Vol.2019-NL-243 No.8
2019/12/4

BERTを利用した文書の特徴ベクトルの作成

田中 裕隆[1,a)]　曹 鋭[2,b)]　白 静[2,c)]　馬 ブン[2,d)]　新納 浩幸[3,e)]

概要：
近年，BERT のような事前学習モデルを利用することで，自然言語処理システムの性能が大きく向上している．BERT は，Transformer の Multi-head Attention を用いることで文脈に応じた単語の埋め込み表現列を得ることのできるモデルである．文書分類のタスクの場合，文書を BERT に入力し，その出力から文書の特徴ベクトルを構築する方法によって処理できる．しかし，BERT に入力できるシーケンスの長さには上限がある．この制限によって，長い文書を扱う場合，標準的な手法では文書分類に必要な情報を十分に得られないと考えられる．そこで，BERT から長い文書内の全ての単語に対応する埋め込み表現を得て，そこから文書の特徴ベクトルを作成する手法を提案する．

キーワード：BERT，文書分類，埋め込み表現，TF-IDF

Construction of document feature vectors using BERT

Tanaka Hirotaka[1,a)]　Cao Rui[2,b)]　Bai Jing[2,c)]　Ma Wen[2,d)]　Shinnou Hiroyuki[3,e)]

1. はじめに

近年，自然言語処理の多くのタスクで，事前学習モデルを利用する有効性が示されている [7][8]．事前学習モデルは様々なものが提案されているが，その中でも BERT[2] が特に優れた性能を示している．

BERT (Bidirectional Encoder Representations from Transformers) は Transformer[10] で用いられた Multi-head attention を 12 層（あるいは 24 層）重ねたモデルであり，パラメータの学習は Masked Language Model と Next Sentence Prediction という 2 つのタスクを解くことで，教師なしの枠組みの下で行われる．学習できたモデルを利用すると，入力文あるいは入力文対に対して，その単語埋め込み表現列を得ることができる．

ただし，BERT はその入力となるシーケンスの最大長が固定である．したがって，BERT の事前学習時の最大長より長い文書を入力とすると，最大長を超える部分の単語に対して埋め込み表現を得ることはできない．そのために，通常の手法で BERT を利用する場合，実際のタスクを解くために必要な情報を文書全体から得ることができない．BERT の事前学習時に十分大きな最大長を設定することで，長い文書に対応することもできるが，入力長に上限が存在することに変わりはなく，またその事前学習のコストも高い．

そこで，任意の長さの入力シーケンスから BERT を利用した埋め込み表現列を得て，そこから文書の特徴ベクトルを作成する手法を提案する．

実験では Webis-CLS-10[*1]の日本語の感情分析のデータを利用して，提案手法の有効性を示した．考察では，提案手法による BERT からの特徴ベクトルと，word2vec による単語の分散表現から得られる特徴ベクトルとの比較を行う．

1 茨城大学工学部情報工学科
2 茨城大学大学院理工学研究科情報工学専攻
3 茨城大学大学院理工学研究科情報科学領域
Ibaraki University, Nakanarusawa 4-12-1, Hiachi, Ibaraki 316–8511, Japan
a) 16t4032n@vc.ibaraki.ac.jp
b) 18nd305g@vc.ibaraki.ac.jp
c) 19nd301r@vc.ibaraki.ac.jp
d) 19nd302h@vc.ibaraki.ac.jp
e) hiroyuki.shinnou.0828@vc.ibaraki.ac.jp

[*1] https://webis.de/data/webis-cls-10.html

2. 関連研究

2.1 BERT

BERT の基本のパーツは Multi-head attention である．Multi-head attention は n 単語埋め込み表現列を入力として，各埋め込み表現をより適切なものに変換して出力する．つまり出力は変換された n 単語埋め込み表現列である．

Multi-head attention の概略を述べる．基本は self attention なので Q, K, V の 3 組が入力である．今，単語埋め込み表現が m 次元であったとする．Multi-head attention では m 次元ベクトルを $d_k(= m/k)$ 次元に圧縮する線形変換器を Q, K, V それぞれに対して用意する．Q, K, V の実体は $d_k \times d_k$ の線形変換行列である．Multi-head attention の入力は n 個の m 次元ベクトルであるが，これが先の圧縮機で $n \times d_k$ の行列 X に変換され，Q, K, V に渡され $n \times d_k$ の行列 XQ, XK, XV ができる．これらを Q', K', V' とおき，以下の式[*2]により self attention を行う．

$$\mathtt{softmax}\left(\frac{Q'K'^T}{\sqrt{d_k}}\right)V'$$

これは $n \times d_k$ の行列である．上記の処理を k 個並行して行うと，$n \times d_k$ の行列が k 個作成され，これらを横に連結することで，$n \times m$ の行列が作成できる．これを更に同次元に線形変換することで Multi-head attention の出力が作られる．

BERT はこの Multi-head attention を 12 層（あるいは 24 層）重ねたモデルである．結局，BERT は n 単語埋め込み表現列を入力とし，それをより文脈に合った n 単語埋め込み表現列に変換していると捉えることができる．

2.2 文書分類

文書分類は分類問題の一種であり，一般に教師あり学習を用いることで解決できる．そのため従来より数多くの研究がある．またディープラーニングを利用する場合でも，CNN[5] や RNN [6] を利用するなど多くの研究がある．

一方，文書分類を含め自然言語処理の多くのタスクにおいて，事前学習モデルを利用する有効性が示されている．事前学習モデルを利用する場合，大きく 2 つの利用法がある．一つは fine tuning である．これは事前学習モデルが出力する情報を，タスクを解決するためのネットワークの入力とし，その事前学習モデルを含めたネットワーク全体を学習の対象とするものである．この場合，事前学習モデルの部分は既に大量のデータから学習できた形となっているため，比較的少量のデータを用いるだけで，連結したネットワークを学習できる．OpenAI GPT [8] はニューラルネット翻訳である Transformer [10] の decoder 部分を利用した言語モデル[*3]あり，このような fine tuning の利用を念頭においている．ULMFiT [3] においても事前学習モデルを言語モデルに設定し，目的のタスクに対して fine tuning を行う．

事前学習モデルのもう一つの利用法は feature based のものである．これは事前学習モデルが出力する情報を，目的のタスクを解くための素性として利用するものである．word2vec のような単語分散表現も事前学習モデルと捉えることができる．単語の分散表現をタスク解決のための素性とした研究には文書分類を含め多くの研究がある．fastText は Subword [1] に対する分散表現を構築し，高速かつ高精度な文書分類が行えることを実験で示している [4]．ELMo [7] は文脈を考慮した単語の分散表現を導くモデルである．実体は 2 層の双方向 LSTM であり，大規模コーパスを利用して言語モデルを学習する．これが事前学習モデルとなり，feature based の形で利用できる．

本論文で利用する BERT は従来の事前学習モデルを改善しており，様々なタスクで従来の事前学習モデルの性能を上回っている．このため本論文で扱う文書分類であっても，その効果が期待できる．

2.3 BERT と TF-IDF を併用した文書の特徴ベクトル

我々は既に BERT による単語埋め込み表現列を用いた文書分類において，文書分類を解く場合の BERT による特徴ベクトルを扱う手法について提案した [9]．

そこでは，BERT による単語埋め込み表現列から求めた平均ベクトルと，TF-IDF によって求めた特徴ベクトルを求め，これらを単位ベクトルに正規化し連結したベクトルを文書の特徴ベクトルとする（図 1 参照）．

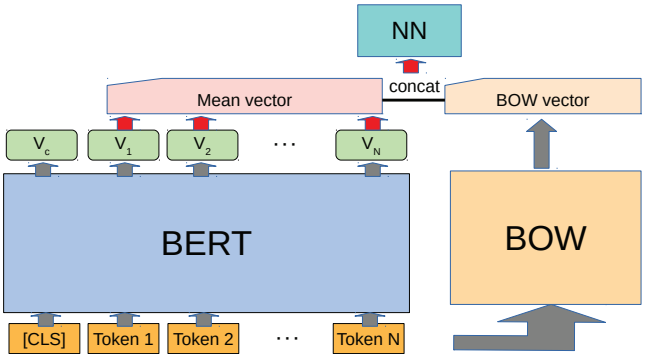

図 1 BERT と TF-IDF による特徴ベクトル

ただし上記の手法は BERT に入力する単語列の最大長の制限から，最大長を超える長い文書に対しては，最大長以降を切り捨てた文書にしてから BERT への入力を行っている（図 2 参照）．上記の手法は BERT 単独の手法と比較して，大きな改善を果たしているが，それは切り捨てられた単語列の情報を BOW のモデルに取り込めているからと考えられる．

[*2] Scaled Dot-Product Attention
[*3] 言語モデルも一種の事前学習モデルである．

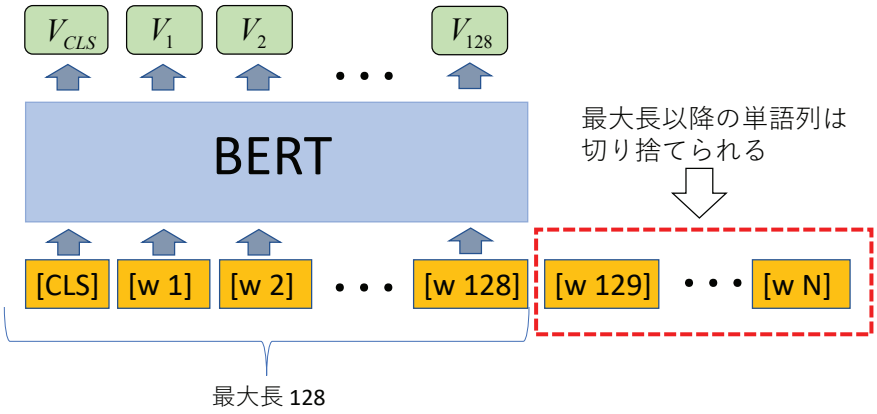

図 2　最大長を超える単語列部分の切り捨て

3. 提案手法

標準的な手法では，BERT の学習時に設定した最大長を超えた入力シーケンスからは，最大長を超えた部分の単語列に対してその埋め込み表現を得ることはできない．ここでは，BERT の最大長を超えるシーケンスに対して，全ての単語に対する埋め込み表現を得る手法について示す．(図 3 参照)

提案手法は，以下の手順で任意の長さの文書に対して，その特徴ベクトルを得る．

(1) 対象となる文書の先頭から BERT の最大長分を入力として BERT の出力を得る．
(2) 文書の全てが BERT の入力列に入りきらなかった場合，文書の先頭から BERT の最大長の半分だけずらした位置から最大長分を入力として BERT の出力を得る．
(3) 文書の全てから BERT の出力を得られるまで手順 (2) を繰り返す．
(4) 得られた BERT の出力である単語埋め込み表現のベクトル全てを足し合わせる．

このようにして得られたベクトルを BERT による文書の特徴ベクトルとする．

数式で表すと次のようになる．入力となる文書を d として，先頭から i 番目の単語トークンを d_i とする．文書の長さを N，BERT の最大長を L とする．入力の d_i に対応する BERT の出力を v_i^j とする．BERT による特徴ベクトル b_l は，次のように表せる．

$$\begin{aligned} b_l = & v_0^0 + v_1^0 + \cdots + v_{L/2-1}^0 \\ & + v_{L/2}^0 + v_{L/2}^1 + v_{L/2+1}^0 + v_{L/2+1}^1 + \cdots \\ & + v_{L-1}^0 + v_{L-1}^1 + v_L^1 + v_L^2 + \cdots \\ & + v_i^j + \cdots + v_N^{1+(N-1)/L} \end{aligned}$$

実験では，TF-IDF によるベクトル t と連結した $[b_l; t]$ を文書 d の特徴ベクトルとして文書分類を行う．

文書分類のための分類器には 3 層ニューラルネットワークを用いた．具体的には各層は全結合であり，順に 400 次元，50 次元，2 次元ベクトルへと線形変換される．活性化関数にはシグモイド関数を用いており，出力層に対しては softmax 関数を用いている．交差エントロピー誤差を損失関数として損失を求め，Adam によって最適化した．

4. 実験

4.1 日本語 BERT 事前モデル

公開されている BERT の多言語モデル[*4]には日本語も含まれており，日本語のタスクに対して多言語の事前学習モデルを利用することも可能である．しかし，これを利用すると基本単位が文字になってしまい，適切ではないと考えられる．そこでここでは，日本語に対応した事前学習モデルとして，京都大学黒橋・河原研究室が以下のサイトで公開している日本語事前学習モデルを使用する．

http://nlp.ist.i.kyoto-u.ac.jp/index.php?BERT%E6%97%A5%E6%9C%AC%E8%AA%9EPretrained%E3%83%A2%E3%83%87%E3%83%AB

このモデルは，訓練コーパスを日本語 Wikipedia(約 1800 万文) とし，モデルの構造は BASE と同じ 12 層，語彙数は 32000，最大長は 128 としている．

また，この事前学習モデルの入力となるテキストは，同じく京都大学黒橋・河原研究室が公開している Juman++[*5]で形態素解析を行い，形態素単位に分割する．

4.2 実験データ

実験で使用したデータセットは，以下のサイトで公開されている Amazon のレビュー文書である．評価の 4 と 5 を positive，1 と 2 を negative とした感情分析データとして用いる．

[*4] https://storage.googleapis.com/bert_models/2018_11_23/multi_cased_L-12_H-768_A-12.zip
[*5] http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN++

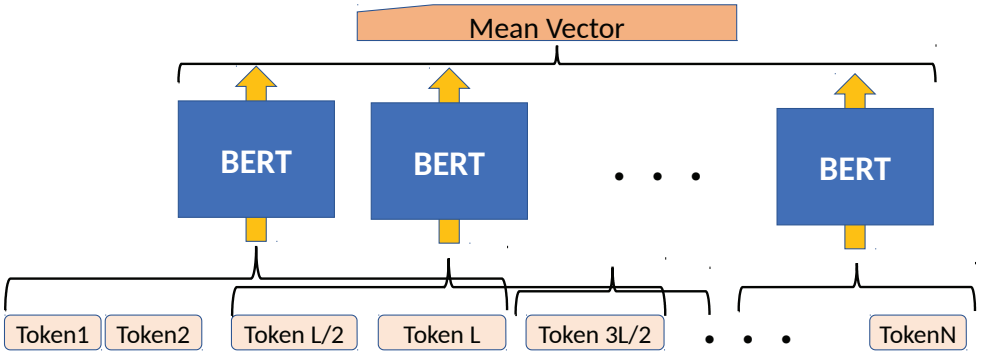

図 3　BERT による特徴ベクトル作成の提案手法

https://webis.de/data/webis-cls-10.html

このデータセットは books，DVD，music の 3 つの領域がある．領域毎に訓練データとして 2,000 文書，テストデータとして 2,000 文書が存在する．

この実験で使用する BERT 事前学習モデルの最大長は 128 である．この実験のテストデータには，トークン列が 128 を超える文書が各領域にそれぞれ 905 文書, 872 文書及び 727 文書存在する．つまりデータ全体に対して約 40 % 程度の文書が，標準的な手法では扱いきれない長さの文書である．

TF-IDF で扱う特徴語は，全訓練データ 6,000 文書に出現する 41,400 語とする．

4.3　実験結果

実験結果を表 1 と図 4 に示す．この結果は，3 領域それぞれ 10 回ずつ試行した平均値である．

b_s は，BERT の最大長を超える情報は切り捨てて，求めた BERT の単語埋め込み表現列の平均ベクトルを文書の特徴ベクトルとする手法による結果である．t は，TF-IDF によるベクトルを文書の特徴ベクトルとする手法による結果である．$[b_s; t]$ は，b_s と t を連結して得られた特徴ベクトルを用いる手法の結果である．b_l は，提案手法のように BERT から求めたベクトルを文書の特徴ベクトルとする手法である．b_s より良い結果となっている．$[b_l; t]$ は，b_l と t を連結して得られた特徴ベクトルを用いる提案手法の場合の結果である．この提案手法は，実験の中で最も良い結果となった．

表 1　実験結果（各手法の正解率）

	books	DVD	music	3 領域の平均
b_s	0.7859	0.7818	0.8086	0.7921
t	0.7816	0.8135	0.8224	0.8058
$[b_s; t]$	0.8156	0.8229	0.8427	0.8271
b_l	0.8086	0.8014	0.8203	0.8101
$[b_l; t]$	**0.8192**	**0.8338**	**0.8516**	**0.8349**

また，提案手法 $[b_l; t]$ が最大長 128 を超える文書 (以下長文とする) に対して切り捨てる手法 $[b_s; t]$ と比較して有効となっていることを確かめるために，長文のみのテストデータで評価を行った結果を表 2 に示す．表 2 より，提案手法が長文に対して有効であることがわかる．

表 2　長文に対する結果（正解率）

	books	DVD	music	3 領域の平均
$[b_s; t]$	0.8079	0.8360	0.8355	0.8265
$[b_l; t]$	**0.8202**	**0.8464**	**0.8488**	**0.8385**

5.　考察

5.1　分散表現列との比較

実験では BERT による単語埋め込み表現列を用いた．ここでは，word2vec による分散表現列を用いて同様に実験を行い，その結果を比較する．ここで用いる word2vec は，以下のサイトで公開されている学習済みのモデルである．このモデルは，本論文で用いた BERT の日本語事前学習モデルと同様に，日本語 Wikipedia データから学習している [12].

http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/

入力文書に対して形態素解析を行い，得られた単語列から word2vec により分散表現列を求めた．次にそれら分散表現の平均ベクトル w を求め正規化し，TF-IDF によるベクトル t と連結して $[w; t]$ を作成した．この $[w; t]$ を入力文書の特徴ベクトルとし，先の実験を行った．結果を表 3 に示す．

表 3　分散表現列を用いた手法との比較（正解率）

	books	DVD	music	3 領域の平均
$[w; t]$	0.7899	0.8207	0.8330	0.8146
$[b_s; t]$	0.8156	0.8229	0.8427	0.8271
$[b_l; t]$	**0.8192**	**0.8338**	**0.8516**	**0.8349**

この実験から，分散表現列を用いた場合と比較して，BERT を用いた手法はより良い結果となった．分散表現列と BERT による単語埋め込み表現列は，同じような情報を表現しているが，BERT の方が分散表現よりも有用であることが言える．

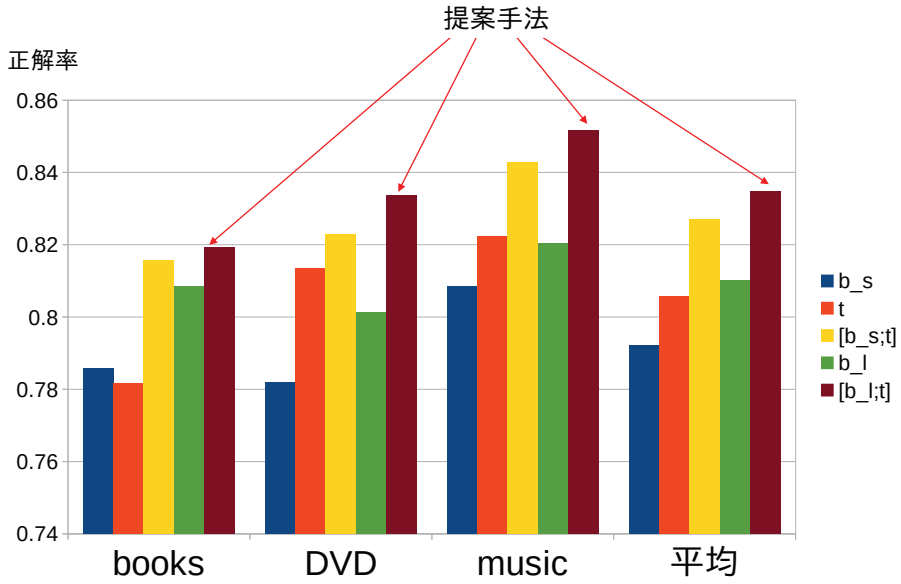

図 4　実験結果のグラフ

5.2 Fine-tuning

BERT は，fine-tuning を行うことによって高いパフォーマンスを出すことのできるモデルである．ここでは，BERT における fine-tuning の標準的な手法で実験を行い，結果を比較する．fine-tuning の標準的な手法とは，BERT で入力シーケンスに付与する特殊トークンである [CLS] の埋め込み表現を文書の特徴ベクトルとして学習する手法である．

ここでは BERT のソースと一緒に公開されている `run_classifier.py` [*6]を使うことで，実験データに対して fine tuning を行った．学習率は 2e-5，エポック数は 3 に設定した．その結果を表 4 に示す．f_{bert} が fine tuning の結果である．f_{bert} は $b_{[CLS]}$ よりも改善されてはいるが b_l に及ばない．当然，提案手法である $[b_l; t]$ よりも正解率ははるかに低い．

表 4　Fine Tuning との比較（正解率）

	books	DVD	music	3 領域の平均
$b_{[CLS]}$	0.7629	0.7567	0.7779	0.7658
b_l	0.8086	0.8014	0.8203	0.8101
t	0.7816	0.8135	0.8224	0.8058
$[b_l; t]$	**0.8192**	**0.8338**	**0.8516**	**0.8349**
f_{bert}	0.7894	0.7799	0.8019	0.7904

文書分類では単純に [CLS] の埋め込み表現を用いた BERT の fine tuning よりも，提案手法のように BERT を feature based で利用する方が有効であることが確認できた．ただし本論文の提案手法のように文書の特徴ベクトルを構築し，そこから fine tuning することも可能だと思われる．今後はそれも試したい．

[*6] `https://github.com/google-research/bert`

5.3 XLNet

XLNet[11] は，BERT の問題点を改良したモデルである．その問題点の一つは，BERT の事前学習手法の一つの Masked Language Model である．XLNet では，より純粋な自己回帰モデルベースの言語モデルに改良することで，この問題点に対処している．また，XLNet のもう一つの特徴は任意の入力長に対応していることである．本論文では，BERT が任意の入力長を扱えないことを問題点としてあげているが，XLNet も同様にこの問題に対処している．

以下で公開モデルされている日本語 XLNet のモデル (NFKC 版) を利用した実験を行った．このモデルは tokenizer に MeCab+NEologd 及び Sentencepiece を利用している．MeCab を利用することで形態素として正しく分割し，その上で Sentencepiece を行うことで，語彙のカバー範囲を広げている．

`https://qiita.com/mkt3/items\`
`/4d0ae36f3f212aee8002`

結果を表 5 に示す．表中の f_{xlnet} が XLNet を用いて fine tuning を行った結果である．BERT を用いた fine tuning の結果である f_{bert} よりもわずかに正解率が低く，XLNet を用いた効果はなかった．

XLNet の fine tuning の利用法では，本タスクに対しての効果はなかったが，feature base の利用法では良い結果を得ることができる可能性がある．XLNet の feature base

表 5　XLNet との比較（正解率）

	books	DVD	music	3 領域の平均
$[b_l; t]$	**0.8192**	**0.8338**	**0.8516**	**0.8349**
f_{bert}	0.7894	0.7799	0.8019	0.7904
f_{xlnet}	0.7750	0.7750	0.8000	0.7833

の利用法に関しては今後の課題である．

6. おわりに

本論文では BERT の最大長が固定である制約に対して，最大長より長い文書に対しても BERT の情報を有効に用いる手法を提案した．入力シーケンスに対して最大長の半分ずつスライドして複数回 BERT の出力を求めることによって，最大長を超える文書に対しても BERT の情報を用いて文書の特徴ベクトルを作成した．Amazon データセットを利用した実験により，提案手法の有効性を示した．提案手法を fine tuning に拡張することや，XLNet の feature base の利用を試すことを今後の課題とする，

参考文献

[1] Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T.: Enriching Word vectors with Subword Information, *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146 (2017).
[2] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *NAACL-2019*, pp. 4171 – 4186 (2019).
[3] Howard, J. and Ruder, S.: Universal Language Model Fine-tuning for Text Classification, *ACL-2018*, pp. 328–339 (2018).
[4] Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T.: Bag of Tricks for Efficient Text Classification, *arXiv preprint arXiv:1607.01759* (2016).
[5] Kim, Y.: Convolutional Neural Networks for Sentence Classification, *EMNLP-2014*, pp. 1746–1751 (2014).
[6] Lai, S., Xu, L., Liu, K. and Zhao, J.: Recurrent convolutional neural networks for text classification, *AAAI-2015*, pp. 2267–2273 (2015).
[7] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. and Zettlemoyer, L.: Deep Contextualized Word Representations, *NAACL-2018*, pp. 2227–2237 (2018).
[8] Radford, A., Narasimhan, K., Salimans, T. and Sutskever, I.: Improving language understanding by generative pre-training, *Technical report, OpenAI.* (2018).
[9] Tanaka, H., Shinnou, H., Cao, R., Bai, J. and Ma, W.: Document Classification by Word Embeddings of BERT, *PACLING-2019*, paper# 11 (2019).
[10] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I.: Attention is all you need, *Advances in neural information processing systems*, pp. 5998–6008 (2017).
[11] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. and Le, Q. V.: XLNet: Generalized Autoregressive Pretraining for Language Understanding, *arXiv preprint arXiv:1906.08237* (2019).
[12] 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).