ニューラル文法誤り訂正のための多様な規則を用いる人工誤り生成

古山翔太 [1,2]　高村大也 [1,2]　岡崎直観 [1,2]
[1] 東京工業大学　[2] 産業技術総合研究所
shota.koyama[at]nlp.c.titech.ac.jp, takamura.hiroya[at]aist.go.jp,
okazaki[at]c.titech.ac.jp

1 はじめに

文法誤り訂正は，与えられた文書から綴りの間違いや，単語の誤用などの文法的な誤りを見つけ，正しい表現へと修正する技術である．言語学習者支援や文書校正への応用が期待されており，自然言語処理の重要な課題のひとつである．

文法誤り訂正では，誤り注釈付きの学習者コーパスを教師データとして，ニューラルネットワークのモデルを学習する手法が主流である [1]．さらに，人工誤り生成による人工誤りデータを用いてニューラルネットワークのモデルを事前学習したのちに，学習者コーパスを用いて再学習（ファインチューニング）を行うアプローチの有効性が報告されている [2]．そのため，事前学習に用いる学習データを生成するための人工誤り生成手法の改善は，ニューラル文法誤り訂正における重要な研究課題である．

人工誤り生成は，文法誤り訂正固有のデータ拡張手法であり，単語を置き換える規則や文法誤りを生成する手法を用い，文法的な文に誤りを導入することで，人工誤りデータを構築する．ニューラル文法誤り訂正のための人工誤り生成には，機械翻訳による誤り生成と，誤り生成規則による誤り生成がある．機械翻訳による誤り生成は，系列変換モデルを用いて誤りデータを作成する手法であり，逆翻訳を用いる手法 [3, 4, 5] や，折り返し翻訳を用いる手法 [2, 6]，初級・上級翻訳器を用いる手法 [7] が提案されている．一方，規則による誤り生成では，各規則は特定の種類の誤りのみしか生成できず，文法カテゴリごとに異なる規則を準備する必要がある．このため，英語文法誤り訂正においては，誤りの種類ごとに様々な誤り生成規則が提案されている．

英語を対象とした人工誤り生成において，誤り生成規則は，機能語誤り，語形変化誤り，表記体系の誤り，単語選択の誤り，語順誤りの生成に関するものに大別できる．

機能語誤りは，例えば “I went for Tokyo.” の for（訂正は to）のような，前置詞や代名詞などの機能語の用法に関する誤りである．機能語誤りの生成手法として，冠詞を置き換える手法 [8] や，学習者の誤り傾向を反映し，冠詞や前置詞の誤りを生成する手法 [9, 10] などが提案されている．

語形変化誤りは，例えば “I goed to Tokyo.” の goed（訂正は went）のような，内容語の語形変化における誤りである．語形変化誤りの生成手法として，不可算名詞の数に関する誤りを生成する手法 [11] や，品詞解析の結果に基づき名詞数や動詞変化の誤りを生成する手法 [12] などが提案されている．

表記体系の誤りは，例えば “I want to Tokyo.” の want（訂正は went）のような綴りの誤りや，句読法，複合語，分かち書き，縮約，大小文字の規則に関する誤りである．表記体系の誤り生成手法として，文中のコンマを削除する手法 [13] や，綴り誤り訂正器を逆適用する手法 [14] が提案されている．

単語選択の誤りは，例えば “I lost my flight.” の lost（訂正は missed）のような類義語の誤用や，接辞が異なる語を用いる誤りである．単語選択の誤り生成手法として，品詞解析の結果に基づき接辞に関する誤りを生成する手法が提案されている [15]．

語順誤りは，例えば “I my flight missed.” の my flight の位置（missed の後ろが正しい）のように，文中の語や句の位置に関する誤りである．語順誤りの生成手法として，隣り合う単語を入れ替える手法が提案されている [16, 14]．

以上のように，人工誤り生成のための規則による誤り生成手法は数多く存在している．しかし，いずれの手法も，特定の種類か，あるいは少数の誤り生成規則を適用して人工誤り生成を行っている．この

ため，これらの手法で生成されるデータは，多様な種類の誤りを含む実際の文書と比較して，非常に限られた種類の誤りのみを含む．このような手法で生成される誤りデータを用いて訂正モデルを学習する場合，人工誤り生成が対応していない種類の誤りに対しては，誤り訂正の性能向上が期待できない．

本研究では，英文中の様々な文法誤りに対して誤り生成規則を設計し，それらを組み合わせて多様な誤り文を生成できる人工誤り生成の枠組みを提案する．提案手法では，複数の誤り生成規則を文ごとに異なる確率で適用することで，多様な誤り文を生成する．生成された誤りデータで文法誤り訂正モデルを事前学習し，さらに学習者コーパスで再学習する実験を行う．提案手法の誤り生成規則や適用確率はすべて人手で調整しているが，この方法で生成した誤りデータを用いて事前学習を行ったモデルは，事前学習を行わないベースラインよりも高い訂正性能を示した．特に，CoNLL-14 タスクのテストデータでは，最高性能を達成した．さらに，特定の誤り種類に関する誤り生成規則を人工誤り生成から取り除くことによる影響を評価し，多様な誤り生成規則を用いる人工誤り生成の有効性を検証する．

2 手法

提案手法では，単言語コーパスの各文に対して複数の誤り生成規則を順番に，かつ独立に適用する．単言語コーパスの各文には，SpaCy v2.3 [1] を用いてトークナイズを施し，品詞タグ，係り受け解析タグ，見出し語，固有表現の IOB タグと種類を，単語ごとに付与しておく．誤り生成規則は単語ごとに適用され，ある誤り生成規則が，ある語に対して誤り生成可能であるかは，その語やその周辺の語に付与された情報から判定される．各誤り生成規則は，固有のベータ分布を保持しており，文ごとにしきい値をサンプルする．誤り生成規則は，誤りを生成可能な単語ごとに，一様分布から値をサンプルし，その値がしきい値を超えた場合に誤りを生成する．この仕組みのため，文ごとに異なる割合で誤りが生成され，多様な誤り文を得ることができる．入力文に対してすべての誤り生成規則を適用した文を誤り文，入力文を修正文とし，誤りデータを生成する．

誤り生成規則は，計 188 種類を作成した．このうち，154 種類が機能語誤りに，5 種類が語形変化誤りに，19 種類が表記体系誤りに，6 種類が語順誤りに，2 種類が単語選択誤りに関する誤り生成規則である．その他，記号や頻度の高い機能語の削除を行う誤り生成規則と，マスクトークン予測のための規則がある．誤り生成規則の多くは，条件に合う単語を削除したり，置き換えたりするものである．例えば，前置詞 than に関する誤り生成規則は，前置詞の than に対して確率 0.2 で削除操作を行い，またそれぞれ確率 0.4, 0.2, 0.1, 0.1 で than を to, from, over, beyond に置き換える．単語を挿入する誤り生成規則もあり，例えば，冠詞を挿入する誤り生成規則では，文頭，あるいはある単語間の位置において，左の単語が存在しないか，あるいは左の単語の Penn Treebank 品詞タグが，VB, VBD, VBG, VBN, VBP, VBZ, IN のいずれかであり，かつ右の単語の Penn Treebank 品詞タグが，NN, NNS, JJ, JJN, JJS のいずれかである場合に，a, an, the, this, that, these, those を，それぞれ 0.3, 0.3, 0.3, 0.025, 0.025, 0.025, 0.025 の確率でサンプルし，挿入する．人工誤り生成に用いるソースコードは，GitHub 上で公開した[2]．その他の誤り生成規則の詳細はそちらを参照してほしい．

表 1 学習者コーパス

コーパス名	文数
FCE train [17]	28,350
NUCLE [18]	57,113
Lang-8 [19]	498,325
W&I train [20]	34,308

表 2 単言語コーパス

コーパス名	文数
News Commentary v15	594,269
News Crawl 15	24,334,700
News Crawl 16	18,238,693
News Crawl 17	26,861,081
News Crawl 18	18,113,311
News Crawl 19	33,600,797
Europarl v9	2,295,044

表 3 評価コーパス

検証/評価コーパス	評価尺度 (評価器)	文数 (検証/評価)
BEA-19 valid/test [20]	$F_{0.5}$ (ERRANT)	4,384 / 4,477
CoNLL-13/14 [21]	$F_{0.5}$ (MaxMatch)	1,381 / 1,312
FCE valid/test [17]	$F_{0.5}$ (ERRANT)	2,191 / 2,695
JFLEG valid/test [22]	GLEU	754 / 747

3 実験

再学習に用いる学習者コーパスの種類と規模を表 1 に示す．学習者コーパスの入手は，BEA-19 Shared Task [3] の規定に従う．Lang-8 コーパスは，誤り文と修正文が同一な事例を除いた文のみを学習に用い，Lang-8 コーパス以外のコーパスは，学習時に標本数を 3 倍にする．評価に用いるコーパスの種類と規模，評価尺度を表 3 に示す．事前学習に用いる単言語コーパスの種類と規模を表 2 に示す．表 2 中の

1) https://spacy.io/

2) https://github.com/nymwa/arteraro

3) https://www.cl.cam.ac.uk/research/nl/bea2019st/

表 4　誤り訂正実験の結果と既存手法との比較

	BEA-19 test ($F_{0.5}$)	CoNLL 14 ($F_{0.5}$)	FCE test ($F_{0.5}$)	JFLEG test (GLEU)
既存手法				
Choe ら [12]	69.06	60.33	-	-
Grundkiewicz ら [14]	69.47	64.16	55.81	61.22
Omelianchuk ら [23]	**73.7**	66.5	-	-
Lichtarge ら [6]	73.0	66.8	-	**64.9**
提案手法				
ベースライン	-	56.53	49.91	57.37
+ アンサンブル	63.78	59.24	52.94	58.02
+ リスコア	65.90	64.29	54.47	61.04
事前学習	-	57.34	46.24	58.35
+ アンサンブル	60.28	57.73	46.77	58.55
+ リスコア	62.37	59.48	48.75	60.10
+ 再学習	-	65.69	56.40	62.23
+ アンサンブル	72.19	67.52	57.70	62.72
+ リスコア	72.51	**67.84**	57.87	63.69
+ ドメイン適応	-	61.59	57.59	-
+ アンサンブル	72.57	63.29	59.21	-
+ リスコア	72.76	65.50	**59.05**	-

コーパスは，WMT19 のニュースタスクのもの[4]を用いる．誤り生成は，各エポックごとに行う．すべてのデータは，記号の正規化などの前処理が施されたのち，BPE-dropout [24] が施される．dropout 確率は，誤り文で 0.1，修正文で 0 とする．

ニューラル文法誤り訂正には，Transformer Big [25] モデルを用いる．実装は `fairseq` v0.10.1 [5] を用いる．生成は，幅 12 のビーム探索で行い，スコアは長さ正規化をする．実験結果は，5 モデルの平均値と，アンサンブル，RoBERTa Large を用いるリスコア [26] によるものを報告する．

4　結果

実験結果を表 4 に示す．ベースラインでは，学習者コーパスを用いて 50 エポック訓練する．事前学習では，単言語コーパスに誤り生成を施した誤りデータを用いて 20 エポック訓練する．再学習では，学習者コーパスを用いて 30 エポック訓練する．ドメイン適応は，再学習を行ったのちに，評価データとドメインの近い訓練データで追加の学習を行うことである．ドメイン適応では，事前学習済みモデルを学習者コーパスすべてで 5 エポック訓練し，さらに，ドメインが同じ学習者コーパスのみを用いて 20 エポック訓練する．ドメインが同じ学習者コーパスは，BEA-19 valid/test に対しては，W&I train，CoNLL 13/14 は NUCLE，FCE valid/test は FCE train である．

4）http://www.statmt.org/wmt19/translation-task.html
5）https://github.com/pytorch/fairseq

表 5　誤り生成規則の有無が性能に与える影響の比較

	BEA-19 valid	CoNLL 14
ベースライン	51.17	64.29
1 カテゴリのみ		
+機能語誤り	50.83	64.44
+活用誤り	50.25	63.94
+表記体系誤り	51.93	64.50
+単語選択誤り	50.87	63.04
+語順誤り	50.00	62.19
1 カテゴリ除く		
-機能語誤り	52.87	65.22
-活用誤り	52.98	63.92
-表記体系誤り	52.38	65.93
-単語選択誤り	53.33	66.01
-語順誤り	52.72	66.54
全誤り生成規則	53.05	66.14

ベースラインと比較すると，提案手法による人工誤りデータを用いて事前学習し，再学習するモデルの性能がすべての評価データセットで高くなっており，多様な規則を用いる人工誤り生成が有効であることがわかる．アンサンブル，リスコアはともに性能向上に大きく寄与している．事前学習済みモデルのスコアは，ベースラインのそれに迫る結果となっているが，アンサンブルやリスコアによる性能向上は小さい．これは，事前学習済みモデルが，ベースラインモデルと比較して，良い言語モデル的性質を持っており，アンサンブルやリスコアによる恩恵を受けにくいためだと考えられる．ドメイン適応による性能向上は，句読法の規約などの，評価データ間で異なっている特徴に適応する効果によると考えられる．NUCLE コーパスは CoNLL-14 タスク [21] の訓練データセットであるが，CoNLL-14 データセットへのドメイン適応の効果は見られなかった．

誤り生成において，特定のカテゴリの誤り生成規則のみを用いる場合と，特定のカテゴリの誤り生成規則を除く場合の性能の変化をカテゴリごとに調べるために，誤り生成規則の 5 カテゴリについて，1 つのみを用いるか，1 つを除く 800 万文で，事前学習を 10 エポック行い，再学習，アンサンブル，リスコアを行った結果を表 5 にまとめた．1 カテゴリの規則のみを用いる場合は，ベースラインと同程度かそれ以下となる．これは，事前学習に用いるデータでの誤りの種類が多様でない場合には，規則による誤り生成が有効ではない可能性を示している．1 カテゴリの規則のみを取り除く場合の性能は，すべての規則を用いる場合と比較して，性能が大きくは変

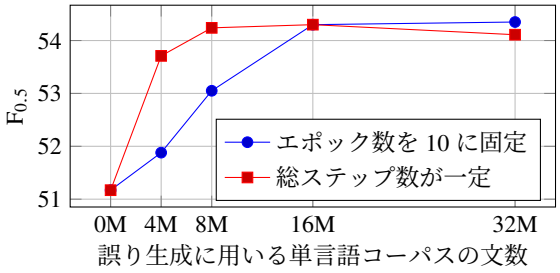

図 1　事前学習データのエポック数・ステップ数による BEA-19 valid データセット上での性能変化

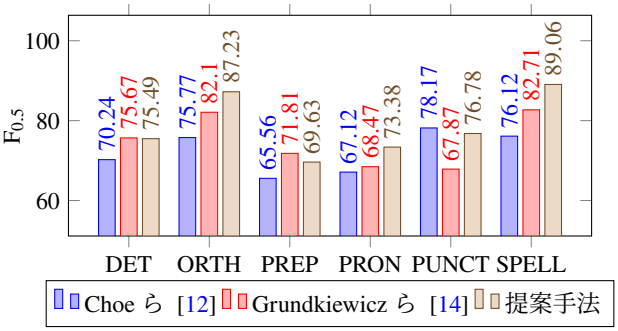

図 2　従来手法との BEA-19 test データセットでの ERRANT 誤り種類ごとの性能比較

わらない．これは事前学習に用いるデータにおける誤りの偏りが小さいことが有効であることを支持している．また，評価データによっては，特定の誤りカテゴリがない方がスコアが高くなる．この原因を調べるためには，各誤り生成規則がスコアに与える影響を評価する方法を検討する必要がある．

逆翻訳を用いる人工誤り生成手法では，事前学習に用いるデータを大きくするとモデルの訂正性能が向上することが確認されている [5]．図 1 の青線「エポック数を 10 に固定」は，事前学習に用いる単言語コーパスの文数を 400 万文，800 万文，1,600 万文，3,200 万文に変化させたとき，エポック数を 10 回に固定することで，総ステップ数が単言語コーパスの文数に比例する条件で事前学習を行い，再学習，アンサンブル，リスコアし，BEA-19 valid データセット上で評価した結果である．実験結果から，既存の報告と同様に，事前学習に用いるデータの規模が性能向上に寄与することがわかる．

しかし，この実験設定では，訂正性能の向上は単言語コーパスの規模の増加によるものか，事前学習の総ステップ数の増加によるものかの区別がつかない．そこで，単言語コーパスの規模を大きくしても学習の総ステップ数がおおよそ一定になるように，エポック数を調整した条件で実験を行った．図 1 の赤線「総ステップ数が一定」は，事前学習に用いる単言語コーパスの文数を 400 万文，800 万文，1,600 万文，3,200 万文と変化させたとき，エポック数をそれぞれ 40, 20, 10, 5 回に設定することで，総ステップ数が同程度になる条件で事前学習を行った結果である．実験結果から，総ステップ数を揃えた場合，単言語コーパスが比較的小規模（400 万文や 800 万文）の場合も，高い性能を示すことがわかる．これは，事前学習に用いる誤りデータでは，同じ文に対してもエポックごとに異なる誤りを生成させているためだと考えられる．また，このことは，単言語コーパスの規模が小さな言語に対しても，適切な誤り生成規則を設計すれば，高性能なモデルを学習できる可能性を示している．

従来の規則による誤り生成手法と，提案手法を比較するために，表 4 の BEA-19 test データセットでの，ERRANT [27] 誤り種類の一部に対しての訂正性能の比較を図 2 に示す．正書法の誤り (ORTH) や綴り誤り (SPELL) では，提案手法は従来手法よりも高い性能を示している．これは，提案手法の表記体系の誤りの生成が有効であることを示している．Choe らの手法 [12] と提案手法は，ドメイン適応を行っているが，句読法の誤り (PUNCT) の性能が高い．これは，句読法はドメインごとに規約が異なるためだと考えられる．冠詞誤り (DET) や，前置詞誤り (PREP)，代名詞誤り (PRON) などの機能語誤りでは，規則による誤り生成を行う従来手法との単純な比較は難しい．機能語誤りのように訂正に前後の文脈を要する誤りの生成においては，規則による誤り生成は機械翻訳による誤り生成と比較して質が劣ると考えられる．その検証は，今後の課題である．

5　おわりに

本研究では，多様な誤りに対して誤り生成規則を定義し，それらを組み合わせた人工誤り生成の枠組みを提案した．実験を通して，多様な規則を用いる誤り生成は，ニューラル文法誤り訂正の事前学習のためのデータ拡張手法として効果的であることが示された．さらに，誤り生成規則の多様性は，文法誤り訂正の事前学習の効果を向上させることがわかった．また，人工誤りデータの生成に用いる単言語コーパスが小規模な場合でも，条件によっては高性能な訂正モデルを学習できる可能性も判明した．

謝辞　本研究成果は，「産総研・東工大 実社会ビッグデータ活用 オープンイノベーションラボラトリ」により得られたものです．

参考文献

[1] 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, 2018.

[2] Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. Corpora generation for grammatical error correction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3291–3301, 2019.

[3] Marek Rei, Mariano Felice, Zheng Yuan, and Ted Briscoe. Artificial error generation with machine translation and syntactic patterns. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 287–292, 2017.

[4] Tao Ge, Furu Wei, and Ming Zhou. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1055–1065, 2018.

[5] Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 1236–1242, 2019.

[6] Jared Lichtarge, Chris Alberti, and Shankar Kumar. Data weighted training strategies for grammatical error correction. *Transactions of the Association for Computational Linguistics*, Vol. 8, pp. 634–646, 2020.

[7] Wangchunshu Zhou, Tao Ge, Chang Mu, Ke Xu, Furu Wei, and Ming Zhou. Improving grammatical error correction with machine translation pairs. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 318–328, 2020.

[8] Emi Izumi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. Automatic error detection in the Japanese learners' English spoken data. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pp. 145–148, 2003.

[9] Alla Rozovskaya and Dan Roth. Training paradigms for correcting errors in grammar and usage. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 154–162, 2010.

[10] Alla Rozovskaya and Dan Roth. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 961–970, 2010.

[11] Chris Brockett, William B. Dolan, and Michael Gamon. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 249–256, 2006.

[12] Yo Joong Choe, Jiyeon Ham, Kyubyong Park, and Yeoil Yoon. A neural grammatical error correction system built on better pre-training and sequential transfer learning. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 213–227, 2019.

[13] Simon Flachs, Ophélie Lacroix, and Anders Søgaard. Noisy channel for low resource grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 191–196, 2019.

[14] Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 252–263, 2019.

[15] Shuyao Xu, Jiehao Zhang, Jin Chen, and Long Qin. Erroneous data generation for grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 149–158, 2019.

[16] Jonas Sjöbergh and Ola Knutsson. Faking errors to avoid making errors: Very weakly supervised learning for error detection in writing. 2005.

[17] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 180–189, 2011.

[18] Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. Building a large annotated corpus of learner English: The NUS corpus of learner English. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 22–31, 2013.

[19] Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. The effect of learner corpus size in grammatical error correction of ESL writings. In *Proceedings of COLING 2012: Posters*, pp. 863–872, 2012.

[20] Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. The BEA-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 52–75, 2019.

[21] 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, 2014.

[22] Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 229–234, 2017.

[23] 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, 2020.

[24] Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1882–1892, 2020.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, Vol. 30, pp. 5998–6008, 2017.

[26] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2699–2712, 2020.

[27] Christopher Bryant, Mariano Felice, and Ted Briscoe. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 793–805, 2017.

表 6　誤り訂正実験の結果と既存手法との比較

	BEA-19 valid			BEA-19 test			CoNLL-13 (valid)	CoNLL-14 (test)			FCE valid	FCE test			JFLEG valid	JFLEG test
	P	R	$F_{0.5}$	P	R	$F_{0.5}$	$F_{0.5}$	P	R	$F_{0.5}$	$F_{0.5}$	P	R	$F_{0.5}$	GLEU	GLEU
既存手法																
Choe ら [12]	63.54	31.48	52.79	76.19	50.25	69.06	-	74.76	34.05	60.33	-	-	-	-	-	-
Grundkiewicz ら [14]	59.1	36.8	53.00	72.28	60.12	69.47	-	-	-	64.16	-	-	-	55.81	-	61.22
Xu ら [15]	-	-	55.37	70.14	57.57	67.21	-	73.0	41.1	63.2	-	-	-	-	-	62.6
Kiyono ら [5]	-	-	-	74.7	56.7	70.2	-	72.4	46.1	65.0	-	-	-	-	-	61.4
Omelianchuk ら [23]	-	-	-	79.4	57.2	**73.7**	-	78.2	41.5	66.5	-	-	-	-	-	-
Lichtarge ら [6]	-	-	-	75.4	64.7	73.0	-	74.7	46.9	66.8	-	-	-	-	-	**64.9**
提案手法																
ベースライン	52.13	28.54	44.73	-	-	-	38.06	69.03	32.83	56.53	51.47	58.56	31.46	49.91	52.57	57.37
+ アンサンブル	58.37	28.47	48.24	70.07	46.91	63.78	38.49	75.05	32.15	59.24	55.20	63.68	31.61	52.94	52.77	58.02
++ リスコア	59.15	33.23	51.17	69.97	53.45	65.90	44.49	72.85	43.74	64.29	55.64	63.78	34.38	54.47	55.55	61.04
事前学習	52.58	22.96	41.79	-	-	-	38.58	68.82	34.40	57.34	43.61	56.53	26.77	46.24	52.97	58.35
+ アンサンブル	53.51	22.77	42.14	65.93	44.89	60.28	38.37	69.41	34.15	57.73	44.58	57.51	26.78	46.77	53.05	58.55
++ リスコア	53.35	27.19	44.74	66.01	51.10	62.37	41.29	67.22	40.71	59.48	46.95	56.77	31.15	48.75	54.79	60.10
+ 再学習	60.61	35.03	52.83	-	-	-	47.87	73.37	46.44	65.69	56.12	64.25	38.02	56.40	56.45	62.23
++ アンサンブル	62.38	34.94	53.91	76.61	58.67	72.19	49.30	75.97	46.73	67.52	57.33	66.32	37.96	57.70	56.79	62.72
+++ リスコア	61.75	37.50	54.68	75.79	61.81	72.51	50.84	73.34	52.18	**67.84**	57.48	65.50	39.48	57.87	58.17	63.69
++ ドメイン適応	58.33	42.36	54.24	-	-	-	43.13	74.96	35.97	61.59	56.69	64.21	40.83	57.59	-	-
+++ アンサンブル	61.11	43.09	56.39	74.89	64.54	72.57	43.53	78.75	35.45	63.29	58.38	66.63	40.95	59.21	-	-
++++ リスコア	60.93	43.75	56.49	74.84	65.51	72.76	48.46	72.13	47.89	65.50	58.51	65.39	42.54	**59.05**	-	-

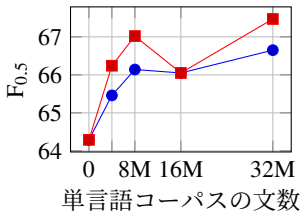

図 3　事前学習データのエポック数・ステップ数による CoNLL-14 test データセット上での性能変化

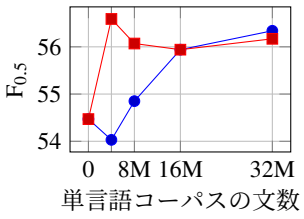

図 4　事前学習データのエポック数・ステップ数による FCE test データセット上での性能変化

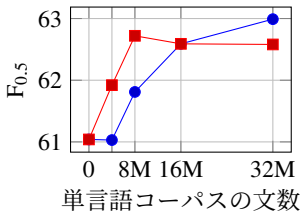

図 5　事前学習データのエポック数・ステップ数による JFLEG test データセット上での性能変化

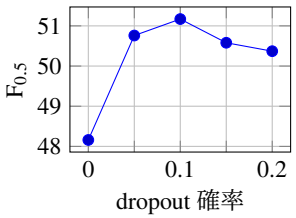

図 6　BPE-dropout の dropout 率による BEA-19 valid データセット上での性能変化

A　実験結果の詳細

表 4 のより詳細な結果を表 6 に示す．再学習を行った場合，ベースラインより特に再現率が高くなる．多様な人工誤り生成を行った誤りデータでの事前学習が，多様な誤りの訂正性能に寄与することが確認できる．ドメイン適応は，評価データ間の規約の差異を埋め，再現率を向上させる．

表 5 の詳細な結果を表 7 に示す．

図 1 と同じ実験を BEA-19 valid データセット以外で行った結果を図 3,4,5 に示す．他のデータセットでも図 1 と同様の結果が確認できる．

BPE-dropout の dropout 確率を変化させたときのベースラインモデルの BEA-19 valid データセットでの性能変化を図 6 に示す．

表 7　誤り生成規則の有無が性能に与える影響の比較

	BEA-19 valid			CoNLL-14		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
ベースライン	59.15	33.23	51.17	72.85	43.74	64.29
事前学習（人工誤り・1 カテゴリのみ）+再学習						
+機能語誤り	59.05	32.65	50.83	73.29	43.44	64.44
+活用誤り	58.49	32.13	50.25	71.69	44.64	63.94
+表記体系誤り	60.30	33.39	51.93	73.15	43.81	64.50
+単語選択誤り	58.80	33.05	50.87	70.45	44.37	63.04
+語順誤り	57.80	32.48	50.00	68.90	44.76	62.19
事前学習（人工誤り・1 カテゴリ除く）+再学習						
-機能語誤り	58.95	37.42	52.87	71.79	47.74	65.22
-活用誤り	59.36	37.05	52.98	69.55	48.28	63.92
-表記体系誤り	60.10	34.61	52.38	74.79	44.74	65.93
-単語選択誤り	61.03	35.45	53.33	74.21	45.78	66.01
-語順誤り	61.13	34.00	52.72	75.45	45.16	66.54
事前学習+再学習	60.03	36.21	53.05	73.21	47.71	66.14