K-009

U-Net を用いた電車走行音雑音除去のための学習モデルの作成方法とその評価
Method and evaluation of learning model for train running noise reduction using U-Net

島田 紘武 † 川喜田 佑介 † 宮崎 剛 † 田中 博 †
Hiromu Shimada Yuusuke Kawakita Tsuyoshi Miyazaki Hiroshi Tanaka

1. はじめに

多様な雑音の除去に対応するためには，学習時に多くの種類の雑音を用いる必要があり，大きな学習コストが必要となる．しかし，対象とする雑音を限定することで，学習コストを抑え，より性能が高い雑音除去モデルが作成可能と思われる．例えば電車の走行区間であれば GNSS，機械の振動状態は加速度センサなどで検知可能である．

したがって，これらのセンサ情報に応じて最適な雑音除去モデルを選択することで，モデルの作成負荷を抑えつつ性能を確保できるシステム，アプリが実現できると考えられる．そのコンセプトを図 1 に示す．本検討では列車の走行音を取り上げて，その効果を調べることとした．

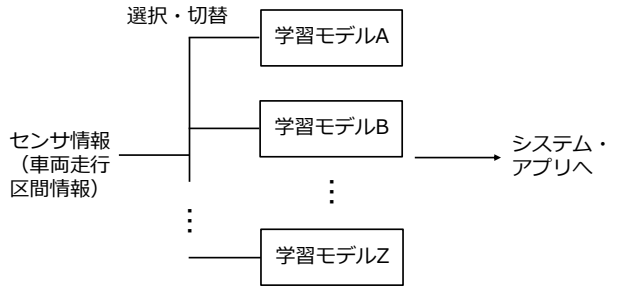

図 1　センサ情報による学習モデルの切り替え

2. 雑音除去手法

雑音除去手法は，筆者らがこれまで適用してきた図 2 に示す方法を適用する．音声データを画像データ（スペクトログラム）に変換することで U-Net[1]を利用する．ここで，U-Net への入力は雑音が重畳された音声データ（雑音重畳データ）であり，U-Net の出力が教師データ，すなわち，雑音が混入していないクリーンな音声と一致するように U-Net を学習する．実利用時は，学習後の雑音除去モデルを用いて雑音を取り除く．

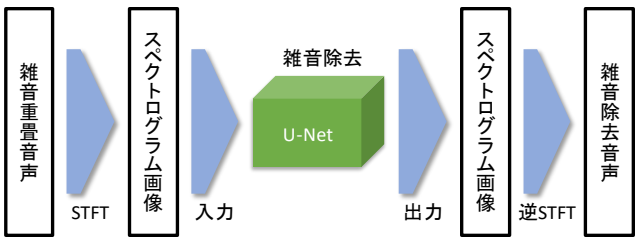

図 2　雑音除去の流れ

† 神奈川工科大学情報学部情報工学科　Dept. of Information and Computer Sciences, Kanagawa Institute of Technology

3. データの取得と雑音除去モデルの作成

本検討では，男性の音声に重畳された電車の走行雑音を除去することを目的に走行区間に応じた雑音除去モデルを作成しその性能を比較する．

3.1 走行雑音の取得

走行雑音は相模鉄道本線の電車内にて録音した．録音には Audacity の録音機能を用いた．データ形式は WAV，サンプリング周波数は 16kHz，チャネル数はモノラルである．本検討では録音した区間のうち，走行雑音が大きくはっきり聞こえ，走行雑音以外の雑音が少ない区間である A：大和～瀬谷，B：希望ヶ丘～二俣川，C：二俣川～横浜の 3 区間における走行雑音を用いた．以降、各区間をそれぞれ区間 A，区間 B，区間 C とする．

3.2 利用音声

学習モデルの作成と評価の際に雑音を重畳する音声には「日本音響学会新聞記事読み上げ音声コーパス」[2]の音声データを使用した．使用した音声のデータ形式は WAV，サンプリング周波数は 16kHz，チャネル数はモノラルである．10～15 分程度の男性 5 名の音声を雑音除去モデルの作成に，雑音除去モデルの作成に用いた男性以外の 1 名の音声を評価に用いた．

3.3 学習データの作成

本検討では，図 3 のように，教師データとして走行雑音を重畳する前のクリーンな音声，学習データとして走行雑音を重畳した後の音声を用いる．そして，学習データが教師データに一致するように学習させ雑音除去モデルを作成する．

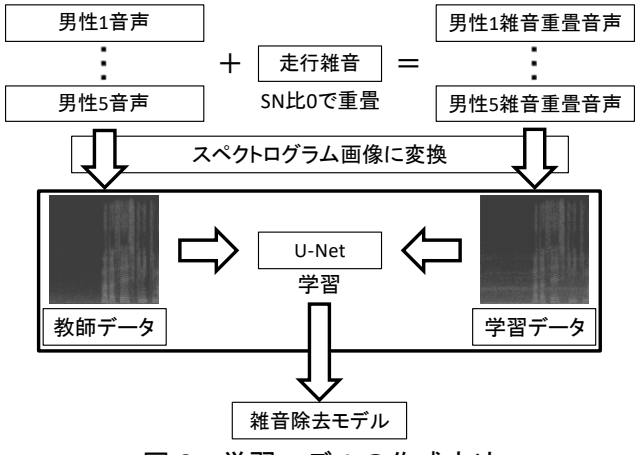

図 3　学習モデルの作成方法

重畳する走行雑音は図 4 のように区間ごとに 40 秒ずつ切り出して複製し，結合することによって区間ごとに 20 分の長さの雑音データとして作成している．各区間 40 秒ずつ切り出したのは，乗客の会話や車内アナウンス，その他走行雑音以外の雑音が入っていない部分を切り出した際に，一番短い区間の走行雑音が 40 秒になるので，条件をそろえるためである．また，複製して 20 分にしたのは，走行雑音を重畳するにあたって男性の音声の長さと同等の時間の走行雑音を確保するためである．今回の検討では，走行雑音を重畳する際の SN 比は 0dB としている．

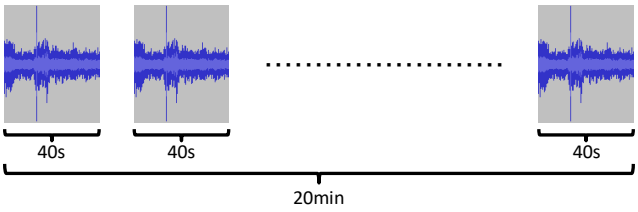

図 4　重畳する走行雑音の作成方法

U-Net を用いて学習するため，雑音重畳前の音声と雑音重畳後の音声を STFT によってスペクトログラムに変換した．今回の検討では，表 1 のように，STFT する際の窓関数は hamming，窓関数の幅は 1024，hop length は 128 とした．

表 1　学習データ作成時のパラメータ

項目	パラメータ
窓関数	hamming
窓関数幅	1024
hop length	128
画像サイズ	256×256
画像切り出しスライド幅	205

図 5 に示した方法で学習データを作成した．窓を 128 ポイントずつずらしながら，幅が 1024 ポイントの窓関数をかけて STFT を行ない，音声データをスペクトログラムに変換した．このスペクトログラムからサイズが入力画像と同じ 256×256 の画像を切り出した．切り出す際のスライド幅は 205(256×0.8)としている．

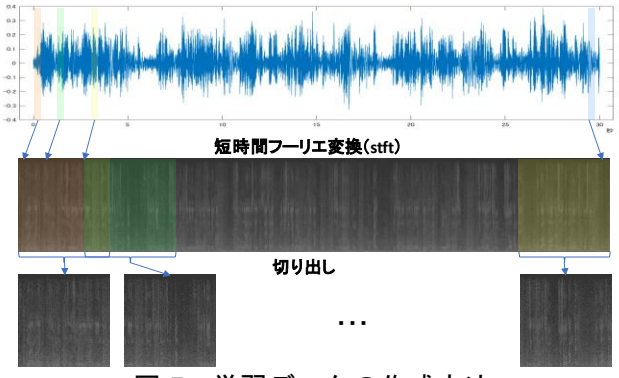

図 5　学習データの作成方法

使用した走行雑音の区間とモデル名，データ数を表 2 に示す．特化モデルでは，雑音を除去したい区間の雑音のみを学習に用いており，汎用モデルでは 3 区間すべての雑音を学習に用いている．今回の検討では，各特化モデルの学習データは 1880 枚，検証データは 470 枚，汎用モデルの学習データは 5640 枚，検証データは 1410 枚作成した．筆者が各区間の雑音を聞いた際の認識としては，区間 A が最も雑音が大きく特徴的な雑音も多かった．また，区間 B は，平坦で特徴が見られず，区間 C は区間 A,B の中間であると感じた．

表 2　各モデルの使用雑音とデータ数

モデル名	使用雑音	学習データ数	検証データ数
特化モデル A	区間 A	1880	470
特化モデル B	区間 B	1880	470
特化モデル C	区間 C	1880	470
汎用モデル	区間 A, B, C	1880	1410

3.4 雑音除去モデルの作成

雑音を除去したい区間の雑音のみを学習に用いた特化モデル，雑音を除去したいすべての区間の雑音を同時に学習に用いた汎用モデルを作成した．学習のパラメータは，表 3 に示す．特化モデルと汎用モデルでバッチサイズが異なっているのは，汎用モデルの作成時にバッチサイズを 16 にすると学習を実行することができなかったためである．

表 3　学習モデル作成時のパラメータ

	エポック	バッチサイズ	最適化手法	学習率	損失関数
特化モデル	200	16	Adam	5e-4	MSE
汎用モデル		8			

特化モデルの学習の収束状況の一例を図 6 に，汎用モデルの学習の収束状況を図 7 に示す（実行環境：CPU: Intel Core i5-12400，メモリ：16GB，GPU: NVIDIA GeForce RTX 3060 メモリ：12GB）．特化モデルと汎用モデルのどちらにおいてもおおむね同様の値に収束していることが確認できる．特化モデルの学習時間はどの区間においても約 6 時間 30 分であり，汎用モデルの学習時間は約 19 時間 30 分であった．

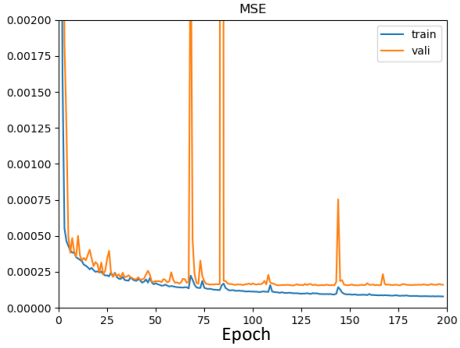

図 6　特化モデル A の学習の収束状況

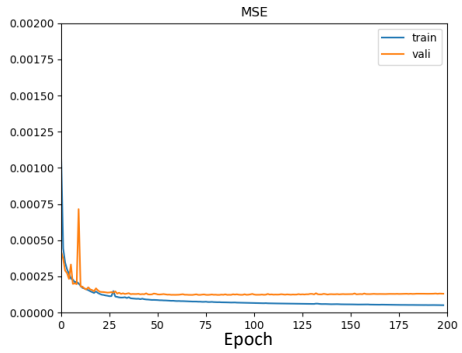

図 7　汎用モデルの学習の収束状況

4. 雑音除去モデルの評価

4.1 画像による評価

4.1.1 スペクトログラムによる評価

雑音重畳前の音声のスペクトログラムを図 8 に，区間 A，B，C の雑音を重畳した際のスペクトログラムとモデル適用後のスペクトログラムをそれぞれ図 9，図 10，図 11 に示す．各区間のスペクトログラムは雑音の特徴が見えやすくするために音声の開始から 10 秒間を表示している．雑音重畳前のスペクトログラムと各区間の雑音重畳後の音声を比較すると 0〜3kHz の周波数帯に強く雑音の特徴が出ていることが確認できる．また，どの区間も汎用モデルを適用した場合と特化モデルを適用した場合のどちらにおいても雑音の特徴を除去することができており，スペクトログラムでは明確な差は見られない．

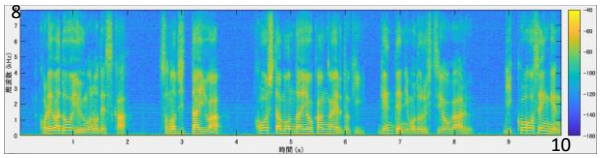

図 8　雑音重畳前スペクトログラム

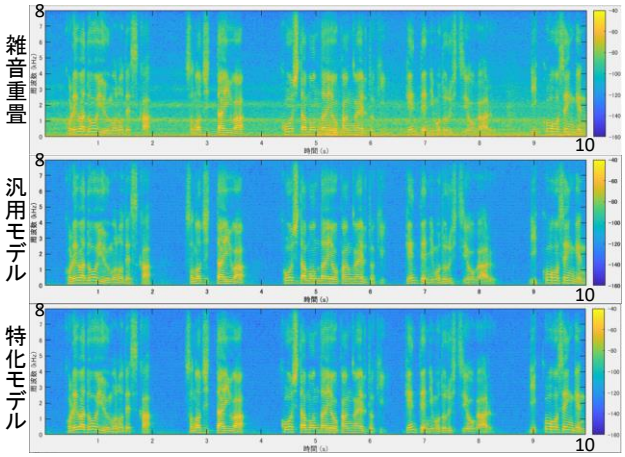

図 9　区間 A スペクトログラム

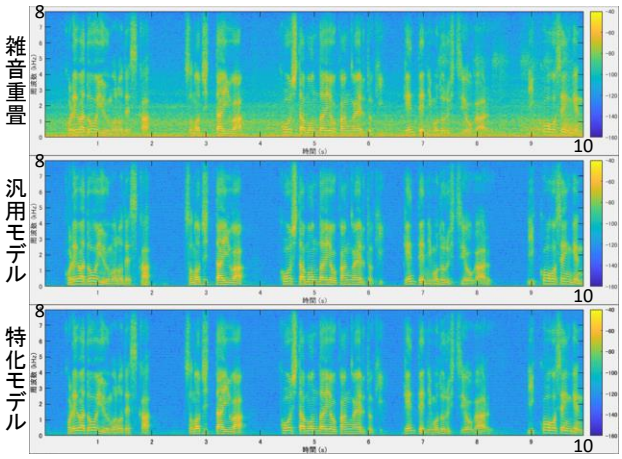

図 10　区間 B スペクトログラム

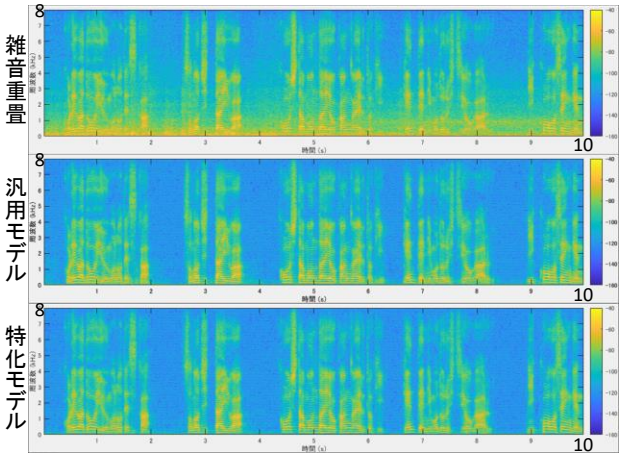

図 11　区間 C スペクトログラム

4.1.2 音声波形による評価

雑音重畳前の音声波形を図 12 に，区間 A，B，C の雑音を重畳した際の音声波形とモデル適用後の音声波形をそれぞれ図 13，図 14，図 15 に示す．各区間の音声波形は雑音の特徴が見えやすいスケールにするために音声の開始から 10 秒間を表示している．どの区間も汎用モデルを適用した場合と特化モデルを適用した場合のどちらおいても音声を発していない部分の雑音は除去できていることが確認できる．一方で，音声を発している部分は極端に波形の特徴が変化している部分はないものの，全体的に元の音声を除去しすぎているように見える．音声波形においても汎用モデルと特化モデルを適用した場合の明確な差は見られない．

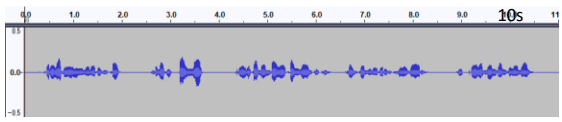

図 12　雑音重畳前音声波形

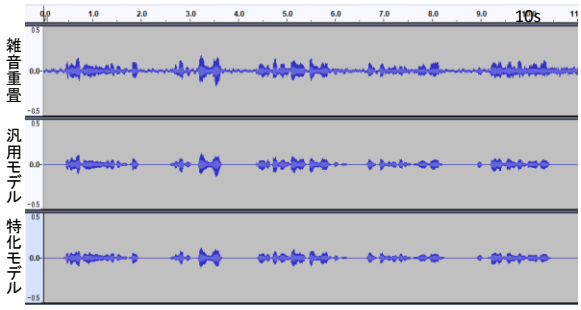

図 13　区間 A 音声波形

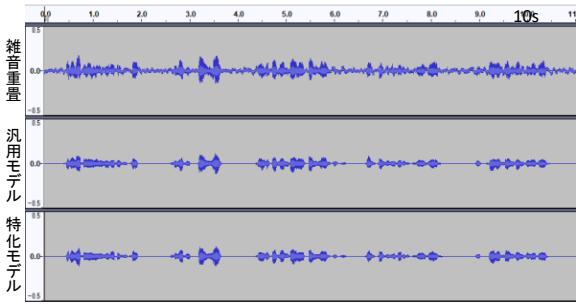

図 14　区間 B 音声波形

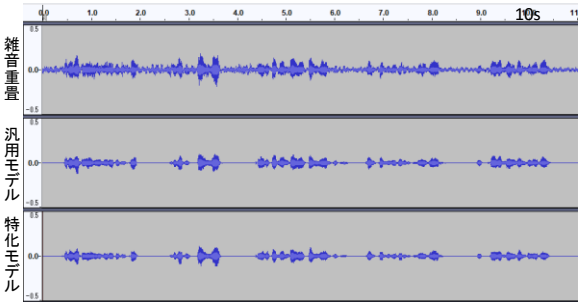

図 15　区間 C 音声波形

4.2 音声認識による評価

4.2.1 評価方法

各区間の雑音重畳音声，特化モデル適用後の音声，汎用モデル適用後の音声を 40 秒ずつ合計 3 分の長さの音声を切り出して Julius[3]を用いて文字起こしを行った．Julius のバージョンは 4.6 を使用している．音声の切れ目では Julius の文字起こしが正常に行われない場合があるため，40 秒のうち 10 秒は重なるように切り出し補完している．

文字起こしした文章から，句読点や空白を取り除く処理を行った後，文字誤り率の測定ツール[4]を用いて精度比較を行った．文字誤り率の計算式は以下のとおりである．文字誤り率の最大値は 1 であり，数値の小さい方が，文字認識精度が良い．今回の検討では，同じ雑音を用いて同じ条件ごとに 2 回ずつモデルを作成し，文字誤り率を算出している．

$$CER = (ins + rep + del) / cor \quad (1)$$

CER：文字誤り率
ins　：挿入語数
rep　：置換語数
del　：削除語数
cor　：正解語数

4.2.2 評価結果

各区間の雑音を重畳した音声と各雑音除去モデル適用後の音声の文字認識精度を表 4 に示す．特化モデルと汎用モデルのどちらを適用した場合にも雑音重畳音声の文字認識精度は向上していることが確認できる．また，どの条件のモデルにおいても 1 回目と 2 回目で文字認識の精度にばらつきが出ているが，特に汎用モデルにおいては，極端な差があることが確認できる．区間 A，B においては，汎用モデルの方が精度が良い傾向が見られ，区間 C においては，特化モデルの方が精度が良い傾向がみられる．

先述のとおり，筆者が各区間の雑音を聞いた際の認識では，区間 A が最も雑音が大きく特徴的な雑音が多く，区間 B は，平坦で特徴が見られず，区間 C は区間 A,B の中間であった．この雑音の差が，汎用モデルと特化モデルの雑音除去効果の差や傾向に影響を与えていることが考えられる．

表 4　文字認識精度の比較

	区間 A	区間 B	区間 C
雑音重畳音声	0.190 (159)	0.204 (171)	0.189 (158)
	0.214 (179)	0.206 (173)	0.125 (180)
汎用モデル適用後音声	0.097 (81)	0.121 (101)	0.121 (101)
	0.075 (63)	0.075 (63)	0.075 (63)
特化モデル適用後音声	0.122 (102)	0.124 (104)	0.078 (65)
	0.095 (80)	0.120 (98)	0.080 (67)

誤り率(総誤り文字数)
総文字数 838 文字

5. まとめと今後の課題

本検討では，音声をスペクトログラムに変換し，区間ごとに雑音除去モデルを作成する手法を提案した．また，汎用モデル，特化モデルのどちらにおいても雑音除去効果を確認できた．

スペクトログラムと音声波形の比較では，汎用モデルと特化モデルを適用した場合の明確な差は見られなかった．文字誤り率による文字認識精度の比較では，区間ごとにみると差が出ている部分もあったが，全体として汎用モデル特化モデルのどちらが良いと結論づけることができるような傾向は今回の検討では確認できなかった．したがって，切り出した 40 秒の走行音データはあくまで区間全体の雑音データの一部であるため，切り出し方を変えてのモデル作成，区間やモデル作成の試行回数を増やして再度検討等をする必要がある．

また，筆者が雑音除去後の音声を聞いた際の認識としては，雑音が除去しきれていないと感じた区間はないが，少し歪が生じていると感じる区間が多かった．雑音除去後に歪が出ないようにパラメータ等の調整を行う必要がある．

今回の検討では，最終的な文字誤り率での比較のみで終わったため，文字の挿入，置換，削除数の傾向も調査する必要があると考えられる．

参考文献

[1] O. Ronneberger, et al. “U-Net: Convolutional Net-works for Biomedical Image Segmentation,” MIC-CAI 2015, Springer, LNCS 9351, pp.234–241, 2015.
[2] JNAS 音声資源コンソーシアム新聞記事読み上げ音声コーパス, http://research.nii.ac.jp/src/JNAS.html,参照 2022/12/13.
[3] Julius, https://julius.osdn.jp/, 参照 2023/4/28
[4] https://github.com/kchan7/WER-CER,参照 2022/12/13