Tokenizer の違いによる日本語 BERT モデルの性能評価

築地俊平
茨城大学工学部
情報工学科
17t4055g@vc.ibaraki.ac.jp

新納浩幸
茨城大学大学院理工学研究科
情報科学領域
hiroyuki.shinnou.0828@vc.ibaraki.ac.jp

1 はじめに

近年、事前学習済みモデル BERT を利用することで自然言語処理システムの性能が飛躍的に向上している。当初、実質英語だけが対象であった BERT だが、現在は日本語 BERT も数多く構築・公開され、一般に利用できるようになってきた。様々な日本語 BERT がある中で、どの BERT を利用すべきかは明らかではないが、選択の要因の一つは単語分割にあると考えられる。日本語 BERT は各々の Tokenizer を持ち、そこで利用される語彙リストが異なっている。そのためタスクの目的に合った Tokenizer を持つ BERT を利用した方がよいと考えられる。一方逆に利用する BERT を固定したい場合も多いと思われる。その場合、その BERT の持つ Tokenizer とは異なる Tokenizer を利用した際に、性能への影響がどの程度あるのかは興味深い。

ここでは東北大版の 'bert-base-japanese' と呼ばれる BERT を利用して、上記の調査を行った。'bert-base-japanese' の BERT は Tokenizer として MeCab が利用されている。入力された文は MeCab により単語分割され、各単語が token id に変換される。この際、語彙リストに登録されていない単語については WordPiece が起動されて単語分割が行われる。つまり BERT への入力として別の Tokenizer を利用して単語分割を行っても、token id 列に変換され、以後の処理は正常に行われる。

本論文では BERT を利用した文書分類のタスクを行う。利用する BERT は東北大版日本語 BERT である。文書から BERT の入力となる token id 列を得るのに、様々な Tokenizer を利用して、Tokenizer の違いによる性能の評価を行う。

2 BERT と Tokenizer

2.1 BERT

BERT は汎用的な言語モデルの事前学習を行う手法の一つである． 大量のテキストデータを事前学習したモデルに少量の教師データを追加学習させることで、テキスト分類などの様々なタスクで先行研を超える分類性能を達成している．BERT は双方向 Transformer をベースとした汎用言語モデルであり、大規模コーパスで事前学習を行い、その後タスクに応じたファインチューニングを行うことで、様々なタスクに応用できるモデルである本実験では BERT の学習済みモデルには東北大学乾・鈴木研究所で作成された日本語 BERT を用いる [1].

2.2 Tokenizer

日本語は英語と比べると単語間に明確な区切りが存在しない．そのため、日本語の解析をする場合には単語、文字など何らかの部分に分ける必要がある．さまざま分け方はあるが、分割された文字列のことをトークンという．Tokenizer は文書を単語分割をして、それに ID という数値に変換するモジュールである．単語を分割する方法としては

- N-graum を使って単語に分割
- MeCab、Juman などを使って形態素解析して形態素に分割
- 事前に単語単位に分割して各単語の頻度を求め、高頻度で現れる単語を一つの単語とするサブワード

がある．また、サブワードについてくわしくは後述する．

表 1 Tokenizer による単語分割

Tokenizer	分割
生データのまま	コンビニ店員になりきってお釣りを返していこう！
MeCab	コンビニ/店員/に/なり/きっ/て/お釣り/を/返し/て/いこ/う/！
MeCab+NEologd	コンビニ店/員/に/なり/きっ/て/お釣り/を/返し/て/いこ/う/！
Juman++	コンビニ/店員/に/なりきって/お釣り/を/返して/いこう/！
Sudachi	コンビニ/店員/に/なり/きっ/て/お釣り/を/返し/て/いこう/！
nagisa	コンビニ/店/員/に/なり/きっ/て/お/釣り/を/返し/て/いこう/!
SentencePiece	コンビ/ニ/店/員/になり/き/って/お/釣り/を/返し/てい/こう/!

表 2 Tokenizer による token id 列

Tokenizer	token id 列
生データのまま	20617, 20333, 7, 297, 2551, 16, 73, 30406, 28477, 11, 13043, 16
MeCab	20617, 20333, 7, 297, 2551, 16, 73, 30406, 28477, 11, 13043, 16
MeCab+NEologd	20617, 805, 1151, 7, 297, 2551, 16, 73, 30406, 28477, 11, 13043, 16
Juman++	20617, 20333, 7, 297, 2551, 16, 73, 30406, 28477, 11, 13043, 16
Sudachi	20617, 20333, 7, 297, 2551, 16, 73, 30406, 28477, 11, 13043, 16
nagisa	20617, 805, 1151, 7, 297, 2551, 16, 73, 8639, 11, 13043, 16
SentencePiece	3900, 353, 805, 1151, 7, 297, 322, 6172, 73, 8639, 11, 13043, 16

3 実験

本研究では 6 種類の Tokenizer と東北大学乾・鈴木研究室が公開している日本語 BERT を使う．その中で Toknizer の SentencePiece は日本語 Wikipedia データを使用する．使用する日本語 BERT の Tokenizer は MeCab+WordPiece である．

3.1 実験条件

本研究では訓練データ、テストデータを livedoornews から使用した．テストデータは 736 文、訓練データは 5887 文が用意し、それぞれ以下の 9 つのカテゴリに属するものを収集したデータセットである．

- トピックニュース
- SportsWatch
- IT ライフハック
- 家電チャンネル
- MOVIE ENTRY
- 独女通信
- エスマックス
- livedoor HOMME
- Peachy

本実験では 9 種類の文書を文書分類をするタスクを行う．まずは通常の単語分割がされていないなめデータの文書を訓練データとして学習させ、単語分割していない生データである、テストデータを実行し精度を確認した．次に各 Tokenizer で訓練データ、テストデータを単語分割した文書を使い精度を確認した．続けて訓練データのみ Tokenizer で単語分割した文書を使用し、テストデータは単語分割のない生データの文書で精度を確認した．最後に訓練データは単語分割されていない生データを使用し、テストデータは各 Tokenizer で単語分割した文章を使いそれぞれ精度を確認した．

3.2 性能評価に使用する Tokenizer

今回使用する Tokenizer の特徴は以下の通りである．Tokenizer ごとにアルゴリズムがかわってくるため、特徴と違いについて触れる．

MeCab+IPAdic、MeCab+IPAdic+NEologd ラティスを構築して、その中から最適なパスを選択する．辞書である NEologd が追加されたため複数の形態素に分割されてしまう固有表現をもっているニュース記事から抽出した新語や固有名詞や道語、ネット上で流行した単語や慣用句やハッシュタグなども行っている．文書クラスが定義する以下についての変更は禁止とする．

Juman++ 京都大学黒橋研究室で作成された,RNN を使用した形態素解析機である．ニューラル LM を用いることで精度が高くなっている．辞

表 3 Sudachi の分割情報と 3 種類の形態素解析結果

登録見出し	選挙/管理/委員会	カンヌ/国際/映画祭
A 単位	選挙|管理|委員|会	カンヌ|国際|映画祭
B 単位	選挙|管理|委員会	カンヌ|国際|映画祭
C 単位	選挙管理委員会	カンヌ国際映画祭

書は基本辞書、Wikipedia、Wikiionary、Web text であり、合計で 90 万語ほどになっている．

Sudachi [2]

ワークスアプリケーションズ　ワークス徳島人工知能 NLP 研究所が開発した形態素解析機である．

UniDic 由来の単単位語句と NEologd 由来の膨大な固有名称をベースに大規模な辞書データをを構築している．約 280 万語を超えている．最適な形態素の長さは、アプリケーションによって異なるという考えから 3 種類の形態素単位が用意されている（表 3 参照）

A 単位とはほぼ UniDic の単単位規定と同じであるが、一部のものは更に短くしているものを A 単位としている．B 単位はとは A 単位に説示及び漢字 1 文字の名詞が結合したもの、及び複合動詞である．C 単位とは更に多くの語句が結合したもので、複合名詞や固有名称、慣用句などがこれに相当し、アプリケーションごとに形態素単位を選択している．

Neologd 由来の語句は長単位のものが多いため、基本的にすべて分割情報を付与している。ただし、量が膨大なため、まず機械的に分割情報を付与し、人手でチェックする

さまざまなツールがある中、商業利用に耐えうる、より高品質で使い勝手の良い形態素解析器であり、辞書の継続的なメンテナンスがされる．

nagisa [3]

Bidrectional-LSTM を用いた日本語単語分割ト品詞推定が特徴である．Bi-LSTM を用いた文字単位の系列ラベリングによる解析を行うので、顔文字や URL、ネット用語に対し頑健な解析ができる．多言語に対応している。

SentencePiece [4]

サブワードの考え方を用いた分割方法である．

事前にテキストを単語分割し、各単語の頻度を求めておく．高頻度の単語は 1 単語として扱い、低頻度後は短い単位に分割する．最終的なトークン数が事前に与えられた数千から数万のサイズになるように分割をする．サブワードによって未知語がなくなり、低頻度語も最終的には文字になり、語彙数を小さくすることができる．

SentencePiece は文書をサブワードに分割するモデルであり、コーパスから教師なし学習が可能になる．特徴の一つとして、End to End 処理に向けたテキストの可逆分割ができます．通常の形態素解析の単語分割では、分割はできてもそれを復元することは困難である．この単語分割の逆を脱トークン化という．脱トークン化を可能にするために、SentencePiece ではスペースを通常のトークンとして扱う、便宜的にスペースをメタ文字 (_) に書き換える．言語依存の処理がなく、完全な End-to-End 処理が可能になる．

参考として各 Tokenizer で得られる単語分割と token ID 列の例を表 1 と表 2 に示す。

3.3 実験結果と考察

今回の実験では、訓練データとテストデータの文書を生データのまま東北大版 BERT で文書分類し、評価値を調べたところ、結果は 87.44 となった．これを基準として、3 つの実験結果を見ていく．

訓練データとテストデータを各々同じ Tokenizer で単語分割したデータでタスクを行った．文書分類をした結果は表 4 となった．全体を通じて、生データのまま文書分類をした場合と比べて、評価値が飛躍的に高くなる、低くなることはなかった．SentencePiece で単語分割したデータの評価値がほかと比べて低くなっていた．

訓練データは生データ、各 Tokenizer で単語分割したテストデータで文書分類のタスクを行った結果は表 5 となった．ここでも生データのままタスクを行ったときの評価値と大きく差は出なかったが、SentencePiece のみ 84.96 と大幅に評価値を落としていた．

テストデータは生データのままで、各 Tokenizer で単語分割した訓練データでの文書分類のタスクを行った結果は表 6 となった．同じように 88.00 を超えることはなく、SentencePiece が評価値落としていた．Tokenizer が同じで、訓練データやテストデータで比較すると、大きな違いはなく、SentencePiece 以外は 87.00 から 87.90 の間の評価値であった．

東北大版日本語 BERT の Tokenizer が MeCab+WordPiece である．他の Tokenizer では分割が異なるため、ID が変わる．そのため、MeCab で単語分割したタスクは生データの評価値と近くなる

ことが予想されていた．逆に、他の Tokenizer で単語分割し、タスクを行ったときは評価値が下がると予想していたが評価値は MeCab での評価値とあまり変わらなかった．これは、多少 Tokenizer の仕組みが変わっても、ID 化したときにはタスクにはあまり影響が出ない程度の違いでしかないと考えられる．Tokenizer で分割した文書を ID にしたときの違いの割合と関係していると考察する．

表 4 訓練データとテストデータを Tokenizer で単語分割

両方のデータに使用した Tokenizer	accuracy
生データのまま	87.44
MeCab+IPAdic	87.39
MeCab+IPAdic+NEologd	87.85
Juman++	87.76
Sudachi	87.59
nagisa	87.71
SentencePiece	86.37

表 5 テストデータを Tokenizer で単語分割

テストデータに使用した Tokenizer	accuracy
生データのまま	87.44
MeCab+IPAdic	87.77
MeCab+IPAdic+NEologd	87.26
Juman++	87.77
Sudachi	87.26
nagisa	87.45
SentencePiece	84.96

表 6 訓練データを Tokenizer で単語分割

訓練データに使用した Tokenizer	accuracy
生データのまま	87.44
MeCab+IPAdic	87.21
MeCab+IPAdic+NEologd	87.53
Juman++	87.80
Sudachi	87.58
nagisa	87.30
SentencePiece	85.11

4 おわりに

本研究では、訓練データとテストデータの文書を様々な Tokenizer を使用することによって、日本語 BERT の評価値の変化を確認することができた．

単語ごとに分割する形態素解析器での文章の分割の仕方によって、文章のトークン化に多少の違いはあるが、日本語 BERT の Tokenizer(今回は MeCab+NEologd) でトークン化した ID と結果が類似したためか精度は大きく変化はしなかった．

単語を ID にしてもところどころの変化することがあっても、その程度であれば学習に影響が出にくくなっている．もしくは単語分割、今回であれば形態素解析器らは精度が上がり、かなり正確に同じように単語分割ができており、その結果、評価値に変化が小さくなってると考えた．そうであれば単語分割のアプローチは様々あれど形態素解析であれば、結局はあまり変わらないことが予想される．

課題としては、今回は日本語 BERT の東北大版のみを実験で使用したが、日本語事前学習モデル全体でも今回の実験と同じように、結果の差が出るかはわからない．

今後は、異なる日本語 BERT での精度の変化を調べたい．SentenceBERT など Tokinizer がサブワードで単語分割されている日本語 BERT では違いがある可能性がある．

謝辞

本研究は JSPS 科研費 JP19K12093 および 2020 年度国立情報学研究所公募型共同研究（2020-FC03）の助成を受けています．

参考文献

[1]Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018. https://arxiv.org/abs/1810.04805.

[2]坂本美保, 川原典子, 久本空海, 一馬, 内田 佳孝（株式会社ワークスアプリケーションズ ワークス徳島人工知能 NLP 研究所）. 形態素解析器『sudachi』のための大規模辞書開発. 言語資源活用ワークショップ 2018, 2018.

[3]池田大志. Python による日本語自然言語処理系統ラベリングによる実世界テキスト分析 , 2019. https://speakerdeck.com/taishii/pycon-jp-2019.

[4]John Richardson Taku Kudo. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. 2018. https://arxiv.org/abs/1810.04805.