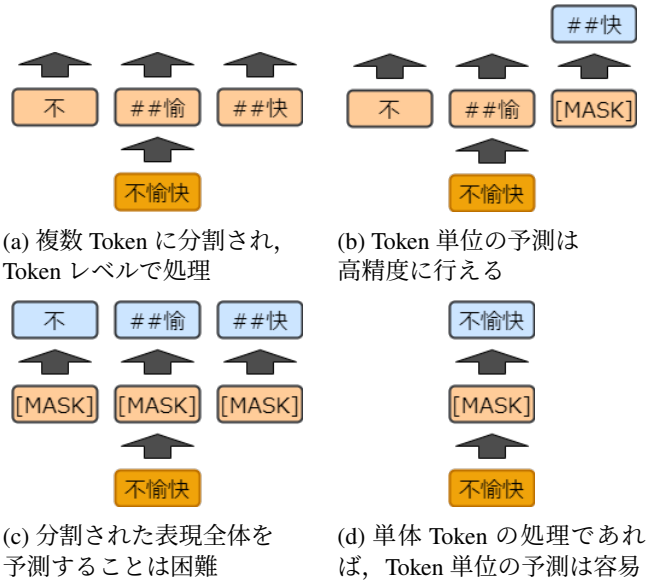
BERT の領域適応における複合語の語彙拡張

田中裕隆 [1]　新納浩幸 [2]
[1] 茨城大学大学院理工学研究科情報工学専攻　[2] 茨城大学大学院理工学研究科情報科学領域
20nm716y@vc.ibaraki.ac.jp
hiroyuki.shinnou.0828@vc.ibaraki.ac.jp

概要

BERT のような事前学習モデルは，大規模コーパスで事前学習し応用タスクデータで finetuning することで，様々な自然言語処理タスクを高精度に処理できる．しかし，Token レベルで処理する BERT が，BPE によって分割される未知語や複数の語で構成される表現を学習することは困難である．本研究では，そのような複数 Token で表現される語を対象にした BERT の語彙拡張における，追加語彙の埋め込み表現の構築について考える．提案手法では，追加の事前学習による領域適応を前提に，類義語による近似的な埋め込み表現で語彙追加を行った．各語彙拡張手法について実験を行い，Masked LM による追加語彙の予測精度で評価した．



(a) 複数 Token に分割され，Token レベルで処理
(b) Token 単位の予測は高精度に行える
(c) 分割された表現全体を予測することは困難
(d) 単体 Token の処理であれば，Token 単位の予測は容易

図 1: 語彙追加の対象となる語の例

1　はじめに

事前学習モデルは，多くの自然言語処理システムの性能を向上させている [1][2]．事前学習モデルの一つである BERT[3] は，Transformer[4] モデルの Multi-head Attention を多層に積み重ねたモデルであり，入力単語列に対する文脈を考慮した単語埋め込み表現を出力することができる．

事前学習モデルは，大規模コーパスで事前学習したモデルを応用タスクデータに適用することで，各応用タスクを高精度に処理する．その性質上，応用タスクのドメインが事前学習したコーパスのドメインと大きく異なると，応用タスクを解く精度が高くならない領域適応の問題が存在する．Gururangan ら [5] は，応用タスクドメインのコーパスで追加の事前学習を行うことで応用タスクを高精度に解くことができる手法を提案している．

事前学習モデルの扱う語彙は事前学習したコーパスに依存して決定されるため，応用タスクデータに現れる語彙に適応するためには，語彙を拡張する必要がある．事前学習モデルの領域適応における語彙の拡張手法については，Yao らの Adapt-and-Distill[6] や，Hong らの AVocaDo[7] といった研究がある．

標準的な BERT における日本語の語彙は，形態素解析と Byte-Pair Encoding によって分割された Token 単位で処理される．未知語が既知の語に分割されて処理されることで，少ない語彙数で多くの単語を処理できる．しかし BERT のような事前学習モデルは，Token レベルの表現を学習するために，複数の語で構成される複合語や固有表現，句の表現を学習することは困難である．(図 1 参照)

複数の語で構成されるエンティティの埋め込み表現を学習するための事前学習モデルとして，LUKE[8] がある．LUKE は，通常の Token の埋め込み表現とは別にエンティティのための埋め込み表現を用意し，通常の Token とエンティティ間の関係を Entity-aware Self-attention によってモデル化・学習する．ただし LUKE は大規模コーパスによる高コストな事前学習を要する．

本研究では，事前学習済みモデルの BERT に語彙

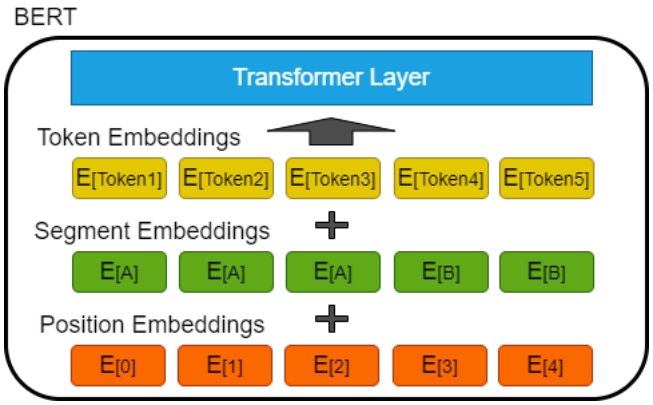


図 2: BERT の入力ベクトル

追加を行う手法を提案する．語彙の追加では，追加語彙の埋め込み表現を得ることができればよい．応用タスクデータによって追加の事前学習を行い，領域適応させることを前提とすれば，その前の追加語彙の埋め込み表現は近似的な表現であっても学習することができると考えられる．

2 BERT

2.1 BERT の Token Embeddings

入力文の Tokenize 処理によって得られた Token 列は，それぞれ BERT のモデルへの入力のための埋め込み表現に変換される．BERT の入力ベクトルは単語を表す Token Embeddings と文を識別する Segment Embeddings，位置情報を埋め込む Position Embeddings の 3 種類ある．(図 2 参照)

Token Embeddings は，各々の単語 Token に対応した埋め込み表現である．BERT の最終的な出力は，文脈に応じて変化する文脈埋め込み表現である．ただし，BERT への入力ベクトルとなる最初の埋め込み表現は，Token 毎に固有の埋め込み表現となる．

2.2 Masked Language Model (MLM)

BRET の事前学習の手法の一つである MLM は，文の中のいくつかの単語を MASK に置き換えて，そこにあるべき単語を文脈から予測する問題である．標準的な手法では，入力トークンの内 15%に対して，以下のように置き換える．

- 80%は，特殊トークンである [MASK] に置き換える．
- 10%は，ランダムな別のトークンに置き換える．
- 残り 10%は，そのまま残す．

3 BERT への語彙追加手法

学習済み BERT モデルに語彙を追加する場合，BERT の Token Embeddings で用いられる埋め込み表現に追加語彙に対応する埋め込み表現を追加すればよい．したがって語彙追加手法は，BERT の埋め込み表現空間における追加語彙の埋め込み表現をどのように得るかが課題となる．

3.1 静的な単語埋め込み表現を利用した手法

追加語彙の埋め込み表現を得る方法の一つは，word2vec や fastText[9] [10] のような分散表現モデルによる静的な単語埋め込み表現を利用する方法である．

具体的には，まず追加語彙を学習した分散表現モデルを用意する．次に，BERT と分散表現モデルで共通して学習している語彙集合を基に，分散表現モデルの埋め込み表現を BERT の埋め込み表現に写像するための変換行列を学習する．学習手法としては，変換のソース単語ベクトルとターゲット単語ベクトルの平均二乗誤差を小さくするように確率的勾配降下法で学習を行う Mikolov ら [11] の手法や，ベクトルの補正や正規化によってより効率的に学習を行う平子ら [12] の手法がある．

3.2 subword の平均ベクトル

事前学習済み BERT モデルのみを利用して追加語彙の埋め込み表現を得る手法としては，subword の平均ベクトルを用いる手法がある．この手法は Yao らの Adapt-and-Distill[6] においても用いられている．

追加の対象となる語彙は，事前学習済みモデルにおいて複数 Token で処理される．例えば「不愉快」という語であれば「不」「##愉」「##快」の 3Token である．事前学習済み BERT モデルからこれら 3Token の埋め込み表現の平均ベクトルを求め，それを追加語彙の埋め込み表現として追加する．

3.3 BERTRAM

BERTRAM[13] は，学習済み BERT モデルの出力を利用して追加語彙の埋め込み表現を得る手法である．BERTRAM の研究では，BERT の事前学習コーパスにおける出現率が低頻度な語の追加を対象にしている．

BERTRAM では，追加語彙の埋め込み表現を Form-Context Model で学習する．Form Model では文

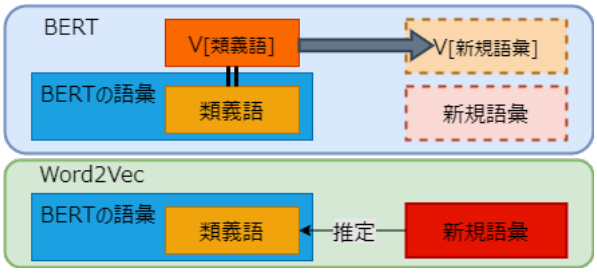


図 3: 提案手法の概念図

字ベースの学習を行い，Context Model では文脈ベースの学習をする．Context Model の学習は標準的な BERT で文脈埋め込み表現を獲得するのと同様である．これに加えて文字ベースの学習を行うことで，未知語に対して埋め込む情報量を確保している．

BERTRAM では，追加語彙「○○」を「<BERTRAM: ○○>」と表記した特殊トークンとして追加する．BERTRAM によって追加された埋め込み表現を利用する場合，データ中の追加語彙を「<BERTRAM: ○○>」の表記に置き換えて処理する．

4 提案手法

本提案手法 (図 3 参照) では，新規語彙の類義語の埋め込み表現を基に，MLM で finetuning された埋め込み表現を用いる．

具体的な手順は次のようになる．

1. 追加語彙を学習済みの Word2Vec のような分散表現モデルを用意する．
2. 追加語彙の類義語を，分散表現モデルで推定する．
3. 推定した類義語の内，BERT の語彙に含まれる語の BERT の埋め込み表現を，追加語彙の埋め込み表現として BERT に追加する．
4. 追加語彙を含むコーパスで語彙追加後の BERT を Masked Language Model で追加事前学習する．

5 実験

5.1 実験設定

本実験では，事前学習済みの日本語 BERT モデルに対する語彙拡張を行う．(図 4 参照) 提案手法の他に，静的な単語埋め込み表現を利用する手法と，subword の平均ベクトルの手法と，BERTRAM の手法を試みる．

BERT の事前学習済みモデルは，東北大学乾研究室が公開しているモデルを用いる．これは，

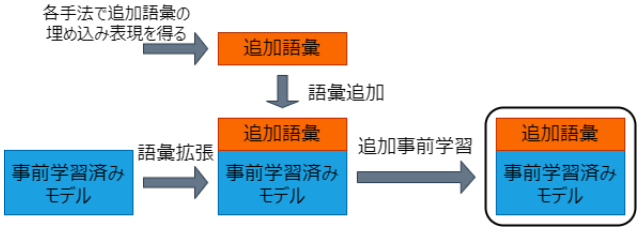


図 4: 語彙拡張の流れ

Hugging Face 社の Transformers ライブラリから，モデル名'cl-tohoku/bert-base-japanese' で利用できるモデルである．事前学習コーパスには日本語 Wikipedia が用いられている．

静的な単語埋め込み表現には，以下のサイトで公開されている日本語 Wikipedia エンティティベクトル [14] を用いる．[1)]

提案手法で，追加語彙の類義語を求める時の類似度の計算は cos 類似度で求める．

形態素解析は Mecab で処理する．

各手法で語彙追加を行った BERT モデルは，いずれも応用タスクデータの学習データについて BERT の事前学習で標準的に行われる Masked Language Model によって追加の事前学習を行う．

本実験で学習・評価を行う応用タスクデータには，以下のサイトで公開されている Amazon レビューコーパスを用いる．[2)]

追加する語彙は，表 1 に示す 20 語とする．これらの語は，事前学習済み BERT では複数 Token で処理される語である．また，Wikipedia コーパスでの出現頻度が低く，Amazon レビューコーパスに複数回出現する語である．学習・テストデータは，これらの追加語彙を含む文のみであり，それぞれ 118 文，205 文である．

評価は，新規語彙のみを MASK トークンに置き換えて，新規語彙を推定できる精度を測る．

5.2 実験結果

提案手法において求めた追加語彙の類義語とその類似度は表 1 に示す．

各手法の実験結果の正解率は表 2 に示す．表 2 では，静的な単語埋め込み表現を利用した手法の結果を分散表現として示している．提案手法と分散表現の手法および平均ベクトルの手法は，追加語彙を Masked LM で同程度に予測することができている．

1) http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/
2) https://webis.de/data/webis-cls-10.html

表 1: 追加する語彙と
提案手法で利用した類義語

追加語彙	類義語	cos 類似度
不愉快	不快	0.7330
羅列	単語	0.5981
読み応え	読者	0.6012
読後感	文体	0.5903
駄作	酷評	0.6094
鵜呑み	間違っ	0.6639
苦笑	困惑	0.7338
読み手	難解	0.5126
蛇足	筆	0.4621
名著	叢書	0.6759
好き嫌い	嫌い	0.7076
後味	味わい	0.6447
いい加減	真面目	0.7478
話し方	話す	0.6276
長文	文章	0.6828
醜悪	邪悪	0.6625
脱帽	絶賛	0.5795
誤字	誤り	0.6326
金儲け	金持ち	0.6791
敷居	棚	0.5344

表 2: 全体と追加語彙毎の予測精度（正解率）

追加語彙	提案手法	分散表現	平均ベクトル	BERTRAM
不愉快	0.7429	0.7429	0.6571	0.4000
羅列	0.3692	0.4615	0.4462	0.0154
読み応え	0.6875	0.6625	0.6375	0.1250
読後感	0.1059	0.0941	0.2117	0.0000
駄作	0.1500	0.0000	0.0750	0.0000
鵜呑み	0.8400	0.9400	1.0000	0.9200
苦笑	0.6889	0.5556	0.5556	0.1111
読み手	0.0400	0.1400	0.1800	0.0000
蛇足	0.0333	0.1667	0.1000	0.0000
名著	0.0250	0.0875	0.1000	0.0000
好き嫌い	0.5000	0.4800	0.4200	0.0000
後味	0.6000	0.8364	0.7818	0.8545
いい加減	0.2769	0.2615	0.2923	0.0000
話し方	0.0000	0.0000	0.0000	0.0200
長文	0.0250	0.0000	0.0000	0.0000
醜悪	0.4400	0.0200	0.2400	0.1600
脱帽	0.5333	0.6222	0.4222	0.0000
誤字	0.4667	0.6000	0.5333	0.0000
金儲け	0.6286	0.4857	0.4286	0.0000
敷居	0.5667	0.5667	0.6333	0.0000
全体	0.3703	0.3723	0.3792	0.1320

しかし，BERTRAM の手法の実験では，追加語彙の予測精度は著しく低くなった．

6 考察

Masked Language Model による追加語彙の予測精度について，表 2 に示す各語彙毎の結果から分析する．

Tokenize 処理において，唯一 BERTRAM の手法では，追加語彙を特殊トークンとして扱っているため，その影響度を調査する必要があると考えている．語彙別に見ると，BERTRAM が高精度に予測できる語彙は，その他の手法においても概ね高精度に予測することができている．

全体の精度としては平均ベクトルの手法が僅かに高いものの，語彙別の精度を見るとそれぞれ特徴がある．「駄作」「醜悪」などの語彙については，提案手法では比較的高い精度で予測できている．提案手法で用いたこれらの類義語はそれぞれ「酷評」「邪悪」となっており，cos 類似度は 0.60 を超える．一方で，「読み手」「蛇足」などの語彙については，提案手法の予測精度は比較的低い結果となった．提案手法で用いたこれらの類義語はそれぞれ「難解」「筆」となっており，cos 類似度は 0.5126 と 0.4621 で，比較的低い類似度である．このような結果から，類似度の高い類義語を利用できる語彙は，他の手法に比べて提案手法が有効であると考えられる．

7 おわりに

本研究では，複数 Token で表現される語を対象に，応用タスクデータに対して追加の事前学習を行うことを前提として，事前学習済み BERT モデルへ語彙追加を行う手法を提案した．提案手法で用いた類義語と追加語彙との類似度と予測精度の関係について分析し，類似度が比較的高い類義語を用いた場合には他の手法と同等以上の有効性があることを示した．本実験で対象とした語彙は，BPE によって複数 Token に分割される語であった．今後の課題としては，複数の語で構成される複合語や固有表現，句といった表現に対しても有効な埋め込み表現を得ることのできる語彙追加手法への拡張が考えられる．

謝辞

本研究は JSPS 科研費 JP19K12093 および 2021 年度国立情報学研究所公募型共同研究（2021-FC05）の助成を受けています.

参考文献

[1] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In **NAACL-2018**, pp. 2227–2237, 2018.

[2] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. **Technical report, OpenAI.**, 2018.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In **NAACL-2019**, pp. 4171 –4186, 2019.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In **Advances in neural information processing systems**, pp. 5998–6008, 2017.

[5] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 8342–8360, Online, July 2020. Association for Computational Linguistics.

[6] Yunzhi Yao, Shaohan Huang, Wenhui Wang, Li Dong, and Furu Wei. Adapt-and-distill: Developing small, fast and effective pretrained language models for domains. In **Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021**, pp. 460–470, Online, August 2021. Association for Computational Linguistics.

[7] Jimin Hong, TaeHee Kim, Hyesu Lim, and Jaegul Choo. AVocaDo: Strategy for adapting vocabulary to downstream domain. In **Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing**, pp. 4692–4700, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[8] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. LUKE: Deep contextualized entity representations with entity-aware self-attention. In **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**, pp. 6442–6454, Online, November 2020. Association for Computational Linguistics.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, Vol. 5, pp. 135–146, 2017.

[10] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. **arXiv preprint arXiv:1607.01759**, 2016.

[11] Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. **CoRR**, Vol. abs/1309.4168, , 2013.

[12] 平子潤, 笹野遼平, 武田浩一. 静的な単語埋め込みによるカタカナ語を対象とした bert の語彙拡張. 言語処理学会第 27 回年次大会 (NLP2021), 2021.

[13] Timo Schick and Hinrich Schütze. BERTRAM: Improved word embeddings have big impact on contextualized model performance. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**, pp. 3996–4007, Online, July 2020. Association for Computational Linguistics.

[14] 鈴木正敏, 松田耕史, 関根聡, 岡崎直観, 乾健太郎. Wikipedia 記事に対する拡張固有表現ラベルの多重付与. 言語処理学会第 22 回年次大会 (NLP2016), 2016.