英語学習者のための解説文生成手法の調査

塙 一晃 †,‡,1　永田 亮 §,†,¶,2　乾 健太郎 ‡,†,3
† 東北大学　‡ 理化学研究所　§ 甲南大学　¶JST さきがけ
[1]kazuaki.hanawa@riken.jp　[2]nagata-nlp2021@ml.hyogo-u.ac.jp　[3]inui@tohoku.ac.jp

1 はじめに

解説文生成とは，与えられた文章に対してライティングに関するヒントや説明を生成するタスクのことである．例えば，*We reached to the station. の下線部に対して，

(1)reach は他動詞ですので、目的語の前には前置詞は必要ありません。

のような解説を生成するタスクである．文献 [1] で，英語学習者が書いた文章に対して，前置詞に関する解説とより一般的な解説の二種類を付与したコーパスが構築，公開されている．このデータセットにより，様々な機械学習による解説文生成が可能となっている．例えば，本タスクは，解説対象文章と解説箇所を入力とする言語生成問題であるので，各種のニューラル言語生成モデルが適用可能である．

このような状況にもかからず，解説文生成手法について分かっていることはとても少ない．存在する知見は，前置詞解説文では検索に基づいた手法が有効である [2] ぐらいである．ニューラル言語生成モデルには，幅広い選択肢があるが，研究例は非常に少ない．文献 [3] では，検索結果に基づいて，ニューラル言語生成モデルで生成を行うが，他のモデルとの比較は行われていない．また，これらの研究は前置詞解説文をのみを対象にしており，一般解説文に対する知見は皆無である．後者のほうが解説の種類が多くより難しい問題であるため，手法の振る舞いが大きく変わる可能性がある．

そこで，本稿では，解説文生成の研究が今後進むべき方向性を模索することを目的として，各種手法の性能比較と生成結果の詳細な分析を行う．具体的には，検索に基づく手法，単純な Encoder-decoder 言語生成モデル，両者を組み合わせた検索編集手法（以下，それぞれ，Retrieval，Simple Generation，Retrieve & Edit と表記）について比較を行う．前置詞解説文に加えて，今まで研究例がない一般解説文も対象とする．

各手法の性能は次のように予想される．Retrieval は，検索した解説文をそのまま出力するため柔軟性に欠け，最も性能が低い．Simple generation は，言語生成モデルであるので柔軟性が高く，より性能が高い Retrieve & Edit は，Retrieval と Simple Generation を組み合わせたような手法であるので，最も性能が高い．以上をまとめると，性能順は， Retrieval < Simple Generation < Retrieve & Edit と予想される．

興味深いことに，実際に性能を評価したところ上の予想とは全く異なる結果が得られた．更に，前置詞解説文と一般解説文では，異なる性能順位となることも明らかになった．具体的には，前置詞解説文では， Retrieve & Edit < Retrieval < Simple Generation，一方，一般解説文では Simple Generation < Retrieve & Edit < Retrieval となった．

生成結果を詳細に分析したところ，いくつかの要因が明らかになった．結果は次のように要約される．第一に，Retrieve & Edit では，“不要な生成”（over-editing と呼ぶ）が頻繁に発生する．Over-editing は，言語生成モデルの柔軟性を上回り，性能低下を招く．その結果，Retrieval よりも低い性能となる．第二に，Simple Generation では，複数の解説内容を混ぜて生成するような現象（mutant と呼ぶ）が起こる．複数の解説内容を混ぜたものは，当然，適切な解説文とはならない．相対的に解説の種類が少ない前置詞解説文では mutant は発生しにくく，Simple generation の性能が最も良くなる．また，Retrieve & Edit では，検索結果を利用するため mutant は起こりにくい．以上のことに基づき，性能向上のためのアイデアも議論する．

2 関連研究

解説文生成研究のためのデータセットが少しずつ増えてきている．文献 [2] で，前置詞の用法を対象にした解説文データセットが公開された．その後，文献 [1] で，その他の文法誤り，構成，語彙選択を含

表 1　各生成手法の全体の正解率.

	前置詞解説文	一般解説文
Simple Generation	0.434	0.194
Retrieval	0.345	0.286
Retrieve & Edit	0.315	0.224

む一般解説文のデータが公開された．本稿では，この前置詞解説文と一般解説文のデータを利用する．そのほか，文献 [4] では，linking word の用法に着目した解説文を含む学習者コーパスが公開された．

データの公開と共に，解説文生成手法の数も増えつつある．人手で作成した規則に基づいて解説文を生成する手法 [5, 6, 7]，テンプレートに基づいて解説文を作成する手法 [8]，検索に基づいた手法 [2] などが知られている．これらの手法は広い範囲の誤りに柔軟に対応することが困難である．

より一般的には，解説文生成は，解説対象文と解説箇所を入力とした言語生成問題と捉えることができる．したがって，各種のニューラル言語生成モデルが本タスクに効果的であると予想される．そのような研究に文献 [3] がある．しかしながら，解説文生成において，各種ニューラル言語生成モデルの性能比較を行った研究は我々が知る限り存在しない．

3　解説文生成手法

3.1　タスク定義と表記

文献 [2] に基づきタスクを定義する．入力は，解説対象文と解説文の対象となるトークン位置（以下，解説位置と表記）である．一方，出力は解説位置に対する解説文である．

形式的にタスクを定義するため，次の記号を導入する．解説対象文，その長さ（トークン数），i 番目のトークンをそれぞれ S，N，w_i と表記する．すなわち，$S = w_1, \cdots, w_i, \cdots, w_N$ である．また，解説位置を o と表記する．更に，S と o を組にしたものを $\boldsymbol{x}$ と表記する．すなわち，$\boldsymbol{x}$ は入力である．これに対応させ，出力となる解説文を $\boldsymbol{y}$ と表記する．一方，生成（予測）された解説文は $\hat{\boldsymbol{y}}$ のように^をつけて表す．トークン w_i に対応する隠れベクトルは $\boldsymbol{h}_i$ と表記する．また，$\boldsymbol{x} = (S, o)$ をエンコードしたベクトルを $\boldsymbol{c}$ と表記する．

Retrieval と Retrieve & Edit では次の記号も使用する．入力 $\boldsymbol{x}$ に対する検索事例を $\boldsymbol{x}'$ と表記する．また，$\boldsymbol{x}'$ に紐づいた解説文を $\boldsymbol{y}'$ と表記する．

3.2　Simple Generation

例 (1) のように S 中の単語は $\boldsymbol{y}$ 中にもしばしば出現する．したがって生成手法は S 中の単語をコピーできる機構をもっていることが望ましい．そのため，我々は pointer-generator network [9] を Encoder-Decoder として採用する．このネットワークは入力文中の単語のコピーと語彙からの単語の生成を p_{gen} の値に応じて制御する．

文献 [9] と違う点として，本タスクでは解説位置 o の情報を $\boldsymbol{c}$ に含める必要がある．具体的には，まず Bi-LSTM によって S 中の各トークン w_i を隠れベクトル $\boldsymbol{h}_i$ に変換する．次に o 番目の隠れベクトル $\boldsymbol{h}_o$ を文脈ベクトル $\boldsymbol{c}$ として使用する．$\boldsymbol{c}$ はデコーダへの初期入力として用いられる．

3.3　Retrieval

この手法はニューラルネットワークを用いた検索に基づく．具体的には，まず入力 $\boldsymbol{x}$ と最も類似した $\boldsymbol{x}'$ をそれに対応する $\boldsymbol{y}'$ とともに訓練データから検索する．そして，検索した $\boldsymbol{y}'$ を $\hat{\boldsymbol{y}}$ として出力する．$\boldsymbol{x}$ と $\boldsymbol{x}'$ の類似度はそれぞれに対応する文脈ベクトル $\boldsymbol{c}$，$\boldsymbol{c}'$ の余弦類似度によって計算される．

$\boldsymbol{x}$ を $\boldsymbol{c}$ にエンコードするためのネットワークとして Simple Generation と同様なアーキテクチャを用いる．ただし，$\boldsymbol{x}$ の情報を全て $\boldsymbol{c}$ に含めるために，アテンション機構は用いない．

3.4　Retrieve & Edit

本手法は文献 [10] で提案された手法に基づく．1 節で述べたように Retrieve & Edit は Retrieval と Simple Generation を組み合わせたような手法である．本手法は Retriever と Editor と呼ばれる二つの独立したネットワークからなる．Retrieval と全く同じネットワークを Retriever として使用する．ただし，訓練時には，$\boldsymbol{x}$ 自身を除いて最も類似した事例を検索結果とする．また Simple Generation と概ね同じネットワークを Editor として使用する．Simple Generation との違いは検索した事例を考慮する必要がある点である．すなわち，Editor は Simple Generation の入力 $\boldsymbol{x}$ に加えて，Retriever により検索された $\boldsymbol{x}'$，$\boldsymbol{y}'$ を入力として生成を行う．これは，(1)$\boldsymbol{x}$，$\boldsymbol{x}'$，$\boldsymbol{y}'$ をそれぞれ別のネットワークで文脈ベクトルに変換し，(2) これらの三つのベクトルを結合し，(3) それをデコーダへの入力として用いる，という工程によって行

	対象文	検索された解説文	Retrieve & Edit の出力
(1)	above reasons , it is necessary ...	＜名詞＞1語では＜副詞句＞として機能しませんので、辞書で＜名詞＞«conclusion»を引いて前に適切な＜前置詞＞をつけましょう。	＜名詞＞«reasons»の＜目的 語＞は＜副詞＞として機能しません。適切な＜前置詞＞を補いましょう。
(2)	... society include value of money .	＜動詞＞«include»は＜ing形＞になってはじめて＜前置詞＞として機能します。	＜動詞＞«include»はそのままの形では＜前置詞＞として機能しません。＜動名詞＞になってはじめて＜動詞派生前置詞＞となります。

図 1　Retrieve & Edit における生成とコピーの確率の可視化.

われる．より詳細には，$\boldsymbol{x}$ と $\boldsymbol{x}'$ は Simple Generation と同じ方法でエンコードし，$\boldsymbol{y}'$ のエンコードには Bi-LSTM の最終状態を用いる．

また Editor は一つのアテンション機構を持ち，$\boldsymbol{x}$, $\boldsymbol{x}'$, $\boldsymbol{y}'$ の全体に対して同時にアテンションを計算する．これにより，Editor は $\boldsymbol{x}$, $\boldsymbol{x}'$, $\boldsymbol{y}'$ のいずれからもコピーを行うことができる．

4　実験

本実験では，前置詞解説文および一般解説文を収録したデータセット [1] を用いる．このデータセットは学習者が書いたエッセイからなる．各エッセイは，トピック “アルバイト”（PTJ）と “喫煙”（SMK) のいずれかである．本実験では，トピック別に各手法の性能評価を行う．データセットの統計値を付録 A に示す．また，訓練時の設定は付録 B に示す．

評価尺度は生成正解率（解説箇所数に対する適切な生成結果の割合）とした．適切，不適切の判断は，英語指導（2 年間）と英語統語アノテーション（10 年以上），両方に経験がある作業者に依頼した．

表 1 に，PTJ と SMK をまとめた生成正解率を示す（個別の結果は付録 C に示す）．表 1 の生成性能は，前置詞解説文でも一般解説文でも，当初の予想とことなる．具体的には， Retrieve & Edit は，その名の通り，手法の中に，Retrieval を含むにもかかわらず，Retrieval の性能を超えられていない．また，前置詞解説文では最も性能がよい Simple Generation が，一般解説文では最も性能が悪い．このような結果が得られた理由を 5.2 で考察する．

5　考察

5.1　なぜ Retrieve & Edit の性能は低いのか

Retrieve & Edit の出力結果を分析すると，不必要な編集をしてしまう現象（以下，over-editing と呼ぶ）が確認できた．ここで，over-editing とは，解説文の適切さには影響しない単語の編集（生成）のことをさす．訓練時に，検索された解説文と目的の解説文が同一内容であっても，表層が異なると，そのような不必要な編集が学習される．言い換えれば，over-editing は，解説文における表層バリエーションの多さに起因するといえる．次の例を考える：

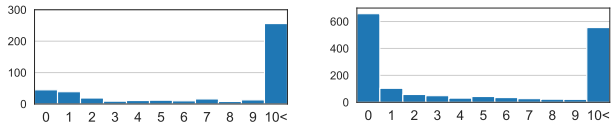
(2)動詞 agree は that 節を目的語にとりうる動詞なので、目的語の前に前置詞 with は不要です。

(3)動詞 agree が同意する内容が節で書かれている場合は前置詞 with は不要です。

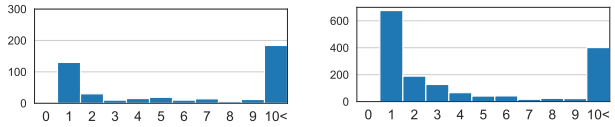
この二つの解説文は，同じ文法規則についての解説であるが表層は大きく異なる．解説文は自然言語で記述されるため，このような表層的なバリエーションは避けられない．例 (2) に対して例 (3) が検索された場合には，そのまま出力すれば十分であるにも関わらず，例 (2) のように書き換えるように訓練が行われる．以上の例では二つの解説文のみを考えているが，実際の訓練データでは，表層は異なるが同種の誤りに対する解説文が多数存在しうる [2]．そのような場合，over-editing はより顕著となる．

実際に over-editing を観察するために，Retrieve & Edit における生成とコピーを制御する確率値 p_{gen} を用いて生成結果の可視化を行った（図 1）．図 1 では p_{gen} の値が大きいほど，すなわち生成の確率が高い単語ほど濃い赤となっている．全体的に p_{gen} の値が大きく，検索してきた事例からコピーを行わず，語彙からの生成を行っている様子が観察できる．例えば，図 1(1) は，検索された解説文の conclusion を reasons 書き換えるだけで適切な解説文となる例である．それにもかかわらず，Retrieve & Edit では，単語のコピーはほとんど行わず，多くの単語を生成している．その結果，誤生成となっている．また，図 1(2) では，生成された解説文は正しいが，多くの単語を生成してしまっている．理想的には，検索結果の少数の単語だけを置き換えることが好ましいが，over-editing により妨げられている．

どのような方法で，over-editing を抑制できるだろうか．Over-editing を抑制し，少数の単語だけを生成するモデルにするためには，少数の単語だけが置き換わる訓練事例を Editor に与えるべきである．その実現方法の一つに，解説文間の表層類似度を用いる方法がある．訓練時に Retrieve & Edit で用いている隠れ状態の類似度（余弦類似度）に加えて，解説文



(a) 前置詞/Simple Generation　(b) 一般/Simple Generation



(c) 前置詞/Retrieval　(d) 一般/Retrieval

図 2　各手法が出力した解説文と同一の解説文の訓練データ中の数．横軸: 同一の解説文の訓練データの数．縦軸: テストデータにおける頻度．

間の表層類似度を考慮して検索を行えば，表層の差異が少ない事例が Editor の訓練に用いられる．そうすることで，検索結果を生かしつつ，少数の書き換え規則のみが学習されやすくなると期待できる．

5.2　Simple Generation の順位の入れ替わり

Simple Generation の生成結果を分析すると，前置詞解説文と一般解説文では異なる振る舞いが見られた．前者では訓練データと表層的に非常に類似した解説文を生成することが多い．一方，後者では，複数の解説内容を融合したような生成結果が頻繁に観測された（以下，この現象を mutant と呼ぶ）．例えば，“* ··· this activity is will not disturb their ···” に対して，次のような mutant が生成された：

(4)主語に合わせて動詞を適切な形で用いましょう。be 動詞は必要か確認しましょう。

一文目は主語と動詞の一致についての解説，二文目は be 動詞と一般動詞の併記に関する解説である．Be 動詞が解説位置である場合，この二種類の解説がなされることが多いことに起因すると分析できる．

この現象は，Encoder-Decoder による生成を次のようなプロセスだと捉えると説明できる：(1) 入力 $\boldsymbol{x}$ により，ベクトル空間のある点が指定される，(2) その点に近い訓練事例があれば，それに対応した $\boldsymbol{y}$ が生成される，(3) ただし，その点に近い（解説内容が異なる）訓練事例が複数あると，混合されて $\hat{\boldsymbol{y}}$ に反映される．解説内容の種類が相対的に少ない前置詞解説文では，(3) の状況が起こりにくい．相対的に種類数が多い一般解説文では例 (4) のような別種の事例が $\boldsymbol{x}$ 周辺に配置されるため mutant ができやすい．mutant は基本的に全て誤生成となる．一方，Retrieval では複数事例の何れかを出力するため，一定の確率で生成に成功する．この点で，Simple generation は，Retrieve より性能が低くなる．

では，なぜ前置詞解説文において Simple generation は最も高い性能を示すのだろうか．一つの理由としては，Simple generation の柔軟性を挙げることができる．Retrieval で生成に失敗した事例のうち，数単語を書き換えたら正しくなるものは，それなりの数存在する（解説位置の 6%が該当する）．これらの失敗は，完全に一致する事例が訓練データに存在しないことに起因する．一方，Simple generation ではそのような事例にも対応可能である．

別の理由として，Simple Generation は解説文の条件付き生成確率を学習するため，頻度の高い解説文を好むことを挙げることができる．正解率をあげるという観点では，解説の種類が認識できていない場合には，より頻度の高い解説文を出力することが良い戦略である．このことに起因して Simple Generation の性能が高くなったと予想できる．

このことを確認するために，生成した解説文と同一の解説文が訓練データ中にどれだけ存在するかを調べた．ただし，同一の解説文かどうかを人手で判定することは容易ではないので，正規化した編集距離が 0.1 未満の場合に同一の解説文だと疑似的にみなした．その結果を図 2 に示す．図 2a, 2c を見ると，Retrieval は訓練データ中で頻度 1 の解説文を出力することが多いが，Simple Generation では頻度 2 以上の解説文を出力することが多くなるのがわかる．このことから Simple Generation は頻度の低い解説文を生成しにくく，それが正解率の向上に寄与したと考えられる．一方で，図 2b，2d を見ると一般解説文では同様の傾向は見られない．逆に，一般誤解説文では，Simple Generation は，訓練データに一度も出現しない解説文を生成することが多いこともわかる．この中に，上で述べた mutant も含まれるであろう．

6　おわりに

本稿では英語学習のための解説文生成タスクにおける生成手法について比較を行った．直感に反し Retrieve & Edit の性能が低いことや，タスクによって生成手法の良し悪しに変化があることを示し，その原因について考察を行った．以上のことから，(1) 特定の項目に対する解説文生成では，Simple generation の拡張が有効である；(2) 一般解説文生成では，Retrieve & Edit で over-editing を低減させることが，性能向上につながると予想される．

参考文献

[1] Ryo Nagata, Kentaro Inui, and Shin'ichiro Ishikawa. Creating Corpora for Research in Feedback Comment Generation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 340–345, 2020.

[2] Ryo Nagata. Toward a Task of Feedback Comment Generation for Writing Learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3206–3215, 2019.

[3] 一晃塙, 亮永田, 健太郎乾. 高信頼度な文法誤り解説生成のための生成制御手法 . 2020 年度 人工知能学会全国大会（第 34 回）, 2020.

[4] Ildiko Pilan, John Lee, Chak Yan Yeung, and Jonathan Webster. A Dataset for Investigating the Impact of Feedback on Student Revision Outcome. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 332–339, 2020.

[5] Kathleen F Mccoy and Christopher A Pennington. English error correction: A syntactic user model based on principled mal-rule scoring. In *In Proceedings of the Fifth International Conference on User Modeling*, pp. 59–66, 1996.

[6] Jun'ichi Kakegawa, Hisayuki Kanda, Eitaro Fujioka, Makoto Itami, and Kohji Itoh. Diagnostic Processing of Japanese for Computer-Assisted Second Language Learning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pp. 537–546, 2000.

[7] Ryo Nagata, Mikko Vilenius, and Edward Whittaker. Correcting Preposition Errors in Learner English Using Error Case Frames and Feedback Messages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 754–764, 2014.

[8] Yi-Huei Lai and Jason Chang. TellMeWhy: Learning to Explain Corrective Feedback for Second Language Learners. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 235–240, 2019.

[9] Abigail See, Peter J Liu, and Christopher D Manning. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1073–1083, 2017.

[10] Tatsunori B Hashimoto, Kelvin Guu, Yonatan Oren, and Percy S Liang. A Retrieve-and-Edit Framework for Predicting Structured Outputs. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors, *Advances in Neural Information Processing Systems 31*, pp. 10052–10062, 2018.

[11] Shinichiro Ishikawa. The ICNALE and sophisticated contrastive interlanguage analysis of Asian learners of English. *Learner corpus studies in Asia and the world*, Vol. 1, pp. 91–118, 2013.

表 2　使用したデータセットの統計量.

前置詞誤り	PTJ		SMK	
	英文数	解説文数	英文数	解説文数
訓練	12163	2439	12312	2341
開発	1129	245	1160	230
テスト	1042	224	1023	214

一般誤り	PTJ		SMK	
	英文数	解説文数	英文数	解説文数
訓練	18251	11510	19957	11792
開発	1315	848	1413	837
テスト	1304	833	1328	772

表 3　PTJ と SMK における生成手法の正解率.

	前置詞誤り		一般誤り	
	PTJ	SMK	PTJ	SMK
Simple Generation	0.408	0.460	0.192	0.196
Retrieval	0.321	0.370	0.276	0.296
Retrieve & Edit	0.289	0.341	0.236	0.212

A　データセットの統計

実験に使用したデータセットの統計は表 2 の通りである.

B　使用したハイパーパラメータ

実験に使用したハイパーパラメータは以下の通りである．全ての解説文生成手法中の LSTM は一層で中間層の次元は 300 である．単語の分散表現の次元数は 200 で，ICNALE[11] で事前学習したものを用いる．訓練は学習率 0.001 の Adam を用いて 50 epoch 行う．別のランダムシードを用いて五回ずつ訓練を行い開発セット上で BLEU が最大になったものを採用する.

C　全ての評価結果

PTJ と SMK それぞれの評価結果は表 3 の通りである.