人間からの誤り訂正フィードバックに基づく適応的なEnd-to-End日本語音声合成に関する検討 *

☆藤井一貴, 齋藤佑樹, 猿渡洋 (東大院・情報理工)

1　はじめに

テキスト音声合成（Text to Speech: TTS） [1] は，テキストを入力としそれに対応した明瞭で自然な音声を合成する技術である．End-to-End TTS [2] は，従来の統計的パラメトリック音声合成 [3] におけるパイプライン処理を用いず，テキストからの音声予測をDeep Neural Network (DNN) による推論に置き換えた一気通貫方式に基づく手法である．この手法により音声合成に必要な専門知識が必ずしも必要でなくなったため，非専門家でも肉声に近い音声を合成できるようになりつつある [4]．しかし，End-to-End TTS はモデル全体が DNN の積み重ねで表現されているためモデルの解釈性が低く，合成音声に誤りがあった場合非専門家を含むユーザーが修正を行うことは困難である．日本語をはじめとするピッチアクセント言語はアクセントが変わると意味も変わるため，合成音声にアクセント誤りが発生すると音声の意味が変わり，適切なコミュニケーションが取れなくなる．Kurihara ら [5] は，テキスト解析由来の韻律情報を入力して合成音声の操作性を向上させる手法を提案した．合成音声の操作性向上のための検討は行われているものの，誤り訂正を容易にするための適応性に関してはまだ検討されていない．合成音声の操作性と適応性は，人間とコンピュータが音声を主要なコミュニケーション手段として利用する高度な社会を実現するために不可欠な要素である．

本研究では合成音声のアクセント誤りに着目し，人間からのフィードバックにより容易に誤り訂正可能な End-to-End TTS を提案する．本手法は，人間にとって解釈容易な韻律記号を予測する韻律予測器を導入し，合成音声の自然性と制御性を両立させる．また，合成音声の適応性を改善するために，合成音声のアクセント誤りを集合知を用いて修正する Human-In-The-Loop (HITL) フレームワークを提案し，クラウドソーシングで多数の作業者を対象とした実験によりその有効性を検証する．ピッチアクセント言語に対する評価の結果，提案手法は韻律予測器によるアクセント予測誤差が合成音声の品質を著しく低下させるが，我々の HITL フレームワークによりこれらの誤差をうまく補正し品質向上に貢献できることが示された．

2　関連研究

End-to-End TTS ではテキストから音声を生成する単一の DNN を学習するため，音声合成システム全体としての最適化が可能であり，従来方式よりも高い品質の音声を合成できる．ここで，合成対象の言語に着目すると，日本語をはじめとするピッチアクセント言語はアクセントの違いにより同じ読みで異なる意味を持つ単語を区別している．例えば,「箸」と「橋」,「端」は全て同じ「はし」という読みであるが，異なるアクセントにより意味を区別している．従って，アクセントの誤りが発生すると言葉の意味が変わってしまい，正しいコミュニケーションを取ることが困難となる．ここで，End-to-End TTS の社会実装が進んだ先の実際の応用について考えると，TTS を利用した人間同士のコミュニケーションが活発に行われることが考えられる．そのような将来を見据えて TTS を用いたコミュニケーションに着目すると，TTS は発話誤りを起こした際の修正能力を持たないため，間違えることによるコミュニケーションの阻害が起こってしまう．TTS の誤りを訂正することを考えても，人間はニューラルネットワークの積み重ねとして表現されるブラックボックス化された End-to-End TTS システムを完全に解釈することはできないため，合成音声に誤りが発生した際の制御が困難である．そのため，特にピッチアクセント言語における End-to-End TTS は操作性と適応性が必要となる．

End-to-End TTS における操作性を向上させるための取り組みがこれまでに行われている．Kurihara ら [5] は，音素と韻律を表す単一の情報を入力とした，音質と韻律の制御性を向上させる手法を提案し，誤り削減のための韻律制御性を向上させた．しかし，End-to-End TTS において誤りが発生することを前提とした誤りへの対処法を論じた研究は少なく，体系的な方法は確立されていない．

3　提案手法

End-to-End TTS の制御性と適応性の両方を向上させるために，合成音声の正解アクセントが存在しないなど，より実際のアプリケーションに近い状況で使用できる音声合成システムを提案する．具体的には，韻律記号 [5] を導入し，テキストから韻律記号を予測する韻律予測器を新たに導入する．また，集合知を用いてアクセント誤りを修正する HITL フレームワークを提案し，韻律予測器の誤りを修正することで適応性を向上させる．

3.1　ベースラインとなる TTS モデル

提案手法の概要を Fig. 1 に示す．本研究では，学習・推論の速度と推論の安定性を重視し，非自己回帰型 End-to-End TTS である FastSpeech 2 [6] を提案手法のベースライン TTS モデルとして採用した．FastSpeech 2 のモデルは，(1) 音素埋め込み，(2) 文脈を考慮した単語埋め込み，(3) 韻律予測器の出力として得られる韻律埋め込みによって条件付けされる．

---

[*] Investigation of Adaptive End-to-End Text-to-Speech Synthesis Based on Error Correction Feedback from Humans, by FUJII Kazuki , SAITO Yuki, SARUWATARI Hiroshi (UTokyo).

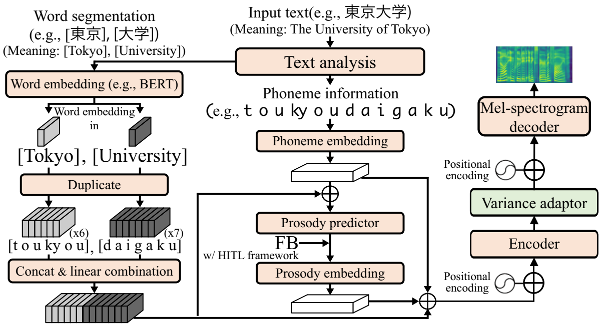

Fig. 1　提案手法の TTS モデルに関する概要図．韻律予測器は，音素埋め込みと BERT による文脈を考慮した単語埋め込みから韻律記号を予測する．

Table 1　韻律記号の定義

記号	定義	韻律予測器の出力値
[	上昇	$+1$
]	下降（アクセント核）	-1
_	変化なし	0

3.2　韻律予測器

提案手法に導入された韻律予測器は，テキストのアクセントの変化を表す韻律記号列を出力する DNN である．この DNN は，Fig. 1 の右側に記載されている TTS モデルのベースラインとなる FastSpeech 2 [6] の Variance Adaptor 内に記述されている，ピッチやエネルギーなどを離散化した特徴表現を予測する予測器と同じアーキテクチャを採用している．韻律予測器によって予測された韻律に対して TTS モデルを条件付けすることで，合成音声の制御性が向上する．

3.2.1　入力：音素埋め込みと文脈を考慮した単語埋め込み

韻律記号予測において入力テキストの文脈情報を考慮するために，音素埋め込みに加えて BERT [7] から得られる単語埋め込みを韻律予測器の入力として使用する．BERT とその Tokenizer を用いて入力テキストを単語ごとに分割し，各単語について単語埋め込みを得る．本研究では，Tokenizer で分割された各サブワードから埋め込みを抽出するため，単語を構成する複数のサブワード埋め込みを平均して一つの単語埋め込みを得る．音素埋め込みと単語埋め込みの異なる時間分解能を揃えるため，単語埋め込みは単語中の音素の数と同じだけ重複させる．最後にこれらの埋め込みの要素和をとり，韻律予測器への入力として使用する．

3.2.2　出力：解釈可能な韻律記号

韻律予測器の出力値を Table 1 に示す．Kurihara ら [5] が用いた韻律記号のうち，人間がアクセントを付与する際に意識するピッチ上昇，ピッチ下降の記号のみを本研究では採用した．ここで，FastSpeech 2 ではテキストに付与された韻律記号に対する継続長が定義できない．そこで，音素列と同じ長さを持つ韻律記号列を得るために，アクセントに変化がないことを意味する記号 “_” を新たに導入した．韻律予測器が予測すべき正解を得るために，OpenJTalk [8] を用いたテキスト解析を行った後に，Kurihara らのアルゴリズムを適用することで正解の韻律記号を抽出した．また，実際には韻律予測器の出力は $\{+1, -1, 0\}$ のいずれかであり，-1 がピッチ下降，$+1$ がピッチ上昇，0 がアクセントに変化なしを表す．韻律予測器の学習における損失は，予測された韻律記号と正解の韻律記号の平均二乗誤差であり，後者はテキスト解析の結果や人間によるフィードバックとして得ることができる．本手法では，End-to-End の TTS モデル全体が DNN として記述されるため，単一の誤差逆伝播アルゴリズムに基づく学習が可能である．

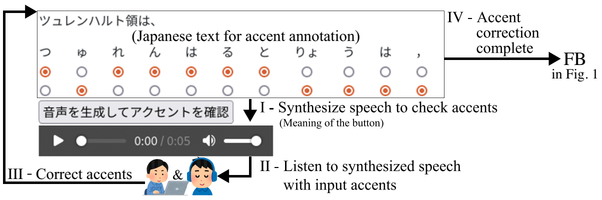

Fig. 2　提案手法の HITL フレームワークに関する概要図．韻律予測器は，音素埋め込みと BERT による文脈を考慮した単語埋め込みから韻律記号を予測する．

3.3　HITL アクセント誤り訂正

提案法における集合知を用いた HITL フレームワークは，誤り訂正に人間を取り入れることで適応性を向上させることを目的としている．HITL における誤り訂正フィードバックは，クラウドソーシングを利用して対象文にアクセントアノテーションを行うクラウドワーカーを募集することで成立する．日本語のアクセントは音の高低で表現され，最小単位はモーラである．そのため，本手法では文章をモーラ単位に区切り，それぞれに対してアクセントを付与させる．このとき，モーラ単位の情報だけでなく生テキストも表示する．加えて，クラウドワーカーのほとんどは音声合成の専門家ではなく，不自然な音声となり得るアクセントの付与を行う可能性があるため，誤り訂正の信頼性を高めることが必要である．そこで，入力されたアクセントを用いて実際に音声合成を行い，クラウドワーカーに聴覚的なフィードバックを行うことで誤り訂正の信頼性を高めている．

誤り訂正フィードバックのインタフェースと概要図を Fig. 2 に示す．クラウドワーカーは，アクセントの高低をモーラごとにラジオボタンで選択してアクセントの付与を行う．ラジオボタンにはテキスト解析由来の高低を示す韻律が初期状態として設定されている．このとき，クラウドワーカーは入力されたアクセントで合成された音声を聞くことができ，アクセントを修正することができる．ここで，End-to-End TTS の実応用を考えた時，合成したいテキストに対応する音声が存在するケースは稀であるため，収集されたクラウドワーカーによる訂正済み韻律列は，以下の手法を用いて 1 つの韻律列に統合を行い検討する．また，この際モーラ単位の韻律列を音素単位の韻律列に変換する．

- **Mode**: 各モーラごとにアクセント高低の最頻値をとる手法
- **MACE**: 期待値最大化法に基づく Multi Annotator Competence Estimation (MACE) [9] によ

る手法．未知の回答とそれを回答するワーカーの能力を交互に推定し，能力の低いワーカーの影響を排除することによって 1 つの韻律列に統合

4　評価実験

4.1　TTS モデルの実験条件

本実験では，単一の女性話者により構成される JSUT コーパス [10] の BASIC5000 サブセットを用いた．音声データは 22,050 Hz にダウンサンプルされ，メルスペクトログラムの次元は 80 とした．また，F0 分析には WORLD [11] を用いた．学習データと評価データの数はそれぞれ 4,488 文，512 文とした．本研究で使用する FastSpeech 2 は，GitHub 上の実装[1]を基にした．音素アライメント情報は Julius [12] で取得した．テキスト解析の辞書には，多様なテキストに対して正しいアクセントを推定可能にするために tdmelodic [13] を導入した．FastSpeech 2 の埋め込み表現の次元は全て 256 に設定し，DNN 学習の Optimizer は学習率が 0.001 の Adam [14] とした．ニューラルボコーダには HiFi-GAN [15] を使用した．BERT のモデルには，bert-base-japanese-v2[2]を用いた．また，単語埋め込み次元は線形変換により 256 次元に圧縮を行う．比較手法として，以下の 3 手法を提案手法と比較した．

- **FS2**: FastSpeech 2 を韻律で条件付けせず学習した手法
- **FS2+PS**: FastSpeech 2 をテキスト解析由来の韻律で条件付けして学習した手法
- **FS2+PP(GT)**: FastSpeech 2 を提案手法の枠組みで学習を行うが，韻律予測器が予測した韻律を使用せず，テキスト解析由来の韻律を用いる手法

音声を合成する際に使用する各手法のモデルの学習ステップは 100,000 とした．

4.2　HITL フレームワークの実験条件

集合知を用いた HITL フレームワークの実験には，クラウドソーシングプラットフォームであるランサーズ [16] を用いてワーカーを募集した．本実験では，JSUT コーパスのうち，VOICEACTRESS100 サブセットを用いた．VOICEACTRESS100 は，BASIC5000 には含まれない固有名詞や造語などのアクセントの推定が困難な文章を多く含むため，アクセント誤りの修正の妥当性を示すのに十分な修正フィードバックを得ることができる．本実験では，VOICEACTRESS100 の全文に対し，1 文あたり 15 人のクラウドワーカーがアクセントを付与するように実験を実施した．収集された 1 文あたり 15 個の韻律列は，Mode と MACE を用いて 1 つの韻律列に統合を行い検討を行う．さらに，クラウドワーカーより得られたアクセントに関する評価のために，収集された 15 個の韻律列全てを使用し音声を合成し，それらと正解音声の対数基本周波数 (logF0) の Root Mean Squared Error (RMSE) をとり，値が最も小さいサンプル (Best)，中央に位置するサンプル (Median)，最も大きいサンプル (Worst) に関しても検討を行う．

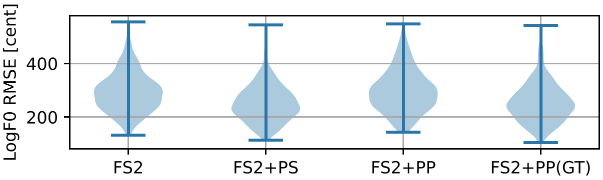

Fig. 3　TTS モデルに関する客観評価実験結果

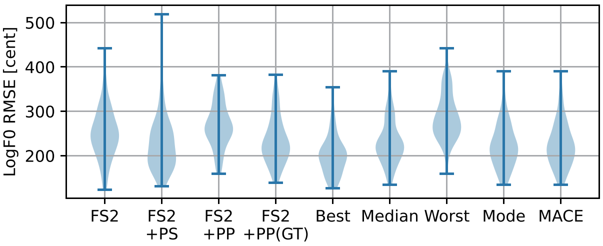

Fig. 4　HITL フレームワークに関する客観評価結果

4.3　客観評価実験

End-to-End TTS の韻律予測精度について客観評価を行った．本実験ではピッチ予測精度にのみ着目するため，音声合成時に自然音声の音素継続長を使用する．また，評価基準として logF0 RMSE [cent] を用いる．Fig. 3 に客観評価結果を示す．FS2+PP は FS2+PS よりも性能が低く，FS2+PP(GT) は FS2+PS と同程度の結果を示していることがわかる．これらの結果から，韻律予測器の予測誤差が合成音声の品質を大きく劣化させるが，正しい予測により自然な韻律を持つ音声を合成することができることがわかる．

続いて，HITL フレームワークの客観評価実験を Fig. 4 に示す．FS2+PP は FS2 や FS2+PS よりも悪いスコアを示しているが，FS2 や FS2+PS で悪いスコアを示していたサンプルに改善傾向が見られ，外れ値の予測性能が向上していることがわかる．これは，新たに韻律予測器を導入し，音響モデルとともに学習を行った影響によるものだと考えられる．また，Best, Median, Worst の結果から，クラウドワーカーのスキルに大きな違いがあることが確認された．ここで，FS2+PP(GT) で使用される韻律とアクセント付与インタフェースの韻律初期値は同等であるが，この結果は FS2+PP(GT) よりも Worst の方が明らかに品質が劣化している．これは，クラウドワーカーのアクセント付与に対する認識の齟齬が起こった結果という可能性が考えられる．さらに，Mode と MACE 間の違いは少なく，これらは正解データを用いる FS2+PP(GT) と同等の結果をもたらすことが示唆された．

4.4　主観評価実験

続いて，合成音声の韻律自然性に関する主観評価実験を行い，客観評価で示された傾向が主観評価評価でも同様かを確認する．主観評価実験は全てランサーズを用いて実施した．

4.4.1　TTS モデルに関する主観評価実験

プリファレンス AB テストと MOS テストを実施する．プリファレンス AB テストでは，受聴者に対して比較手法もしくは提案手法で合成した 2 音声をラン

---

[1] https://github.com/Wataru-Nakata/FastSpeech2-JSUT

[2] https://huggingface.co/cl-tohoku/bert-base-japanese-v2

Table 2　合成音声の韻律自然性に対するプリファレンス AB スコア．**太字**は 2 手法の間に有意差があることを示す ($p < 0.05$)．

比較手法	Preference score
FS2 vs. FS2+PP	**0.552 vs. 0.448**
FS2 vs. FS2+PP(GT)	**0.268 vs. 0.732**
FS2+PS vs. FS2+PP	**0.768 vs. 0.232**
FS2+PS vs. FS2+PP(GT)	0.520 vs. 0.480
FS2 vs. FS2+PS	**0.248 vs. 0.752**

Table 3　TTS モデルにおける合成音声の自然性に関する MOS スコア．**太字**の値は FS2+PP(GT) と同等の値であることを示す ($p < 0.05$)．

手法	MOS ± 95% 信頼区間
JSUT	4.24 ± 0.139
FS2+PP	2.76 ± 0.143
FS2+PP(GT)	3.35 ± 0.163
FS2+PS	**3.45 ± 0.158**
FS2	2.71 ± 0.157

ダムに提示し，韻律が自然な音声を選択させた．各評価の受聴者数は 25 人で，1 人の評価回数は 10 回である．プリファレンス AB テストの結果を Table 2 に示す．FS2+PP(GT) の方が FS2+PP よりもスコアが良く，FS2+PS と FS2+PP(GT) に有意差がないことから，韻律予測が正しければ提案手法は自然な韻律の音声合成が可能であることが分かった．MOS テストでは，各手法の合成音声をランダムに提示し，その韻律自然性を 5 段階（1：非常に悪い–5：非常に良い）で評価させた．この評価での受聴者数は 50 人で，1 人の評価回数は 20 である．MOS テストの結果を Table 3 に示す．これらの結果は，客観評価実験の結果と同様の結果を示している．また，FS2+PS と FS2+PP(GT) 間には，t 検定の結果有意差がないことが示された．TTS モデルの評価結果より，韻律情報で条件付けすることの有効性と韻律予測器を導入することによる性能劣化がないことが示された．

4.4.2　HITL フレームワークに関する主観評価実験

MOS テストを実施する．この評価での受聴者数は 50 人で，1 人の評価回数は 28 である．MOS テストの結果を Table 4 に示す．t 検定の結果得られた p 値より，提案手法における韻律予測器の予測誤差が合成音声の品質を大きく劣化させることが確認された．さらに，Best，Median，Worst は **4.3** 節の結果と同じ傾向を示し，クラウドワーカーのスキルに大きな差があることの裏付けとなった．また，Mode と MACE の結果には同様の傾向があったことから，クラウドワーカーの大半は高い能力を持ち，低い能力の影響は軽微であることが示された．本実験の結果，HITL フレームワークは韻律予測器の予測誤差をうまく補正し合成音声の品質向上に貢献できることが示された．

5　おわりに

本論文では, 合成音声のアクセント誤りを集合知で訂正可能な HITL の枠組みを構築し，フィードバックにより合成音声のアクセント誤りを容易に修正できる End-to-End TTS を提案した．実験では，提案する HITL フレームワークによって合成音声の誤りをうまく補正することで品質向上に貢献できることがわかった．今後は，フィードバックのユーザーインタフェースや得られた韻律列の統合方法に関する更なる検討を行う予定である．

Table 4　合成音声の自然性に関する MOS スコア．**太字**の値は FS2+PP よりも高い値であることを示す ($p < 0.05$)．

手法	MOS ± 95% 信頼区間
FS2+PP	2.87 ± 0.163
FS2+PP(GT)	**3.54 ± 0.156**
Best	**3.62 ± 0.151**
Median	**3.38 ± 0.156**
Worst	2.84 ± 0.174
Mode	**3.63 ± 0.159**
MACE	**3.49 ± 0.161**

謝辞: 本研究は，JST，ムーンショット型研究開発事業，JPMJMS2011 の支援を受けたものです．

参考文献

[1] Y. Sagisaka, “Speech synthesis by rule using an optimal selection of non-uniform synthesis units,” in *Proc. ICASSP*, New York, U.S.A., Apr. 1988, pp. 679–682.

[2] Y. Wang et al., “Tacotron: Towards end-to-end speech synthesis,” in *Proc. INTERSPEECH*, Stockholm, Sweden, Aug. 2017, pp. 4006–4010.

[3] H. Zen et al., “Statistical parametric speech synthesis,” *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, Nov. 2009.

[4] J. Shen et al., “Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions,” in *Proc. ICASSP*, Calgary, Canada, Apr. 2018, pp. 4779–4783.

[5] K. Kurihara et al., “Prosodic features control by symbols as input of sequence-to-sequence acoustic modeling for neural TTS,” *IEICE Transactions on Information and Systems*, vol. E104.D, no. 2, pp. 302–311, Feb. 2021.

[6] Y. Ren et al., “FastSpeech 2: Fast and high-quality end-to-end text to speech,” in *Proc. ICLR*, Vienna, Austria, May 2021.

[7] J. Devlin et al., “BERT: pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL-HLT*, Minneapolis, USA, June 2019, pp. 4171–4186.

[8] “OpenJTalk,” http://open-jtalk.sourceforge.net/.

[9] D. Hovy et al., “Learning whom to trust with mace,” in *Proc. NAACL-HLT*, 2013, pp. 1120–1130.

[10] S. Takamichi et al., “JSUT and JVS: Free Japanese voice corpora for accelerating speech synthesis research,” *Acoustical Science and Technology*, vol. 41, no. 5, pp. 761–768, Sep. 2020.

[11] M. Morise et al., “WORLD: a vocoder-based high-quality speech synthesis system for real-time applications,” *IEICE Transactions on information and systems*, vol. E99-D, no. 7, pp. 1877–1884, Jul. 2016.

[12] A. Lee et al., “Julius — an open source real-time large vocabulary recognition engine,” in *Proc. EUROSPEECH*, Aalborg, Denmark, Sep. 2001, pp. 1691–1694.

[13] H. Tachibana, Y. Katayama, “Accent estimation of Japanese words from their surfaces and romanizations for building large vocabulary accent dictionaries,” in *Proc. ICASSP*, May 2020, pp. 8059–8063.

[14] D. P. Kingma, J. Ba, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, San Diego, California, U.S.A., May 2015.

[15] J. Kong et al., “HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis,” in *Proc. NeurIPS*, vol. 33, Virtual Conference, Dec. 2020, pp. 17022–17033.

[16] “Lancers,” https://www.lancers.jp/.