部分木の類似性に基づく言語生成技術の自動評価

帖佐 克己, 平尾 努
NTT コミュニケーション科学基礎研究所
{katsuki.chousa.bg, tsutomu.hirao.kp}@hco.ntt.co.jp

概要

系列変換モデルの発展により，言語生成技術の性能は飛躍的に向上し，多様な表現で流暢な文を生成できるようになってきた．言語生成技術をさらに発展させるには自動評価法が必要となるが，単語の一致に基づく従来の自動評価法は参照テキストと同じ意味を持つテキストであっても表現が異なれば低いスコアしか与えられない．一方，単語埋め込みベクトルを活用した自動評価法は，単語の類似性を重視するため，似たような単語列であれば意味が違っても高いスコアを与える．こうした問題を解決するため，本稿では，2 つの文の間の類似性を構文木の類似性に基づき決定する手法を提案する．WMT Metric Shared Task のデータセットを用いて人手評価との間の相関を調べた結果，提案法は教師データを必要にしないにもかかわらず，既存手法と同等以上の高い相関を達成した．

1 はじめに

言語生成技術の性能は系列変換モデル，特に Transformer モデル [1] の発展とともに大きく向上した．ニューラルネットワークを用いることで語の類似性をよく捉えることができ，多様な表現のテキストを生成できる．言語生成技術の自動評価指標には，生成されたテキストとあらかじめ用意された参照テキストとの間の単語列の一致に基づく類似度が広く用いられている．機械翻訳の場合は，BLEU [2]，自動要約の場合は ROUGE [3] という N グラムの一致率を利用した手法がデファクトスタンダードであるが，N グラムの一致率という表層的な情報に基づく手法には，参照テキストと同じ意味のテキストであっても，表現が異なれば不当に低いスコアを与えるという問題がある．この問題に対して，近年提案された BERTScore [4] や BLEURT [5] では，単なる表層上の一致に頼らず，BERT [6] などの事前学習済みモデルから得られる単語や文の文脈埋め込みベクトルを活用し，参照テキストと生成されたテキストとの間の類似度を計算する．

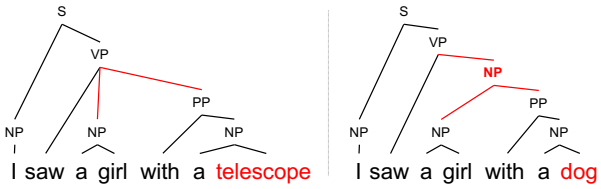

図 1 表層情報がほとんど同じなのに構造および意味が異なるテキスト対の例

しかし，これらの研究ではテキストがその背後に持つ句構造や依存構造などの構文構造，修辞構造のような談話構造を利用していない．たとえば図 1 に示したテキスト対では，表層情報のみに着目すると telescope と dog という名詞 1 つが異なるだけの意味の似た文に見えるが，その構造にも着目すると左のテキストでは前置詞句（PP）が動詞句（VP）の直接的な子であるのに対して，右のテキストでは前置詞句は名詞句（NP）の子となっていることから，その意味も異なることがわかる．このようなテキストを構成する単語がほぼ同じなのに意味が異なるようなテキスト対などに対して，BLEU などの表層情報の一致に基づく手法では曖昧性を解消できないことから不当に高いスコアを与えてしまう．また，BERTScore などでも単語の類似性を重視するため，同様に不当に高いスコアを与えてしまう．この問題に対して，テキストの構造は曖昧性を解消し，テキスト間の差異を捉える有用なヒントとなり得るが，その有用性は明らかにはなっていない．

本稿では，構文木の間のアラインメントとその埋め込みベクトルの類似度に基づいた言語生成技術の自動評価法を提案する．2 つのテキストの部分木を列挙した後，いくつかの制約の下での部分木間のアラインメントを決定し，最後にアラインメントに基づいた部分木間のベクトル類似度を元にテキスト間の類似度を計算する．WMT20 および WMT21 の Metrics Shared Task のデータセットに対して，提案

手法と人手による評価との間の相関を調べたところ，提案手法は教師データを必要としないにも関わらず，タスクに参加した既存手法と比べて同程度もしくは高い相関を達成した．

2 関連研究

BLEU [2] や ROUGE [3] に代表される N グラムの一致率に基づく自動評価法では言い換えを正しく同定できないことから，近年は大規模事前学習済みモデルによる単語や文の文脈埋め込みベクトルに利用した手法が提案されている．BERTScore [4] は，テキストのトークンの埋め込みベクトルのコサイン類似度に基づいてテキスト間のトークンのアラインメントを決定し，そのアラインメントに基づいてテキスト間の類似度を計算する．この際，埋め込みベクトルの計算に用いられる事前学習済みモデルはファインチューニングを行わないため，教師データを必要としない．WMT18 metrics shared task ではシステム単位での自動評価において人間の評価結果と高い相関があることが報告されている．

一方で，BLEURT [5] では，BERT [6] の最終層の [CLS] ベクトルの上に線形層を追加して類似度を計算する．機械翻訳の自動評価に利用するため，BERT および線形層のパラメータは，人手でアノテートされた翻訳評価データを用いて追加学習される．よって，人手で作成された教師データが必要となる．

また，系列変換モデルによる言い換え生成を利用した手法も提案されている．Prism [7] は，Seq2Seq モデルとして作成した言い換え生成器を用いて，参照訳から翻訳文，もしくはその逆の方向で言い換えを生成した際のスコアを評価値とする手法を提案している．このとき，言い換え生成器は多言語翻訳モデルを学習することで実現され，それによりゼロショット翻訳タスクと同様の枠組みで同一言語間の言い換えの生成を行う．また，Prism は 39 言語に対応しており，WMT19 metrics shared task のセグメント単位での自動評価において他のすべての手法と同程度もしくは高い相関を達成している．

事前学習済みモデルには潜在表現として構文情報が保持されているという分析 [8] が行われているものの，先行研究の評価はサブワードや文という単位に基づいて計算されることから，得られるスコアもその計算の単位により制限されてしまい，表層情報が似ているが意味が異なる文対に対して不当に高いスコアを与えてしまう恐れがある．こうした問題の解決のため，構文情報は有用であると考えられるが，文対の間の構文の違いを明示的に利用した自動評価法は提案されていない．もちろん，構文解析分野において，2 つの構文木の違いを評価する指標は提案されている [9, 10, 11]．ただし，これらは，同じ文に対する異なる構文木を評価するものである．つまり，文が一致している場合にしか利用できないので言語生成技術の自動評価には利用できない．

3 提案手法

本稿では，構文木の部分木の間のアラインメントとその埋め込みベクトルの類似度に基づく自動評価法を提案する．提案法は（1）2 つの構文木の部分木を列挙した後，（2）部分木間のアラインメントを求め，最後に（3）アラインメントに基づいた部分木間のベクトル類似度を用いて 2 つの文の間の評価スコアを計算する．

(1) テキストの部分木の列挙 2 つのテキスト $x = \langle x_1, \cdots, x_i \rangle$ と $y = \langle y_1, \cdots, y_j \rangle$，およびその埋め込みベクトルである $\langle \boldsymbol{x}_1, \cdots, \boldsymbol{x}_i \rangle$ と $\langle \boldsymbol{y}_1, \cdots, \boldsymbol{y}_j \rangle$ が与えられたとする．

まず，与えられた 2 文の構文を解析する．次に，アラインメントを取る対象とする部分木を列挙する．構文木として依存構造木を採用し，部分木として単一のノード，葉から各ノードへのパス，任意のノードとその子孫を列挙する．このとき，テキスト x の部分木を s_m，部分木 s_m に含まれる単語の添字列を $\mathrm{idx}(s_m)$ と定義し，部分木 s_m に対応するベクトルを

$$\boldsymbol{s}_m = \frac{\sum_{k \in \mathrm{idx}(s_m)} \boldsymbol{x}_k}{\|\mathrm{idx}(s_m))\|} \tag{1}$$

のように計算する．また，テキスト y の部分木 t_n に対しても同様に定義，計算を行う．

(2) 部分木間のアラインメント 次に，列挙した部分木 s_m, t_n の間のアラインメントを求める．このとき，列挙されたすべての部分木の間で対応付けを行うと，木構造の上での親子関係が逆転するような対応や否定の関係にある対応が得られてしまうという問題が考えられる．この問題の影響を低減するため，本手法では対応付けられるのは同じ単語長 l を持つ部分木間のみに限るという制約を課す．

本手法では，BERTScore [4] を参考にして，個々の部分木に対してもう一方のテキストの部分木から最もスコアの高いものを選択することでアライ

ンメントを求める．すなわち，単語長 l の s_m^l, t_n^l のそれぞれに対応するもう一方のテキストの部分木 $a(s_m^l), a'(t_n^l)$ は以下の通り求める．

$$a(s_m^l) = \max_{t_n^l} sim(\boldsymbol{s}_m^l, \boldsymbol{t}_n^l) \tag{2}$$

$$a'(t_n^l) = \max_{s_m^l} sim(\boldsymbol{t}_n^l, \boldsymbol{s}_m^l) \tag{3}$$

ここで，$sim(s,t)$ は部分木 s,t 間のスコアを表し，部分木 s の主辞 $\mathrm{head}(s)$ を用いて以下のように表す．

$$sim(s,t) = \begin{cases} \frac{st}{\|s\|\|t\|} & (\mathrm{head}(s) = \mathrm{head}(t)) \\ 0 & (\mathrm{otherwise}) \end{cases} \tag{4}$$

上記で定義した部分木間のスコアでは，部分木の主辞の一致による対応付けのフィルタリングを行っている．事前に対応付けられる候補をフィルタリングすることにより，対応関係を求める際の計算量を少なく抑えることができる．また，フィルタリングを行わない場合には，対応しない部分木同士の対応に対しても 0 より大きいスコアを割り当ててしまい，類似度尺度に対してノイズとなってしまうことが考えられる．事前にフィルタリングを行って，明確に対応しない部分木間のスコアに 0 を割り当てることで，この問題を軽減されることが期待できる．このことより，本手法では主辞に基づく対応付けのフィルタリングを導入している．

(3) アラインメントに基づくテキスト間類似度

最後に，各単語長 l の部分木間のアラインメントに基づいたスコアを計算し，それらのスコアから最終的なテキスト間類似度を計算する．

まず，テキスト x,y の間の単語長 l のアラインメントに対するスコア F_l を，アラインメントの x に対する平均スコア P_l と y に対する平均スコア R_l から，以下のように定義する．

$$F_l = \frac{2P_lR_l}{P_l + R_l} \tag{5}$$

$$P_l = \frac{\sum_{s_m^l} sim(s_m^l, a(s_m^l))}{\text{the number of } s_m^l} \tag{6}$$

$$R_l = \frac{\sum_{t_m^l} sim(a'(t_m^l), t_m^l)}{\text{the number of } t_m^l} \tag{7}$$

次に，上記で得られた複数の単語長のスコアに基づいて，最終的なテキスト間類似度 F を以下の通り求める．

$$F = \frac{2PR}{P+R}, P = \frac{\sum_{l\in I} F_l}{\|I_x\|}, R = \frac{\sum_{l\in I} F_l}{\|I_y\|} \tag{8}$$

ここで，I_x, I_y はそれぞれ x,y の部分木のもつ単語長の種類の集合であり，$I = I_x \bigcup I_y$ である．

4 実験

言語生成タスクの 1 つである機械翻訳を対象として人間の評価結果との間の相関で提案法のメタ評価を行った．

4.1 設定

データセットには WMT20 および WMT21 の Metrics Shared Task [12, 13] のシステムレベルの評価タスクのデータセットを用いた．言語対は目的言語が英語であるもののみ，すなわち WMT20 では {cs,de,iu,ja,km,pl,ps,ru}-en，WMT21 では {cs,de,is,ja,ru,zh}-en の ref-A と呼ばれる参照訳を用いた[1)]．ベースラインはそれぞれの年度でタスクに参加していた尺度の中から，WMT20 では Prism [7] と BLEURT [5]，WMT21 では Prism と BERTScore [4] を採用した．評価尺度には，WMT20 に従って，ピアソンの積率相関係数を用いた[2)]．

提案手法における部分木の間の類似度を計算するための事前学習済み言語モデルには，SpanBERT [14] を使用した．WMT20 の ja-en を開発用データセットとし，そのデータセットで最も良い結果を示したことから，構造情報には依存構造を用いた．依存構造解析器には spaCy の RoBERTa ベースのパイプライン [3)] を使用した．

4.2 結果

結果を表 1 に示す．WMT20 では pl-en および ps-en 以外，WMT21 では zh-en 以外の言語対において，ベースラインと少なくとも同等以上の性能を達成していることがわかる．ベースラインである Prism や BLEURT は大規模な多言語対訳コーパスや人手による評価データを学習データとして必要とするのに対して，提案手法は比較的入手が用意な単言語データで事前学習された言語モデルしか必要としない．こうした教師なし手法にもかかわらず高い性能を達成したことは，言語生成タスクの自動評価において構文構造を考慮することの有効性を示している．一方，提案手法と同様に単言語データによる事前学習済み言語モデルに基づく BERTScore と比べると，一部のデータにおいて相関が大きく低下して

1) WMT21 の ha-en に関しては事前学習済みモデルで扱えない長さの翻訳文が含まれていたため今回は使用しなかった．
2) データの取得，評価には以下のライブラリを用いた: https://github.com/google-research/mt-metrics-eval
3) https://spacy.io/models/en#en_core_web_trf

言語対 (翻訳システム数)	WMT20 cs-en (10)	de-en (9)	iu-en (9)	ja-en† (7)	km-en (7)	pl-en (13)	ps-en (6)	ru-en (10)
提案手法	**.816**	.770	.416	**.930**	.973	.200	.873	**.879**
Prism	.720	**.775**	**.616**	.869	.950	.269	**.966**	.839
BLEURT	.725	.770	.320	.820	**.984**	**.371**	.955	.844

言語対 (翻訳システム数)	WMT21 cs-en (8)	de-en (19)	is-en (10)	ja-en (16)	ru-en (10)	zh-en (28)
提案手法	.488	**.382**	**.874**	.798	.559	.531
Prism	**.651**	.349	.846	**.827**	.657	**.643**
BERTScore	.629	.336	.867	.819	**.668**	.589

表 1　WMT20/WMT21 での各言語対における相関係数．**太字**は 3 つの尺度の中で最も良いスコアであることを示す．また，† は開発用データセットであることを示す．

システム	相関係数
依存構造+主辞フィルタリング	0.930
- 依存構造 + 句構造	0.885
- 主辞フィルタリング	0.757

表 2　提案手法を構成する個々の要素のみを変化させた際の相関係数の変化

いる．BERTScore がトークンレベルのアラインメントに基づくのに対して，提案手法はそのアラインメントを構造レベルに拡張したものだとみなすと，部分木という単語よりも大きな系列を対象とするとアラインメントがうまくいかないことがあるのではないかと考える．

4.3　分析

提案手法を構成する個々の要素の貢献を確認するために，構文情報として句構造を用いた際やフィルタリングを行わない際の WMT20 の日英でのデータに対する相関係数を調べた．句構造解析器には Berkeley Neural Parser [15] [4] を使用し，主辞は Collins のルール [16] に従って決定した．また，句構造木に対しては，任意のノードとその子孫を部分木として列挙した．結果を表 2 に示す．

まず，使用する構文情報の違いに着目すると，句構造より依存構造を用いたほうが相関が高い．これは，直接的に主辞をモデリングしている依存構造のほうが誤った主辞を用いることが少なくなり，より正確に主辞によるフィルタリングを行えているからだ考える．また，主辞フィルタリングを行わない場合に着目すると，大きく相関係数が劣化している．このことより，事前に対応候補をフィルタリングしておくことが自動評価の性能に大きく寄与することがわかる．

4）　https://github.com/nikitakit/self-attentive-parser

5　まとめ

本稿では，構文木の間のアラインメントとその埋め込みベクトルの類似度に基づく言語生成技術の自動評価法を提案した．WMT Metrics Shared Task のデータセットを用いたメタ評価の結果，提案手法は大規模な追加データを必要としないにも関わらず既存手法と同程度に人手評価と相関することがわかった．このことより，言語生成タスクの自動評価において構文構造を考慮することが有効であることが示唆された．

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In **Proceedings of the NIPS 2017**, pp. 5998–6008, 2017.

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

[3] 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.

[4] 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.

[5] 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**, pp. 7881–7892, Online, July 2020. Association for Computational Linguistics.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In **Proceedings of the NAACL-2019**, pp. 4171–4186, 2019.

[7] Brian Thompson and Matt Post. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, Online, November 2020. Association for Computational Linguistics.

[8] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. 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. 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] Satoshi Sekine and Michael Collins. Evalb, 1997.

[10] Joakim Nivre, Johan Hall, and Jens Nilsson. Memory-based dependency parsing. In **Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004**, pp. 49–56, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics.

[11] Jason M. Eisner. Three new probabilistic models for dependency parsing: An exploration. In **COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics**, 1996.

[12] Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. Results of the WMT20 metrics shared task. In **Proceedings of the Fifth Conference on Machine Translation**, pp. 688–725, Online, November 2020. Association for Computational Linguistics.

[13] Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, George Foster, Alon Lavie, and Ondřej Bojar. Results of the WMT21 metrics shared task: Evaluating metrics with expert-based human evaluations on TED and news domain. In **Proceedings of the Sixth Conference on Machine Translation**, pp. 733–774, Online, November 2021. Association for Computational Linguistics.

[14] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving pre-training by representing and predicting spans. **Transactions of the Association for Computational Linguistics**, Vol. 8, pp. 64–77, 2020.

[15] Nikita Kitaev and Dan Klein. Constituency parsing with a self-attentive encoder. In **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**, pp. 2676–2686, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[16] Michael Collins. Head-driven statistical models for natural language parsing. **Computational Linguistics**, Vol. 29, No. 4, pp. 589–637, 2003.