読み情報を利用した ニューラル日本語入力誤り訂正モデルの構築と評価

金本 勝吉　　麻生 貴裕
株式会社ベルシステム 24 ホールディングス
{kanemoto.katsuyosh, aso.takahiro}@bell24.co.jp

概要

日本語の入力誤りは読み手にとって負担を強いたり文章の信頼性を損ねたりするため，計算処理によりその訂正を支援することが求められる．本研究では読み情報を推定させたり，読み情報も含めて訂正させたりするなど，読み情報を利用する多様な条件で Seq2Seq 形式の誤り訂正モデルを構築した．比較実験の結果，モデルに読みを推定させると精度が悪化し，別途読みを与えつつ，2 つの Decoder で原文の訂正と併せて読みの訂正を行うことで誤り訂正精度が向上した．

1　はじめに

コンピューターを用いて日本語を入力する際，キーボード操作や日本語 IME における，かな漢字変換の選択誤りといった操作上の誤りで意図通りに入力できなかったり，入力を意図した文章自体に文法等の誤りが含まれ，結果的に誤った文章として入力されてしまうことがある．

これらの誤った文章は読み手にとって不自然に感じられ，なめらかに読み進めることを阻害するだけでなく，書き手が本来意図した内容が何であるかの推測を強いたり，意図と異なる内容として伝わったり，文章に対する信頼性を下げてしまったりするという問題がある．一方，このような誤りを人の手で確認・訂正することは，継続的な集中を必要とする作業であるため，計算処理により自動化したり支援したりすることが求められる．

このような誤りを含む文章を訂正する試みは，主に外国語の学習支援を目的とした，文法誤り訂正（Grammatical Error Correction: GEC）を中心にいくつかのシェアードタスクが提案され研究されてきた [1]．また，その周辺ではスペルミスの訂正，文章校正の自動化，入力誤りの訂正などのタスクやデータセットが提案されている．これらのタスクは文章の誤り自体だけでなく，修正すべき箇所や正解の定義が難しく，主に提案されているデータセットの作成プロセスに依存したより広範囲な訂正タスクとなっている．

本研究では主に入力誤りを訂正した履歴に基づくデータセットを学習・評価に利用するため，結果的に日本語の入力誤りにフォーカスしたものである．その中でも日本語入力については，その読みをローマ字で入力し，かな漢字変換するのが一般的であったり読みの情報が誤り修正に有用であるという報告 [2] から，読み情報を多様な形で利用するいくつかの訂正モデルを構築して比較評価した．

2　関連研究

2.1　文章の誤りの訂正

この分野の研究において近年はニューラルモデルを用いた手法が主流になっているが，主に分類問題として解く方法と Seq2Seq 形式で解く方法の 2 種類に大別される．

1 つ目は分類器を用いるアプローチである．GECToR[3] は GEC を Token Classification，具体的には入力文章の各 Token に対し，KEEP, DELETE, APPEND, REPLACE に分類するとともに，カンマの追加，複数形への変更，置き換え後の単語といった具体的な訂正内容の分類を並行して行う．これにより単に訂正できるだけでなく，分類結果により具体的にどのような文法的な誤りだったかを示すこともできる．

2 つ目は機械翻訳で利用される Seq2Seq 形式のモデルを利用するアプローチである．原文から翻訳文への出力の代わりに，誤文から訂正文を出力する Seq2Seq モデルを構築するアプローチであり，機械翻訳で用いられているノウハウの多くを利用するこ

表 1　構築条件の一覧

条件名	Encoder (入力)	→	Decoder (出力)
1. baseline	誤文		訂正文
2. text2(fix+yomi)	誤文		訂正文+読み (訂正文)
3. text2{fix, yomi}	誤文		異なる 2 つの Decoder: 訂正文 / 読み (訂正文)
4. text2fix+text2yomi	prefix でタスクを分けたデータで同時に学習: 誤文 / 訂正文		訂正文 / 読み (訂正文)
5. (text+yomi)2fix	誤文+読み (誤文)		訂正文
6. (text+yomi)2(fix+yomi)	誤文+読み (誤文)		訂正文+読み (訂正文)
7. (text+yomi)2{fix, yomi}	誤文+読み (誤文)		異なる 2 つの Decoder: 訂正文 / 読み (訂正文)

とができる [4].

2.2　マルチタスク学習

本研究では，日本語の入力誤りの一因が読みからの変換誤りに起因することから，読みの情報を利用するために，読みを推定したり読みも含めて誤りを訂正したりするマルチタスク学習をすることで改善を目指した．マルチタスクを解くためのアプローチはいくつか提案されており，1. prefix に目的とするタスクを入れる方法．2. 入力や出力をテキストとして連結して複数の情報を入出力する方法．3. Decoder を複数用意し，タスクごとに Decoder を学習する方法などが提案されている．

それぞれ 1 つ目の手法は，T5 と呼ばれるマルチタスクで学習された事前学習モデルにも採用されている手法であり，例えば “translate English to German: ” という prefix を与えた上で翻訳タスクを，“summarize: ” という prefix を与えて要約を学習させることでそれぞれの能力を同時に持ったモデルを獲得している [5].

2 つ目の手法は，出力に性質の異なる複数の情報を与え，それらも含めた入出力を学習する手法である．例えば Encoder-Decoder 構造の音声認識モデルにおいて，Decoder にて音声認識結果のテキスト系列だけでなく言語の判別，音声の開始終了時刻を示すタイムスタンプなど，多様な情報を出力系列に埋め込むことで同時に学習している事例がある [6].

3 つ目の手法は，1 つの Encoder の出力を利用して，個別のタスクに特化した複数の Decoder で出力する手法である．例えば，入力の音声を Encoder で符号化し，その符号情報を入力にして，異なる 2 つの Decoder で音声認識結果，および音声認識と同時に翻訳結果を出力するタスクを組み合わせて学習した事例が報告されている [7].

3　実験

3.1　構築したモデル

読みの情報を併用して入力誤りを訂正するため，表 1 に示すような多様な構成でのモデル構築を行った．条件 2–4 は読みを推定する能力自体をモデルが学習することを期待し，条件 5–7 は読みの情報を別途与えることで読みの情報も利用する能力をモデルが学習することを期待している．

共通して事前学習モデルは retrieva-jp/t5-base-long[1] を利用し，学習には日本語 Wikipedia 入力誤りデータセット (v2)[2] の train 分割データを利用した．また，読みは MeCab(UniDic Lite) にて付与されたカタカナ表記のものをひらがなに変換して利用した．ひらがなに変換したのは予備実験によりカタカナ表記のままより，ひらがな表記に変換して利用した方が概ね良い結果が得られたためである．

ファインチューニングにおいて入力として訂正前の誤文を与える際 prefix に “訂正: ” を付与した．入力に読みも含めて与える場合，この prefix と誤文の後に続く形で “ 読み: ” を続け，その後に読みの文章を与えた．同様に出力について訂正文を出力する際は “訂正文: ” を，読みを出力する場合は “読み方: ” を prefix として出力するように学習した．条件 2, 6 については訂正文と読みを一つの文章として出力するが，文字列として連結して出力するように学習した．その他，学習時のパラメータは各構築条件で共通して，エポック数=2, 勾配累積=2, 学習率=1e-4, バッチサイズ=3 で実施した．

いくつかの構築条件について補足すると，条件 3, 7 における異なる 2 つの Decoder については，1 つの同じ Encoder の出力を，それぞれ独立した 2 つの

1)　https://huggingface.co/retrieva-jp/t5-base-long

表 2 test, gold 分割に対する評価結果 (値は%).

	test 分割				gold 分割			
条件名	適合率	再現率	F 値	GLEU+	適合率	再現率	F 値	GLEU+
1. baseline	75.9	68.5	72.1	97.5	**69.9**	50.0	58.3	96.1
2. text2(fix+yomi)	74.3	57.9	65.1	96.5	62.6	39.6	48.5	95.1
3. text2{fix, yomi}	75.2	65.5	70.0	97.1	65.0	47.2	54.7	95.7
3'. text2{fix, yomi(0.5)}	76.7	66.1	71.0	97.2	68.2	47.1	55.7	95.8
4. text2fix+text2yomi	77.4	65.8	71.2	97.2	67.5	47.1	55.4	95.8
5. (text+yomi)2fix	76.9	68.6	72.5	97.4	68.5	50.6	58.2	96.1
6. (text+yomi)2(fix+yomi)	73.2	68.0	70.5	97.4	64.4	51.0	56.9	96.1
7. (text+yomi)2{fix, yomi}	75.6	69.0	72.1	97.5	68.6	51.3	58.7	96.1
7'. (text+yomi)2{fix, yomi(0.5)}	**78.3**	**71.6**	**74.8**	**97.7**	69.2	**52.4**	**59.6**	**96.2**

Decoder に入力して訂正文，および訂正文の読みを生成する学習を行った．なお，学習時の loss はこれら 2 つの Decoder の loss の合計を利用している．また予備実験において訂正文を生成する能力と訂正文の読みを生成する能力をバランスよく学習できない状況が見られたため，同じ構成においてパラメータ調整として読み（訂正文）を出力する Decoder の loss を 0.5 倍して学習する条件も追加して比較検討した．以降これらを条件 3', 7' と表記する．

条件 4 については誤文から訂正文を生成するデータと，訂正文由来の文章を入力にその読みを生成するデータを prefix にてタスクを分けた上で混合して同時に学習した．

3.2 評価

評価は日本語 Wikipedia 入力誤りデータセット (v2) の test，および gold 分割に対して適合率，再現率，F 値，GLEU+ にて行った．また，評価前にはそれぞれ Unicode NFKC 正規化をかけた．

適合率，再現率の計算は文字単位の追加，削除，置換の最小編集について，誤文からシステムの出力に対する編集と，誤文から正解となるリファレンスに対する編集を比べて編集箇所と内容が一致した編集数を分子に，誤文からシステムの出力の編集数を分母にしたものを適合率，誤文からリファレンスのテキストの編集数を分母にしたものを再現率とした．GLEU+ は GEC の評価に用いられる評価指標で [8]，文字単位での評価を行った．

4 結果

4.1 定量評価

日本語 Wikipedia 入力誤りデータセット (v2) の test, gold 分割における評価結果を表 2 に示す．条件 1 の baseline と比較して読みの推定も含めて学習した条件 2, 3, 3', 4 や条件 6 は悪化，読み情報を入力として与えた条件 5, 7, 7' は同程度〜精度向上する結果となった．

表 3 誤りカテゴリの概要

誤字	1 文字のひらがな・カタカナの入れ替わり
脱字	1 文字のひらがな・カタカナの抜け
衍字 (a)	余分なひらがな・カタカナ 1 文字の挿入
衍字 (b)	直前と一致する余分な 2 文字以上の挿入
転字	ひらがな・カタカナ 2 文字の入れ替わり
誤変換 (a)	同じ読みを持つ漢字の入れ替わり
誤変換 (b)	似た読みを持つ漢字の入れ替わり
その他	その他

条件 2, 6 について，それぞれ一連の文字列の中で訂正文と読みを生成する構成であり，この形式での生成が他の条件より困難であり精度が上がらなかった可能性が考えられる．条件 2, 3, 3', 4 は読みを生成する構成であるが，学習データの特徴量やサンプル数などが不十分で読みが推定できるまで学習できずに精度が上がらなかった可能性が考えられる．一方，条件 5, 7, 7' は読み情報を別途入力として与え，この情報を利用することで精度が向上した可能性があると考えられる．特に条件 7' については，読みの情報を使いつつ最終的な目的タスクの学習が促進されるように loss の調整を行ったことが精度の向上に寄与したと考えられる．

日本語 Wikipedia 入力誤りデータセット (v2) にはマイニングした結果として誤りの分類が付与されている．カテゴリの定義の概要を表 3 に，test 分割に付与されているカテゴリごとの評価結果 (F 値のみ) を表 4 に示す．

この結果より，条件 1 の baseline に対して，特に条件 5, 7, 7' において実験前に期待していた誤変換の精度向上が確認できる．一方誤変換以外の誤字や脱字といったカテゴリについても精度向上が見ら

表 4 test 分割に付与されたカテゴリ別の F 値 (値は%).

条件名	誤字	脱字	衍字 (a)	衍字 (b)	転字	誤変換 (a)	誤変換 (b)	その他
1. baseline	65.0	73.0	86.0	89.0	56.8	69.2	33.0	66.4
2. text2(fix+yomi)	55.1	66.2	81.3	86.1	56.4	61.1	17.9	57.6
3. text2{fix, yomi}	63.4	69.3	83.7	87.5	53.3	67.8	33.1	61.7
3'. text2{fix, yomi(0.5)}	62.3	73.2	84.6	90.8	**60.1**	67.6	31.3	63.0
4. text2fix+text2yomi	63.2	71.2	84.2	**90.9**	55.9	69.1	32.7	61.3
5. (text+yomi)2fix	64.6	73.4	85.6	88.2	57.1	70.7	33.9	66.1
6. (text+yomi)2(fix+yomi)	61.7	71.2	83.9	90.7	53.8	67.5	33.9	67.8
7. (text+yomi)2{fix, yomi}	66.5	73.1	84.6	90.6	50.0	69.8	33.1	66.2
7'. (text+yomi)2{fix, yomi(0.5)}	**68.9**	**76.4**	**86.4**	89.9	60.0	**73.3**	**34.4**	**68.0**

表 5 構築条件 7' における入出力事例

	入出力文章（訂正すべき箇所とその入出力を【入力 → 出力】で示す）	訂正の正誤
例 1	... が多いが【豚国 → 豚国】の消費量の割合が...	誤
	... がおおいが【ぶたこく → ぶたこく】のしょうひりょうのわりあいが...	誤
例 2	... 耐久度が設定されており、【耐久尾 → 耐久値】がなくなると...	誤
	... たいきゅうどがせっていされており【たいきゅうお → たいきゅうど】がなくなると...	正
例 3	ハバードは後になって自らを【書く → 核】物理学者と自称した。	正
	はばーどはあとになってみずからを【かく → かく】ぶつりがくしゃとじしょうした	正
例 4	展示品は 2 万年前から 1700 年まで時代【純の → 順に】並べられ...	正
	てんじひんは 2 まんねんまえから 1700 ねんまでじだい【じゅんの → じゅんに】ならべられ...	正

れ，読みの情報が他のカテゴリの誤り訂正にも寄与した可能性が考えられる．

4.2 入出力事例

最も精度が良かった構築条件 7' における入出力の事例を表 5 に示す．

例 1 においては本来「豚肉」と訂正されるべきものであり，入力時に「ぶたくに」と誤って入力・変換されたものと推察されるが，入力自体が誤りであるため読みの推定に正解を与えること自体困難であることが伺える．このような課題に対しては入力時の変換前の情報や，推定される複数の読み候補の導入が有効であると考えられる．

例 2 は読みの訂正のみが正解した事例である．いくつかのエラーを分析したところ，この例のように元の文章や読みの片方だけ正解する事例も散見された．このことは元の文章での訂正と，読みにおける訂正が独立した Decoder で行われている事が一つの要因として考えられる．

例 3, 4 はそれぞれ正解した事例である．これらの事例だけを見て読み情報が訂正の精度向上に寄与したことを示すのは困難であるが，構築条件 1. の baseline では【書く →(削除)】や【純の → 純の】と誤った事例である．

5 終わりに

本研究では読みの情報を用いる事で日本語の入力誤り訂正モデルの精度向上を目指した．結果的に読みを推定させるより，別途与えた上で読みも含めた訂正を行う事で精度向上を達成することができた．

モデルに読みを推定させる構築条件について，読み推定能力を十分に獲得できなかったため精度が下がったと考えられるが，その能力を獲得できた場合，別途読み情報を与えずとも精度向上することが期待される．これには，例えば読みで構成されたデータセットを併用した事前学習モデルや，より大規模なデータに基づき，読み推定が十分にできるような追加事前学習をしたモデルをもとにファインチューニングすることで達成できる可能性がある．

また，読み情報を与えた条件 5, 7, 7' について，読みも含めた訂正が精度向上に寄与したが，エラー分析結果より読みだけが正解したり，その逆のケースも存在した．条件 3, 3', 7, 7' の構成について，2 つの Decoder は独立して訂正している．これは，誤り訂正を利用する際に 2 のうち 1 つの Decoder だけで良いという計算量上のメリットがあるが，相互結合することで双方の訂正の結果を受けた訂正により精度向上する可能性があると考えられる．

参考文献

[1] Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. The CoNLL-2014 shared task on grammatical error correction. In **Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task**, pp. 1–14. Association for Computational Linguistics, 2014.

[2] 田中佑, 村脇有吾, 河原大輔, 黒橋禎夫. 日本語 wikipedia の編集履歴に基づく入力誤りデータセットと訂正システムの構築. 自然言語処理, Vol. 28, No. 4, pp. 995–1033, 2021.

[3] Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. GECToR – grammatical error correction: Tag, not rewrite. In **Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications**, pp. 163–170. Association for Computational Linguistics, 2020.

[4] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. Approaching neural grammatical error correction as a low-resource machine translation task. In **Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)**, pp. 595–606. Association for Computational Linguistics, 2018.

[5] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. **Journal of Machine Learning Research**, Vol. 21, No. 140, pp. 1–67, 2020.

[6] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In **Proceedings of the 40th International Conference on Machine Learning**, Vol. 202 of **Proceedings of Machine Learning Research**, pp. 28492–28518. PMLR, 2023.

[7] Hang Le, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In **Proceedings of the 28th International Conference on Computational Linguistics**, pp. 3520–3533. International Committee on Computational Linguistics, 2020.

[8] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. GLEU without tuning, 2016. https://arxiv.org/abs/1605.02592.