訂正文の流暢性向上を目的とした系列タグ付け文法誤り訂正器の強化学習手法

五藤巧 [1]　渡辺太郎 [1]
[1] 奈良先端科学技術大学院大学
{goto.takumi.gv7, taro}@is.naist.jp

概要

テキストに対して編集のタグ付けを行う文法誤り訂正器である GECToR は，出力の説明性が高い一方で訂正文の流暢性が低い問題がある．この問題の原因として，GECToR がタグの情報のみから最適化されており，推定した訂正文の情報を考慮できていないことが考えられる．本研究では，訂正文の流暢性を考慮してタグ推定の最適化を行うために，GPT-2 言語モデルが与える文の perplexity を報酬とした強化学習を行なった．実験の結果，訂正性能を維持しつつ，訂正文の流暢性が系列変換モデルと同等もしくは上回ることを示した．

1　はじめに

文法誤り訂正は，入力文に含まれる文法誤りや表記誤りを訂正するタスクである．近年では系列タグ付けモデルに基づいた，推論が高速で説明性が高いモデルが提案されている．同モデルでは，入力の各トークンについて編集操作を示すタグを推定し，タグに従って後処理として入力文を編集することで訂正文を獲得する．代表的なモデルである GECToR [1] は，無編集・置換・挿入・削除を示す基本的なタグに加えて，「動詞を過去形に」といった言語情報を考慮したタグを推定する．従来の系列変換モデルに基づく訂正手法が訂正文を直接推定することと比較して，タグの情報が直接ユーザへのフィードバックとして使用できるため有用である．

一方で，GECToR の訂正文は流暢性が低いことが問題である．詳細は 2 節で述べるが，系列変換モデルとの比較を行った結果，訂正性能が同等であるにもかかわらず GECToR の訂正文の流暢性は低いことが明らかになった．Sakaguchi ら [2] によって流暢性の重要性が指摘されていることや，流暢性の低さは GECToR の実応用の幅を狭めることを考慮すると，改善されることが望ましい．ここで，表 1 に Napoles ら [3] に倣い流暢性の高い訂正例を挙げる．最小限の訂正では冠詞 *the* の挿入と，主語動詞の一致の訂正（*affects* → *affect*）が起こっている．一方，流暢な訂正では動詞の語彙選択に関する訂正（*affect* → *cause*）と，それに伴う語順の入れ替えによって大きな書き換えが起こっている．本研究の大局的な目的は，後者のような流暢な訂正の性能向上である．

表 1　最小限の訂正と流暢な訂正の例．誤り文における誤り単語を赤字で，訂正文において訂正された単語を青字で示した．

誤り文	Both of _ hyperparameters affects more conservative correction for the model.
最小限の訂正文 (Minimal edit)	Both of the hyperparameters affect more conservative correction for the model.
流暢な訂正文 (Fluency edit)	Both of the hyperparameters cause the model to be more conservative when making corrections.

本研究は，GECToR の流暢性が低い原因は，最適化がタグ情報のみから行われており，得られた訂正文の情報を考慮していないことにあるという仮説に基づいている．そこで，言語モデルの生成確率に基づいて流暢性の観点から訂正文の報酬を計算し，報酬を強化学習のアルゴリズムを用いてタグの最適化に反映する学習手法を提案する．

2　GECToR の流暢性

流暢性に関して，GECToR と系列変換モデルの比較実験を行った．GECToR には RoBERTa [4] ベースの訓練済みモデルを用いる．また，推論時のパラメータである無編集タグへのバイアスと文レベルの誤り検出確率の閾値は共に 0 とした．これらの値を小さくすることで訂正が積極的になり，より大きな書き換えを必要とする流暢な訂正が行えると考えられる．系列変換モデルには，Kiyono ら [5] と Kaneko ら [6] のモデルを用いる．Kiyono

らは，Transformer [7] 系列変換モデルを大量の擬似誤りで事前学習する点が特徴である．Kaneko らも Transformer 系列変換モデルであり，エンコーダにより得られた表現に加えて BERT [8] から得られた表現も加えて学習する．なお，いずれも著者らが公開する訓練済みモデルを用いて[1)]，ビーム幅を 5 として生成した．これらの 3 種類のモデルは，訂正性能そのものが乖離しないように，最小限の訂正性能が近いものを恣意的に選択した．具体的には，CoNLL-2014 ベンチマーク [9] における M^2 スコアラ [10] の $F_{0.5}$ は GECToR が 62.43，Kiyono らが 62.03，Kaneko らが 62.77 である．

流暢性の評価に用いるデータセットには，JFLEG [3] の開発データを用いる．JFLEG は流暢な訂正性能を評価するためのデータセットである．複数の観点から流暢性を評価するため，評価尺度には GLEU [11, 12]，SOME [13]，および GPT-2 [14] の perplexity の 3 種類を用いる．GLEU は n-gram の一致率に基づく JFLEG の標準的な参照あり評価尺度である．SOME は文法誤り訂正の参照なし評価尺度の一つであり，訂正文の人手評価に直接最適化するように評価器を学習する．SOME は複数の観点から評価するが，本研究では流暢性の評価結果のみを報告する．最後に，Asano ら [15] が訂正文の流暢性を RNN 言語モデルを用いて計算したことを動機とし，より高性能な言語モデルとして GPT-2 を用いて評価する．評価データの訂正文それぞれについてトークン数で正規化した perplexity を独立に算出し，その平均値を報告する．

実験結果を表 2 に示す．表 2 の編集率は，入力文と出力文の単語レベルの編集距離を，入力文の単語数で割った値である．表 2 の結果は，最小限の訂正性能は同等であるにもかかわらず，GECToR は系列変換モデルよりも訂正文の流暢性が低いことを示唆する．また，系列変換モデルと同程度の編集率であることから，十分に書き換えを行なっているのにもかかわらず訂正文の流暢性が低いことが分かる．

3 流暢性を報酬とした強化学習

本研究では，流暢性が低い原因は，GECToR の最適化がタグの情報のみから行われており，推定したタグを用いて編集することで得られる訂正文の情報を考慮できていないことにあると考える．したがって，訂正文を流暢性の観点から考慮した最適化を行うための手法を提案する．

表 2 GECToR と系列変換モデルの比較結果

	JFLEG-dev			
	GLEU ↑	SOME ↑	GPT-2 ↓	編集率
Kiyono ら	56.2	0.794	278.9	17.6
Kaneko ら	56.1	0.794	213.9	17.6
GECToR	54.3	0.787	308.7	18.1

3.1 定式化

GECToR はトークン単位のタグ付けと誤り検出をマルチタスク問題として最適化する．入力文を $\boldsymbol{x} = (x_1, x_2, \dots)$，$\boldsymbol{x}$ に対する正解のタグ列を $\boldsymbol{y}^t = (y_1^t, y_2^t, \dots)$ および正解の誤り検出ラベルを $\boldsymbol{y}^d = (y_1^d, y_2^d, \dots)$ とおく．ここで，$|\boldsymbol{x}| = |\boldsymbol{y}^t| = |\boldsymbol{y}^d|$ であり，事前に定義したタグの集合を $\mathcal{Y}$ として $y^t \in \mathcal{Y}$ である．また誤り検出ラベルは $y^d \in \{0, 1\}$ であり，1 のとき誤りであることを示す．GECToR は $\boldsymbol{x}$ を入力したとき，何らかのエンコーダ（BERT [8] など）により獲得した $\boldsymbol{x}$ の表現ベクトルを，2 種類の線形層にそれぞれ独立に入力し，タグの推定確率および誤り検出確率を計算する．この処理をタグの推定関数 $P^t(\cdot)$ および誤り検出確率の推定関数 $P^d(\cdot)$ と定義するとき，従来の損失関数 $\mathcal{L}_{mle}$ は次式で表される．

$$\mathcal{L}_{mle} = -\sum_{i=1}^{|\boldsymbol{x}|} (\log P^t(y_i^t | x_i, \boldsymbol{x}) + \log P^d(y_i^d | x_i, \boldsymbol{x})) \quad (1)$$

一方で，本研究では訂正文から計算される報酬を考慮するため，貪欲に生成した系列に対する損失を考える．まず $x \in \boldsymbol{x}$ について，貪欲に推定したタグ $\hat{y}^t \in \mathcal{Y}$ および誤り検出ラベル $\hat{y}^d \in \{0, 1\}$ を得る．

$$\hat{y}^t = \underset{y^t \in \mathcal{Y}}{\arg\max}\ P^t(y^t | x, \boldsymbol{x}),\ \hat{y}^d = \underset{y^d \in \{0,1\}}{\arg\max}\ P^d(y^d | x, \boldsymbol{x}) \quad (2)$$

得られたタグ系列 $\hat{\boldsymbol{y}}^t = (\hat{y}_1^t, \hat{y}_2^t, \dots)$ をもとに $\boldsymbol{x}$ を編集する[2)]ことで，訂正文 $\hat{\boldsymbol{t}} = (\hat{t}_1, \hat{t}_2, \dots)$ が得られる．ここで，訂正文の流暢性を計算する何らかの報酬関数を $r(\cdot) \in \mathbb{R}$ としたとき，REINFORCE アルゴリズム [16] によって報酬を考慮した損失 $\mathcal{L}_{rl}$ を次式で計

1) Kiyono らのモデルには pretlarge+SSE (finetuned)(https://github.com/butsugiri/gec-pseudodata) を，Kaneko らのモデルには https://drive.google.com/drive/folders/1h_r46EswcT1q75qwje6h6yJpOxzAG8gP?usp=sharing を用いた．

2) この編集処理では誤り検出確率の閾値を 0 としているため，タグのみの情報から訂正文が獲得される．

算する．

$$\mathcal{L}_{rl} = -(r(\hat{\boldsymbol{t}}) - \hat{r}) \sum_{i=1}^{|\boldsymbol{x}|} (\log P^t(\hat{y}_i^t | x_i, \boldsymbol{x}) + \log P^d(\hat{y}_i^d | x_i, \boldsymbol{x})) \tag{3}$$

$\hat{r}$ は報酬の分散を抑えるためのベースライン報酬である．式 3 で重要なのは，損失はタグ（$\hat{y}^t$）および誤り検出ラベル（$\hat{y}^d$）の情報から計算するが，報酬は訂正文（$\hat{\boldsymbol{t}}$）の情報から計算することである．これにより，訂正文の流暢性を考慮しながら最適化することができる．

最終的な損失 $\mathcal{L}$ は式 4 に示すように，パラメータ λ $(0 \leq \lambda \leq 1)$ を用いて重み付き和を取る．この理由は，訂正性能および訂正文の流暢性の両方の側面からモデルを最適化するためである．$\mathcal{L}_{mle}$ 項が訂正性能の向上に，$\mathcal{L}_{rl}$ 項が訂正文の流暢性の向上に貢献することを期待する．

$$\mathcal{L} = \lambda \mathcal{L}_{mle} + (1 - \lambda) \mathcal{L}_{rl} \tag{4}$$

3.2 報酬

式 3 の損失でモデルを学習するためには，報酬関数 $r(\cdot)$ が必要である．本研究ではこの報酬関数として，トークン数で正規化した GPT-2 の perplexity を用いる．これにより，言語モデルが捉える流暢性を反映した訂正文が生成できると考えられる．ただし，より流暢であるほど報酬が高くなるように設計するため，式 5 のように負の perplextiy を Min-Max 法によって正規化した値を報酬とする．ここで，$ppl(\cdot)$ は GPT-2 のパープレキシティであり，min および max は正規化における最小値と最大値に対応するハイパーパラメタである．ただし，負の perplexity が最小値を下回る場合は最小値に丸めることとする．

$$r(\boldsymbol{t}) = \frac{\max(min, -ppl(\boldsymbol{t})) - min}{max - min} \tag{5}$$

4 実験

4.1 実験設定

データセット 実験に用いるデータを表 3 に示す．訓練データは擬似誤りデータである PIE-synthetic [17][3] および実誤りデータである FCE-train [18], Lang-8 [19], NUCLE [20], W&I+LOCNESS-train [21] を用いる．実験では Omelianchuk ら [1] と同様に 3 段階での訓練を行うため，データを用いる段階も表に併記する．評価データには，JFLEG-test [3] および CoNLL-2014 [9] を用いる．

3) 著者らが公開するデータセットのうち a1 のセットを用いた．

表 3 実験に使用するデータセットの概要

データセット	文ペア数	分割	段階
PIE-synthetic	8,865,347	訓練	1
Lang-8	1,037,651	訓練	2
NUCLE	57,151	訓練	2
FCE-train	28,350	訓練	2
W&I+LOCNESS-train	34,308	訓練	2,3
JFLEG-dev	754	開発	-
JFLEG-test	747	評価	-
CoNLL-2014	1,312	評価	-

評価方法 訂正文の流暢性や流暢な訂正性能は JFLEG-test に対する出力を 2 節で述べた評価尺度で評価する．また，最小限の訂正性能を評価するために CoNLL-2014 に対する出力を M^2 スコアラ [10] により評価する．なお，推論時の 2 種類のパラメータ（2 節を参照）は JFLEG-dev の性能を元に探索する．

学習方法 Omelianchuk ら [1] と同様に 3 段階での訓練を行う．1 段階目は擬似誤りによる訓練，2 段階目は無編集の文ペアを除いた実誤りによる訓練，3 段階目は無編集の文ペアも含めた W&I+LOCNESS-train による訓練である．1 段階目では $\mathcal{L}_{mle}$ の損失のみで最適化し，2・3 段階目では $\mathcal{L}$ の損失で最適化する．入力文の表現ベクトルを得るためのエンコーダは RoBERTa [4] とする．式 4 の λ は $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ を試行する．また，式 3 のベースライン報酬 $\hat{r}$ にはミニバッチ内の報酬の平均を用いた．式 5 の min と max はそれぞれ-300 と-1 とした．その他の詳細なパラメータについては，付録 A を参照されたい．

4.2 実験結果

実験結果を表 4 に示す．表 4 では，系列変換モデルに基づく既存手法と，$\mathcal{L}_{mle}$ 項のみで学習することに等しい $\lambda = 1$ の場合，および開発データで最も高い性能を達成した $\lambda = 0.5$ の場合の性能を記載する．まず，$\lambda = 0.5$ と $\lambda = 1$ の場合を比較すると，$\lambda = 0.5$ の場合が JFLEG-test の評価値全てにおいて上回っている．このことは，強化学習に基づく損失 $\mathcal{L}_{rl}$ が流暢性の向上に寄与していることを示す．また，編集率は 0.2 ポイント向上しており，より大きく書き換えることで流暢性が向上したと考えられる．提案手法（$\lambda = 0.5$）と系列変換モデル（Kiyono らおよび Kaneko ら）との比較では，GLEU の値は提案手法が

表 4　提案手法と既存手法との性能比較．いずれもシングルモデルの評価結果である．上のブロックは系列変換モデル，下のブロックは系列タグ付けモデルに基づく訂正器の評価結果を示す．

	JFLEG-test				CoNLL-2014 ↑		
モデル	GLEU ↑	SOME ↑	GPT-2 ↓	編集率	Prec.	Rec.	$F_{0.5}$
Kiyono ら [5]	61.0	0.797	193.1	12.8	68.5	44.8	62.0
Kaneko ら [6]	61.0	0.797	175.5	13.0	68.9	46.1	62.7
従来手法（GECToR, $\lambda = 1$）	58.6	0.799	207.3	14.2	65.7	48.7	61.4
提案手法（$\lambda = 0.5$）	58.9	0.805	185.5	14.4	65.0	48.2	60.8

劣っているものの，SOME や GPT-2 のパープレキシティの値は競合する結果となった．このことは，提案手法は訂正文の流暢性を向上させるが，そのことは流暢な訂正性能の向上には必ずしもつながらないことを示唆する．

CoNLL-2014 において $\lambda = 0.5$ と $\lambda = 1$ の場合の性能を比較すると，提案手法は $F_{0.5}$ が 0.6 ポイント低下した．しかしながら，大きな悪化ではないと考えられるため，提案手法は訂正性能を維持しつつ訂正文の流暢性を高められると考えられる．

5　分析

λ の影響　λ を $\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9, 1\}$ の中で変化させたときの JFLEG-test における実験結果を表 5 に示す．$\lambda = 0.5$ として，$\mathcal{L}_{mle}$ と $\mathcal{L}_{rl}$ を同じ重みで扱うことで最良の結果を得た．GPT-2 の値からは，λ の値が小さいほど（$\mathcal{L}_{rl}$ 項の影響が大きくなるほど）訂正文が流暢になる傾向にあることが分かる．一方で，SOME の評価結果は逆の傾向を示した．このことは，訂正文の流暢性について一貫した評価は難しいことを示唆する．

表 5　λ と流暢性に関する性能値の関係

	JFLEG-test		
λ	GLEU ↑	SOME ↑	GPT-2 ↓
0.1	58.5	0.619	74.1
0.3	57.6	0.788	392.3
0.5	58.9	0.805	185.5
0.7	58.8	0.801	186.6
0.9	58.6	0.799	207.5
1.0	58.6	0.799	207.3

流暢性が向上する要因　どのような訂正が行われたことで流暢性が向上したか分析するため，ERRANT [22, 23] を用いて JFLEG-test の性能を誤りタイプ別に評価した．$\lambda = 1$ の場合を基準として $\lambda = 0.5$ の場合の訂正傾向の違いを分析したところ，名詞や動詞といった内容語の訂正が増加している一方，冠詞・綴り・前置詞といった機能語もしくは表層の訂正はほとんど増加していなかった．内容語に関する訂正は機能語や表記の訂正よりも訂正文の perplexity に大きく影響すると考えられるため，perplexity を報酬とする提案手法は内容語を積極的に書き換えることで流暢性を向上させたと考えられる．ただし，訂正の総数は増えているが偽陽性が増える傾向にあったため，表 4 の GLEU や M^2 による参照あり評価では性能が伸び悩んだと考えられる．誤りタイプ別の評価の詳細は付録 B に示す．

6　関連研究

強化学習に基づく文法誤り訂正手法には，Sakaguchi ら [24] と Raheja ら [25] の研究がある．Sakaguchi らは，GRU 系列変換モデルの学習において，評価尺度である GLEU を報酬関数とした学習手法を提案した．Raheja らは，報酬関数として訂正文の作成者が人間か訂正器かを判別する分類器を用いて，Transformer 系列変換モデルと敵対的に学習させる手法を提案した．

これらの関連研究と比較して，本研究は系列タグ付けに基づく訂正器に強化学習を導入した点が異なる．また，提案手法では報酬関数を GPT-2 言語モデルとしたが，関連研究に倣い GLEU や SOME といった他の尺度を用いたり，新たな報酬関数を設計して敵対的に学習することには考慮の余地がある．

7　おわりに

本研究では，系列タグ付けに基づく文法誤り訂正器である GECToR は，訂正文の流暢性が低い問題を指摘した．問題の原因は最適化に訂正文の情報が使われていないことであると考えられる．したがって，言語モデルに基づいて計算した報酬を用いて強化学習を行うことで，訂正文の流暢性を考慮する学習手法を提案した．実験の結果，訂正性能を維持しながら訂正文の流暢性が向上した．提案手法と異なる報酬関数による学習や，GECToR 以外の系列タグ付けに基づく訂正器への応用は今後の課題とする．

参考文献

[1] Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. GECToR – grammatical error correction: Tag, not rewrite. In **BEA**, pp. 163–170, Seattle, WA, USA → Online, July 2020. Association for Computational Linguistics.

[2] Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. **TACL**, Vol. 4, pp. 169–182, 2016.

[3] Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. JFLEG: A fluency corpus and benchmark for grammatical error correction. In **EACL**, pp. 229–234, Valencia, Spain, April 2017. Association for Computational Linguistics.

[4] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. **arXiv preprint arXiv:1907.11692**, 2019.

[5] Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. An empirical study of incorporating pseudo data into grammatical error correction. In **EMNLP-IJCNLP**, pp. 1236–1242, Hong Kong, China, November 2019. Association for Computational Linguistics.

[6] Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In **ACL**, pp. 4248–4254, Online, July 2020. Association for Computational Linguistics.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. **Advances in neural information processing systems**, Vol. 30, , 2017.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In **NAACL**, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] 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 **CoNLL**, pp. 1–14, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[10] Daniel Dahlmeier and Hwee Tou Ng. Better evaluation for grammatical error correction. In **NAACL**, pp. 568–572, Montréal, Canada, June 2012. Association for Computational Linguistics.

[11] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. Ground truth for grammatical error correction metrics. In **ACL**, pp. 588–593, Beijing, China, July 2015. Association for Computational Linguistics.

[12] Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. Gleu without tuning, 2016.

[13] Ryoma Yoshimura, Masahiro Kaneko, Tomoyuki Kajiwara, and Mamoru Komachi. SOME: Reference-less sub-metrics optimized for manual evaluations of grammatical error correction. In **COLING**, pp. 6516–6522, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. **OpenAI blog**, Vol. 1, No. 8, p. 9, 2019.

[15] Hiroki Asano, Tomoya Mizumoto, and Kentaro Inui. Reference-based metrics can be replaced with reference-less metrics in evaluating grammatical error correction systems. In **IJCNLP**, pp. 343–348, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[16] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. **Machine learning**, Vol. 8, No. 3, pp. 229–256, 1992.

[17] Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. Parallel iterative edit models for local sequence transduction. In **EMNLP-IJCNLP**, pp. 4260–4270, Hong Kong, China, November 2019. Association for Computational Linguistics.

[18] Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A new dataset and method for automatically grading ESOL texts. In **ACL**, pp. 180–189, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[19] Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In **ICJNLP**, pp. 147–155, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[20] Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. Building a large annotated corpus of learner English: The NUS corpus of learner English. In **BEA**, pp. 22–31, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[21] Helen Yannakoudakis, Øistein E Andersen, Ardeshir Geranpayeh, Ted Briscoe, and Diane Nicholls. Developing an automated writing placement system for esl learners. **Applied Measurement in Education**, Vol. 31, No. 3, pp. 251–267, 2018.

[22] Mariano Felice, Christopher Bryant, and Ted Briscoe. Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments. In **COLING**, pp. 825–835, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[23] Christopher Bryant, Mariano Felice, and Ted Briscoe. Automatic annotation and evaluation of error types for grammatical error correction. In **ACL**, pp. 793–805, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[24] Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. Grammatical error correction with neural reinforcement learning. In **ICJNLP**, pp. 366–372, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[25] Vipul Raheja and Dimitris Alikaniotis. Adversarial Grammatical Error Correction. In **Findings of EMNLP**, pp. 3075–3087, Online, November 2020. Association for Computational Linguistics.

A　実験における詳細なパラメータ

学習時　4 節で述べたように，モデルの学習は 3 段階で行う．訓練時のパラメータは表 6 のとおりである．特に表記のない限り，3 段階全てで同じパラメータを用いた．

表 6　実験に用いたパラメータの詳細

バッチサイズ	64
勾配累積するバッチ数	1 段階目　4 2・3 段階目　2
損失	CrossEntropyLoss
label smoothing	0.0
エポック数	5
学習率	1e-5
タグセットのサイズ	5000
分類層のみ訓練するエポック数	1 段階目　2 2・3 段階目　0

B　誤りタイプに関する詳細な性能

5 節では，提案手法により流暢性が向上する原因として，内容語の訂正をより積極的に行っていることを述べた．より詳細な性能を表 7 に示す．モデルが行った訂正に注目するため，表 7 には一部の誤りタイプについて，True Positive に該当する訂正数（TP カラム），False Positive に該当する訂正数（FP カラム）および適合率を示す．さらに，提案手法が従来手法に比べて何倍の訂正数かを示すため，訂正数の合計（True Positive と False Positive の和）について，$\lambda = 0.5$ の場合の値を $\lambda = 1$ の場合の値で割った値を示す（Δ カラム）．

5 節で述べたように，機能語や表層の訂正と内容語の訂正には訂正数の傾向に違いがある．名詞については，語彙選択に関する訂正である「名詞」は提案手法により 1.47 倍訂正数が増加した．一方，「名詞の数」は訂正数に変化がない．名詞の数に関する訂正は文意を大きく変えないため，提案手法を適用しても変化がなかったと考えられる．同様のことは，「動詞」や「動詞の時制」は若干訂正数が増加するが，「動詞の形態」については変わらないことからも言える．「形容詞」や「副詞」も訂正数が増加していることが分かる．ただし，「形容詞」や「副詞」は頻度が少ないため Δ に関して他の誤りタイプと比較するのは難しい．

表 7　誤りタイプ別の訂正数と性能値

	$\lambda = 1$			$\lambda = 0.5$			Δ
誤りタイプ	TP	FP	Prec.	TP	FP	Prec.	
前置詞	185	83	69.0	187	88	68.0	1.02
正書法	114	8	93.4	113	8	93.3	0.99
前置詞	85	67	55.9	86	68	55.8	1.01
綴り	183	33	84.7	190	33	85.2	1.03
名詞	13	29	30.9	13	49	20.9	1.47
名詞の数	102	18	85.0	102	18	85.0	1.00
動詞	37	47	44.0	35	54	39.3	1.05
動詞の形態	41	25	62.1	38	28	57.5	1.00
動詞の時制	50	44	53.1	50	50	50.0	1.06
形容詞	9	14	39.1	10	17	37.0	1.17
副詞	15	16	48.3	16	21	43.2	1.19