K-006

# Julius と Chainer による非語の音声認識
# Non-word speech recognition by Julius and Chainer.

多々納 俊治 [1]　縄手 雅彦 [2]　伊藤 史人 [2]　酒向 慎司 [3]　門脇 和央 [2]
Toshiharu Tadano  Masahiko Nawate  Fumihito Ito  Shinji Sako  Kazuhisa Kadowaki

## 1. はじめに

「読み書き困難児のための音読・音韻処理能力簡易スクリーニング検査ソフト ELC」 [1]はディスレクシアと呼ばれる読み書き困難児をスクリーニングする検査ソフトである．この検査ソフトは児童の読みに疑問を持った教師が学校で簡便に使用できることを目的に開発されたものであり，PC を使用して 10 分程度で検査を行うことができる．検査項目は音韻特徴を捉える課題として「短文音読課題」，音韻意識を評価する課題として「音韻操作課題（逆唱・削除）」，デコーディングを評価する課題として「単語非語音読課題」の 3 部から構成されている．読みの能力や音韻操作の能力は流暢性が評価の重要な点であるため，回答時間を自動計測することが特徴となっている．しかし，現時点では回答された音声の正誤判定は検査者自身が行わなければならず，音声認識技術の活用による自動化が期待されている．ELCの「音韻操作課題」で出題される16の単語，もしくは非語（無意味単語）に逆唱，削除を施したものを **Table 1** に示す．

最近のコンピュータを利用した音声認識の技術の進歩は目覚ましく丁寧に発声された音声認識率は単語認識率に換算して 95%前後を達成している [2]．一般的な音声認識エンジンの構造は各単語とそれを発声した時の音声の特徴パラメータ系列との関係を表す音響モデルと単語の並びの制約を表す言語モデルからなり，観測された音響特徴パラメータに最も適合する単語列を膨大な検索空間上を検索することによって実現されている [2]．しかし音響モデルでの単独モーラの認識は難しく，ある程度のモーラ数を意味のあるものとしてとして登録された言葉だけが言語モデルを通して認識され，語彙どうしの繋がりがどのような形になるかの解析に重点がおかれる．したがって辞書に登録する語彙は意味のある塊（コーパス）をベースとしたものが基準で非語を対象とした登録は殆ど行われていない．文章を基準としたコーパス・ベースの音声認識の方式では単語一語の認識率は下がり，特に非語を認識の対象とした場合70%～80%に留まる [2]．また音声認識は多数の辞書データから正しい言葉を分類し選出する仕組みが基本であり，登録データが少ない場合は未知の入力に対して既知の誤りデータを正解データと誤認識する可能性が高くなる．したがって仮に非語を辞書に登録した場合には誤認識をいかに防ぐかが課題になる．しかし全ての誤り語を誤りデータとして網羅し登録するのは現実的ではない．

そこで非語を辞書に登録する方法以外に音声認識エンジンの性能を補強する手段を考えた場合，機械学習の機構を組み込むことが候補に挙がる．音声認識に機械学習を取り入れる試みは最近にわかに行われつつある手法で，結果として認識率が向上したという事例がいくつか報告されている [3].

| | 逆唱 | | 削除 | |
|---|---|---|---|---|
| 単語 | あたま | またあ | ねくたい | ねたい |
| | かめら | らめか | たまねぎ | たねぎ |
| | たまご | ごまた | あさがお | あさが |
| | つくえ | えくつ | ひまわり | ひまり |
| 非語 | みしけ | けしみ | いそれす | いそす |
| | たぐめ | めぐた | なゆかた | なゆか |
| | たちの | のちた | ぶとみご | ぶとご |
| | せとく | くとせ | よですち | よすち |

**Table 1　ELC の音韻操作課題**

1 出雲医療看護専門学校
2 島根大学大学院総合理工学研究科
3 名古屋工業大学大学院

## 2. 目的

ELC の検査項目である非語の正誤判定を自動化するには，非語にたいして精度の高い音声認識の仕組みが要求される．本研究では既存の音声認識エンジンに機械学習の機構を取り入れることで，その機能を補強し，非語に対する認識率を向上させることを目的とする．

## 3. SVM による実験

はじめに機械学習の部分にサポートベクタマシン（SVM）[4]を組み込む実験を行った．SVM は線形入力素子を利用して 2 クラスのパターン識別器を構成する手法であり，識別境界を決める学習データをサポートベクトルとし，各クラスのサポートベクトルと識別境界との距離であるマージンが最大になるような識別境界面を得るように線形入力素子のパラメータを学習するアルゴリズムである．

### 3.1. 方法

SVM の実装は C++のライブラリである LIBLINER [5]により行った．LIBLINER は線形予測に特化しており，従来のライブラリである LIBSVM [6]に比べ処理時間が速く学習作業時間の大幅な短縮が期待できる．

研究目的で既存の音声認識エンジンを選ぶ場合，現在フリーで利用できるものは名古屋工業大学が開発管理し公開している大語彙連続音声認識エンジン Julius [7]のみであるが，本研究ではシステムの前面に置く音声認識エンジンに Julius を利用した．Julius は別途オープン・ソースで公開されている JuliusLib をプログラムに組み込むことでその機能を自由に使用することができ，人間の発声知覚の特徴を考慮した声道特性であるメル周波数ケプストラム係数（MFCC）を代表値とし，音響モデルと言語モデルの尤度からなる Julius 独自のスコア出力やその他，音量，発話時間など様々な出力を取得することができる．それらの音声特徴量を取得し LIBLINEAR で機械学習させる．

またメイン・インターフェースは Ruby で作成し，学習段階で認識率の推移を記録しグラフで可視化するためオンライン方式による逐次学習機能を実現した．

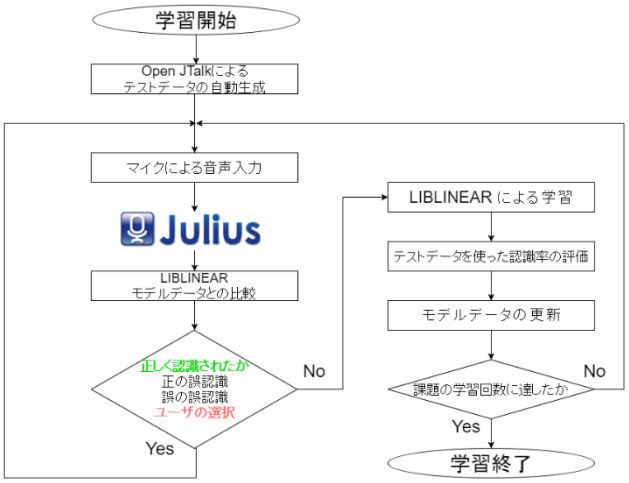


**Fig. 1　システムの流れ図**

学習させる非語は ELC の音韻操作課題から選んだ 8 つの非語である．

システムの流れ図を **Fig.1** に示す．入力された音声はまず Julius を通して正誤判定される．さらに LIBLINER が JuliusLib の出力した特徴量と更新前の非語の正誤モデルを比較して再び正誤判定を施す．その時点でその結果が正しい音声を間違いと認識した場合，または間違った音声を正しいと認識した場合にのみ LIBLINEAR にその間違いを学習させるようにユーザが操作する．その後 LIBLINER は正しい特徴量を学習し， 非語の正誤モデルを更新する．次回の予測にはその更新されたモデルが利用され，学習ループの中で逐次的に認識率が推移していく．

学習データは 20 代から 30 代の男女含め 13 人から採取を行った．まず 8 つの非語のうちの一つ「あたま」の逆唱である「またあ」について，複数人で非語の正誤モデルを更新する学習作業を一人 20 試行ほど順次交代で合計 160 回程度行った．次に 8 つの非語のすべて「またあ」，「らめか」，「ごまた」，「えくつ」，「けしみ」，「めぐた」，「のちた」，「くとせ」について，一人でそれぞれ 40 回程度の学習作業を行った．

認識率は対象の非語について正しく発音された音声，または間違って発音された音声で構成されるテストデータを機械学習が，自らの更新した正誤モデルを使ってどの程度の割合で正しく認識したか，という指標で測定する．

用意するテストデータは，まず先に，8 つの非語について名古屋工業大学が開発した音声合成エンジン Open JTalk [8]を用いて作成した WAV ファイルを使用した．音声モデルは mei [9] の normal 及び，hts-voice- nitech-jp-atr503-m001 の両方である．そして後に，人間の音声を学習データとして採取したものの中から 20%程度をテストデータに回し「えくつ」について認識率を測定した．

### 3.2.　結果

学習した 8 つの非語のすべてにおいて最大認識率が 80％台に達した．しかし，それぞれの認識率の推移に様々な違いが出た．「またあ」について複数人から採取した音声データを 160 回程度学習した場合の認識率の学習回数による推移のグラフを **Fig.2** に示す．途中までは単調な増加が見られるが，学習回数 70 前後で認識率の向上が頭打ちになり，その後大きく振動を繰り返している．また 8 つの非語

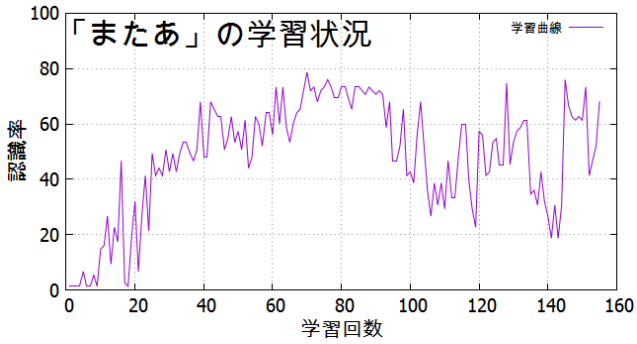


**Fig. 2　「またあ」の学習結果**

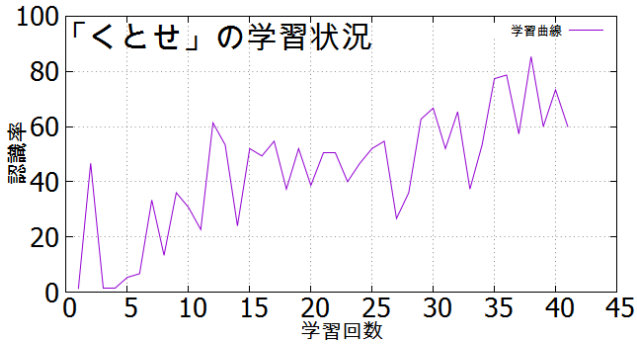


**Fig. 3　「くとせ」の学習結果**

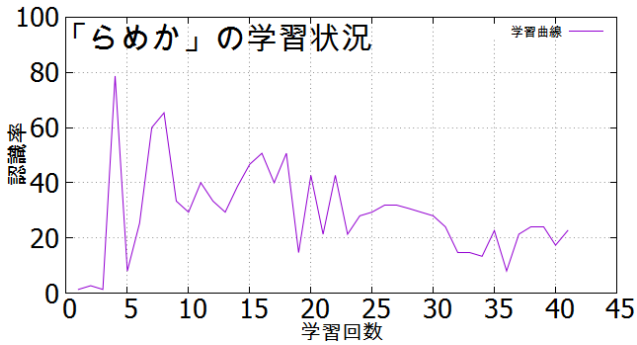


**Fig. 4　「らめか」の学習結果**

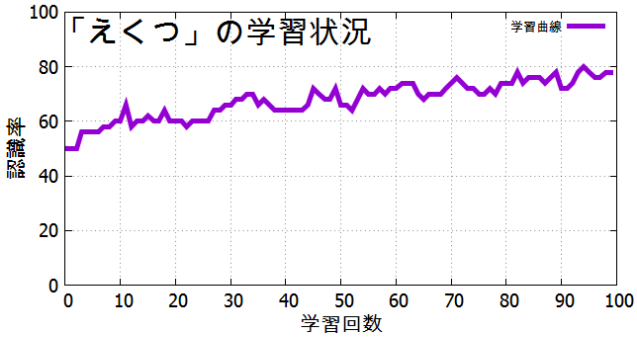


**Fig. 5　「えくつ」の学習結果**

について単独で採取した音声データを 40 回程度学習させた場合の「くとせ」と「らめか」についてのグラフを **Fig.3** 及び **Fig.4** に示す．

「くとせ」については振動しながらではあるが，ほぼ直線的に単調増加している．それについては「またあ」，「ごまた」，「えくつ」，「けしみ」，「めぐた」，「のちた」も同様の変化を見せた．しかし「らめか」については減衰振動の様な変化が見られた．いずれも全体として軌跡が振動を繰り返し，収束する様子は見られない．

また人間の音声をテストデータとした「えくつ」についてのグラフは **Fig.5** に示す様に振動が比較的なく，なだらかな単調増加となった．

### 3.3. 考察

学習した 8 つの非語の認識率の推移のグラフを見る限り，すべてにおいて最大認識率が 80 %台に達する結果となった．

テストデータに音声合成音を利用する場合，音声合成の音声はまれに発音が不適当なものがあり不安定である．音声合成音をテストデータに使用したことで，各々の学習曲線の軌跡が乱れた可能性が高い．「えくつ」については人間の音声をテストデータに利用したので比較的きれいな学習曲線が得られた．

また，学習回数がある程度になると認識率の伸びが止まり降下することがある．これは過学習によるものだと考えられる．

## 4. Deep Learning による実験

機械学習に SVM を使用した場合，認識率は 80%に留まった．しかし実用的なレベルの認識率として意味単語の認識率と同様の 95%前後が必要である．そこで SVM を Deep Learning に差し替えて実験を行った．

Deep Learning は SVM に比べ，より複雑なアルゴリズムで出来ている．基本構造は人間の脳内のニューロンを模したユニットからなるニューラルネットワークを並列・多層に連結させたものであり，特徴量の抽出を自らが行うことが最大の特徴である [10]．様々なユニットの組み合わせによるネットワークのモデルは新しいものが次々と提案されているが，非語の音声認識に適応したネットワークモデルを構築することで SVM よりも高い認識率を得ることが期待できる．

### 4.1. 方法

Deep Learning を機械学習に使う実験では学習させる音声の特徴量を複数選択可能とし，それぞれの組み合わせによる学習で認識率向上に寄与する最適なものを探索することに重みを置いた．また認識率の変動をすぐに確認できるようグラフの多重プロットによる比較を行った．機械学習は Python のライブラリである Chainer [11]を使用した．Chainer は音声認識のためのニューラルネットワークをシンプルに実装できる柔軟性と直感性を備えており，GPU で計算をすることにより，おおよそ CPU の 40 倍程度の速さで学習を行うことが出来る．

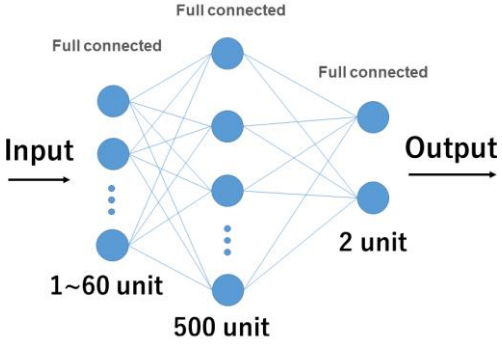


**Fig. 6　構築したネットワークモデルの図**

構築したネットワークモデルは全結合層 3 層のシンプルな構成で，ユニット数は，入力層を選択した特徴量に応じて 1 から 60 までの可変長とし，中間層を 500，出力数を 2 値分類のために 2 とした．構築したネットワークモデルの図は **Fig.6** の様になる．

すべての操作を担うインタフェースを Python の GUI ライブラリである PyQt で作成した．複数の特徴量をチェックボタン形式で選択し，選択した組み合わせでの学習に切り替えて実行できる機能を備えた．

Deep Learning の学習方法は逐次学習法ではなく，学習データを一回で全て読み込んだ後パラメータ調整をする試行を 1epoch とし，200epoch まとめて行った．

学習用データは「えくつ」について SVM による実験で使用した音声データに新たに採取した 300 サンプルを付け加えたものであり，それを利用して「えくつ」について学習を行った．

実験手順はまず始めに MFCC を代表値として単独で学習させる．その後，新たに動的時間伸縮法（DTW） [12]の結果も特徴量として学習させる．DTW は見本の音声と学習データの音声の波形パターンを伸縮同期させ，波形の類似度を算出する手法である．例えば「えくつ」の見本の音声と採取した音声の MFCC の 3 次元要素に DTW を施すと **Fig.7** の様になる．実験では MFCC の 25 次元のそれぞれの波形同士の連なりのうち最も近接した距離を学習させた．

ところで Julius には別機能としてセグメンテーション・キットがある．セグメンテーション・キットは見本の音素列を参考に学習データの音素列を時間単位で分離しそれぞれの音素の類似度を見積もる機能である．


**Fig. 7　「えくつ」についての DTW の結果**

| 開始時間 | 終了時間 | 類似度 | 発話言語 |
|---|---|---|---|
| 0.000 | 0.042 | -34.376373 | silB |
| 0.042 | 0.142 | -25.251791 | E |
| 0.142 | 0.172 | -33.933491 | k |
| 0.172 | 0.312 | -29.119303 | u |
| 0.312 | 0.522 | -27.422319 | ts |
| 0.522 | 0.612 | -23.890150 | u |
| 0.612 | 0.792 | -24.736803 | silE |

**Table 2　「えくつ」についてのセグメンテーションの結果**

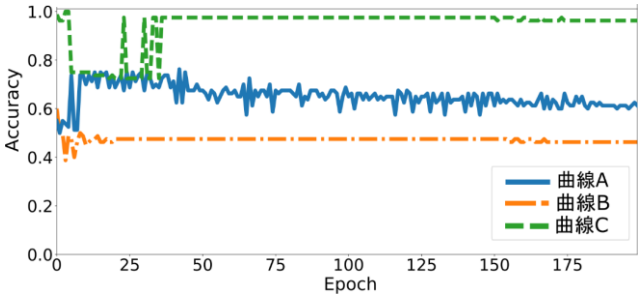


**Fig. 8　特徴量の組み合わせと認識率の関係**

**Table 2** はセグメンテーション・キットを使い「えくつ」について任意の学習データにセグメンテーションを行ったときに出力されたログである．「えくつ」は母音と子音の音素に分割され，それぞれの音素について発音開始時間と発話終了時間の情報，および見本の音素との類似度の情報が付加されている．

このセグメンテーションの結果で，ログの中の類似度のみを読み取り特徴量として学習させる．

SVM で学習させた Julius の出力する特徴量に DTW とセグメンテーションの結果を含め，MFCC，音響モデルと言語モデルの尤度からなる Julius 独自のスコア，発音時間，およびモーラ数の計 6 つの特徴量の組み合わせについて学習し，それぞれの動向を見て，認識率の向上に寄与する特徴量の最適な組み合わせを探索した．

### 4.2.　結果

6 つの特徴量の候補の内，Julius 独自のスコアと発音時間，およびモーラ数を学習させた結果，認識率の動向に殆ど差は出なかった．しかし MFCC と DTW，およびセグメンテーションの組み合わせを学習させた場合には認識率の動向に差が生じた．**Fig.8** に MFCC と DTW，およびセグメンテーションを学習させる特徴量の候補として選択した場合の認識率の推移を示す．曲線 A の様に MFCC 単独の学習では認識率が 60%程度に留まった．次に MFCC に加え DTW の結果を学習させたところ，曲線 B の様に認識率が 50%程度に降下した．最後に MFCC に加えセグメンテーションの結果を学習させたところ，曲線 C の様に認識率が最大 97%に上昇した．

### 4.3.　考察

6 つの特徴量の候補のうち，Julius 独自のスコアと発音時間，およびモーラ数は認識率の動向に影響を及ぼすことないと考えられる．DTW は認識率を降下させる要因であり特徴量の候補としては不適切と考えられる．現状では MFCC に加え，セグメンテーションの学習が認識率向上の決め手になっていると考えられる．

## 5.　まとめ

非語の音声認識の認識率を向上させるため Julius に機械学習の機構を組み込む実験を行った．機械学習は SVM と Deep Learning の両方のライブラリを試した結果，SVM において最大認識率 80%を得た．

Deep Learning においては，学習させる 6 種類の音声特徴量の候補を挙げ，最適な組み合わせを探索した．その結果，MFCC にセグメンテーションの結果を付け加えた組み合わせによる学習で最大認識率が 97%になった．

## 6.　今後の課題

DTW は学習させる特徴量としてではなく適切なフィルタとして Julius の前後に置く．

フィルタとして同様の機能をはたす可能性のあるものとして，フォルマントによる母音列の絞り込みを考える．その場合まず非語を構成する音素内の母音区域を特定する手法に課題があるが，Julius のセグメンテーション機能が利用できる可能性があるので試行する．

ELC の検査で使用される他の非語の学習について，音素列の違いにより認識率の推移に差が出る場合は，それぞれの語について個別に手法を考案する必要がある．

### 謝辞

本研究を進める上で音声データの採取に協力していただいた「山陰発達障害当事者会スモステの会」の方々，ならび島根大学教育学部のボランティアの方々に御礼申し上げます．

### 参考文献

[1] 加藤醇子，安藤壽子，原恵子，縄手雅彦, “読み書き困難児のための音読・音韻処理能力簡易スクリーニング検査 ELC Easy Literacy Check”, 図書文化, (2016).

[2] 中川聖一, “「ここまでできる音声ドキュメント処理技術」”, 平成 24 年度 電気関係学会東海支部連合大会, S3-1, (2012).

[3] 河原 達也, “[特別講演] 音声認識技術の展開”, IEICE technical report : 信学技報, Vol.115, No.388, pp111-116, (2015).

[4] 荒木 雅弘, “フリーソフトでつくる音声認識システム”, 森北出版, (2007).

[5] National Taiwan University, “LIBLINEAR - A Library for Large Linear Classification.”, https://www.csie.ntu.edu.tw/~cjlin/liblinear/

[6] National Taiwan University, “LIBSVM - A Library for Support Vector Machines.”, https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[7] 名古屋工業大学, “Julius now on GitHub.”, http://julius.osdn.jp/

[8] 名古屋工業大学, “OpenJtalk”, http://open-jtalk.sp.nitech.ac.jp/

[9] 名古屋工業大学, “メイちゃんの公式ウェブサイト”, http://mei.web.nitech.ac.jp/

[10] 斎藤 康毅, “ゼロから作る Deep Learning-python で学ぶディープラーニングの理論と実装”, オーム社, (2016).

[11] Preferred Networks, “Chainer”, https://chainer.org/

[12] haripo, “Dynamic Time Warping（動的時間伸縮法）”, https://haripo.com/articles/2017/dynamic-time-warping/

[13] 伊福部 達ほか, “進化するヒトと機械の音声コミュニケーション”, エヌ・ティー・エス, (2015).