Langsmith: 人とシステムの協働による論文執筆

伊藤拓海 *,1,2　栗林樹生 *,1,2　日高雅俊 *,3　鈴木潤 1,4　乾健太郎 1,4
[1] 東北大学 [2]Langsmith 株式会社 [3]Edge Intelligence Systems [4] 理化学研究所
{t-ito, kuribayashi, jun.suzuki, inui}@ecei.tohoku.ac.jp
hidaka@edgeintelligence.jp

1 はじめに

自然言語処理の有望な応用先として，エッセイや物語，論文などの作成支援技術が盛んに研究されている [1, 2, 3]．近年特に，効率的な支援の実現のため，人とシステムがインタラクションを取りながら協働的に言語活動を行う枠組みに期待が集まっている [4, 5, 6, 7, 8]．本研究では，表現が十分に練られていない草稿を人がシステムと協働的に推敲するという場面に焦点を当てる．

そもそも推敲とは，（人の頭の中にある）書きたい内容と現在書いている文章を比較し，内容をわかりやすく，正確に伝えるために文章を練ることである．推敲は語彙や文法，修辞法などに対する深い知識が必要であり，人（特に未熟な書き手）にとって負荷の高い作業である．一方システムは大規模コーパスから適切な表現や文章のパターンを獲得することでき，表現に関する支援が期待できる．しかしながら，システムが草稿から人の書きたい内容を推測するのは困難であり，システムが自動で推敲を行うのは現実的でない．推敲支援には，人がシステムに書きたい内容を伝え，それに対してシステムが適切な表現をフィードバックするというインタラクションが重要であり，インタフェースの設計が鍵となる（図 1）．

既存研究 [4, 5] では，「ここに単語を挿入して欲しい」や「この表現を書き換えて欲しい」といった指示を考慮して書き換えを行う生成モデルの開発がされている．しかしこれらの研究は，人をシミュレートした環境で，指示に従った生成ができるかどうかというモデルの性能に関する評価しか行っておらず，人とシステムの協働的執筆が文章の質の向上に繋がるのか，どういったインタラクションの取り方が効率的なのかといった人を系に入れた議論が不足している．

* 三者の貢献は同等である．

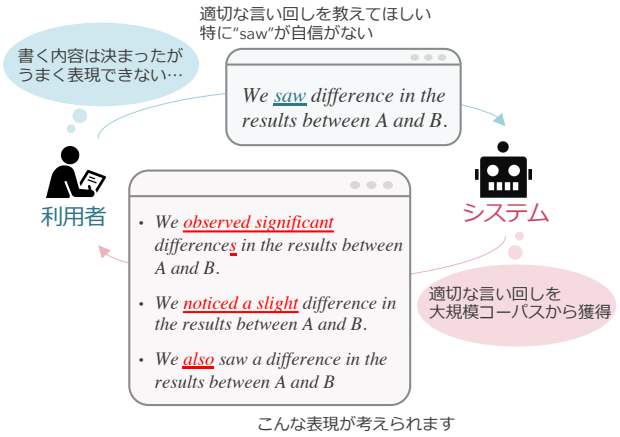


図 1 人とシステムによる協働的推敲の概要．

本研究では，自然言語処理分野の英語論文執筆支援を想定し，既存研究 [4, 5] の成果をもとに，人とシステムが協働で文章を書く場として Langsmith エディタ[1)] を作成した．英語を母語としない学生に利用してもらい，人とシステムの協働的な推敲に関する事例研究を行った．実験結果から，システムとの協働的推敲によって文章の質の向上が確認された．また，ユーザ調査により実用的なインタフェースに関する知見が得られた．

2 関連研究

文法誤りの訂正や言い換え表現の提案を行う執筆支援システムとして，Grammarly[2)] や Beewriter[3)] が挙げられる．これらのツールはシステムが一方的にエラーを検知するいわば校正型であるのに対し，Langsmith エディタでは利用者がシステムの編集を制御できるなどシステムとの協働を重視しており，協力して文章を書いていく執筆初期段階の支援を目指している点で Langsmith エディタはこれらのツールと異なる．

1) https://editor.langsmith.co.jp/
2) https://www.grammarly.com/
3) http://beewriter.com/

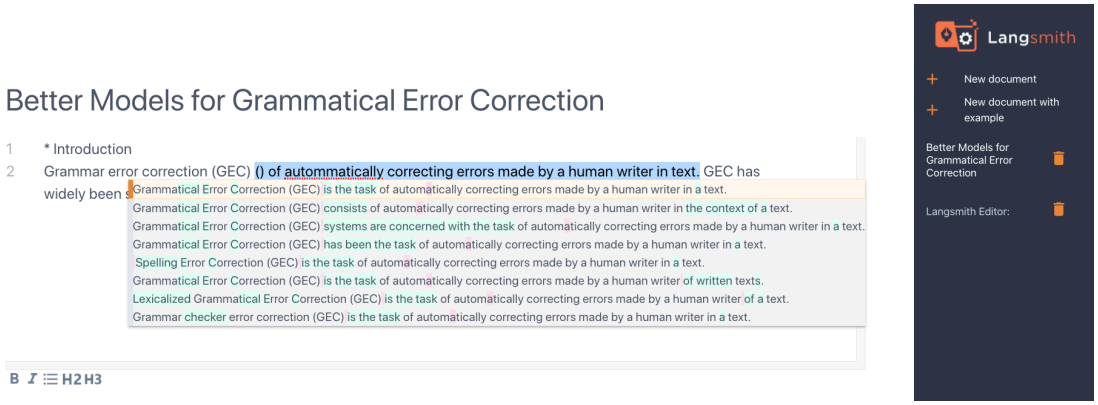


図 2　Langsmith エディタのスクリーンショット．

3　Langsmit エディタ

3.1　協働による書き換え

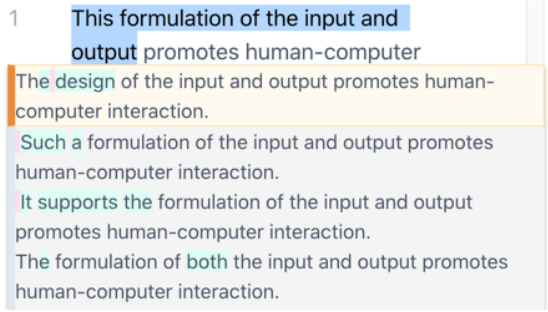
人が与えた文に対してシステムが論文に適した流暢な文を提案する機能を実現する（図 2）．

人による制御可能性:　人とシステムの協働の効率化のため，利用者が書き換えを制御できるよう 2 つの手段を提供する．1 つ目は，重点的に書き換えて欲しい箇所の選択機能（**編集箇所指定**）である．利用者は自分が書いた文について「この表現が不自然」といった箇所を自覚している可能性があり，そこを修正すべきか，またどう修正すべきかについてフィードバックが得られれば満足する場合が想定される．図 3 に示すように，本システムでは利用者がカーソルで選択した箇所に対して集中的に編集を加える機能を実現した．

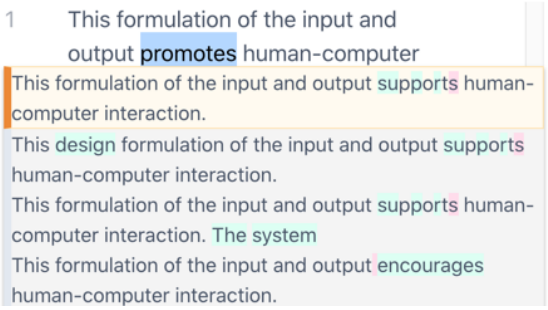
2 つ目は，表現を挿入してほしい位置の指定機能（**挿入箇所指定**）である．文脈付き索引のように，文脈に応じた自然な表現を探す時に役立つと考えられる．利用者は文中の特定の位置に特殊記号 “()” を挿入でき，システムは特殊記号を適切な表現に置き換えて文を書き換える（図 2）．

システムによる多様な提案:　流暢さやスタイルに関わる修正では適切な推敲結果が一意に絞り込めない場合があるため，複数の書き換え候補を提供する設計とした（図 2）．推論時の生成確率の上位数件を提示した場合非常に似たような候補が複数提示される傾向が観察されたため，推論結果に多様さを促している [9]．

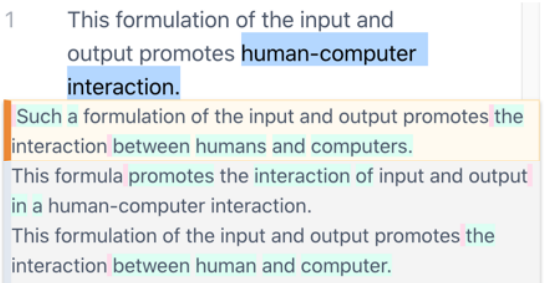
モデルの実装:　Ito ら [5] と同様に疑似学習データを用いて Encoder-Decoder 型系列変換モデル [10] を訓練した．また編集箇所指定を実現するため，入力文中のスパンのうち参照文と比較して多くの編集が含まれるものに特殊記号をつけて，編集箇所のヒントを与える形でモデルを学習した（付録 A）．システムは印がある箇所を重点的に編集するように学習され（5.1 節），エディタ上では利用者が印の位置を指



(a) *This formulation ··· output.* に焦点を当てた書き換え．



(b) *promote.* に焦点を当てた書き換え．



(c) *human–computer interaction.* に焦点を当てた書き換え．

図 3　書き換え機能．カーソル選択した箇所が中心的に書き換えられる．

定することで編集箇所の指示が可能になっている．

エディタ上での実装: エディタ上では，執筆者が文章の一部を範囲選択した際に書き換え機能が起動する．モデルへの入力について，ユーザがカーソル選択をした箇所に特殊記号が付与されることで編集箇所指定が実現される．システムによって提案された候補は，各候補の違いを区別しやすくするため，入力文と比較して追加されたトークンは青く，削除されたトークンが存在した箇所は赤く強調表示される．

3.2 その他の機能

補完機能: 文章中の適当な箇所から，続く文章を補完することができる．文章内の先行文脈と共に論文タイトルやセクションを考慮することができる．エディタ上では，利用者が Tab キーを押すと補完機能が実行される．ACL Anthology から収集した論文データでチューニングしたニューラル言語モデル [11] を使用している．詳細は付録 B に記載する．

誤り訂正機能: オープンソースの文法・綴り誤り訂正ツールである LanguageTool[4] を使用した．エディタ上では検出された誤りが下線で強調表示される．

4 実験

人とシステムの協働を見据えて実装したエディタが，想定利用者に対して有用であるかを調査する．

4.1 評価用データの作成

実際に論文執筆中の研究者を集めシステムの有効性を検証することは困難であるため，研究者が論文執筆の途中であるという状況を再現して実験を行う．執筆中の原稿として言語処理学会論文誌 LaTeX コーパス中の 8 つの日本語抄録を機械翻訳システム[5]で英訳した文章（以降草稿と呼ぶ）を用い，草稿の推敲を英語を母語としない研究者に依頼する．また，翻訳業者[6]が英訳した抄録を参照訳とした．論文執筆者として自然言語処理の研究を行う 16 名の学部生と修士課程の学生に推敲作業を依頼した．

4.2 実験設定

本エディタ上で実現される人とシステムの協働の有効性を調査するため，(i) 本エディタが提供する書き換え・補完機能を使わずに推敲を行う場合[7]（**人のみ**），(ii) 推敲に人が介入せず，入力に対して自動的に書き換え機能を適用した場合（**システムのみ**），(iii) 人がエディタの機能を活用して推敲した場合（**協働**）の 3 つの条件を比較する．システムのみの設定では，入力の各文に対して編集箇所指定を行わずに書き換え機能を適用し，もっとも生成確率の高い書き換え結果を採用した．

参加者には 2 つの草稿をそれぞれ人のみと協働設定で推敲してもらった．参加者の半数が最初に人のみの設定で原稿を推敲し，その後協働設定で別の原稿を推敲した．残りの半分の参加者は逆の順序で同じ作業を行った．制限時間は設けず，参加者には草稿とともにオリジナルの日本語抄録を提示した．完成した文章と参照訳を BLEURT [12] で比較した[8]．なお，BLEURT が出力する値については [0,1] といった値域が設定されておらず，参照訳と近いと値が大きくなる．またベースラインとして草稿（**編集なし**）についても参照訳との BLEURT を計算した．

表 1 各設定で書かれた文章の比較．

設定	BLEURT
協働	**-0.08**
人のみ	-0.14
システムのみ	-0.18
編集なし	-0.36

4.3 結果

表 1 に結果を示す．協働の設定で書いた文章の方が，人のみおよびシステムのみで書いたものよりも有意にスコアが高く[9]，本エディタにおける人とシステムの協働のための機能について，有用性が示唆された．また付録 C に，各設定で推敲された文章の統計を記載する．

5 分析

5.1 編集箇所指定機能の挙動

まず最初に 3.1 節で紹介した編集箇所指定機能（図 3）について実装の妥当性を検証する．具体的には，編集箇所として指定した範囲において，範囲指定されていない箇所と比べて頻繁に編集が行われて

4） https://github.com/languagetool-org/languagetool/releases/tag/v3.2
5） https://translate.google.com/
6） https://www.ulatus.com/
7） 通常の執筆環境を想定し誤り訂正機能は使えるものとした．
8） BLEURT は文の比較に用いられる．完成した文章と参照訳について予め文に分割し，BLEURT の値が最も大きくなる文を組とみなして各文の BLEURT を計算した（付録 D）．
9） ブートストラップ法による仮説検定 [13] ($p < 0.05$)

表 2 質問 1-6 に対するユーザー調査の結果．各値はその選択肢を選んだ参加者の割合を示す．

質問	そう思う	ややそう思う	あまり思わない	そう思わない
1	87.5	12.5	0	0
2	50.0	50.0	0	0
3	62.5	31.3	6.3	0
4	12.5	50.0	31.3	6.3
5	75.0	12.5	6.3	6.3
6	43.8	43.8	12.5	0

表 3 各機能が執筆に役立ったかのアンケート結果．

機能	割合
書き換え機能	100
補完機能	31.3
誤り修正機能	62.5

いることを確認する．

T 個のトークンからなる文 $\boldsymbol{x} = (w_1, \cdots, w_T)$ について特定の範囲 $s = (i, j)$ $(1 \leq i < j \leq T, 1 \leq j - i \leq 5)$ を無作為に決め，w_i の前と w_j の後に編集範囲の開始と終了を示す特殊記号を挿入する．記号が挿入された文を $\boldsymbol{x}^{\text{edit}}$ とする．$\boldsymbol{x}^{\text{edit}}$ に対して書き換え機能を適用し出力の生成確率上位 10 文 $(\boldsymbol{y}_1^{\text{edit}}, \cdots, \boldsymbol{y}_{10}^{\text{edit}})$ を得て，以下のスコアを算出する：

$$r = |\{\boldsymbol{y}_k^{\text{edit}} \mid \boldsymbol{x}_{i:j} \in \text{ngram}(\boldsymbol{y}_k^{\text{edit}}), 1 \leq k \leq 10\}| \ .$$

ここで $\boldsymbol{x}_{i:j}$ は編集範囲として指定された $\boldsymbol{x}$ の部分列 $(w_i, \cdots, w_j)$ である．また関数 ngram$(\cdot)$ は与えられた系列のすべての n-gram の集合を返す関数である．r は 10 個の出力のうち言い換えられて**いない**ものの数を示し，r が小さいほど指定範囲で書き換えが生じているとみなした．特殊記号を挿入せずに書き換え機能を適用した出力上位 10 文についても同様に上記のスコア r' を計算し r と r' を比較する．r' は範囲指定しない場合に上位 10 出力に置いてその範囲が偶然書き換わる回数である．

草稿からランダムに収集した 1,000 文を用い，r と r' の大小を比較する試行を 1000 回行った．結果は $r < r'$ が 340 回，$r = r'$ が 555 回，$r > r'$ が 105 回となった． r' より r が小さくなることが有意に高頻度に起きた[10] ことから本アプローチによって書き換え箇所の制御が行えているとみなした．

5.2 ユーザ調査

4.2 節の実験後，参加者に以下の項目についてアンケート調査を行った．

1. 協働の設定の方が執筆作業が快適．
2. 協働の設定の方が良い文章が作成できた．
3. 編集箇所指定機能が役立った．
4. 挿入箇所指定機能が役立った．
5. 書き換え機能で候補が複数提示されると役立った．
6. 補完機能で候補が複数提示されると役立った．

表 2 に結果を示す．1 と 2 に対する回答から本エディタ上での協働の有効性が示唆された．4 に対する回答より，挿入箇所指定機能については比較的利用者が恩恵を受けていないことが示唆された．複数の多様な候補を提示することについては「どれを選べばよいか分からない」といった利用者の負荷も懸念されていたが，5，6 に対する回答結果から利用者の印象は良かった．

また本エディタの各機能について，執筆で役立ったかを回答してもらった（表 3）．書き換え機能が最も役に立ったという評価を得ており，逆に言語モデルによる補完は比較的役立ったという回答が少なかった．少なくとも本実験や実際の論文執筆では何を書くかという内容は決まっており，言語モデルによる内容レベルの補完は役立つ場面が少ないことが示唆される．

6 おわりに

英語論文執筆支援システム Langsmith エディタを開発した．実験では．英語非母語話者が英語論文を推敲する際に本システムが有用であるかを検証した．16 名の被験者実験により本システムの有効性が示唆された．ただし，本稿の実験は参加者が限定的で草稿も擬似的なものであるため，実情を把握するためにはより大規模な実証実験が求められる．また，より効率的な支援システムの構築のため，人とシステムの円滑なインタラクションの方法を模索していく必要がある．

現在，医学・化学など様々な分野に特化したシステムも公開中であり，今後さらにモデルや機能を開発・追加していく．本エディタが学術界の言語的障壁の解決に寄与することを期待する．

謝辞

本研究は JSPS 科研費 JP19H04425，JP20J22697 の助成を受けたものである．

10） 符号検定を行った（p<0.05）

参考文献

[1] Robert Dale and Adam Kilgarriff. Helping our own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG 2011)*, pp. 242–249, 2011.

[2] Chung-Ting Tsai, Jhih-Jie Chen, Ching-Yu Yang, and Jason S. Chang. LinggleWrite: a coaching system for essay writing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL 2020)*, pp. 127–133, 2020.

[3] Seid Muhie Yimam, Gopalakrishnan Venkatesh, John Lee, and Chris Biemann. Automatic compilation of resources for academic writing and evaluating with informal word identification and paraphrasing system. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*, pp. 5896–5904, 2020.

[4] David Grangier and Michael Auli. QuickEdit: Editing text & translations by crossing words out. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2018)*, pp. 272–282, 2018.

[5] Takumi Ito, Tatsuki Kuribayashi, Hayato Kobayashi, Ana Brassard, Masato Hagiwara, Jun Suzuki, and Kentaro Inui. Diamonds in the rough: Generating fluent sentences from early-stage drafts for academic writing assistance. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG 2019)*, pp. 40–53, 2019.

[6] Qian Wang, Jiajun Zhang, Lemao Liu, Guoping Huang, and Chengqing Zong. Touch editing: A flexible one-time interaction approach for translation. 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 (AACL 2020)*, pp. 1–11, December 2020.

[7] Faeze Brahman, Alexandru Petrusca, and Snigdha Chaturvedi. Cue me in: Content-inducing approaches to interactive story generation. 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 (AACL 2020)*, pp. 588–597, December 2020.

[8] Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019) : Demonstrations*, pp. 89–97, June 2019.

[9] Ashwin K. Vijayakumar, Michael Cogswell, Ramprasaath R. Selvaraju, Qing Sun, Stefan Lee, David J. Crandall, and Dhruv Batra. Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018)*, pp. 7371–7379, 2018.

[10] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. Pay Less Attention with Lightweight and Dynamic Convolutions. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*, 2019.

[11] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. 2019.

[12] Thibault Sellam, Dipanjan Das, and Ankur Parikh. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pp. 7881–7892, 2020.

[13] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, pp. 388–395, July 2004.

[14] Takumi Ito, Tatsuki Kuribayashi, Masatoshi Hidaka, Jun Suzuki, and Kentaro Inui. Langsmith: An interactive academic text revision system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): System Demonstrations*, pp. 216–226, Online, October 2020.

[15] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. In *Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)*, 2020.

A　書き換え機能の詳細

Ito ら [5] と同様に疑似学習データを用いて書き換えモデルを学習した．具体的には，(i) 乱択削除モデル，(ii) 文法誤り付与モデル，(iii) 文体変換モデル，(iv) 含意文生成モデルの 4 種類の生成モデルを用いて，ACL Anthology Sentence Corpus[11] にノイズを加え疑似的な草稿を作成した．なお、Langsmith エディタでは誤り訂正機能が書き換え機能とは別に実装されているため，文法誤り付与モデルで生成されたデータは使用していない．乱択削除モデルはランダムに単語を削除したり，入れ替えたり，単語を “()” に置き換えたりするモデルである．この乱択削除モデルによる疑似的な草稿により，挿入箇所指定に実現した．編集箇所指定を実現するため，文体変換モデルで作成した草稿文と参照文を比較し，編集が必要な箇所を編集記号（<? ?>）で囲んだ．具体的には，参照文になく，草稿文にある表現に編集記号を挿入した．なお，草稿文と参照文に複数の編集箇所がある場合は，最長の編集箇所に編集記号を挿入する．より詳細な編集記号挿入アルゴリズムは文献 [14] の付録 A を参照されたし．

また．利用者に呈示される書き換え候補はニューラル言語モデルの生成確率によって，リランキングされている．ここで用いている言語モデルは補完機能で用いている言語モデルと同一である．

B　補完機能の詳細

モデルには，事前訓練されたニューラル言語モデル GPT-2 small（117M）を言語処理分野の論文でチューニングした．チューニング用のデータは，2019 年までに ACL Anthology に掲載された 234,830 件の論文を使用した．学習のパラメータについては文献 [14] の付録 B を参照されたし．また，補完機能でもサンプリング [15] を用いて，複数の候補を呈示している (図 4).

C　4 節の評価用データと推敲後の文章の統計量

表 4 に 4 節で作成した評価用データと各実験設定で書かれた文章の統計情報を示す．各値は平均値で，“±” に続く値は標準偏差を表している．

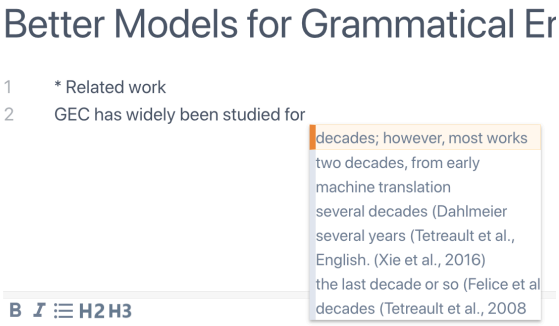


図 4　補完機能．これらの提案は，左のコンテキストとセクション名，論文タイトルによって条件づけられている．

表 4　4 節で作成した評価用データと各実験設定書かれた文章の統計量．

	長さ	単語の種類数
参照訳	199 ± 52	108 ± 17
協働	192 ± 40	101 ± 17
人のみ	192 ± 43	100 ± 16
システムのみ	199 ± 58	105 ± 22
草稿（編集なし）	202 ± 56	104 ± 22

D　4 節の評価方法

WMT Metrics ratings data でチューニングされた BLEURT-Base を用いた．[12] BLEURT は文ペアの類似度を評価する指標である．節 4 の実験では，実験参加者に抄録全体の推敲を依頼しており，文数や構成が参照訳と必ずしも一致しない．推敲後の文章と参照訳について文分割し，BLEURT の値が最も大きくなる文を組とみなして各文の BLEURT を計算した．なお，文分割には spaCy[13] を用いた．

11）https://github.com/KMCS-NII/AASC

12）https://storage.googleapis.com/bleurt-oss/bleurt-base-128.zip

13）https://spacy.io/