R2T: 言語モデルの確率操作による学習なし中間文生成

城戸晴輝　前川在　小杉哲　船越孝太郎　奥村学
東京工業大学
{haruki, maekawa, kosugi, funakoshi, oku}@lr.pi.titech.ac.jp

概要

中間文生成は，損傷した文章の復元，記事の執筆のような多くの状況で役立つ一般的なタスクであるものの十分な研究が行われていない [1]．本研究では，表面的なパターンを学習してしまうことや，近年の言語モデルのサイズの増加に伴い学習にコストがかかることなどの，教師あり学習モデルの持つ問題を回避するため，学習なしでの中間文生成に取り組む．手法としては，最後の文の埋め込みと各語彙トークンの埋め込みとのコサイン類似度を，自己回帰言語モデルが次トークンを予測する際に使用する語彙の対数確率に足し合わせるという単純なものである．これはあらゆる自己回帰言語モデル，復号化アルゴリズムに対して適用可能な Plug-and-Play な手法である．実験の結果，これまでの学習なし中間文生成手法と比較して，計算コストが十分に少なく，自動評価において大幅に上回る指標があるなど，新たな中間文生成の手法としての可能性を示した．

1　はじめに

中間文生成は，文や段落の欠落した文章を補完するテキストを生成することを目的としたタスクである．中間文生成の主な手法として，GPT2 [2] のような事前学習済みモデルの fine-tuning [3, 4, 5] や，ゼロから学習を行う [3, 4, 6] などの方法が挙げられる．しかし，大きなデータセットにより教師あり学習を行なってもなお，データセットに固有の表面的なパターンを学習してしまう [7] ことや，現在の言語モデルは 540B パラメータのもの [8] が出てくるなど，非常に巨大になっていることから，それと性能的に匹敵するモデルをゼロから学習したり，既存の事前学習済みモデルを fine-tuning するのにも多くのコストがかかることなど多くの問題が存在する．そこで本研究では，現在最も支配的なパラダイムである GPT のような，左から右への推論を行い，与えられたテキストの次に来るトークンを予測するモデルである自己回帰言語モデルにおいて，追加での学習なしに中間文生成を行うことが可能であり，あらゆる自己回帰言語モデルや復号化アルゴリズムに対して適用可能な手法を提案する．

追加での学習を必要としないこれまでの中間文生成手法として，TIGS [9] や DELOREAN [10]，COLD [11] が挙げられる．しかし，これらの手法は，最適化のために多くのステップを必要としたり，複数の候補文の生成を行う必要があるなど計算コストが大きいという問題がある．本提案手法では，自己回帰言語モデルが事前学習により獲得しているトークンの埋め込みベクトルを活用し，追加の学習を必要としない中間文生成を行う．具体的には，最後の文の各トークンとのコサイン類似度をモデルが出力する次トークンの対数確率に加算することで，これまでの文との継続性とともに最後の文を考慮して中間文を生成するという単純なものであり，通常の左から右への推論と同等の計算コストで生成を行うことができる．

実験は，1 文の中間文からなる *ART*Dataset と，3 文の中間文からなる ROCStories の 2 つのデータセットを使用して行なった．*ART*Dataset を使用した実験では，学習なし中間文生成の先行手法である COLD と DELOREAN との比較を行い，自動評価において提案手法の一定の有効性を確認した．さらに，計算コストの面でも DELOREAN や COLD に比べて 100 倍以上高速であることを確認した．ROCStories を使用した実験では先行手法と同様に複数の中間文生成にも拡張可能であることを確認し，今後の課題をより明確にした．

2　関連研究

TIGS は勾配情報を利用して中間文のベクトルを決定し，それを語彙ベクトルに離散化することで中間文を生成するものである．DELOREAN は通常の順伝播でこれまでの文との流暢な継続性を保証するだけでなく，逆伝播によりこれからの文に制約を

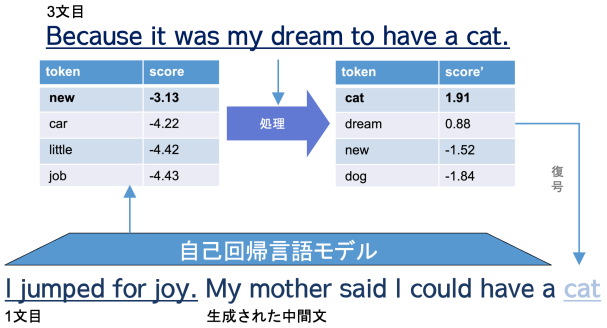

図 1　R2T の図解．1 文目として *I jumped for joy.* が，3 文目として *Because it was my dream to have a cat.* が与えられ，中間文として *My mother said I could have a* まで生成された時の score の値（$\lambda = 7$）．自己回帰言語モデルに *I jumped for joy. My mother said I could have a* を入力として与え score を得る．これに対し *Because it was my dream to have a cat.* に含まれる全トークンとのコサイン類似度の最大値に λ をかけたものを足し合わせることで確率分布を編集し中間文生成を行う．

付与することで，中間文の生成を実現する．COLD は，エネルギーベースモデルを利用し，これにこれまでの文脈と一貫性があり，最後の文とも一貫性があることを保証するような制約関数を導入することで中間文の生成を実現している．これらの学習なし中間文生成手法における共通の課題として，計算コストが大きいことが挙げられる．こうした既存手法の問題点を解決するために，この論文では生成時に最適化を一切必要としない単純な手法を提案する．

3　提案手法

本研究では，従来の左から右への推論を行う自己回帰言語モデルに対して，中間文生成，つまり，最後の文を考慮してこれまでの文の続きを生成するための手法として Right sentence to Text（R2T）を提案する（図 1）．これは，K2T [12] を中間文生成タスクに応用する試みである．K2T は，word2vec [13] などの単語埋め込みを利用し，ガイド単語に意味的に近い単語の対数確率を増加させることにより，一切追加での学習をすることなくガイド単語の出現制約を満たすテキスト生成を行うものである．さらに，中間文生成に関連する先行研究 [14] によると，コンテキスト RNN の入力に対し，入力された文から生成された文の分散表現だけでなく，最後の文の分散表現を加算することで，より良い中間文生成が可能になるという結果が得られている．

これらの先行研究を組み合わせ，自己回帰言語モデルが次のトークンを予測する際に使用する語彙の

表 1　*ART*Dataset の例

	Story
1 文目	The Smiths were having holidays done of the children.
2 文目	The Smiths bought toys for the kids.
3 文目	Ty's face lit up as he ran to the new toy, happily posing for photos.

表 2　ROCStories の例

	Story
1 文目	The Smiths were having holidays done of the children.
2 文目	Ty's face lit up as he ran to the new toy, happily posing for photos.
3 文目	The Smiths bought toys for the kids.
4 文目	Ty's face lit up as he ran to the new toy, happily posing for photos.
5 文目	The Smiths bought toys for the kids.

対数確率 $\mathrm{score}(y_t \mid \mathbf{y}_{<t})$ に対して，最後の文 $\mathbf{s}$ の埋め込みとのコサイン類似度を加算することを考える．ここで $\mathrm{score}(y_t \mid \mathbf{y}_{<t})$ は Logits に対して log_softmax を適用したものであり，$\mathbf{y}_{<t}$ は 1 文目とそれまでに生成された文，y_t は次に生成するトークンの候補である．score は Softmax 関数にかけられ，次トークン y_t の確率として機能する．ここで，最後の文の埋め込みとのコサイン類似度を，最後の文に含まれる全トークンの埋め込みとのコサイン類似度のうち最大のものとすると，次のように score の編集を定式化することができる．また，$\gamma(\mathrm{token})$ は token を事前学習済み自己回帰言語モデルが獲得しているトークンの埋め込みベクトルに変換したものである．

$$\mathrm{score}'(y_t, \mathbf{s}|\mathbf{y}_{<t}) = \mathrm{score}(y_t|\mathbf{y}_{<t}) + \lambda_i \cdot \max(0, \max_{\mathrm{token}\in\mathbf{s}} \cos(\gamma(y_t), \gamma(\mathrm{token})))$$

ここで，最後の文の影響の強弱は中間文の各文目ごとに λ_i で制御することができる．特に，$\lambda_i = 0$ においては $\mathrm{score}'(y_t, \mathbf{s}|\mathbf{y}_{<t}) = \mathrm{score}(y_t|\mathbf{y}_{<t})$ であり，score の編集の影響を受けず，素の自己回帰言語モデルと同じ score を取る．また，生成する中間文が N 文（$N \geq 2$）の場合，λ_i は最後の文からの距離に比例すると近似し，$\lambda_i = a + (i-1) \times \frac{b-a}{N-1}$ で求まるとする．特徴としては，モデルが持つトークンの埋め込みを使用するため，追加での学習なしに中間文生成が可能であることと，任意の自己回帰言語モデルや復号化アルゴリズムに適用できることが挙げられる．また，モデルが出力する score に対し，最後の文に含まれる全トークンの埋め込みとのコサイン類似度のうち最大のものに λ をかけたものを足し合わせるだけの単純な手法であるため，計算コストは素の自己回帰言語モデルと同等である．

表 3 *ART*Dataset のテストデータにおける R2T の自動評価結果と先行研究との比較．上から 3 つの手法は [11] における Reported Score である．LEFT-ONLY に関しては，R2T と同じ $p = 0.9$ の核サンプリングを用いた場合と比較した．また，実際に 1 つの物語の中間文の生成に要した時間を計測した．

Model	BLEU4	ROUGE_L	CIDEr	BERTScore	METEOR	時間
LEFT-ONLY	0.88	16.26	3.49	38.48	-	-
DELOREAN [10]	1.60	19.06	7.88	41.74	-	3min
COLD [11]	**1.79**	**19.50**	10.68	**42.67**	-	19min
LEFT-ONLY	0.51	12.85	3.69	38.02	10.46	1sec
R2T	1.35	17.81	**11.63**	40.02	**12.69**	1sec

4 実験

4.1 データセット

実験には，*ART*Dataset [15] と ROCStories [16] の 2 つのデータセットを使用した．*ART*Dataset は，学習なしで中間文生成を行う先行研究である DELOREAN，COLD の実験環境と一致させ，Reported Score との比較を行う目的で使用し，ROCStories は先行手法と同様に複数文の中間文生成にも拡張可能かを確認する目的で使用した．

*ART*Dataset は，ROCStories から抽出された，1 文目と 3 文目に対応する Obs_1，Obs_2 と，その中間文である 2 文目に対応する Hyp などが含まれたデータセットである（表 1）．λ のチューニングのため検証データ（7,252 件）を，評価のためテストデータ（14,313 件）を使用した．

ROCStories は，5 文のコモンセンスストーリーからなるデータセットである（表 2）．ROCStories Winter 2017 の全データ 52,665 件のうち，ランダムに 1,000 件を抽出したものを検証データとして使用し，さらに検証データを除いた 51,665 件の中からランダムに 1,000 件を抽出したものをテストデータとして使用した．また，同じデータセットに対し 10 回生成と評価を行い，その平均を結果とした．

4.2 評価指標

先行研究 [11] を参考に，BLEU4 [17]，METEOR [18]，ROUGE [19]，CIDEr [20]，BERTScore [21] の 5 つを評価指標とした．また，生成テキストの 3-gram の繰り返しを測る Repetition Percentage（RP），最初と最後の文の 3-gram と生成テキストの 3-gram の重複率を測る Overlap Percentage (OP) の 2 つを追加で導入した．

表 4 ROCStories のテストデータにおける素の gpt2-xl と R2T の自動評価の結果．

Model	BLEU4	ROUGE_L	CIDEr	BERTScore	METEOR
LEFT-ONLY	0.22	10.35	4.85	40.49	10.00
R2T	**0.48**	**12.50**	**8.32**	**41.79**	**10.43**

4.3 実験設定

先行研究 [11] を参考に，ベースの自己回帰言語モデルとして gpt2-xl を使用した．復号化アルゴリズムとしては $p = 0.9$ の核サンプリング [22] を使用した．

文の終端の判定には，nltk ライブラリの sent_tokenize[1] を用いた（付録 A）．なお，データセットが有する問題（付録 B）に対処するため，生成されたテキストの 1 トークン目がピリオドである場合は，削除し文の終端の判定を行っている．付録 A により定義された条件を満たさないテキストが生成された場合は，生成されるまで繰り返し生成を行った．また，λ が大きくなると，最後の文にピリオドがなく生成されたテキストが文として終了しないという問題が発生した（$\lambda = 10$ においては 0.2%程度の確率）．これに対処するためにピリオドトークンの有無に関わらず，最後の文に対してはピリオドトークンを追加した．

*ART*Dataset については，検証データにおける結果（付録 C）から，$\lambda_1 = 7$ を採用し，ROCStories については，$(a, b) = (2, 8)$，つまり $(\lambda_1, \lambda_2, \lambda_3) = (2, 5, 8)$ を採用した（付録 D）．

5 結果

***ART*Dataset** 表 3 は DELOREAN と COLD の Reported Score と本手法 R2T のテストデータにおける評価結果である．これから，提案手法である R2T は，素の gpt2-xl と比べて中間文生成の自動評価が全て改善されていることがわかる．また，提案手法と COLD を比較すると，CIDEr を除いて劣るものの，CIDEr に関しては先行手法を大幅に上回っている．また，改善幅で見ると，BERTScore の改善幅は先行手法の半分程度であるものの，BLEU4 と ROUGE_L に関してはほぼ同等である．

CIDEr において提案手法 R2T が高い性能を示している理由として，CIDEr は TF-IDF という他のテキストにおける出現確率を考慮した n-gram の一致率を評価していることが挙げられる．R2T は最後の

1) https://www.nltk.org/api/nltk.tokenize.sent_tokenize.html

表 5 *ART*Dataset のテストデータにおける R2T による生成例. 3 文目との共通単語を太字で強調している

1 文目	3 文目	生成テキスト	参照テキスト
Aurelia was given an ice cream maker for her birthday.	**She** began brainstorming a new **ice cream** recipe.	**She** liked **ice cream**.	Aurella loved **ice cream.**
Matt wanted to travel.	**He decided to** stay in **New Zealand** forever.	**He** took a trip **to New Zealand**.	Matt **decided to** travel **to New Zealand**.
I was born in a small town.	**I was** buried **in a small town**.	**I was a small town**.	**I** grew up **in a small town**.

表 6 ROCStories のテストデータにおける R2T による生成例. 5 文目との共通単語を太字で強調している.

1 文目	5 文目	生成テキスト	参照テキスト
The boy teased the girl.	**The girl got in** trouble.	He began to be bold. He **got** up **in** front of **the girl**. She **got in** front of **the** boy.	**The girl got** mad. She punched **the** boy. **The** boy told on **the girl**.
Yesterday I went on a trip to the islands.	Then **I** ate **it**.	**I** wrote a profile of the island **I** visited. **I** finished **it**. **I** went to the hotel.	**I** went deep sea fishing. **It** was very fun. **I** caught a large sea bass.

文に存在するトークンの出現確率を強制的に増加させており，各物語に固有な単語が出現する場合が多い．例えば表 5 の *New Zealand* という単語は他の物語では出現頻度が低いが，提案手法は最後の文の単語の出現確率を増加させることで，このような出現頻度の低い単語も正確に予測できているため，CIDEr は高い値を達成できている．n-gram の一致率が先行手法に比べて低い理由としては，事前学習により獲得している尤もらしいテキストの生成能力が score の編集により減少していることが理由として考えられる．BERTScore が先行手法に比べて半分程度の改善しか見られなかった理由としては最後の文の影響が少ないことが考えられる．現手法では，元の自己回帰言語モデルの確率分布によらずコサイン類似度を足し合わせているので，λ が大きくなるにつれ繰り返しが増加するなどの問題が起きやすく十分に最後の文の影響を強めることができなかったと考えられる．この問題は元の自己回帰言語モデルの出力した確率分布をさらに活用することで解消できる可能性がある．

生成にかかる時間を見ると，DELOREAN や COLD は最適化のために多くのステップを必要とするため，GeForce RTX 3090 においては数分から数十分程度を必要とするが，提案手法においては素の gpt2-xl による左から右への推論にかかる時間と同程度の 1 秒で生成が完了する．このことから，先行手法に比べて 100 倍以上高速に中間文が生成可能であり，より現実のアプリケーションに使用しやすくなることが想定される．

表 5 が *ART*Dataset を元に R2T により生成された実際のテキストである．参照テキストと最後の文との共通部分が多い場合に，この手法が特に有効に働くことがわかる．2 つ目の生成例をみると，従来の左から右への推論では生成されなかったであろう *New Zealand* という具体的な旅行先が正しく生成されていることがわかる．ただし，素の自己回帰言語モデルが獲得している文法的な正しさなどを無視して強制的に対数確率にコサイン類似度を加算しているので，*I was a small town.* のような不適切な文が生成される場合もある．

ROCStories　表 4 より，複数の中間文を生成する場合についても素の gpt2-xl に比べ，中間文生成の自動評価は全て改善されていることがわかる．さらに，複数文からなる中間文については，中間文生成に関連する先行研究 [14] と同様に，最後の文に近づくにつれ最後の文の影響を強めることでより良い中間文生成が可能になることも示している．

表 6 が ROCStories を元に R2T により生成された実際のテキストである．ROCStories は *ART*Dataset に比べ，最初の文と最後の文に距離があり，中間文の自由度が高い．そのため，単純に最後の文に含まれる全トークンとのコサイン類似度を足し合わせるだけでは，中間文が最後の文に対し意味的に一貫していない場合が多いことがわかる．

6　おわりに

本稿では教師あり学習モデルの持つ問題を回避するために，追加での学習を必要としない中間文生成手法である R2T を提案した．実験により，自動評価において R2T が素の自己回帰言語モデルよりも中間文の生成能力が高いことや，先行手法に対して計算コストが十分少ないにも関わらず，評価指標 CIDEr においては大幅な改善が見られることなど新たな中間文生成の手法としての可能性を示した．今後として，単純にコサイン類似度を足し合わせるだけでなく元の確率分布をより効率的に活用した方法によって言語モデルが持つさまざまな能力を活かした中間文生成手法を実現し，最後の文との意味的な一貫性の向上を目指す予定である．

参考文献

[1] Wanrong Zhu, Zhiting Hu, and Eric P. Xing. Text infilling. **arXiv preprint arXiv:1901.00158**, 2019.

[2] 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.

[3] Mohammad Bavarian, Heewoo Jun, Nikolas A. Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. Efficient training of language models to fill in the middle. **arXiv preprint arXiv:2207.14255**, 2022.

[4] Chris Donahue, Mina Lee, and Percy Liang. Enabling language models to fill in the blanks. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 2492–2501, Online, July 2020. Association for Computational Linguistics.

[5] Pengjie Wang, Liyan Wang, and Yves Lepage. Generating the middle sentence of two sentences using pre-trained models: a first step for text morphing. In **Proceedings of the 27th Annual Conference of the Association for Natural Language Processing**, pp. 1481–1485, 2021.

[6] Anh Nguyen, Nikos Karampatziakis, and Weizhu Chen. Meet in the middle: A new pre-training paradigm. **arXiv preprint arXiv:2303.07295**, 2023.

[7] Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. Counterfactual story reasoning and generation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, **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. 5043–5053, Hong Kong, China, November 2019. Association for Computational Linguistics.

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. **J. Mach. Learn. Res.**, Vol. 24, pp. 240:1–240:113, 2022.

[9] Dayiheng Liu, Jie Fu, Pengfei Liu, and Jiancheng Lv. TIGS: An inference algorithm for text infilling with gradient search. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**, pp. 4146–4156, Florence, Italy, July 2019. Association for Computational Linguistics.

[10] Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 794–805, Online, November 2020. Association for Computational Linguistics.

[11] Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, **Advances in Neural Information Processing Systems**, Vol. 35, pp. 9538–9551. Curran Associates, Inc., 2022.

[12] Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. A plug-and-play method for controlled text generation. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, **Findings of the Association for Computational Linguistics: EMNLP 2021**, pp. 3973–3997, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[13] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In **International Conference on Learning Representations**, 2013.

[14] 飯倉陸, 岡田真, 森直樹. 潜在変数付き階層型エンコーダ・デコーダモデルに基づく物語の補完的生成手法の提案. 人工知能学会全国大会論文集, Vol. JSAI2021, pp. 3D4OS12c03–3D4OS12c03, 2021.

[15] Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. Abductive commonsense reasoning. In **International Conference on Learning Representations**, 2020.

[16] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, **Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, pp. 839–849, San Diego, California, June 2016. Association for Computational Linguistics.

[17] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, **Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics**, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[18] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Jade Goldstein, Alon Lavie, Chin-Yew Lin, and Clare Voss, editors, **Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization**, pp. 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[19] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In **Text Summarization Branches Out**, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[20] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In **2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, pp. 4566–4575, 2015.

[21] Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In **International Conference on Learning Representations**, 2020.

[22] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In **International Conference on Learning Representations**, 2020.

表 7　ピリオドがない例

	Story
1 文目	Amy was shopping for new slippers
2 文目	Amy found the comfiest slippers of all time.
3 文目	Amy ended up wearing the slippers until the sole fell out.

表 8　*ART*Dataset の検証データにおける R2T の λ_1 の値による自動評価の結果

λ_1	BLEU4	ROUGE_L	CIDEr	BERTScore	METEOR	RP	OP
0(no ctrl)	0.38	12.96	3.89	38.09	10.45	**0.03**	**0.49**
1	0.70	14.03	4.37	38.44	11.39	0.04	0.63
2	0.45	15.26	5.15	38.96	11.75	0.05	0.90
3	0.00	16.18	6.47	39.26	12.17	0.11	1.22
4	1.05	16.79	7.50	39.65	12.90	0.21	1.93
5	0.76	16.85	8.17	39.84	12.49	0.63	3.45
6	0.75	17.87	10.15	40.12	**12.96**	1.11	5.23
7	**1.27**	**18.08**	11.85	**40.36**	12.67	1.74	7.72
8	1.22	17.88	11.87	40.12	12.40	2.34	9.33
9	1.22	17.46	12.59	40.23	11.93	2.99	11.85
10	1.17	17.23	13.06	40.25	11.80	3.30	13.33
11	1.15	17.46	**13.61**	40.29	11.88	3.35	14.29

A　一文の定義

左から右への推論を行う自己回帰言語モデルは，過去のテキストの続きを生成するのみであるため，どのように文の終端を判定するかは重要な問題である．本実験では，過去の文の続きを 20 トークン生成し，その生成されたテキストが nltk ライブラリの sent_tokenize により，2 文以上に分割され，かつその 1 文目が 2 トークン以上である場合にそれを一文とみなす．例として，*"I have a cat."*を入力としたときに，*"It is cute. The cat is ..."*という文が生成された場合，*"It is cute."*を次の一文とする．

B　*ART*Dataset における問題

本研究において，第 4 節で使用した *ART*Dataset の dev-w-comet-preds と test-w-comet-preds について，表 7 のように，ピリオドが不適切に存在しないテキストが全体の 1%程度見受けられた．本手法において，ピリオドが不適切に存在しないテキストを入力として与えられた時に次のトークンとしてピリオドを生成する場合が多く見られた．これは sent_tokenize により 1 トークン目を文の終端と判断されてしまうものの，1 トークンからなる文であるため本実験における一文の定義（付録 A）と外れてしまうため，再度生成が行われ，ループ状態に陥る．そこで今回は生成された中間文の 1 トークン目がピリオドである場合はそれを削除したのちに文の終端の判定を行っている．これは *ART*Dataset の元のデータセットである ROCStories（ROCStories spring2016）にも共通する問題である．

C　*ART*Dataset における λ の決定

表 8 の結果より，傾向としては以下が見られる．

- BLEU 4: $\lambda_1 = 7$ まで増加
- ROUGE_L: $\lambda_1 = 7$ まで増加し，以降減少
- CIDEr: λ_1 の増加とともに単調に増加
- BERTScore: $\lambda_1 = 7$ まで増加
- METEOR: $\lambda_1 = 6$ まで増加し，以降減少
- RP, OP: λ_1 の増加とともに単調に増加

以上より，$\lambda_1 = 7$ を採用した．

表 9　ROCStories の検証データにおける R2T の (a, b) の値による自動評価の結果．BERTScore について降順で上位 7 件．

(a, b)	BLEU4	ROUGE_L	CIDEr	BERTScore	METEOR	RP	OP
(2, 10)	0.47	12.72	8.77	**42.02**	10.31	4.09	3.73
(2, 8)	0.46	**12.90**	**8.9**	42.01	**10.72**	3.22	**2.97**
(2, 9)	0.43	12.71	8.64	41.96	10.42	3.68	3.34
(1, 10)	0.37	12.51	8.97	41.93	10.46	3.18	3.46
(1, 9)	0.47	12.54	9.07	41.91	10.65	**2.67**	3.03
(3, 9)	0.49	12.79	8.54	41.9	10.08	5.30	3.70
(3, 8)	**0.51**	12.89	8.57	41.86	10.28	4.72	3.34

表 10　*ART*Dataset の少量の検証データにおける R2T の λ_1 の値による自動評価の結果

λ_1	BLEU4	ROUGE	CIDEr	BERTScore	METEOR	RP	OP
0	0.00	7.06	8.89	39.35	5.81	0.06	1.20
1	0.00	7.39	9.73	39.57	5.63	**0.00**	**0.32**
2	0.00	8.63	8.47	40.33	6.31	**0.00**	0.44
3	0.00	8.92	10.02	40.61	6.04	0.08	0.90
4	0.00	8.96	11.50	41.02	6.13	**0.00**	1.48
5	0.00	10.00	13.78	40.66	**6.78**	0.89	2.94
6	0.00	10.67	13.30	41.49	6.31	0.94	3.56
7	0.00	10.18	11.99	41.39	6.14	1.40	2.13
8	0.00	**10.88**	13.91	40.63	6.26	4.31	4.41
9	0.00	10.73	**14.12**	**41.64**	6.66	4.37	5.96
10	0.00	9.04	11.08	40.90	5.57	3.54	7.10
11	**0.23**	9.61	13.51	40.31	5.72	6.04	6.53

D　ROCStories における λ の決定

ROCStories において中間文は 3 文であるので，$\lambda_i = a + (i-1) \times \frac{b-a}{2}$ から，$(\lambda_1, \lambda_2, \lambda_3) = (a, \frac{a+b}{2}, b)$ と置き換えることができるため，この (a, b) の値を決定する．表 9 より $(a, b) = (2, 8)$ と決定され，$(\lambda_1, \lambda_2, \lambda_3) = (2, 5, 8)$ を採用した．

表 11　*ART*Dataset のテストデータにおける少量のデータセットで決定された $\lambda_1 = 6$ を使用した R2T の自動評価の結果．

λ, model	BLEU4	ROUGE	CIDEr	BERTScore	METEOR
LEFT-ONLY ($\lambda_1 = 0$)	0.51	12.85	3.69	38.02	10.46
R2T ($\lambda_1 = 7$)	**1.35**	**17.81**	**11.63**	**40.02**	**12.69**
R2T ($\lambda_1 = 6$)	1.16	17.15	9.82	39.89	12.27

E　少量の検証データによる λ の決定

本手法の最も重要な特徴として，学習なしで実現可能なことを挙げているが，λ を適切な値に設定しないと，繰り返しが増加したり，本研究で示した性能を下回る性能を示してしまうことが考えられる．本研究では λ の値を設定するために，第 4 節において，*ART*Dataset の dev-w-comet-preds（7,252 件）を使用している．しかしこれだけのデータセットを集めること自体にコストが発生する．そこで dev-w-comet-preds の中からランダムに抽出した 20 件の小規模なデータセットを使用し λ の決定を行い，本実験で示した自動評価の値と比較を行う．表 10 より $\lambda = 6$ と決定され，これをテストデータにより評価すると表 11 のようになる．本論文中で使用した $\lambda_1 = 7$ の性能と比較すると BLEU4，CIDEr では 85%程度の性能を示し，ROUGE，BERTScore，METEOR では同程度の性能を示している．これらから，少ないデータセットを使用し λ の値を設定してもなお，本論文で示した性能と類似した性能を示すことが可能であることが確認された．