補助文自動生成を用いた BERT による日本語アスペクトベース感情分析におけるアスペクトカテゴリ検出の精度向上

張 懿陽 [1]　竹下 昌志 [1]　ラファウ・ジェプカ [2]　荒木 健治 [2]
[1] 北海道大学大学院情報科学院　[2] 北海道大学大学院情報科学研究院
{yiyang.zhang,takeshita.masashi,rzepka,araki}@ist.hokudai.ac.jp

概要

アスペクトベース感情分析（Aspect-Based Sentiment Analysis: ABSA）とは，テキスト内の特定のアスペクトに対する意見の極性を特定する感情分析タスクであり，細かい粒度の感情分析である．対象言語を英語にした研究は盛んだが，日本語に特化した研究はまだ少ない．そこで本研究では，日本語 ABSA タスクのアスペクト抽出（Aspect Extraction: AE）サブタスクの精度の向上を目的とし，補助文を自動的に生成し，対象文と補助文を組み合わせた文ペアで BERT を fine-tuning するという英語に対する既存手法を日本語に適用する．「chABSA」というデータセットで性能評価実験を行った結果，先行研究より F1 値が 7.54 ポイント改善し，提案手法の有効性が確認された．

1　はじめに

感情分析とは，書かれた文章を読み取り，ポジティブな内容か，ネガティブな内容か，あるいはニュートラルの内容かを判定するタスクのことである．感情分析というタスクは主にドキュメントレベル，文レベル，アスペクトレベルという 3 つの粒度がある．それぞれドキュメント単位，文単位，アスペクト単位で感情極性を判定するタスクである．アスペクトとは，ある対象の 1 つの側面である．アスペクトレベルの感情分析はアスペクトベース感情分析（Aspect-Based Sentiment Analysis: ABSA）という．ABSA を例で説明すると，例えば，「この店の寿司は美味しいが，ラーメンはまずい」という文の中に，「この店」という対象は「寿司」と「ラーメン」という 2 つの側面がある．話者が「寿司」に対して「美味しい」と評価しているため，「寿司」というアスペクトに対して「肯定的」な感情をもっており，一方で「ラーメン」に対して「まずい」と評価しているため，「ラーメン」というアスペクトに「否定的」な感情をもっている．ABSA は，ドキュメントレベルの感情分析と文レベルの感情分析より細かい粒度で感情分析を行え，提供できる情報が比較的多いため，多くの場面で実用化が期待される．例えば，ある会社が自社が発売した商品の各アスペクトに対し，消費者がどんな感情をもっているかを分析したい場合，アスペクトレベルの感情分析が必要となる．本稿では，アスペクトレベルの感情分析，すなわち ABSA に注目して研究する．ABSA タスクは主に 2 つのサブタスクから構成されている．1 つ目は，文中に含まれているアスペクトを抽出するタスク（Aspect Extraction: AE）である．2 つ目は，各アスペクトに対する感情極性を特定するタスク（Aspect Sentiment Classification: ASC）である．

英語を対象言語とした研究では，ABSA について多くの研究が行われているが，日本語を対象言語とした研究では，我々が知っている範囲では，ABSA についての研究はまだ 4 件 [1][2][3][4] しかなく，より良い手法を用いればより高い精度を達成することが期待できると考えられる．また，我々 [4] は既に日本語 ABSA タスクの ASC サブタスクについて研究した．そのため，本研究では，日本語 ABSA タスクのアスペクトカテゴリ検出サブタスク，すなわち AE サブタスクに注目し，先行研究より高いアスペクトカテゴリ検出精度を達成することを目的としている．我々は補助文を自動的に生成し，対象文と補助文を組み合わせた文ペアで BERT を fine-tuning するという英語のために提案された手法 [5] を用い，補助文の生成方法を日本語に適用し，「chABSA」データセット [6] という日本語のアスペクト感情分析データセットを用いて実験を行う．その結果，同じデータセットに基づいた先行研究 [2] より高いアスペクトカテゴリ検出精度を達成した．

2 関連研究

英語を対象言語とした ABSA の初期の研究では，Wagner ら [7] の研究と Kiritchenko ら [8] の研究は特徴量エンジニアリングに大きく依存していた．その後，Nguyen と Shirai ら [9]，Wang ら [10]，Tang ら [11][12]，Wang ら [13] はニューラルネットワークベースの手法を用い，より高い精度を達成した．その後，Ma ら [14] は有用な常識知識をディープニューラルネットワークに組み込み，モデルの結果をさらに向上させた．Liu ら [15] は言語構造をよりよく捉えるためにメモリネットワークを最適化し，Liu らのモデルに適用した．

最近では，事前学習済み言語モデルについての研究が進みつつあり，Peters ら [16] が提案した ELMo，Radford ら [17] が提案した OpenAI GPT，Devlin ら [18] が提案した BERT のような事前学習された言語モデルが特徴量エンジニアリングの手間を軽減する効果を示している．特に BERT は「隣接文予測」（next sentence prediction）というタスクで事前学習されたため，QA タスクと NLI タスクのような文と文の関係を理解するタスクにおいて優れた成果を上げている．そこで，Sun ら [5] は BERT が QA タスクと NLI タスクに優れるという特徴を活かした手法を提案した．

日本語を対象言語とした研究では，赤井ら [1] は英語を対象言語とした先行研究で使われていた自己注意機構を使ったニューラルネットワークモデルを日本語に適用し，KNB コーパス [19] のうち評判情報の感情タグが付いた文で実験を行った結果，感情分類の正解率（accuracy）は 85%に達した．三浦ら [2] は BERT を組み込んだアスペクトカテゴリ分類ネットとアスペクトセンチメント分析ネットからなる文に含まれている複数アスペクトのセンチメント分析のための自己注意ニューラルネットワークモデルを提案し，「chABSA」データセットで実験を行った結果，AE サブタスクでの F1 値は最高 70.68%まで達した．

我々 [4] は既に日本語の ABSA タスクの ASC サブタスクについて研究し，ASC サブタスクにおいて三浦ら [2] が達成した精度より高い精度を達成できたが，AE サブタスクについてはまだ研究していない．また，三浦ら [2] の研究の AE サブタスクの部分では主に 2 つの問題が存在している．1 つ目は BERT の fine-tuning 手法である．三浦ら [2] の研究では，各文をそのまま BERT に入力して fine-tuning しているため，BERT は QA タスクと NLI タスクのような文ペア分類問題でより優れた性能を発揮できるという特徴を生かしていない．それに対し，我々は BERT のこの特徴を活かしたより適切な fine-tuning 手法を用いて BERT を fine-tuning すれば，より高い分類精度を達成できると考える．2 つ目はモデルの性能評価方法である．三浦ら [2] の研究ではテストデータで dropout 率を変えながら性能評価を行ったが，我々は検証データで最も良い精度を達成できる dropout 率を探し，その dropout 率でモデルの性能を評価するという性能評価方法の方が適切だと考える．

以上の問題点を解決するために，本研究では，BERT のより適切な fine-tuning 手法を探し，より適切な性能評価方法でモデルの性能を評価し，三浦ら [2] が達成したアスペクトカテゴリ検出精度より高い精度を達成することを目標とする．

3 提案手法

本研究では，Sun ら [5] が提案した補助文を自動的に生成し，対象文と補助文を組み合わせて生成した文ペアで BERT モデルを fine-tuning という手法を用い，日本語に適用し，実験を行う．

以下では BERT，補助文の生成方法，文ペアで fine-tuning した BERT-pair モデルについて詳細に述べる．

3.1 BERT

BERT とは，「Bidirectional Encoder Representations from Transformers」の略で，日本語では，「Transformer による双方向のエンコード表現」である．BERT は 2018 年 10 月に Google 社が発表した自然言語処理モデルであり，自然言語処理の多くのタスクで最高水準を達成した．BERT の特徴は，文脈を考慮した上で単語のエンコード表現を得ることができることと，様々なタスクに適用する際に，学習に必要なデータ量が少ないことである．学習に必要なデータ量が少ない理由は，BERT は既に大規模なデータセットで事前学習が行われているからである．事前学習では 2 つの学習方法が施されている．1 つ目は，「マスク予測」（masked language model）で，文章の一部分を穴抜けにしてモデルに穴の部分の単語を予測させることである．2 つ目は，「隣接文予測」で，2 つの文をモデルに与え，隣接した文であるかどうかを判別させることである．この 2 つの学習方法によ

り，BERT は文脈を考慮した上での各単語の意味，及び文と文の関係をよく捉えられるようになっている．

本研究では，東北大学が提供している事前学習済みの日本語版の BERT-base モデルの whole-word-masking バージョン[1)]モデルと whole-word-masking ではないバージョン[2)]モデルを利用する．この 2 つのモデルの両方とも日本語版のウィキペディアで事前学習されたものである．

3.2 補助文の生成方法

Sun ら [5] の研究では ABSA タスクの ASC サブタスクにおいての手法について具体的に説明されているが，AE サブタスクにおいての手法に関する具体的な説明がなかったため，本研究では Sun ら [5] の手法の考え方に基づき，AE サブタスクにおいての日本語に向けた手法を考案した．具体的には，以下で紹介するそれぞれの補助文生成方法を用い，各文をデータセットのアスペクトカテゴリ数に応じて拡張するという手法である．

補助文の生成方法を**表 1** に示す．以下にそれぞれの補助文生成方法について説明する．

- QA: QA は Question Answering（質問に答える）という意味である．この方法では，質問の形式で補助文を生成し，システムにその質問の答えを出すようにする．具体的な生成方法は，アスペクトカテゴリと「が含まれていますか」という文を結合して補助文を生成することである．
- NLI: NLI は Natural Language Inference（自然言語推論）という意味である．この方法では，アスペクトカテゴリだけで補助文を生成する．

表 1 補助文の生成方法

方法	補助文	出力
QA	（アスペクトカテゴリ）が含まれていますか	Yes・No
NLI	（アスペクトカテゴリ）	Yes・No

本研究で扱うデータセットに 14 種類のアスペクトカテゴリが含まれているため，以上のそれぞれの補助文生成方法を用い，各文をアスペクトカテゴリの種類数に応じて 14 文に拡張する．

1） https://huggingface.co/cl-tohoku/bert-base-japanese-whole-word-masking
2） https://huggingface.co/cl-tohoku/bert-base-japanese

3.3 BERT-pair モデル

対象文と前述の補助文生成方法によって生成した補助文を [SEP] というトークンで結合した文で fine-tuning した BERT-base モデルをそれぞれの補助文生成方法に対応して「BERT-pair-QA」，「BERT-pair-NLI」と名付ける．ただし，whole-word-masking バージョンの BERT-base モデルを使用したものは後ろに「(w)」を付加する．

4 実験

4.1 データセット

本研究では，「chABSA」というデータセットを使用している．「chABSA」データセットは上場企業の有価証券報告書（2016 年度）をベースに作成されたデータセットで，文の中に含まれている各アスペクトに対する「Positive」・「Negative」・「Neutral」という感情極性情報が含まれている．日本語の ABSA データセットとしては他に楽天の「楽天トラベル: レビューアスペクト・センチメントタグ付きコーパス」[3] というコーパスがあるが，これは有料で公開されていないため本研究では用いない．

三浦ら [2] の研究と同様に，本研究ではこのデータセットの中の company，business，product という 3 つのエンティティと sales，profit，amount，price，cost という 5 つのアトリビュートの組み合わせ（例：company#sales，business#amount）からなる 15 種類のアスペクトカテゴリを扱う．ただし，「company#price」というアスペクトカテゴリに属するアスペクトの数は 0 個なので，三浦ら [2] の研究と同様に，アスペクトカテゴリ「company#price」を扱わない．したがって，実際に扱うアスペクトカテゴリは 14 種類で，総文章量は 1,077 文で，合計のアスペクトの数は 2,079 個となる．

提案手法では，扱うアスペクトカテゴリの数に応じて各文を 14 個に拡張するため，実際のデータセットでは合計データ数は 15,078 個（1,077 の 14 倍）となる．最初に，このデータセットを 7:1:2 の割合で，訓練データ，検証データ，テストデータに分けて実験を行い，検証データで最も高い精度を達成できるように dropout 率を調整したら，検証データを訓練データに入れて実験を行う．したがって，最終的にこのデータセットを 4:1 の割合で，訓練データ 12,062 個，検証データ 3,016 個に分けて実験を行う．

4.2 ハイパーパラメータ

本研究では，ハイパーパラメータを Appendix の**表 3** の通りに設定して実験を行う．

ここで，検証データを用いて dropout 率を調整した結果，「BERT-pair-QA」，「BERT-pair-QA(w)」，「BERT-pair-NLI」，「BERT-pair-NLI(w)」のどちらも dropout 率を 0 に設定した時に検証データで最も高い F1 値を得られたため，dropout 率を 0 に設定してテストデータでモデルの性能評価を行う．

4.3 評価指標

本研究では，三浦ら [2] の研究と同様に，適合率と再現率の調和平均指標である F1 値を評価指標とする．

適合率は，モデルが「Yes」と予測したデータの中に正解も「Yes」であるデータの数の割合である．再現率は，正解が「Yes」であるデータの中にモデルが「Yes」と予測したデータの数の割合である．F1 値は，適合率と再現率の調和平均で，トレードオフ関係にある適合率と再現率のバランスを取る評価指標である．F1 値を式で表すと**式（1）**となる．

$$F1\text{値} = 2 \cdot \frac{\text{適合率} \cdot \text{再現率}}{\text{適合率} + \text{再現率}} \tag{1}$$

4.4 実験結果

実験結果を**表 2** に示す．

表 2 実験結果

Model	Dropout 率	適合率	再現率	F1 値
三浦らのモデル [2]	0	0.7530	0.6531	0.6985
	0.2	0.7545	0.6667	0.7068
	0.5	0.7313	0.6737	0.7009
BERT-pair-QA	0	0.8182	0.6990	0.7539
BERT-pair-QA(w)	0	0.8094	0.7282	0.7666
BERT-pair-NLI	0	0.8272	0.7282	0.7745
BERT-pair-NLI(w)	0	**0.8321**	**0.7379**	**0.7822**

補助文生成方法 NLI で生成した補助文で fine-tuning した「BERT-pair-NLI(w)」は既存手法のモデルが達成した全ての結果を上回った上で，提案した他の全てのモデルを超え，最も高い適合率，再現率，F1 値が得られた．

また，「BERT-pair-NLI(w)」は「BERT-pair-NLI」より，「BERT-pair-QA(w)」は「BERT-pair-QA」より高い F1 値が得られた．「BERT-pair-NLI(w)」は「BERT-pair-QA(w)」より，「BERT-pair-NLI」は「BERT-pair-QA」より高い F1 値が得られた．

5 考察

実験結果から，提案手法が既存手法より良い精度を達成したことが確認された．その原因は主に 2 つあると考えられる．1 つ目の原因は，本研究の提案手法では，各文をデータセットに含まれているアスペクトカテゴリの種類数（本研究では 14 種類）に応じて拡張しているため，データセットが拡張され，訓練に用いられる文の数が増加し，モデルの学習データが増加したからである．2 つ目の原因は，BERT は「隣接文予測」タスクで事前学習されており，文と文の関係をよく捉えているため，QA タスクと NLI タスクのような文ペア分類問題でより優れた性能を発揮できるためだと考えられる．

また，三浦ら [2] の研究と異なり，dropout 率を 0.2 ではなく，0 に設定した時に最も良い精度を達成できた．この理由は，データセットが拡張されているため，dropout 率を 0 以外の値に設定することにより dropout を実装した場合，モデルの学習が不足してしまうためだと考えられる．

さらに，「BERT-pair-NLI(w)」は「BERT-pair-NLI」より，「BERT-pair-QA(w)」は「BERT-pair-QA」より高い F1 値が得られた理由は，BERT-base モデルの事前学習で「whole-word-masking」を使用した方が BERT-base モデルがより言葉の意味を学習できるためだと考えられる．また，「BERT-pair-NLI(w)」は「BERT-pair-QA(w)」より，「BERT-pair-NLI」は「BERT-pair-QA」より高い F1 値が得られた理由は，BERT は QA タスクより，NLI タスクを解く方がより優れた性能を発揮できるためだと考えられる．

6 まとめ

本研究では，日本語 ABSA タスクのアスペクトカテゴリ検出サブタスク，すなわち AE サブタスクにおいて，補助文を自動的に生成し，対象文と補助文を組み合わせた文ペアで BERT を fine-tuning するという英語のために提案された手法を日本語に適用し，「chABSA」データセットで実験を行った結果，同じデータセットで実験を行った先行研究より高い精度を達成できた．今後，日本語 ABSA タスクの AE サブタスクにおいてより高い精度を達成するために，より良い手法を探し，精度をさらに向上させることに取り込む予定である．

参考文献

[1] 赤井 龍一, 渥美 雅保. 2019. 自己注意機構を利用したアスペクトベースの感情分析の日本語文への適用, 2020 年度人工知能学会全国大会（第 33 回）.

[2] 三浦 義栄, 赤井 龍一, 渥美 雅保. 2020. 文中の複数アスペクトのセンチメント分析のための自己注意ニューラルネットワーク, 2020 年度人工知能学会全国大会（第 34 回）.

[3] Yuki Nakayama, Koji Murakami, Gautam Kumar, Sudha Bhingardive, and Ikuko Hardaway. 2022. A Large-Scale Japanese Dataset for Aspect-based Sentiment Analysis. In Proceedings of the Thirteenth Language Resources and Evaluation Conference, pages 7014–7021, Marseille, France. European Language Resources Association.

[4] 張懿陽, ラファウ・ジェプカ, 荒木 健治. 2022. BERT モデルと補助文自動生成に基づいた日本語アスペクトベース感情分析の精度向上. ISSN 1346-3551, 人工知能学会第 2 種研究会 ことば工学研究会資料, SIG-LSE-C302-3.

[5] Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing BERT for aspect-based sentiment analysis via constructing auxiliary sentence. Proceedings of NAACL-HLT 2019, pages 380–385.

[6] Takahiro Kubo, Hiroki Nakayama. chABSA: Aspect Based Sentiment Analysis dataset in Japanese. URL https://github.com/chakki-works/chABSA-dataset

[7] Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. DCU: Aspect-based polarity classification for SemEval task 4. In Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014), pages 223–229.

[8] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 437-442.

[9] Thien Hai Nguyen and Kiyoaki Shirai. 2015. PhraseRNN: Phrase recursive neural network for aspect-based sentiment analysis. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 2509–2514.

[10] Yequan Wang, Minlie Huang, Li Zhao, et al. 2016. Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the 2016 conference on Empirical Methods in Natural Language Processing, pages 606–615.

[11] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Effective LSTMs for target-dependent sentiment classification. arXiv preprint arXiv:1512.01100.

[12] Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. arXiv preprint arXiv:1605.08900.

[13] Bo Wang, Maria Liakata, Arkaitz Zubiaga, and Rob Procter. 2017. Tdparse: Multi-target-specific sentiment recognition on twitter. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, volume 1, pages 483–493.

[14] Yukun Ma, Haiyun Peng, and Erik Cambria. 2018. Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive LSTM. In Proceedings of AAAI.

[15] Fei Liu, Trevor Cohn, and Timothy Baldwin. 2018. Recurrent entity networks with delayed memory update for targeted aspect-based sentiment analysis. arXiv preprint arXiv:1804.11019.

[16] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. arXiv preprint arXiv:1802.05365.

[17] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[19] 橋本力, 黒橋禎夫, 河原大輔, 新里圭司, 永田昌明. 2011. 構文・照応・評判情報つきブログコーパスの構築. 自然言語処理 Volume 18, Number 2, pp.175-201.

Appendix

表 3 ハイパーパラメータ

ハイパーパラメータ	値
学習率	2e-5
エポック数	3
バッチサイズ	16
dropout 率	0
pandas.DataFrame.sample の random_state	4,040
Pytorch のランダムシード	2,020