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改良

田中 佑 †　村脇 有吾 †　河原 大輔 ‡　黒橋 禎夫 †
† 京都大学 大学院情報学研究科　‡ 早稲田大学 基幹理工学部
{ytanaka, murawaki, kuro}@nlp.ist.i.kyoto-u.ac.jp　dkw@waseda.jp

1 はじめに

文章執筆時に発生する入力誤りによって，人はスムーズに文章を読むことを妨げられ，計算機は正しく解析することを妨げられる．人が文章を書く際，入力誤り訂正を行うシステムを用いれば，その発生を抑えることができ，計算機においても，システムを事前に適用することで，解析誤りの減少が見込めるため，NLP においても重要である．

入力誤り訂正の一種であるスペルミス訂正において，ニューラルモデルは成功を収めており [1]，入力誤り訂正システムの構築に有望であると考えられる．ニューラルモデルは多量の学習データを必要とするため，ニューラル入力誤り訂正システムの実現の重要課題は多量の入力誤りとその訂正ペアを用意することである．

我々は Wikipedia の編集履歴から約 50 万文規模の日本語入力誤りデータセット (JWTD v1; Japanese Wikipedia Typo Dataset version 1)[1] を構築した [2]．JWTD v1 は誤字・脱字・衍字・漢字誤変換の 4 種類の日本語入力誤りからなるが，転字 (transposition) [3] といった代表的な入力誤りを収集していなかった．データセット構築手法の評価をクラウドソーシングを用いて行っているが，再現率については評価できていなかった．また，ベースラインとして LSTM モデルを用いた訂正システム実験も行ったが，不十分な精度であった．

本研究では，JWTD v1 で収集していない転字などの入力誤りを新たに収集し，フィルタリングの一部を改良して，約 70 万文規模の JWTD v2 を構築した．提案手法とは別に入力誤りを網羅的に収集し，それを用いて評価を行った．JWTD v2 を用いて，seq2seq 事前学習モデルである BART [4] を学習し，訂正システム実験を行い，より強力なベースラインとした．漢字の読み推定を同時に学習する実験を行い，ベースラインと比較した．最後に入力誤り認識において他の校正システムとの精度比較を行った．

2 関連研究

公開されている日本語入力誤りデータセットは JWTD の他に，GitHub Typo Corpus [5] がある．これは GitHub[2] の commit ログから構築した多言語入力誤りデータセットである．日本語データのサイズは 1,500 文ほどであり，ニューラルモデルの学習には十分なサイズではない．そのため，本研究ではテストセットの一つとして利用した．

BART は Transformer [6] を用いた seq2seq モデルであり，大規模な生コーパスで事前学習した後に，各タスクで fine-tuning するモデルである．Katsumata らは BART を用いて英語の文法誤り訂正を行っており，事前学習済み BART を文法誤り訂正タスクで fine-tuning するというシンプルな手法ながら，既存の SOTA に近い結果が得られたと報告している [7]．我々は文法誤り訂正を入力誤り訂正に近いタスクと考え，ベースラインとして同様の手法で入力誤り訂正実験を行った．

3 日本語における入力誤り

日本語における入力誤りは様々なバリエーションが考えられるが，本研究でデータ構築に利用する Wikipedia の編集履歴には，編集のタグが存在しないため，入力誤り全てを取り出すことは容易ではない．そこで，本研究では日本語における入力誤りのうち，表 1 のような，誤字・脱字・衍字・転字・漢字誤変換（以下，誤変換と呼ぶ）の 5 種類を対象に収集した．ここで，転字・衍字 (b)・誤変換 (b) は JWTD v1 では収集しておらず，本研究で新規に収集

1） http://nlp.ist.i.kyoto-u.ac.jp/?日本語 Wikipedia 入力誤りデータセット

2） https://GitHub.com

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

誤字	兄の部隊【の → に】所属していた兵士で …
脱字	… 組織をもっていること【+ で】知られる。
衍字 (a)	特に免疫力の差などがそう【– う】である。
衍字 (b)	… １９６３年に虫プロに入社【-に入社】。
転字	現在の【こと → とこ】ろ、大滝最後の …
誤変換 (a)	… 全てが大学院に【以降 → 移行】して …
誤変換 (b)	交代【龍 → 理由】が戦死ではなく、…

表 1　本研究が対象とする入力誤りカテゴリ

した入力誤りである．誤字は 1 文字の入れ替え，脱字は 1 文字の抜け，衍字 (a) は余分な 1 文字の挿入，衍字 (b) は直前の文字列と一致する 2 文字以上の余分な文字の挿入，転字は隣接する 2 文字間の転置，誤変換（a）は同一の読みを持つ漢字の入れ替え，誤変換（b）は近い読みを持つ漢字の入れ替えの誤りである．

4　データセットの構築

データセットの構築では，Wikipedia の編集履歴から入力誤りとその訂正文ペアの候補を取り出し，その後，入力誤り訂正ではないと考えられる編集をフィルタリングした．

4.1　Wikipedia マイニング

Wikipedia の編集履歴から，入力誤り訂正の可能性がある文ペアを以下の方法で取り出す[3]．

1. Wikipedia の全ての記事の全ての版において、編集前と編集後の差分をとり、編集箇所を持つ文ペアを取り出す．
2. 編集箇所が以下の条件を満たす文ペアを取り出す．

   誤字　編集箇所における文字単位編集距離が 1，編集箇所の文字数変化が 0，編集箇所の文字は前後ともひらがな又はカタカナ

   脱字　編集箇所における文字単位編集距離が 1，編集箇所の文字数変化が 0，編集箇所の文字は前後ともひらがな又はカタカナ

   衍字 (a)　編集箇所における文字単位編集距離が 1，編集箇所の文字数変化が –1，削除文字はひらがな又はカタカナ

   衍字 (b)　編集箇所がその直前もしくはその直後の文字列と一致する，編集箇所の文字数変化が –1 以下，削除文字は編集箇所の文字数変化が –1 ならば漢字，–2 以下ならばひらがな又はカタカナ又は漢字

   転字　編集箇所における文字単位編集距離が 2，編集箇所における文字単位ダメラウ・レーベンシュタイン距離 [8] が 1，編集箇所の文字は前後ともひらがな又はカタカナ

   誤変換 (a)　編集箇所における編集前後の読みが一致，編集箇所の文字は前後ともに漢字が含まれる

   誤変換 (b)　編集箇所における編集前後の読みが誤字または脱字または衍字 (a) または転字を含む，編集箇所の文字は前後ともに漢字が含まれる

3）　JWTD v1 と同様に，undo や revert された版を取り除く，編集がループしている文ペアを取り除く，編集が推移している文ペアは統合するなどの操作も行っているが，紙面の都合上，本稿ではその詳細は省略する．

編集前	入力	今日はいい [mask] だ。
	出力	今日はいい転機だ。
編集後	入力	今日はいい [mask] だ。
	出力	今日はいい天気だ。

表 2　「今日はいい【転機 → 天気】だ。」を BART モデルで尤度計算する際に与える入出力

4.2　フィルタリング

上記で得られた文ペアには入力誤り訂正ではないと考えられる編集も含まれる．それらを取り除くため，品詞・形態素解析，リダイレクトデータ，言語モデルを使ったフィルタリングを行った．本稿では，JWTD v1 から変更した言語モデルフィルタについてのみ以下に述べ，それ以外は省略する．

JWTD v1 では文字単位 LSTM 言語モデルにおける編集による尤度の差をもとにフィルタリングを行っている．本研究では，モデルを事前学習済み BART[4] に変更してフィルタリングを行った．表 2 のように編集箇所を mask トークンで置換した文を入力として，編集前・編集後の文が出力される尤度をそれぞれ計算する．編集による尤度の増加が大きいものは入力誤り訂正である可能性が高いと考えられる．それぞれの入力誤りの種類で閾値を定め，編集による尤度の差が閾値より小さい文ペアを取り除いた[5]．

4.3　結果

上記の手法を 2019 年 6 月の日本語 Wikipedia に適用し，JWTD v2 を構築した．そのデータサイズを表

4）　詳細は付録 A.3 章．

5）　衍字 (b)・転字・誤変換 (a) にはこのフィルタリングを適用しなかった．衍字 (b) においては精度が十分高かったため，転字においてはデータ量が少なかったため，誤変換 (a) においては「若干 → 弱冠」などの高頻度語から低頻度語への正しい編集が取り除かれてしまったためである．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

カテゴリ	マイニング後	フィルタリング後
誤字	416,877	103,908
脱字	477,531	140,466
衍字 (a)	402,590	145,787
衍字 (b)	26,467	23,891
転字	10,263	9,289
誤変換 (a)	295,562	268,114
誤変換 (b)	84,677	28,514
全て	1,657,125	691,842

表 3　データセットのサイズ

3 に示す．合計で約 70 万文ペアが得られた．

5　データセット構築手法の評価

JWTD v1 ではデータセット構築手法の評価を，クラウドソーシングを用いてデータセットの一部を評価することで行っていた[6]．しかし，その方法はデータセットの適合率のみ評価ができるものであった．入力誤りのうち，特定のカテゴリを対象として収集する JWTD の手法では，含まれていない入力誤りが存在すると考えられるため，再現率も評価することが望ましい．本研究では，再現率も含めて評価するために，入力誤り評価データを作成し，それを用いて評価を行った．

5.1　入力誤り評価データ

特定のカテゴリに制限されていない入力誤りデータを用意できれば，データセット構築手法の再現率を評価できる．そこで，クラウドソーシングを用いて，そのようなデータを収集し，それを入力誤り評価データとした．

クラウドソーシングに用いたデータは，4.1 節のステップ 1 で取り出した編集箇所を持つ文ペアからランダムに選んだ 11,000 文ペアである．クラウドソーシングのタスク内容については紙面の都合上，付録の A.1 章に付す．タスクの結果より 1,127 個の文ペアが入力誤り評価データとして得られた．

5.2　評価結果

入力誤り評価データを用いて，データセット構築手法の評価を行った．前節でクラウドソーシングに用いた 11,000 文ペアに対して，4 章の方法を適用し，収集された文ペアから，データセット構築手法を評価した．

結果を表 4 に示す．v1 が JWTD v1 における手法であり，v2 が本研究の手法を示している．これより，本研究の手法がマイニング・フィルタリング両面で F 値が高いことが確認できる．

データ	適合率	再現率	F 値
元データ	10.2	100.0	18.6
マイニング v2	41.5	81.8	55.1
マイニング v1	41.3	76.2	53.6
マイニング v2 + フィルタリング v2	73.5	60.8	66.5
マイニング v1 + フィルタリング v2	73.5	56.4	63.9
マイニング v1 + フィルタリング v1	74.1	52.2	61.2

表 4　データセット構築手法の評価結果

6　入力誤り訂正システム

JWTD v2 を利用して，入力誤り訂正実験を行った．BART モデルを事前学習し，その後，入力誤り訂正タスクで，fine-tuning を行った．誤変換の訂正には漢字の読みの情報が役立つと考え，fine-tuning 時に漢字の読み推定を同時に学習する実験も行った．紙面の都合上，事前学習については付録の A.3 章に付す．

6.1　fine-tuning

入力誤り訂正タスクでは，評価に使うデータ (入力誤り評価データ・次節で後述の JWTD v2 テストセット) を含まないページから得られた，入力誤り文ペア 664,287 文を訓練データ，1,000 文を検証データとした．漢字の読み推定タスクでは，BCCWJ [9] のコアデータ 40,584 文を訓練データとして用いた．BCCWJ の原文文字列（原文）を入力して，発音形出現形（読み）を出力させるように学習させる．読み推定タスクの入力単位は BCCWJ の短単位を subword 分割したものとした．同時に学習を行うモデルでは，まず，入力誤り訂正タスクと読み推定タスクで fine-tuning した後，入力誤り訂正タスクのみで fine-tuning した．

6.2　評価方法

評価指標は入力誤り訂正の適合率・再現率・F 値とし，それらは訂正前の文と訂正後の文の文字単位最小編集と，訂正前の文とシステム出力文の文字単位最小編集から計算した．

評価には，入力誤り評価データ・GitHub Typo Corpus・JWTD v2 テストセットの 3 つを用いた．GitHub Typo Corpus は，Hagiwara らの分類器での入力誤りである確率 (prob_typo) が 0.9 より大きい日本語データから，英語のスペルミス訂正を除いた，557 文を用いた．JWTD v2 テストセットは JWTD v2 の

6）同様の方法での JWTD v2 の評価を A.2 章に付す．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

データ	モデル	適合率	再現率	F 値
評価データ	BART	71.8	59.2	64.9
	BART w/ 読み	72.2	59.3	65.1
GitHub	BART	53.2	38.0	44.3
	BART w/ 読み	53.7	39.9	45.8

表 5　入力誤り訂正結果

モデル	カテゴリ	適合率	再現率	F 値
BART	誤変換 (a)	78.6	70.0	74.1
	誤変換 (b)	52.2	39.7	45.1
BART w/ 読み	誤変換 (a)	81.4	74.7	77.9
	誤変換 (b)	56.9	44.7	50.1

表 6　誤変換の訂正結果

一部をクラウドソーシングで評価し，入力誤り訂正として「正しい」と分類されたものである[7]．

6.3　実験結果

入力誤り評価データと GitHub Typo Corpus における結果を表 5 に示す．入力誤り評価データでのスコアは，表 4 にあるデータセットの評価結果と比較して，十分高いスコアといえる．GitHub Typo Corpus でのスコアは入力誤り評価データと比較していずれも低い．これは，Wikipedia と GitHub の commit ログではテキストのドメインが大きく異なることが原因であると考えられる．JWTD v2 テストセットのうち，誤変換カテゴリにおける結果を表 6 に示す．いずれにおいても，スコアの向上が確認できる．これより，誤変換の訂正では読み推定タスクを同時に学習することが有効であると考えられる．

7　他の校正システムとの比較

前章で作成した BART を用いた入力誤り訂正システムと，他の二つの校正システム (Microsoft Word[8] と Yahoo! 校正支援 API[9]) で，入力誤り認識精度の比較を行った．紙面の都合上，校正システムの設定等は付録 A.4 章に付す．

7.1　評価方法

二つの校正システムの出力の多くが，訂正箇所の指摘のみであったため，入力誤り訂正ではなく，訂正すべき箇所を推定するタスク（入力誤り認識）で評価した．校正システムが出力した訂正箇所（区間）が，正解の訂正箇所（区間）を含むかどうかを数え上げ，適合率・再現率・F 値を計算する．前章で作成した訂正システムにおいては，入力とシステム出力の差分箇所を訂正箇所とした．評価には入力誤り評価データと GitHub Typo Corpus を用い，Word においてはそれぞれランダムに 100 個サンプルしたもので我々が人手で評価した．

データ	システム	適合率	再現率	F 値
評価データ	Word	76.7	23.0	35.4
	Yahoo!	14.4	9.8	11.6
	BART	81.6	63.8	71.6
	BART w/ 読み	79.9	66.9	72.8
GitHub	Word	85.2	20.2	32.7
	Yahoo!	30.8	9.2	14.2
	BART	70.1	50.3	58.6
	BART w/ 読み	70.3	52.2	59.9

表 7　入力誤り認識結果

テキスト	Word	Yahoo!	BART
… 自衛隊【– が】の装備として …	○	×	○
… 職に【着 → 就】かせた。	×	×	○
【うるっぷ → ウルップ】島以北 …	○	×	×

表 8　入力誤り認識結果の実例

7.2　結果

結果を表 7 に示す．本研究のシステムが，他の校正システムより再現率・F 値において高いスコアであることが確認できる．

入力誤り認識結果の実例を表 8 に示す．誤変換以外の衍字などの入力誤りは，Word が指摘できているものがあったが，誤変換は，本研究のシステムのみが指摘できているものが多かった．最後の例は JWTD v2 では収集していない編集距離の大きい入力誤りであり，本研究のシステムは指摘できておらず，今後の課題である．

8　おわりに

本研究では JWTD v1 が収集していない入力誤りの収集とフィルタリングの一部を変更し，JWTD v2 を構築した．データセットの再現率を評価できるデータを作成して評価を行い，v2 が v1 より精度が高いことを確認した．事前学習モデルを用いて入力誤り訂正実験を行い，十分高い精度であることを確認した．また，読み推定タスクを同時に学習することで，精度が向上することを確認した．最後に，入力誤り認識において，他の校正システムと比較して，本研究のシステムの精度が高いことを確認した．今後はデータセットのさらなる改良と，入力誤り訂正システムを異なるドメインで適用した際の分析・改良に取り組む予定である．

7）　詳細は A.2 章．
8）　https://www.microsoft.com/ja-jp/microsoft-365/word
9）　https://developer.yahoo.co.jp/webapi/jlp/kousei/v1/kousei.html

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

参考文献

[1] Keisuke Sakaguchi, Kevin Duh, Matt Post, and Benjamin Van Durme. Robsut wrod reocginiton via semi-character recurrent neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31, No. 1, Feb. 2017.

[2] Yu Tanaka, Yugo Murawaki, Daisuke Kawahara, and Sadao Kurohashi. Building a Japanese typo dataset from Wikipedia's revision history. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pp. 230–236, 2020.

[3] Yukino Baba and Hisami Suzuki. How are spelling errors generated and corrected? a study of corrected and uncorrected spelling errors using keystroke logs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 373–377, 2012.

[4] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, Online, July 2020. Association for Computational Linguistics.

[5] Masato Hagiwara and Masato Mita. GitHub typo corpus: A large-scale multilingual dataset of misspellings and grammatical errors. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 6761–6768, Marseille, France, May 2020. European Language Resources Association.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[7] Satoru Katsumata and Mamoru Komachi. Stronger baselines for grammatical error correction using a pretrained encoder-decoder model. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 827–832, Suzhou, China, December 2020. Association for Computational Linguistics.

[8] Fred J Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, Vol. 7, No. 3, pp. 171–176, 1964.

[9] Kikuo Maekawa, Makoto Yamazaki, Toshinobu Ogiso, Takehiko Maruyama, Hideki Ogura, Wakako Kashino, Hanae Koiso, Masaya Yamaguchi, Makiro Tanaka, and Yasuharu Den. Balanced corpus of contemporary written Japanese. *Language resources and evaluation*, Vol. 48, No. 2, pp. 345–371, 2014.

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, 2016.

[11] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, 2018.

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

A 付録

A.1 入力誤り評価データの作成

クラウドソーシングを用いて，入力誤り評価データを作成した．

A.1.1 タスク内容

質の良い評価データを作るために，二段階の方法でクラウドソーシングを行った．どちらもそれぞれ 5 人のワーカーに回答してもらった．

一段回目のタスクは，編集がどのような編集かを分類するタスクである．編集前と編集後の文を提示して，「内容の変更・追加・削除」，「表現の変更・追加・削除」，「誤字・脱字・衍字・漢字の変換ミスの修正」，「その他・わからない」の 4 つの選択肢の中から一つ選んでもらった．「誤字・脱字・衍字・漢字の変換ミスの修正」が過半数を得た文ペアを入力誤りに関係するペアとし，二段階目のタスクの対象データとした．

二段階目のタスクは，編集前と編集後の文を同時に提示して，それぞれ書き言葉として自然かどうかを質問した．このとき，提示する際は単に文 A，文 B として提示し，どちらが編集前か編集後かをわからないようにした．選択肢は「A は書き言葉として自然な文であるが，B は不自然な文である」，「B は書き言葉として自然な文であるが，A は不自然な文である」，「どちらの文も自然である」，「どちらの文も不自然である」，「わからない」の 5 つとした．編集前が不自然・編集後が自然という回答が過半数を得た文ペアを入力誤りに関係するペアとした．

A.1.2 結果

一段階目のタスクにおいて，「誤字・脱字・衍字・漢字の変換ミスの編集」が過半数を占めた文ペアは 1,773 個であった．それらを対象に行った二段階目のタスクで，編集前の文が不自然・編集後の文が自然という回答が過半数を占めた文ペアは 1,127 個であった．この 1,127 個の文ペアを入力誤り評価データとした．

A.2 クラウドソーシングを用いたデータセット構築手法の評価

クラウドソーシングを用いて，JWTD v2 の評価を行った．評価対象はランダムに選んだ記事 3,000 ページ中から得られた文ペアを用いた．同じ編集が数多く存在しているページが存在したため，データの偏りを避けるために，選ばれたページ内に同一の編集が 4 つ以上含まれていた場合，そのうちの 3 つのみをランダムに採用した．

A.2.1 タスク内容

A.1 節の二段階目のタスクと同じ，編集前後の文が自然かどうかを問うタスクである．それぞれの文ペアにつき 10 人のワーカーに回答してもらった．

A.2.2 結果

編集前が不自然で編集後が自然という回答の票が過半数を占める文ペアを入力誤りとして「正しい」とすると，「正しい」割合は，誤字が 77.6%，脱字が 70.3%，衍字 (a) が 84.1%，衍字 (b) が 93.5%，転字が 52.5%，誤変換 (a) が 64.3%，誤変換 (b) が 69.9%で，全体では 71.9%であった．

A.3 BART の事前学習

日本語 Wikipedia の約 1,800 万文を利用して BART の事前学習を行った[10)]．事前学習では，Text infilling タスクを行った．Text infilling は，文章の一部が mask トークンに置き換えられた文章から，元の文章を生成するタスクである．トークン単位は，テキストを Juman++で形態素分割した後にそれらを subword 分割 [10] したものを用いた．subword 分割には SentencePiece[11] を利用し，語彙数は 32,000 とした．encoder と decoder がそれぞれ 12 層である，large モデルで事前学習した．隠れ層の次元は 1,024，バッチサイズは 512 とし，25 万 step 学習した．

A.4 他の校正システムの設定

Microsoft Word 日本語における入力誤り認識のみの評価に限定するために，校正結果のうち言語設定が日本語とされている箇所のみを評価に用いた．校正の設定は「文書のスタイル」を「通常の文」に設定し，他は初期設定のままとした．利用した Word のバージョンは「Microsoft Word for Mac 16.37」である．

Yahoo! 校正支援 API Yahoo! 校正支援 API は，校正のカテゴリを 17 種類に分けて出力を行う．本研究においては，それら 17 種類のうち，F 値が最大となるようなカテゴリを選択して，評価を行った．

10) 事前学習済みモデルは http://nlp.ist.i.kyoto-u.ac.jp/?BART 日本語 Pretrained モデル にて公開している．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).