+1 \,|\, \Theta_\mathrm{rand}) \tag{10}$$

2.2 学習データ量とモデリング精度の関係

モデリング目的のエフェクタを T で表し，エフェクタ T に計 300 秒程度の長い信号を入力した際の学習データを $\mathcal{D}_T^{(\mathrm{L})} = \{(x^{(\mathrm{L})}[n], y_T^{(\mathrm{L})}[n])\}_{n=1}^{N_{(\mathrm{L})}}$ で表す．$\mathcal{D}_T^{(\mathrm{L})}$ を用いて訓練した結果を

$$\Theta_T^{(\mathrm{L})} = \mathcal{A}(\mathcal{D}_T^{(\mathrm{L})}, 0:K+1 \,|\, \Theta_\mathrm{rand}) \tag{11}$$

で表す．以下の検証では，式 (9) の最適化手法に Adam [12] を，ランダム初期値 Θ_rand の生成には Glorot の初期化 [13] を用いた．学習が適切に行われれば，式 (8) によって得られるモデリング音 $f_{\Theta_T^{(\mathrm{L})}}(x[n-R+1], x[n-R+2], \ldots, x[n])$ は真の出力音 $y[n]$ を十分高精度に再現するが，式 (11) のパラメータ $\Theta_T^{(\mathrm{L})}$ を求める際に 4 時間程度の時間がかかる．エフェクタモデリングでは，個体や設定を変えて何回もモデリングを行うことが求められるため，1 回当たりの学習時間を更に短くすることが重要となる．

一方，エフェクタ T に計 30 秒程度の短い信号を入力した際のデータを $\mathcal{D}_T^{(\mathrm{S})} = \{(x^{(\mathrm{S})}[n], y_T^{(\mathrm{S})}[n])\}_{n=1}^{N_{(\mathrm{S})}}$ として，$\mathcal{D}_T^{(\mathrm{S})}$ から

$$\Theta_T^{(\mathrm{S})} = \mathcal{A}(\mathcal{D}_T^{(\mathrm{S})}, 0:K+1 \,|\, \Theta_\mathrm{rand}) \tag{12}$$

を求めると学習時間は 50 分程度で済む．しかし，モデリング音 $f_{\Theta_T^{(\mathrm{S})}}(x[n-R+1], x[n-R+2], \ldots, x[n])$ の再現精度が不十分となる．そこで次節では，大量データ $\mathcal{D}_T^{(\mathrm{L})}$ からのモデリング音 $f_{\Theta_T^{(\mathrm{L})}}(x[n-R+1], x[n-R+2], \ldots, x[n])$ と同程度の再現精度を少量データ $\mathcal{D}_T^{(\mathrm{S})}$ から達成する学習方法を提案する．

3. ファインチューニングによる高速モデリング

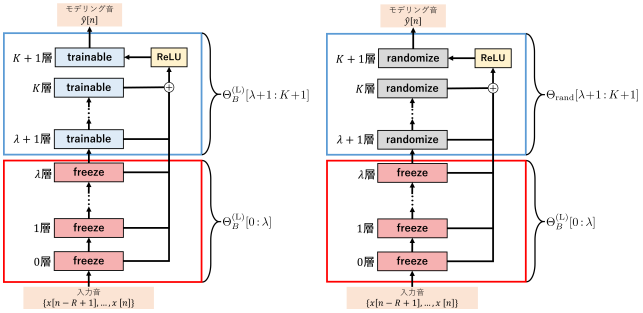
3.1 歪みエフェクタモデリングにおける転移学習

本研究では，精度を保ちながら歪みエフェクタモデリングを高速に行うために，転移学習を利用することを提案する．すなわち，異なるエフェクタ B に対して十分に訓練された WaveNet のパラメータを流用して，目的エフェクタ T を少量データ $\mathcal{D}_T^{(\mathrm{S})}$ から高速高精度にモデリングする．エフェクタ B に計 300 秒程度の信号を入力した際のデータ $\mathcal{D}_B^{(\mathrm{L})} = \{(x^{(\mathrm{L})}[n], y_B^{(\mathrm{L})}[n])\}_{n=1}^{N_{(\mathrm{L})}}$ から事前に訓練された WaveNet のパラメータを $\Theta_B^{(\mathrm{L})}$ とする．

$$\Theta_B^{(\mathrm{L})} = \mathcal{A}(\mathcal{D}_B^{(\mathrm{L})}, 0:K+1 \,|\, \Theta_\mathrm{rand}) \tag{13}$$

一般に，ニューラルネットワークの低層部分では類似タスクに共通する汎用的な処理が行われていると想定される．例えば画像処理であれば，エッジやその方向性などの特徴抽出に相当する．そこで，エフェクタ T の学習においても，第 0〜λ 層までの低層部分には式 (13) のエフェクタ B のパラメータ $\Theta_B^{(\mathrm{L})}[0:\lambda]$ をそのまま用い，残りの第 $\lambda+1$〜$K+1$ 層までの高層部分のパラメータ $\Theta[\lambda+1:K+1]$ のみを更新することを提案する．これにより，少量データ $\mathcal{D}_T^{(\mathrm{S})}$ からでは十分に学習できない汎用的な処理が補われるとともに，更新パラメータ数が削減され，式 (12) の $\Theta_T^{(\mathrm{S})}$ の場合よりも更に学習時間が短くなる．ここで，更新パラメータ数 $\Delta(\lambda)$ は λ に応じて以下のように変化する．

$$\Delta(\lambda) = \begin{cases} 2K(L^2(M+2)+2L) - L^2 & (\lambda = 0) \\ 2(K-\lambda)(L^2(M+2)+2L) - L^2 & (1 \leqq \lambda < K) \\ L & (\lambda = K) \end{cases} \tag{14}$$



(a) 高層のファインチューニング　　(b) 高層のランダム初期化

図 2　提案する 2 種類の転移学習

特に $\lambda=0$ は Pre-Processing Layer のみ固定することを意味し，$\lambda=K$ は最終層の Linear Mixer のみ学習することを意味する．

高層部分のパラメータ $\Theta[\lambda+1:K+1]$ の初期値には，以下の 2 種類を考える．第 1 は，図 2(a) に示すように $\Theta_B^{(\mathrm{L})}[\lambda+1:K+1]$ を用いる場合であり，WaveNet 全体の初期値に式 (13) の $\Theta_B^{(\mathrm{L})}$ を用いることになる．このように類似タスクの訓練済みモデル $\Theta_B^{(\mathrm{L})}$ を目的タスクに適合するように微調整する手法はファインチューニングと呼ばれており，この場合の結果を以下で表す．

$$\Theta_{B\lambda T_{\mathrm{fine}}}^{(\mathrm{S})}=\mathcal{A}(\mathcal{D}_T^{(\mathrm{S})},\lambda+1:K+1\,|\,\Theta_B^{(\mathrm{L})}) \tag{15}$$

ここで，式 (10) と異なり写像 $\mathcal{A}$ の引数が $\lambda+1:K+1$ であるのは，第 $\lambda+1$ 層から第 $K+1$ 層のみを更新することを意味する．第 2 は，図 2(b) に示すようにランダムな値 $\Theta_{\mathrm{rand}}[\lambda+1:K+1]$ (Glorot の初期化) を用いる場合であり，この結果を以下で表す．

$$\Theta_{B\lambda T}^{(\mathrm{S})}=\mathcal{A}(\mathcal{D}_T^{(\mathrm{S})},\lambda{+}1:K{+}1\,|\,(\Theta_B^{(\mathrm{L})}[0:\lambda],\Theta_{\mathrm{rand}}[\lambda{+}1:K{+}1])) \tag{16}$$

3.2　学習アルゴリズム

WaveNet のパラメータ Θ を $\mathcal{D}_B^{(\mathrm{L})}$ や $\mathcal{D}_T^{(\mathrm{S})}$ などの学習データを用いて決定する．ここでは，第 2.1 節のように学習データを一般化して $\mathcal{D}=\{(x[n],y[n])\}_{n=1}^{N}$ で表す．$\mathcal{D}$ のデータ点数 N は，例えばサンプリング周波数が 44.1 kHz の場合，計 30 秒の入出力音で $N=1{,}323{,}000$ となり，計 300 秒の入出力音ではこの 10 倍の値となる．これらを一括して用いてパラメータ Θ を更新すると，計算負荷が高くなり，比較的低精度な局所解に陥る可能性も高くなる場合さえある．そこで，学習データ $\mathcal{D}$ をミニバッチに分割した状態で用いてパラメータ Θ を更新する．

ミニバッチ生成　まず，学習データ $\mathcal{D}$ を 4,096 点ごとに分割し，0.1 秒弱の入出力音を $\lceil\frac{N}{4096}\rceil$ 個作成する．次に，これら 0.1 秒弱の入出力音を P 個ずつランダムに組み合わせてミニバッチを I 個生成する．具体的にデータ点数 $N=1{,}323{,}000$，バッチサイズ $P=16$ の場合，$I=23$ 個のミニバッチが生成される．

パラメータ更新　生成したミニバッチ $\mathcal{D}_{\mathrm{MB},i}$ $(i=1,2,\ldots,I)$ の 1 つを使用し，Adam などの最適化手法によって，式 (9) の ESR を最小化するように WaveNet のパラメータ Θ を更新する．I 個のミニバッチを 1 回ずつ用いてパラメータを更新することを 1 epoch と呼び，1 epoch が終了したらミニバッチ $\mathcal{D}_{\mathrm{MB},i}$ を再度ランダムに生成し直す．上記の過程を一定回数繰り返して，ESR の値が下がり切った状態の Θ を最終的なパラメータとする．

4.　数 値 実 験

4.1　WaveNet の学習条件

実際の歪みエフェクタを用いた実験によって提案手法の有効性を示す．WaveNet の設定は文献 [8, 9] に倣い，チャネル数 $L=16$，残差ブロック数 $K=18$，フィルタサイズ $M+1=3$ $(M=2)$ とする．この場合，WaveNet 全体のパラメータ数は $2K(L^2(M+2)+2L)-L^2)=37{,}792$ となる．dilation 係数を $d_k=2^{(k-1)\bmod 9}$ と設定するので，受容野は $R=2{,}045$ となる．第 3.2 節で述べたように，入力音は 1 データ当たり 4,096 点とし，この長さは受容野 R の約 2 倍である．対応する真の出力音も同様に 1 データ当たり 4,096 点であるが，WaveNet が出力するモデリング音の前半部分は 0 でパディングされた入力音を用いて計算されるため，損失関数の計算には最後の 512 点のみを用いる．損失関数には式 (9) の高域強調後の ESR を用いる．最適化手法として Adam を学習率 0.001 で用い，バッチサイズ $P=16$，更新 epoch 数を 1000 とする．プログラムは Python 3.7.3 で Tensorflow と Keras を用いて実装し，実験に使用した計算機の環境は OS が Windows10 Pro，CPU が Core i9-7980X，メインメモリが 128 GB，GPU が GeForce GTX1080Ti である．

4.2　学習データとテストデータ

事前に訓練済みモデルを作成するエフェクタ B として歪みの強い個体である Ibanez 社製 SD9 を，目的のエフェクタ T として歪みの弱い個体である同社製 TS MINI を用いる．どちらのエフェクタにおいても，音色を調整するつまみを全て中心の 12 時方向に合わせた．学習データの入力音 $x^{(\mathrm{L})}[n]$ と $x^{(\mathrm{S})}[n]$ の作成には，IDMT データセット [14, 15] を用いた．このデータセットから 10 秒程度の約 70 フレーズを選択し，それらをランダムに接続することで計 30 秒と計 300 秒の入力音を生成した．これらをエフェクタ T と B に入力した際の出力音を録音することで，少量データ $\mathcal{D}_T^{(\mathrm{S})}$ と大量データ $\mathcal{D}_T^{(\mathrm{L})}$ 及び $\mathcal{D}_B^{(\mathrm{L})}$ を作成した．入出力音のサンプリング周波数は 44.1 kHz である．

学習後のモデリング精度を評価するためのテストデータには，IDMT データセットには含まれない音源として，エフェクタ T に対する約 5 秒のギター音の入出力関係を収録した自作の 4 つの音源を用いた．各音源の特徴は，音源 1 がアタックの強い低音域の半音階 (E→F→G$^\flat$→G)，音源 2 がアタックの弱い低音域の和音 ($\mathrm{D}_{\mathrm{add9}}$)，音源 3 がアタックの強い高音域の和音 (G→G$^\flat$)，音源 4 がアタックの弱い中音域のオクターブの和音 (C) である．

4.3　比 較 手 法

転移学習を用いない通常の訓練結果を従来手法とする．

- 従来手法 1：式 (11) の $\mathcal{D}_T^{(\mathrm{L})}$ からの訓練結果 $\Theta_T^{(\mathrm{L})}$
- 従来手法 2：式 (12) の $\mathcal{D}_T^{(\mathrm{S})}$ からの訓練結果 $\Theta_T^{(\mathrm{S})}$

従来手法 1 は大量データ $\mathcal{D}_T^{(\mathrm{L})}$ を用いるので，モデリング精度は高いが，学習時間は長いことが予想される．従来手法 2 は少量データ $\mathcal{D}_T^{(\mathrm{S})}$ を用いるので，学習時間は短くなるが，モデリング精度の低下が予想される．式 (13) の $\mathcal{D}_B^{(\mathrm{L})}$ からの訓練結果 $\Theta_B^{(\mathrm{L})}$ を転移して用いる提案手法として初期値の違う 2 つを評価する．

- 提案手法 1：式 (15) の $\mathcal{D}_T^{(\mathrm{S})}$ からの訓練結果 $\Theta_{B\lambda T_{\mathrm{fine}}}^{(\mathrm{S})}$
- 提案手法 2：式 (16) の $\mathcal{D}_T^{(\mathrm{S})}$ からの訓練結果 $\Theta_{B\lambda T}^{(\mathrm{S})}$

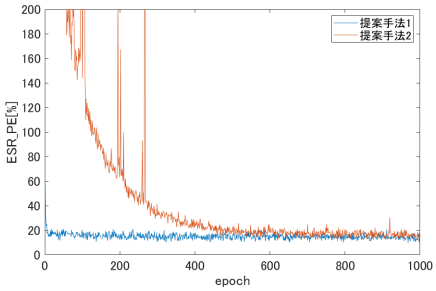


図 3　提案手法 ($\lambda = 9$) における損失関数 (事前強調 ESR) の学習曲線

2 つの提案手法はいずれも少量データ $\mathcal{D}_T^{(\mathrm{S})}$ から学習を行うので，λ の値を変えながら従来手法 1 の精度にどの程度近づけるかを検証し，各手法の学習時間も併せて比較する．その前にまず，更新 epoch 数の検証を行った．例として，$\lambda = 9$ の提案手法の学習曲線を図 3 に示す．高層部分をランダム初期値とする提案手法 2 では，1000 epoch まで損失関数の値が低下し続けており，従来手法と同様に収束に 1000 epoch 程度を要する．一方，初期値に $\Theta_B^{(\mathrm{L})}$ を用いる提案手法 1 では，最初の数十 epoch で損失が急激に低下しており，200 epoch 以降の減少は緩やかである．そこで，提案手法 1 のみ 200 epoch の場合の性能も評価する．

4.4　実 験 結 果

テスト音源に対する高域強調無しの通常の ESR の平均値と各学習時間を図 4 に示す．大量データを用いる従来手法 1 では平均 ESR が 0.35 %と高精度なモデリングを実現したが，少量データを用いる従来手法 2 では平均 ESR が 3.31 %と精度は不十分なものとなった．少量データを用いる提案手法 1 と 2 では $\lambda = 0, 1, \ldots, 9$ のとき平均 ESR が 1 %前後となり，従来手法 2 と比べてモデリング精度の大幅な改善が見られた．これは転移した $\Theta_B^{(\mathrm{L})}$ の低層部分が $\Theta_T^{(\mathrm{L})}$ と同様の特徴抽出を行っており，$\mathcal{D}_T^{(\mathrm{S})}$ からでは十分に学習しきれない汎用的な処理を補うことができたためだと考えられる．一方，$\lambda = 10$ 辺りからは平均 ESR が徐々に増加していき，$\lambda = 15$ 以降は従来手法 2 よりも精度が低くなる．これは $\Theta_B^{(\mathrm{L})}$ の高層部分が $\Theta_T^{(\mathrm{L})}$ と異なる特徴を計算しているためだと考えられる．また，dilation 係数 d_k が $k = 10$ で大きく切り替わることも影響している可能性がある．

テスト音源 3 に対する従来手法と提案手法のモデリング音の波形を図 5 に示す．図 5(a) の従来手法 1 では，橙色のモデリング音が青色の目的音源を高精度に再現しており，実際に視聴した際も 2 つの音を区別することは困難であった．図 5(b) の従来手法 2 では，目的音源の振幅が小さい部分でモデリング音の精度が不十分となり，実際にモデリング音を視聴した際も音が途切れ途切れになっている印象を受けた．図 5(c) の $\lambda = 18$ とした場合の提案手法 1 では，最終層の Linear Mixer のみしか調整されないため，エフェクタ B の特性が残存し，モデリング音は目的音源と比べて小さな振幅値でクリッピングされている．モデリング音を視聴した際も，エフェクタ B として用いた SD9 の強い歪みの残存が確認された．一方で $\lambda = 9$ とした場合は，図 5(d) の 1000 epoch の提案手法 1，図 5(e) の 200 epoch の提案手法 1，図 5(f) の 1000 epoch の提案手法 2，いずれでもモデリング音が高精度に生成されており，音も途切れていない．

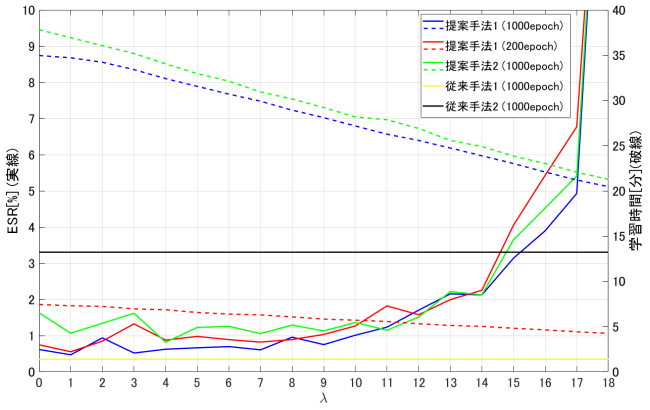


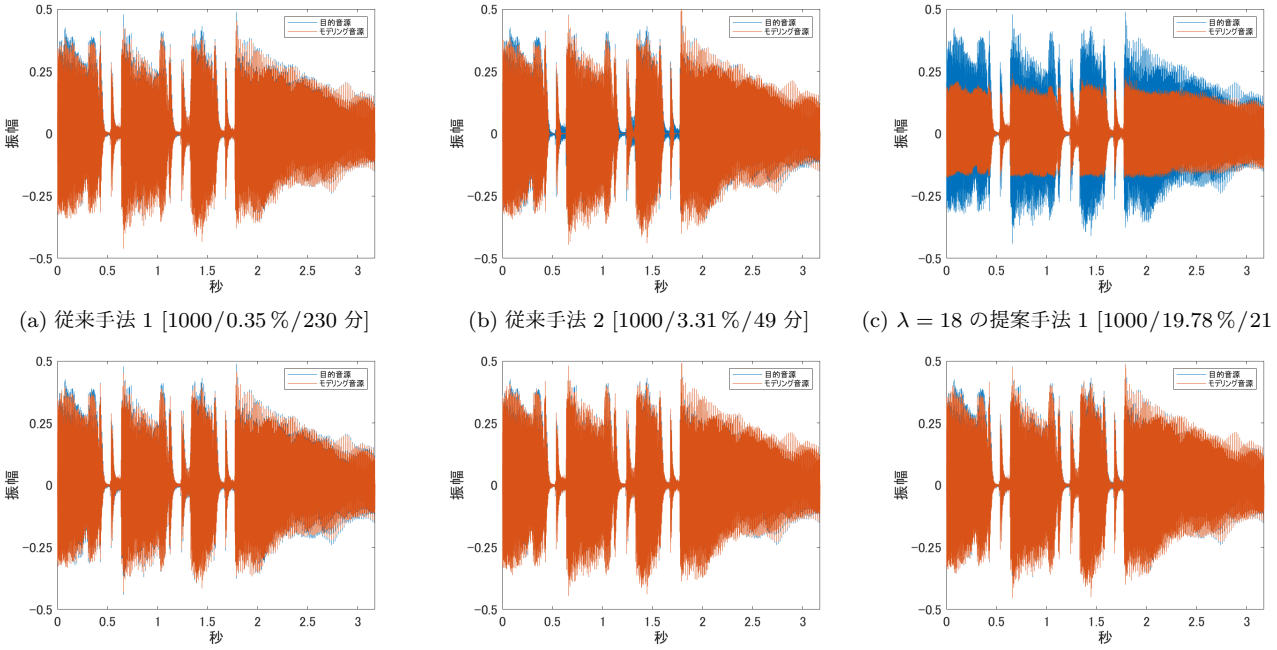
図 4　λ に応じたモデリング精度 (無強調 ESR) と学習時間 (分) の変化

実際に各モデリング音を視聴すると，従来手法 1 の結果と比べても遜色なく，目的音源との区別が困難であった．提案手法 1 と提案手法 2 を比べると，図 4 の平均 ESR では提案手法 1 の方が僅かに高精度だが，視聴すると同じ音のように感じられた．

次に各手法の学習時間を比較する．大量データを用いる従来手法 1 では 230 分，少量データを用いる従来手法 2 では 49 分の学習時間がかかった．一方で，第 0 層のみを固定した場合，1000 epoch の提案手法 1 と提案手法 2 では 35 分と 38 分，200 epoch の提案手法 1 では 7 分の学習時間となり，層を 1 つ固定することで学習時間が短くなることが分かる．更に，提案手法では λ の増加に伴い式 (14) の更新パラメータ数 $\Delta(\lambda)$ が減少し，結果として学習時間はより短くなることが図 4 から確認できる．モデリング精度が低下し始める $\lambda = 10$ の直前である $\lambda = 9$ の場合，1000 epoch の提案手法 1 と提案手法 2 で 28 分と 29 分，200 epoch の提案手法 1 で 6 分の学習時間となり，少量データを用いる従来手法 2 と比べて時間が，1000 epoch の提案手法 1 と提案手法 2 で 43 %と 41 %，200 epoch の提案手法 1 で 88 %も短縮されている．大量データを用いる従来手法 1 と比べると，1000 epoch の提案手法 1 と提案手法 2 でも 88 %と 87 %，200 epoch の提案手法 1 に至っては 97 %も学習時間を短縮している．$\lambda = 9$ のときは 200 epoch の提案手法 1 で聴覚上十分な精度の結果を得ているので，ファインチューニングで少量データから高速高精度に TS MINI をモデリングできることが示された．

4.5　転移前後のエフェクタを入れ替えた場合の実験結果

最後に，エフェクタ B に歪みの弱い TS MINI，エフェクタ T に歪みの強い SD9 を用いた場合の実験結果を簡単に紹介する．前節の結果に基づいて $\lambda = 9$ とし，提案手法 1 のみ更新回数を 200 epoch とした．テスト音源に対しての平均 ESR は，従来手法 1 が 0.35 %，従来手法 2 が 5.38 %，提案手法 1 が 2.61 %，提案手法 2 が 3.67 %であった．従来手法 2 と提案手法の比較から，歪みの強いエフェクタに対しても転移学習によって少量データからのモデリング精度が改善することが分かった．また，提案手法 1 が提案手法 2 よりも有効である傾向が顕著に現れた．一方，従来手法 1 と提案手法の比較から，大量データから訓練した結果には数値的にも聴覚的にも未だ劣っていることが分かった．以上から，転移後のエフェクタ T の方が歪みが強い場合，ファインチューニングに更なる工夫が必要であると考えられる．



(a) 従来手法 1 [1000/0.35 %/230 分]　(b) 従来手法 2 [1000/3.31 %/49 分]　(c) $\lambda = 18$ の提案手法 1 [1000/19.78 %/21 分]

(d) $\lambda = 9$ の提案手法 1 [1000/0.75 %/28 分]　(e) $\lambda = 9$ の提案手法 1 [200/1.03 %/6 分]　(f) $\lambda = 9$ の提案手法 2 [1000/1.13 %/29 分]

図 5　テスト音源 3 (G→G$^\flat$) と従来手法及び提案手法のモデリング音の波形 [エポック数/平均 ESR/学習時間]

5. おわりに

WaveNet を用いた歪みエフェクタモデリングにおいて，少量データから高速かつ高精度にモデルを学習する手法を提案した．提案手法では，目的エフェクタとは異なるエフェクタに対して十分に訓練された WaveNet を用意しておき，そのパラメータを初期値としてファインチューニングを適用する．これにより，少量データからでは十分に学習できないエフェクタ間に共通の汎用的な処理が補われ，モデリング精度が改善する．また，低層部分のパラメータを固定することで，更新パラメータ数を減少させ，少量データからの学習時間を短縮した．ファインチューニングを適用した提案手法では，高層部分をランダムな初期値にする場合と比べて学習曲線の収束が速く，少ない epoch 数で高精度なモデリング音が得られることを確認した．数値実験において提案手法は，少量データから学習する従来手法と比べてテスト音源に対しての平均 ESR を 69 %削減した．また，大量データから学習する従来手法と比べて学習時間を 97 %短縮した．

今後の研究として，転移後のエフェクタ T の方が歪みが強い場合にも少量データから十分高精度にモデリングを行うために，文献 [9] の様に損失関数を変更することが考えられる．また，異なるメーカなど，様々なエフェクタ間での検証が必要である．

文　献

[1] D. T. Yeh, J. Abel, and J. O. Smith, “Simulation of the diode limiter in guitar distortion circuits by numerical solution of ordinary differential equations,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Bordeaux, France, 2007, pp. 197–203.

[2] D. T. Yeh and J. O. Smith, “Simulating guitar distortion circuits using wave digital and nonlinear state-space formulations,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Espoo, Finland, 2008, pp. 19–26.

[3] A. Novak, L. Simon, P. Lotton and J. Gilbert, “Chebyshev model and synchronized swept sine method in nonlinear audio effect modeling,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Graz, Austria, 2010, 4 pages.

[4] F. Eichas and U. Zölzer, “Black-box modeling of distortion circuits with block-oriented models,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Brno, Czechia, 2016, pp. 39–45.

[5] F. Eichas, S. Möller, and U. Zölzer, “Block-oriented gray box modeling of guitar amplifiers,” in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Edinburgh, UK, 2017, pp. 184–191.

[6] Z. Zhang, E. Olbrych, J. Bruchalski, T. J. McCormick, and D. L. Livingston, “A vacuum-tube guitar amplifier model using long/short-term memory networks,” in *Proc. IEEE SoutheastCon*, Saint Petersburg, FL, USA, 2018, 5 pages.

[7] E.-P. Damskägg, L. Juvela, and V. Välimäki, “Deep learning for tube amplifier emulation,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brighton, UK, 2019, pp. 471–475.

[8] E.-P. Damskägg, L. Juvela, and V. Välimäki, “Real-time modeling of audio distortion circuits with deep learning,” in *Proc. Sound Music Comput. Conf. (SMC)*, Málaga, Spain, 2019, pp. 332–339.

[9] K. Yoshimoto, H. Kuroda, D. Kitahara, and A. Hirabayashi, “WaveNet modeling of distortion pedal using spectral features,” *Acoust. Sci. Tech.*, vol. 42, no. 6, pp. 305–313, 2021.

[10] A. van den Oord, S. Dieleman, H. Zen, K. Shimonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, “WaveNet: A generative model for raw audio,” *preprint arXiv:1609.03499*, 15 pages, 2016.

[11] 田中 宏, 亀岡 弘和, 金子 卓弘, 北条 伸克, “WaveCycleGAN2 : 高品質音声生成のためのニューラル波形ポストフィルタ,” 電子情報通信学会技術研究報告, vol. 119, no. 188, pp. 1–6, 2019.

[12] D P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, 2015, 15 pages.

[13] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proc. Int. Conf. Artif. Intell. Stat.*, Sardinia, Italy, 2010, pp. 249–256.

[14] SMT-Guitar, `https://www.idmt.fraunhofer.de/en/publications/datasets/guitar.html`, 2022/1/31 参照.

[15] Bass-Single-Track, `https://www.idmt.fraunhofer.de/en/publications/datasets/bass_lines.html`, 2022/1/31 参照.