単語正規化による表記ゆれに頑健な BERT モデルの構築

勝田 哲弘 [1]　林 政義 [1]　山村 崇 [1]　Tolmachev Arseny[1]
高岡 一馬 [1] 内田 佳孝 [1]　浅原 正幸 [2]
[1] 株式会社ワークスアプリケーションズ・エンタープライズ
[2] 人間文化研究機構 国立国語研究所
katsuta_a@worksap.co.jp

概要

近年、様々な自然言語処理タスクで大規模な事前学習済みの言語モデルである BERT による飛躍的な精度向上が示されている。しかし、日本語テキストには同一の語が複数の表記で出現することが多く、英語テキストでの学習と比べその影響は自明ではない。そこで我々は形態素解析器による単語正規化を行い、表記ゆれを考慮した BERT モデルの構築を行った。また下流タスクにおいて、提案手法が表記ゆれに対して出力の一貫性を保ちつつ、正規化をしない従来の BERT モデルと同程度の精度であることを示した。

1　はじめに

大規模な事前学習済みの言語モデル BERT [1] を利用することで自然言語処理の様々なタスクで飛躍的な精度向上が示されている。その一方、Ribeiro ら [2] は BERT モデルが同義語による摂動などで精度を低下させることを実験の中で示している。

日本語テキストには同一の語が複数の表記で出現することが多い。「引っ越し」「引越し」のように部分文字列を共有するものだけでなく、「酢橘」「すだち」のように全く異なる文字列に変化するものもあり、同義語と同様にモデルへの影響が予想される。BCCWJ を対象とした調査では WEB・書籍上で約 1 割の語に表記ゆれが存在する [3]。

そこで我々はコーパスの前処理として形態素解析器 Sudachi [4] の単語正規化を利用し、表記ゆれに頑健な BERT モデルの構築を行った。また、下流タスクで表記ゆれによる出力の一貫性を担保しつつ正規化をしない BERT モデルと同程度の精度であることを示した。

現在、chiTra[1)] として Sudachi を利用した単語正規化の機能をもつ Hugging Face[2)] 互換のトークナイザーを公開している。今後は本研究で提案する表記ゆれに頑健な事前学習済み BERT モデルを公開する予定である。

2　関連研究

BERT [1] は平文から 2 つの事前学習タスク（Masked Language Model, Next Sentence Prediction）を学習し、様々な応用タスクに適応して精度向上に貢献している。事前学習モデルの評価では GLUE [5] や SQuAD2.0 [6] が用いられることが多く、複数の自然言語処理のタスクで評価される。Ribeiro ら [2] はソフトウェアエンジニアリングの動作テストから着想を得て、モデルを評価するためのチェックリストを提案している。その中でも Invariance では入力の変化に対して出力がどれだけ不変であるかを評価している。入力に対して同義語による置換を行うと彼らが実験で使用した BERT モデルでは 13.1%に予測結果のゆれが生じることが示されている。この結果から日本語における表記ゆれも予測結果に影響を与えると推測される。

3　Sudachi による単語正規化

Sudachi 形態素辞書では表記ゆれに対応するため登録語ごとに正規化表記が付与されている。この正規化表記を用いて単語正規化を行う。例として以下のようなものが正規化の対象となる。

- 送り違い： 打込む → 打ち込む
- 字種： かつ丼 → カツ丼
- 異体字： 附属 → 付属
- 誤用： シュミレーション → シミュレーション
- 縮約： ちゃあ → ては
- 翻字： test → テスト

1） https://github.com/WorksApplications/SudachiTra

2） https://huggingface.co/

表 1　各正規化による出力例

surface	引越	し	て	から	すだち	を	とどけ	ます	。
normalized_and_surface	引っ越し	し	て	から	酢橘	を	とどけ	ます	。
normalized_conjugation	引っ越し	し	て	から	酢橘	を	届け	ます	。
normalized	引っ越し	為る	て	から	酢橘	を	届ける	ます	。

表 2　実験で使用した日本語 Wiki-40B の学習データの統計値

クリーニング処理	総文書数	総文数	平均文書長	平均文長
なし	1,869,067	12,917,970	6.911	52.626
あり	927,692	10,088,563	10.875	53.493

- 活用形：泳い（だ）→ 泳ぐ（た）
- 可能動詞：泳げる → 泳ぐ

ただし、「あく」から「空く、開く、明く」など曖昧性解消を必要とする正規化は行われない。

この正規化には活用形や可能動詞の正規化も含まれており、単純に利用するとこれらの情報が失われてしまう。このような強い正規化は事前学習や下流タスクに悪影響を与える可能性がある。そこで正規化の強さの影響をみるために、下記 4 つのルール毎に学習を行い、正規化の適用範囲を調整する。

surface
　正規化をせず入力をそのまま出力する

normalized_and_surface
　入力単語の内、活用語は見出し語を、それ以外は正規化表記を出力する

normalized_conjugation
　入力単語を全て正規化表記で出力し、活用形のある「動詞、助動詞、形容詞」は活用語尾を入力の活用形に合わせる

normalized
　入力単語を全て辞書の正規化表記で出力する

それぞれのルールを適用した出力例を表 1 に示す。normalized_conjugation では「とどけ」を「届け」に正規化を行いつつも活用情報を残しているが、normalized の出力では動詞の活用情報は表層から消えていることが見て取れる。

4　BERT モデルの事前学習

4.1　コーパスのクリーニング

事前学習用コーパスとして、比較のため既に公開されている日本語 BERT の事前学習モデルでも用いられている Wikipedia を使用した。ただし、Wikipedia 中のページには十分なテキストが含まれていない不要な非コンテンツ（曖昧さ回避・リダイレクト・削除済み・非エンティティ）も存在している。そこで、これらの非コンテンツのページが除去された日本語 Wiki-40B [7] を用いた。実験では、Wiki-40B を段落ごとに分割して 1 文書として扱い、文書内のテキストに対して文分割を適用した。文分割器には、Bunkai [3] [8] を用いた。

Raffel ら [9] は、下流タスクに有用ではないと考えられる定型文や不快な言葉（NG ワード）、ソースコードなどをテキストから除外することで、下流タスクにおける事前学習モデルの精度が向上することを報告している。本実験ではこれに倣ってクリーニング処理を適用した。詳細は付録の表 6 に示す。NG ワードとして、C4 [9] で用いられているリスト[4] と稲葉 [10] が用いたリストから、著者 2 名によって NG ワードとして妥当だと考えられる 210 語を選定し利用した。

表 2 に、本実験で使用した事前学習用データセットの統計値を示す。Train データに対してクリーニング処理を適用すると、総文書数が元の半分になっている一方で、総文数は約 21%しか減少していない。これは、1、2 文からなる文書が元のデータセット中に多く出現していたためと考えられる。

4.2　事前学習

WordPiece による語彙の学習及び BERT の事前学習の際に 3 節で述べた 4 つの正規化手法を適用し、それぞれのモデルごとに WordPiece による 32,000 語の選定とその語彙でサブワード化されたコーパスでの事前学習を行った。形態素解析には SudachiPy[5]

3）https://github.com/megagonlabs/bunkai
4）https://github.com/LDNOOBW/List-of-Dirty-Naughty-Obscene-and-Otherwise-Bad-Words/blob/master/ja
5）https://github.com/WorksApplications/sudachi.rs/tree/develop/python

表 3　BERT の事前学習に使用したハイパーパラメータ

パラメータ	phase1	phase2
batch size	176	22
learning rate	7.5e-4	5e-4
warmup steps	2000	200
num accumulation	64	192
train steps	11374	
precision	tf32	
use xla	true	
num gpus	4	

を使用して C 単位で解析している。

BERT の事前学習は NVIDIA の実装[6]を利用し、NVIDIA RTX A6000 を 4 枚使用して学習した。学習は 2 段階で行われ phase1 では max_length を 128 として学習し、phase2 では max_length を 512 に拡張して再学習した。使用したハイパーパラメータを表 3 に示す。

5　実験設定

5.1　評価対象

本実験では正規化による下流タスクへの影響及び入力の表記ゆれに対する頑健性の評価を行うために以下の 2 つについて調査する。

下流タスクへの影響

正規化を行った影響を調査するため、4.2 節で構築した BERT モデルごとに下流タスクで fine-tuning した際の精度を評価する。下流タスクについては 5.3 節で後述する。

fine-tuning およびハイパーパラメータの探索範囲については BERT 元論文 [1] に従った。

表記ゆれに対する頑健性の評価

提案手法において入力に表記ゆれが発生しても出力が変化しないケースの量を評価する。下流タスクのテストデータの文章を正規化することで表記ゆれを疑似的に生成し、表記ゆれの有無でモデルの出力がどの程度変化するかを計測する。

自然な表記ゆれを生成するため、正規化手法には提案手法中で正規化の度合いが最も弱い normalized_and_surface を用いる。

5.2　ベースライン

ベースラインとして、公開されている日本語 BERT モデルから、東北大 BERT[7]（cl-tohoku/bert-base-japanese-whole-word-masking）、京大 BERT[11]（BASE WWM 版）、NICT BERT（BASE BPE あり）の 3 つを比較に用いた。それぞれの詳細を付録の表 7 に示す。

5.3　評価タスク

日本語の言語モデルの評価においては英語での GLUE や SQuAD のような基準となるタスク群がないため、類似する日本語のデータセットとして以下の 3 つを用いる。

Multilingual Amazon Reviews Corpus (Amazon) [12]

文章分類タスクとしてアマゾンが公開しているデータセットを用いる。ここでは日本語のサブセットのみを対象とし、レビュー本文から評価の星数（1-5）を推定する。評価指標には精度 (Acc) および平均二乗誤差 (MSE) を用いる。

京都大学常識推論データセット (KUCI) [13]

常識推論タスクとして京都大学が公開しているデータセットを用いる。これは文脈文に続く内容として最も適切な文を 4 つの選択肢から選ぶタスクである。本データは Juman++による形態素解析が行われているため、前処理として生文に戻す。評価指標には精度 (Acc) を用いる。

解答可能性付き読解 (RCQA) [14]

読解タスクとして東北大が公開しているデータセットを用いる。これは質問・解答・文章の組に対し、文章から解答を推定できるかのスコアが付与されたデータである。ここでは NICT BERT[8]の評価実験設定にならい、解答可能性スコアが 2 未満のものを解答なしとして SQuAD2.0 形式のデータへと変換し、データの年代によって 2009 年までを訓練、2009 年代を開発、2010 年以降を評価データと分割して用いる。評価指標には出力と正解単語列の完全一致率 (EM) および F1 スコア (F1) を用いる。

6　結果と考察

表 4 に下流タスクでの精度を示す。提案モデルは正規化の有無に関わらず既存モデルと同程度の精度を達成している。全体的に NICT-BERT が良い性能

6）https://github.com/NVIDIA/DeepLearningExamples/tree/master/TensorFlow2/LanguageModeling/BERT

7）https://github.com/cl-tohoku/bert-japanese

8）https://alaginrc.nict.go.jp/nict-bert/index.html

表 4　各タスクでの性能

	Amazon (Acc↑)	Amazon (MSE↓)	KUCI (Acc↑)	RCQA (EM↑)	RCQA (F1↑)
東北大 BERT	**0.595**	0.670	0.762	**0.763**	0.763
京都大 BERT	0.593	0.694	0.737	0.713	0.738
NICT BERT	0.589	**0.634**	**0.785**	0.759	**0.773**
surface	0.592	0.674	0.750	0.736	0.737
normalized_and_surface	**0.594**	0.687	**0.752**	0.739	0.739
normalized_conjugation	0.580	**0.673**	0.746	0.742	0.743
normalized	0.588	0.684	0.750	**0.748**	**0.748**

ベースラインモデルと今回学習したモデルのそれぞれで最も性能が良いものを太字で示している。

表 5　入力の normalized_and_surface 正規化による出力変化 (%)

	Amazon	KUCI	RCQA
サンプル数	5000	10291	6178
東北大 BERT	8.98 (3.78, 3.30)	5.52 (2.62, 1.95)	10.2 (5.76, 2.02)
京都大 BERT	11.1 (4.60, 4.06)	6.46 (3.05, 2.25)	13.4 (6.04, 2.56)
NICT BERT	7.74 (3.18, 2.82)	5.08 (2.43, 1.94)	11.1 (5.54, 1.93)
surface	8.70 (4.10, 3.08)	4.89 (2.35, 1.91)	10.4 (5.84, 1.96)

数値は、出力が変化したサンプル（うち元々正答であったもの、 正規化後に正答となったもの）。いずれも総サンプル数に対する割合。なお出力変化の総計には誤答から誤答への変化も含まれる。

となっているが、これは NICT-BERT の事前学習時の訓練量が多く、より学習が進んでいることによるものと思われる。

提案モデルに注目すると、各タスクについて正規化ありのモデルのいずれかが surface モデルと同等以上の性能を達成しており、正規化を行うことによる精度への悪影響は起こっていないことが確認された。しかしどの正規化手法の性能が高くなるかはタスクによって異なっており、汎用的に精度を向上させる手法の特定にまでは至らなかった。

表 5 に正規化による疑似表記ゆれを施した入力を与えた時の各モデルの出力変化を示す。surface モデルではサンプルのおよそ 5-10%で出力が変化しており、4-8%ではタスクにおける正誤も変化している。各タスクで良い性能であった既存モデルでも同様の傾向が確認でき、通常の学習では表記ゆれには対応しきれないことが推測できる。

提案手法ではモデルに含まれる正規化機能により、これらの出力変化に対する頑健性を得ることができている。

7　まとめ・今後の予定

今回の実験では正規化手法ごとの性能はタスクによって上下した。文書分類、常識推論、読解のタスクについて評価したが、品詞推定などより文法を考慮する必要のあるタスクなどでも正規化の影響を調査したい。また、タスクに依らず良い性能となる正規化手法は存在するのか、特定のタスクに合わせた手法の選択は可能かといったことや同義語、フィラーの処理など正規化の適用範囲を広げることによる影響についても調査したい。

今回利用した表記ゆれの生成方法は、提案手法にやや有利なものになっている。手法やデータセットを工夫することでより適切な表記ゆれへの頑健性の評価を行いたい。

本研究では単語正規化を考慮した表記ゆれに頑健な BERT モデルの構築を行った。そして、下流タスクで表記ゆれによる出力の一貫性を担保しつつ正規化をしない BERT モデルと同程度の精度であることを示した。

学習に使用したスクリプトは GitHub で公開している[9)]。今後、学習したモデルも公開する予定である。本実験では、Wiki-40B を使用して BERT の事前学習をおこなった。言語モデルの学習はより大規模なコーパスであるほど良いとされており [15]、今後本提案手法の正規化を適用した国語研日本語ウェブコーパス (NWJC) [16] で学習した BERT モデルを公開する予定である。

9）https://github.com/WorksApplications/SudachiTra

参考文献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 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. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[2] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4902–4912, Online, July 2020. Association for Computational Linguistics.

[3] 小椋秀樹. コーパスに基づく現代語表記のゆれの調査— bccwj コアデータを資料として—. 第 1 回コーパス日本語学ワークショップ, pp. 321–328, 2012.

[4] Kazuma Takaoka, Sorami Hisamoto, Noriko Kawahara, Miho Sakamoto, Yoshitaka Uchida, and Yuji Matsumoto. Sudachi: a japanese tokenizer for business. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France, may 2018. European Language Resources Association (ELRA).

[5] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

[6] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[7] Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. Wiki-40B: Multilingual language model dataset. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 2440–2452, Marseille, France, May 2020. European Language Resources Association.

[8] Yuta Hayashibe and Kensuke Mitsuzawa. Sentence boundary detection on line breaks in Japanese. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pp. 71–75, Online, November 2020. Association for Computational Linguistics.

[9] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, Vol. 21, No. 140, pp. 1–67, 2020.

[10] 稲葉通将. おーぷん 2 ちゃんねる対話コーパスを用いた用例ベース対話システム. 第 87 回言語・音声理解と対話処理研究会 (第 10 回対話システムシンポジウム), 人工知能学会研究会資料 SIG-SLUD-B902-33, pp. 129–132, 2019.

[11] 柴田知秀, 河原大輔, 黒橋禎夫. Bert による日本語構文解析の精度向上. 言語処理学会第 25 回年次大会発表論文集, 2019.

[12] Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. The multilingual Amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4563–4568, Online, November 2020. Association for Computational Linguistics.

[13] Kazumasa Omura, Daisuke Kawahara, and Sadao Kurohashi. A method for building a commonsense inference dataset based on basic events. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2450–2460, Online, November 2020. Association for Computational Linguistics.

[14] 鈴木正敏, 松田耕史, 岡崎直観, 乾健太郎. 読解による解答可能性を付与した質問応答データセットの構築. 言語処理学会第 24 回年次大会発表論文集, 2018.

[15] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv:2010.14701*, 2020.

[16] Masayuki Asahara, Kikuo Maekawa, Mizuho Imada, Sachi Kato, and Hikari Konishi. Archiving and analysing techniques of the ultra-large-scale web-based corpus project of ninjal, japan. *Alexandria*, Vol. 26, No. 1-2, pp. 129–148, 2014.

A 付録

A.1 コーパスのクリーニング

4.1 節で行っているクリーニング処理を以下に示す。

表 6 学習コーパスのクリーニング処理一覧

引用元	クリーニング処理
東北大 BERT	短すぎる文（10 文未満），長すぎる文（200 語超過）を削除
東北大 BERT	引用記号などのマークアップを削除
東北大 BERT	不可視文字を削除
C4	短すぎる文書（5 文未満）を削除
C4	ソースコードが含まれうる文書（波括弧が含まれている文書）を削除
C4	NG ワードが含まれる文書を削除
本実験	メールアドレスが含まれる文を削除
本実験	URL が含まれる文を削除
本実験	（文分割エラー回避として）2 文字以下の文を前の文の末尾に結合する

A.2 比較対象のモデル詳細

5.2 節でベースラインとして採用したモデルの詳細を以下に示す。

表 7 モデル詳細

モデル	コーパス	形態素解析器	訓練量
東北大 BERT	日本語 Wikipedia (2019/09/01, 17M 文)	MeCab IPA	256 batch size * 1M steps
京都大 BERT	日本語 Wikipedia (18M 文)	Juman++	30 epochs
NICT BERT	日本語 Wikipedia	MeCab Juman	4096 batch size * 1.1M steps

A.3 正規化の例

5.1 節で表記ゆれの生成に用いた normalized_and_surface による正規化の例を以下に示す。

表 8 normalized_and_surface による正規化

元文	ガッカリです。ヨレヨレなのは仕方ないと思えます。
正規化後	がっかりです。よれよれなのは仕方ないと思えます。
元文	なんと瀬戸大橋のたもとまで行けるようだ
正規化後	何と瀬戸大橋の袂まで行けるようだ
元文	オクタカルボニル二コバルト Co(CO)
正規化後	オクトカルボニル 2 コバルト CO（CO）

3 つ目の例でコバルトの元素記号"Co"が"CO"に変換されるなど、不適切な変換も含まれる。