令和 3 年度年度茨城大学大学院理工学研究科情報工学専攻

修士学位論文

簡易小型化 BERT を利用した日本語構文解析

所属　　情報工学専攻

著者　　河野慎司 (20NM709N)

指導教員　　新納浩幸教授

令和 4 年 2 月 4 日 (金)

令和 3 年度年度茨城大学大学院理工学研究科情報工学専攻

修士学位論文

簡易小型化 BERT を利用した日本語構文解析

著者: 河野慎司 (20NM709N)

指導教員: 新納浩幸教授

論文要旨

近年，自然言語処理の分野において単語のベクトルを獲得する様々な手法が登場してきている．その中でも BERT は事前に学習したネットワークの重みを fine-tuning することで様々な NLP タスクに対して高精度を出した事前学習済みモデルである．しかし BERT は多くのパラメータを調整する必要があるため学習や推論に時間がかかるという問題がある．そこで本論文では，自然言語処理のタスクの一つである日本語の構文解析に対して，BERT の一部の層を削した簡易小型化 BERT の利用を提案する．ここでの日本語構文解析とは句同士の係り受け関係を明らかにするタスクのことである．先行研究では，通常版 BERT を日本語構文解析に利用した結果，KNP などの構文解析器や BiLSTM を利用したときよりも高い精度を出せると示されている．

実験では，京都大学ウェブ文書リードコーパスと京都大学テキストコーパスを混合したデータを用いて，京大版の BERT とそれを元に構築した簡易小型化 BERT の解析精度と処理時間を比較した．提案する簡易小型化 BERT では，京大版の BERT からの精度劣化をウェブコーパスで 0.84%，テキストコーパスで 0.91% に押さえながら，学習時間は 83%，推論時間はウェブコーパスで 64%，テキストコーパスで 84% まで削減することができた．

構文解析のテスト精度は BERT の中位層を削除したモデルが削除した層数に関わらず高い精度を維持していたため，日本語構文解析では BERT の上位層と下位層の両方を使うことが効果的だと考えられる．また，BERT の層を削除した各モデルが正解や不正解だった係り受けペアを，構文解析器である KNP を使用し格解析してみた結果，どのモデルにおいても大きな差が出なかったため，格に関しては BERT が考慮していないと推測できる．一方で，BERT の層を削除した各モデルが正解とみなしたペアのうち未知語の割合は，下位層を削除したモデルよりも上位層を削除したモデルに多く含まれていたため，BERT の上位層ほど語彙を捉えて構文解析を行っていると考えられる．

Master's Thesis in Scholastic 2022,
Major in Computer and Information Sciences,
Granduate School of Science and Engineering, Ibaraki University

Japanese Parsing Using Smaller BERT

Author: Shinji Kono (20NM709N)

Adviser: Prof. Hiroyuki Shinnou

Abstract

In recent years, various methods for acquiring word vectors have emerged in the natural language processing. And, BERT is a pre-trained model that achieves high accuracy for various NLP tasks by fine-tuning the weights of the pre-trained network. However, BERT has a problem that it takes a long time to train and infer because many parameters need to be adjusted. In this paper, we propose the use of a simplified miniaturized BERT for Japanese syntactic analysis by dropping some of the layers of BERT. Japanese syntactic parsing refers to the task of clarifying clause-to-clause relations. From previous studies, the results of using BERT for Japanese syntactic parsing were more accurate than those of other parsers such as KNP and BiLSTM.

In this experiment, we compared the parsing accuracy and processing time of the Kyoto University version of BERT and a simplified miniaturized BERTs. The proposed simplified and miniaturized BERT reduces the training time by 83% and the inference time by 64% for the web corpus and 84% for the text corpus while keeping the accuracy degradation from the Kyoto University version of BERT to 0.84% for the web corpus and 0.91% for the text corpus.

The test accuracy showed that the model with the middle layer of BERT dropped maintained high accuracy regardless of the number of dropped layers, suggesting that using both the top and bottom layers of BERT is effective for Japanese syntactic parsing. The results of analysis using KNP, showed that there was no significant difference between the models, so we can assume that BERT does not take the case into account. On the other hand, the percentage of unknown words in the pairs considered correct by the models with the BERT layers dropped was higher in the models with the higher layers dropped than in the models with the bottom layers dropped, suggesting that the higher layers of BERT capture and parse more of the vocabulary.

目次

第 1 章

序論

自然言語処理は私たちが普段利用する日本語や英語などの自然言語をコンピュータを使って分析する研究分野である．多言語間の機械翻訳技術や，質問に対する自動回答など，今日自然言語処理は日常の様々な分野で応用されている．

自然言語処理のタスクの一つに，構文解析というものが挙げられる．構文解析とは句同士の係り受け関係を明らかにするタスクのことである．日本語の構文解析を行う場合，日本語構文解析ツールの KNP[*1] [1] [2] が有名であるが，近年，BERT を利用することで，従来の KNP よりも高い精度を出すことが示されている [3].

BERT [4] は fine-tuning(後述) することで様々な NLP タスクに対して高精度を出せる事前学習済みモデルである．BERT を利用した構文解析では，BERT からの出力ベクトルを順伝播型ニューラルネットワーク (FFNN) に入力し構文解析を行う．ただし，fine-tuning には多くのパラメータを調整する必要があるため学習や推論に時間がかかるという問題がある．

そこで本研究では，事前学習済み BERT の一部の層を削除した簡易小型化 BERT の利用を提案する．実験では，京都大学ウェブ文書リードコーパス [5] と京都大学テキストコーパス [6] を混合したデータを用いて，京大版の BERT[*2,*3]と，それを用いて構築した簡易小型化 BERT の解析精度と処理時間を比較した．提案する簡易小型化 BERT では，3~10 層目を削除した合計 4 層のモデルが，京大版の BERT からの精度劣化をウェブコーパスで 0.84%，テキストコーパスで 0.91% に押さえる結果となり，層を削除した

[*1] https://nlp.ist.i.kyoto-u.ac.jp/?KNP

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

[*3] `https://nlp.ist.i.kyoto-u.ac.jp/?ku_bert_japanes`

後でも高い精度を維持していることが分かった．また学習・推論時間は削除する層を増やすほど速くなり，合計 4 層モデルでは学習時間は 83%，推論時間はウェブコーパスで 64%，テキストコーパスで 84% まで削減することができた．

また BERT のどの位置の層が構文情報を捉えているかを，12 層のうち 1 層のみ fine-tuning に使用しテストを行うことで調査した．その結果，新聞コーパスは上位・下位層が高い精度を出したが，Web コーパスにおいてはどの層も大きな変化は出なかった．BERT はコーパスに含まれるトークン数の多さや未知語の割合などによって，構文解析の精度に変化が出ることを示した．

第 2 章

関連研究

本章では，BERT を利用した構文解析及び文法の特性を捉えた研究，また BERT の層がどういった情報を捉えているかの研究等，これらに関する関連研究を示す．

Zhang ら [7] は構文解析を，単語ごとに係り先を決定する Head Selection 問題として行っている．BiLSTM からの特徴量を用いることで，高精度の構文解析精度を出している．

BERT を使った日本語構文解析の関連研究として柴田ら [3] の研究がある．柴田らは事前学習済み BERT を京都大学テキストコーパスと京都大学ウェブ文章リードコーパスという 2 種類のコーパスで fine-tuning し，BERT による構文解析の精度を調査した．結果，BERT を利用した場合，KNP や CaboCha[*1]，BiLSTM を利用したときよりも高い精度を出している．宇田川ら [8] は，BERT からの出力ベクトルに加え，係り元と係り先の距離などを one-hot ベクトルで表現した基本素性を FFNN に入力し，構文解析を行っている．このとき BERT と基本素性を組み合わせた場合，CaboCha よりも高い精度となっている．

Sajjad ら [9] は，事前学習済み BERT の層を削除し，fine-tuning した場合，GLUE タスクにおいて 12 層の BERT に比べどの程度差が出るのかを調査した．このとき，

- 4 層減らすとき，下位の層を削除するのが一番スコアが低いが，他はあまり差がない．
- 6 層減らすとき，上位の層を削除するのが一番スコアがいい．

[*1] https://taku910.github.io/cabocha/

など，削除する層数や位置によってスコアに差が出ることを述べている．ただし構文情報を確認するデータセットにおいては，結果が安定していなかったため除外したと述べている．

本研究において簡易小型化 BERT を作成する際，Sajjad らのように事前学習済み BERT の層を削除する手法を利用したが，Gordon ら [10] は BERT の重みの値が 0 に近いものを不要と見なし取り除くという枝刈り手法を提案している．枝刈りの割合を 30∼40% にしても，枝刈りをしていないモデルに比べて，GLUE タスクにおいて大きな差は出ないと述べている．

BERT の軽量化は枝刈りだけでなく，蒸留を使用したモデルもある．Victor ら [11] は，BERT を教師モデル，DistilBERT を生徒モデルとして，教師モデルの出力の分布に合うように，DistilBERT を事前学習している．その結果，$\mathrm{BERT_{BASE}}$ よりもパラメータが 40% 少なく，60% 高速に動作し，GLUE で測定された BERT の 97% の性能を維持していると述べている．

BERT の層がどの情報を捉えているかについて，Sajjad らは上位層はタスクに特化した重みになっており，中間層は重要な情報を含んでおらず，下位層は文脈情報を含んでいると述べている．同じく Tenney ら [12] も，基本的な構文情報は BERT の早い段階で処理され，高レベルの意味的情報は より高い層に現れると述べている．

一方で，Jawahar ら [13] は Probing タスクにおいて，BERT が統語的な情報を中間層が捉えているという結果を出している．Hewitt ら [14] も依存構造解析において $\mathrm{BERT_{BASE}}$ は 6∼9 層，$\mathrm{BERT_{LARGE}}$ は 15∼17 層等，比較的中間に近い層が構文情報を捉えていると述べている．このように研究によって，BERT の捉え方に違いが出ているが，計測手法などの実験手順によるためだと考えられる．

Horne ら [15] は twitter の感情分析タスクで高い精度を達成することを目的に，GRUBERT と呼ばれる GRU ベースのアーキテクチャを提案している．GRUBERT を構築する際，BERT の隠れ層をいくつかのグループに分け，それぞれのグループ内でベクトルの結合を行い，それらのベクトルを実験に用いている．結果，BERT の 1∼6，7∼12 層を 2 つのグループに分けてそれぞれ結合したベクトル，また 1∼4，5∼8，9∼12 層の 3 つのグループに分けたものが一番精度が高いと述べている．

Wei ら [16] は，BERT の「主語と動詞の一致」の学習に頻度が与える影響を調査している．学習データに出現しない主語と動詞のペアに対しても一致ができるなど抽象的な

「文法」の学習を示唆する一方で，出現頻度も精度に大きく影響することを指摘した．

白ら [17] は，Amazon データセットを利用した領域適応の実験の際，BERT の最上位層のベクトルを用いるのでなく，その 1 つ下の階層のベクトルを用いている．結果，データセットの半数の領域適応では効果があったが，全体の平均でみると標準的な最上位層のベクトルを用いる手法よりも劣ったと述べている．

Yoav ら [18] は，BERT が文法タスクで高い精度を記録したと述べている．タスクとしては，文を与えて動詞が単数形/複数形になるかを予測している．

第 3 章

BERT

本章での BERT の解説は [19]，[20] を参考にした．

BERT(Bidirectional Encoder Representations from Transformers) とは以前言語処理のタスクのためのネットワークモデルに対する事前学習済みモデルである．BERT は文に対する単語列を埋め込み表現列に変換する．

Transformer は文中の各単語に対して文脈をよく考慮したベクトルを得る機構のことである．詳細は後述の項で説明する．

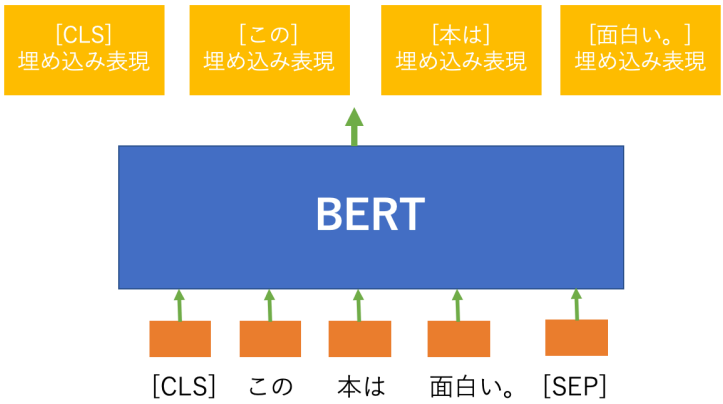

図 3.1: BERT からの埋め込み表現

図 3.1 では，BERT への入力とその出力関係を表している．BERT への入力は [CLS] トークンと呼ばれる文の始まりを表すトークンと，単語のトークン，文の終わりを表す [SEP] トークンである．ただし，各トークンは単語 id に変換する必要がある (後述)．入力後，BERT はそれぞれのトークンに対応する埋め込み表現を出力する．

BERT が出力する単語の埋め込み表現は文脈依存になっている．例えば「私は犬が好き.」の「犬」と「奴は警察の犬だ.」の「犬」は単語の意味が異なる．BERT は文脈依存の埋め込み表現を生成するため，この 2 つの「犬」に対する埋め込み表現は異なるものとなる．

本章では BERT のアーキテクチャから，BERT を使った構文解析，BERT の層を削除することによる簡易小型化について述べる．

3.1　アーキテクチャ

BERT では事前学習とファインチューニングの 2 ステップによってモデルの学習を行う．

事前学習では，以下の 2 つのタスクを行う．

- 文中の単語をランダムに隠し，周りの単語からその隠された (MASK された) 単語を予測する「穴埋め問題」.
- 選んだ 2 文が連続しているかどうかを当てる「次文予測問題」.

これら 2 つのタスクを Wikipedia コーパスなどの大規模なコーパスで BERT を事前学習させる．図 3.2 は事前学習で行うタスクの概要を示す．

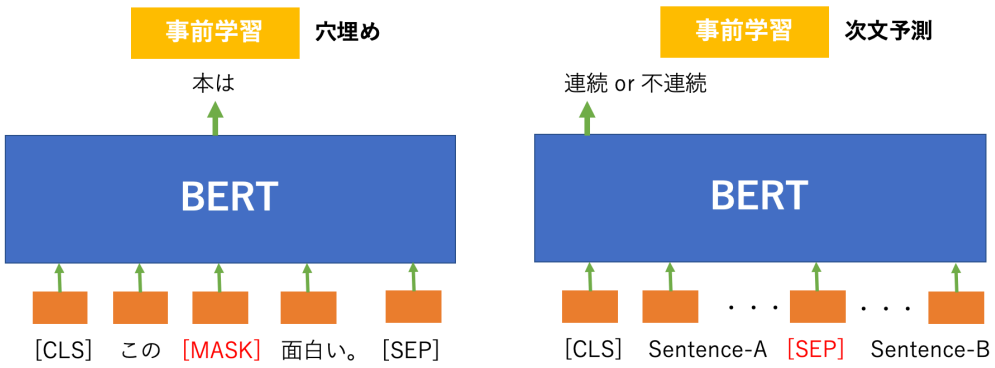

図 3.2: BERT の事前学習

事前学習済みの BERT が出来上がったあと，各々のタスクで fine-tuning を行う (図 3.3)．fine-tuning とは，事前に学習された重みを新しいネットワークの初期値の重みと

して設定し，再学習する方法のことである．

[CLS] に対応する BERT からの埋め込み表現は，その文を表す埋め込み表現となっている．そのため，その埋め込み表現を Linear などの層に代入し，postive/negative などの感情分析が可能になる．

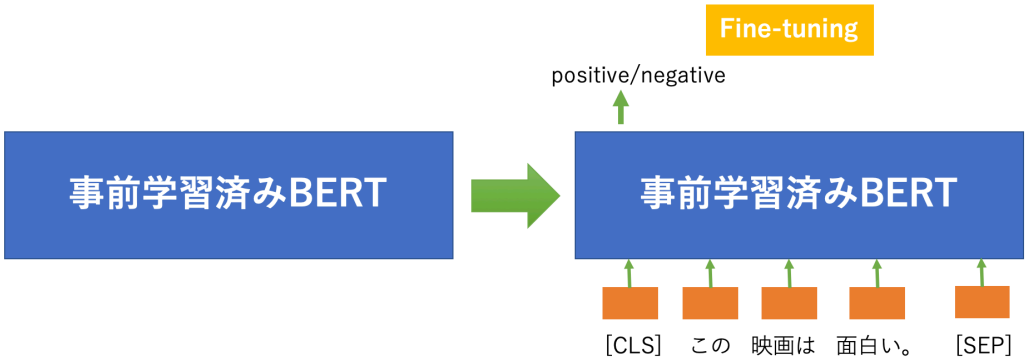

図 3.3: BERT の学習ステップ

BERT は $\mathrm{BERT_{BASE}}$ と $\mathrm{BERT_{LARGE}}$ の 2 種類ある．$\mathrm{BERT_{BASE}}$ は 12 層の Transformer のエンコード部分を連ねたもの (図 3.4)，$\mathrm{BERT_{LARGE}}$ は 24 層連ねたものである．

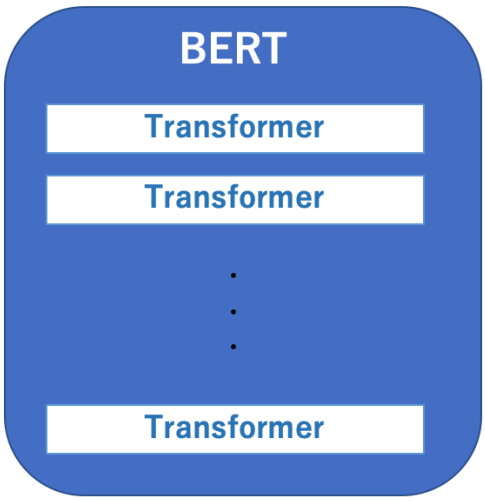

図 3.4: BERT の構造

3.1.1　Transformer

本項は芝山ら [21] の説明を参考にした．

Transformer は Attention ベースの Encoder-Decoder 型モデルである．RNN，CNN を一切使用せず，Attention のみで構成されているモデルであり，特徴的な構造として挙げられるのが，小規模の Attention を一定数作成・計算しそれらを結合することで Attention を計算する Multi-Head Attention である．また，並列化しやすく，RNN や CNN を採用しているモデルに比べ訓練時間が短いという特徴がある．

3.1.2　BERT での単語区切り

BERT を事前学習する際，コーパスをトークン区切りにし，単語を id に変換する必要がある．単語の区切り方や使用するコーパスによって事前学習済み BERT の種類も変わってくる．表 3.1 に日本語で事前学習された一部の BERT モデルの特性を示す．

表 3.1: 事前学習済み BERT の種類

公開名	区切り方	学習に使用するコーパス
京都大学 BERT	Juman++, BPE でサブワード化	日本語 Wikipedia
東北大学 BERT	MeCab	日本語 Wikipedia
NICT BERT	MeCab-Juman	日本語 Wikipedia

BERT の入力に関して，BERT に入力できるシーケンスの長さには上限があることに注意する必要がある．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．田中ら [22] は，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している．

3.2　構文解析

本研究で，BERT による構文解析を行うモデル図を図 3.5 に示す．まず，BERT に基本句区切りにしたトークンを subword に分割し入力する．次に注目したトークンより，

後ろのトークン全てに対して *score* を計算し，*score* を元に係り受けの有無を 2 値分類する．*score* の計算には，宇田川ら [8] の構文解析モデルを元に作成した．

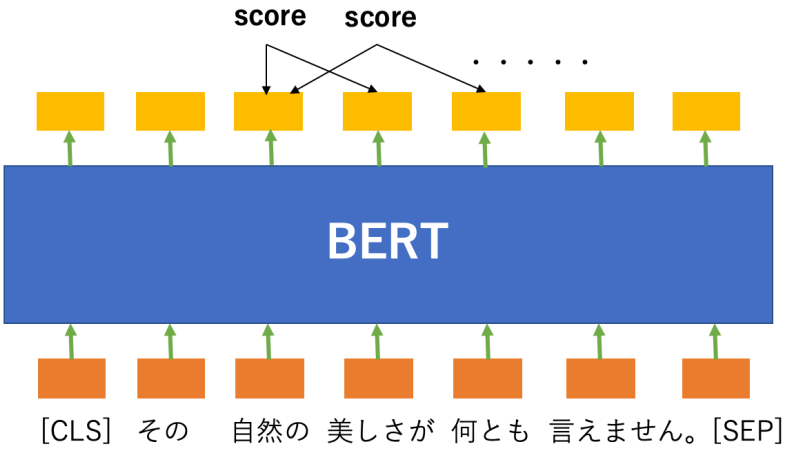

図 3.5: BERT による構文解析のモデル図

具体的に「私は茨城大学の学生です.」という文を本手法で構文解析した手順を述べる．まずこの文を基本句区切りにしたときの係り受け関係は図 3.6 のようになる．矢印の先が係り先で，例えば「私は」は「学生です.」に係っている．

図 3.6: 係り元と係り先の関係

本研究では単語の区切りに基本句区切りを採用した．基本句区切りは 1 つの自立語とそれに続く付属後からなる単位である．例えば「神社の参道は暗かった.」という文を基本句区切りにすると，「神社の｜参道は｜暗かった．」のように区切れる．

本研究では利用しなかったが，基本句区切りよりも細かい単位の区切りは形態素区切りとなる．形態素は意味をもつ表現要素の最小単位となっていて，先程の例文だと「神社｜の｜参道｜は｜暗かった｜.」のように区切れる．

本手法では，注目したトークンより後ろのトークン全てに対して係り受けの 2 値分類

を行うため，注目トークンとそれ以降のトークンに対して，係り受けラベルを付与する．表 3.2 において，「私は」のトークンに注目したときの係り受けラベルの付与について示す．このとき subword に分割されたトークンに対しては係り受けラベルは付与しない．

表 3.2: 係り先ラベルの設定

係り元トークン	係り先トークン	係り受けラベル
私は	茨城	0
私は	大学の	0
私は	学生です	1

次に *score* の計算方法について述べる．v_i と v_j をそれぞれ係り元，係り先の BERT からの出力ベクトル ($v_i, v_j \in \mathbb{R}^{768}$) だとすると，以下の式で score を出力する．

$$
\begin{aligned}
v_{ij} &= concat(v_i, v_j) \\
x &= sigmoid(Uv_{ij}) \\
score &= w^{\mathsf{T}}x
\end{aligned}
$$

このとき，U と w は fine-tuning する際，新しく追加したパラメータであり，それぞれ $U \in \mathbb{R}^{1000\times1536}$, $w \in \mathbb{R}^{1000}$ とする．

最後に，*score* を 2 値クロスエントロピーとして，係り受けの有無を fine-tuning で学習させる．

以下に，BERT から *score* を計算するアーキテクチャーを図 3.7 に示す．BERT からの係り元，係り先の出力ベクトルを v_i と v_j として *concat* する前に Dropout(p=0.35) を適応する．

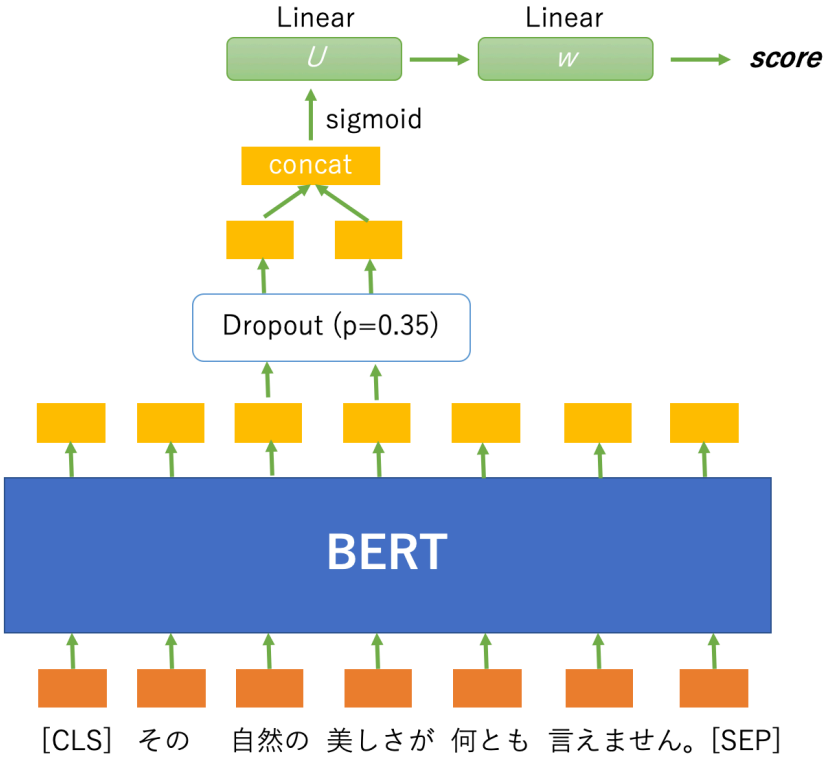

図 3.7: *score* 計算のアーキテクチャ

次節では，*score* の計算に用いた，U，w，そして Dropout などのニューラルネットの基礎ついて記す．

3.3　ニューラルネットワーク

ニューラルネットワークとは人間の脳の神経細胞の結合を表したものである．学習をすることで神経細胞の強度を変化させることによって出力を正解へと導き精度を高めていく．

ニューラルネットワークの学習には誤差逆伝播法という手法を用いる．誤差逆伝播法はニューラルネットワークの重みを学習させる際に使用するアルゴリズムである．ニューラルネットワークを順伝播させ，その最終的な出力の値と正解ラベルの値を比較し，その差を最小化するように学習を行う．

フィードフォワードニューラルネット (FFNN) は単純なニューラルネットワークモデルであり，信号の伝播が単一方向のモデルである．本研究においては，U, w での計算に利用している．また FFNN はループする結合を持たない．信号の伝播方向は入力層→中間層→出力層の順である．

FFNN の出力は以下の式で表される．

$$l^{(k+1)} = \sigma^{(k+1)l^{(k)}+b^{(k+1)}} \tag{3.1}$$

ここでの活性化関数を σ，n 層目の重み W^n，出力を l^n，バイアスを b^n で表されている．つまり現在の層の出力は，前の層の出力に重みをかけてバイアスを足したものである．

本研究で *score* を計算する際に利用した活性化関数は，シグモイド関数である．

図 3.8 はシグモイド関数の入力値に対する出力値の値の変化をグラフに表している．

$$\sigma(x) = \frac{1}{1+e^{ax}} \tag{3.2}$$

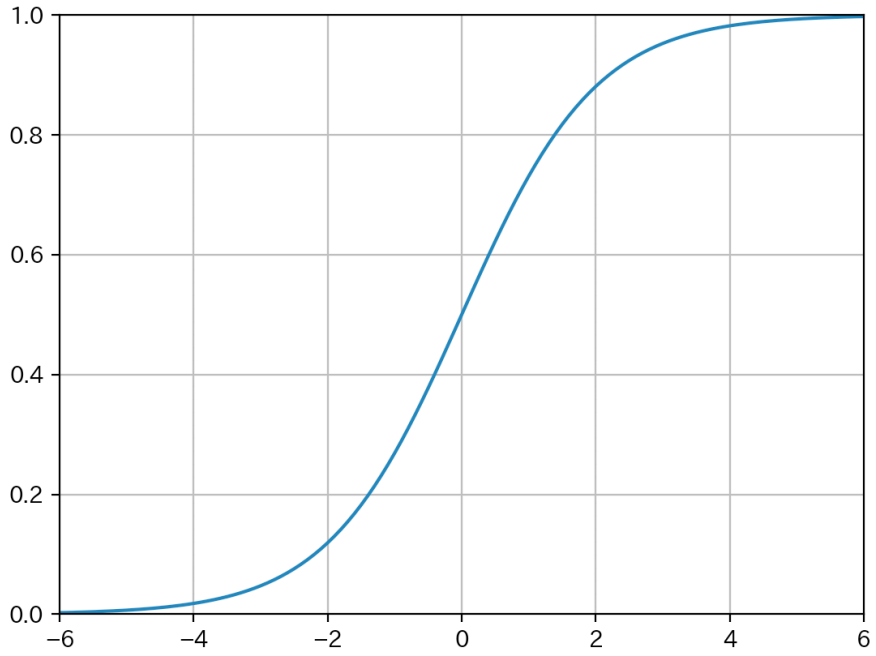

図 3.8: シグモイド関数の出力

Dropout は一定の確率でランダムにニューロンを無視して学習をすることによって過学習を防ぐものである．図 3.9 では 2 つのニューロンを無視し入力を受け付けていない状態である．

本研究においても，BERT からの出力ベクトルに対して Dropout を適応している．

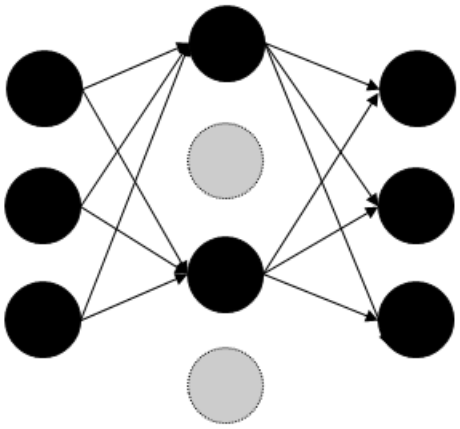

図 3.9: Dropout の形式図

BERT の実装だけでなく，これら Linear や Dropout は，ディープラーニング用フレームワークの PyTorch[*1] を利用することで実装することができる．

3.4　BERT の層削除

構文解析の精度と学習・推論速度の差を調査するため，京大版 BERT の一部層を削除したモデルを複数作成する．

図 3.10 に $\mathrm{BERT}_{\mathrm{BASE}}$ の位置関係を示す．BERT の入力に近い方を下位，出力に近い方を上位と呼ぶこととする．

[*1] https://pytorch.org/

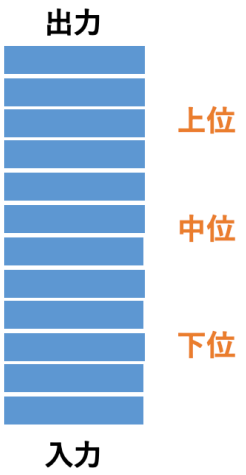

図 3.10: BERT の位置関係

図 3.11 は実験で使用するモデルの種類を表している．図 3.11 の灰色の層が削除した層を表しており，左から上位，中位，下位の位置を削除したものである．削除する層数はそれぞれの位置で 4，6，8 層とした．

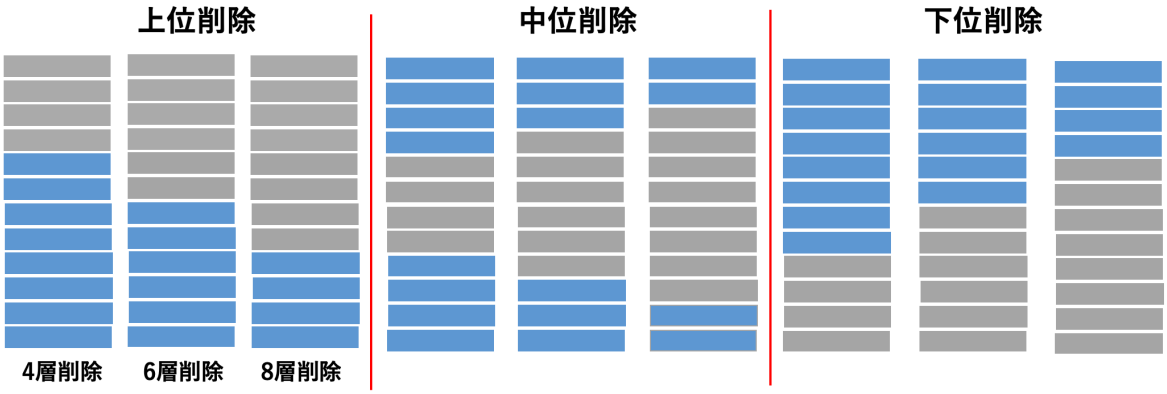

図 3.11: 削除する層の位置

層の削除方法について，事前学習済み BERT の上位 4 層を削除する具体的なソースを以下に示す．

```
1 from transformers import BertConfig, BertModel
2 import torch.nn as nn
3
4 config = BertConfig.from_json_file('/path/bert/Japanese_L-12_H-768_A-12
      _E-30_BPE_WWM_transformers/config.json')
5 model = BertModel.from_pretrained('/path/bert/Japanese_L-12_H-768_A-12
```

```
        _E-30_BPE_WWM_transformers/pytorch_model.bin', config=config)

model.encoder.layer = model.encoder.layer[:8]

model.config.num_hidden_layers = 4
model.save_pretrained("/path/remove_layers/")
```

transformers で BERT モデルを読み込み，model.encoder.layer に抜きたい層を指定している．そして最後に save_pretrained メソッドを利用して，層を削除した BERT モデルを保存する．

第 4 章

実験

本実験では，京都大学ウェブ文書リードコーパス (以降，Web コーパス)[*1]と京都大学テキストコーパス (以降，新聞コーパス)[*2]を混合したデータを用いて fine-tuning を行い，その後，それぞれのコーパスでテストをし，構文解析の精度を調査した．また fine-tuning 時には EarlyStopping を採用した．検証データの精度が最高値を更新しなくなってから 5 エポック後に停止させる．

4.1 使用データ

実験で使用するデータ数を表 4.1 に示す．学習と検証は Web コーパスと新聞コーパスを混合したものを利用するが，テスト時はそれぞれコーパスを分けてテストを行った．

ただし，新聞コーパスは Web コーパスに比べ，1 文のトークン数が長いものが多かったため fine-tuning では学習が収束しなかった．そのため，fine-tuning に用いる新聞コーパスのうち，1 文のトークン数が 24 個以下のもののみを使用した．テスト時のデータはトークン数で制限をかけていない．

表 4.1: 使用したデータ数

学習	Web・新聞 混合	10,000 件
検証	Web・新聞 混合	2,000 件
テスト	Web・新聞 別々	3,000 件

[*1] https://nlp.ist.i.kyoto-u.ac.jp/?KWDLC
[*2] https://github.com/ku-nlp/KyotoCorpus

Web コーパスはまざまなウェブ文書のリード (冒頭)3 文に各種言語情報を人手で付与したテキストコーパスである．ウェブ文書のリード 3 文を収集することによって，ニュース記事，百科事典記事，ブログ，商用ページなど多様なジャンル，文体の文書を含んでいる．

新聞コーパスは毎日新聞の記事に各種言語情報を人手で付与したテキストコーパスである．95 年 1 月 1 日から 17 日までの全記事，約 2 万文，1 月から 12 月までの社説記事，約 2 万文，計約 4 万文に対して，形態素・構文情報を付与している．

どちらのコーパスにも付与されている言語情報は，形態素解析システム JUMAN，構文解析システム KNP で自動解析を行い，その結果を人手で修正したものである．

4.2　fine-tuning

事前学習済み BERT モデルを fine-tuning する際，EarlyStopping で停止したときの各モデルの検証データでのスコアを表 4.2 に示す．損失関数はバイナリークロスエントロピー，最適化関数は SGD を使用した．前章で記述した通り，実装は PyTorch を使用した．

表 4.2: 各 BERT モデルの検証データでの最高精度

モデル名	停止エポック数	検証データの精度
$\mathrm{BERT}_{\mathrm{BASE}}$	6ep	36451.14
上位 4 層削除	3ep	62697.88
中位 4 層削除	6ep	40583.57
下位 4 層削除	8ep	31924.90
上位 6 層削除	4ep	46236.48
中位 6 層削除	6ep	34898.60
下位 6 層削除	9ep	94767.71
上位 8 層削除	12ep	22079.77
中位 8 層削除	9ep	28119.29
下位 8 層削除	5ep	43970.60

4.3　構文解析のテスト結果

各モデルの 2 つのコーパスでのテスト結果を図 4.1 に示す．通常 12 層の $\text{BERT}_{\text{BASE}}$ では Web コーパスで **94.64%**，新聞コーパスで **95.68%** となった．図 4.1 より，削除する層数を増やした場合でも，$\text{BERT}_{\text{BASE}}$ に比べ大きく精度が落ちていないことが分かった．また，削除する層数に関わらず，中位を削除したモデル (上位と下位を使用したとき) が高い精度を維持していることが分かる．

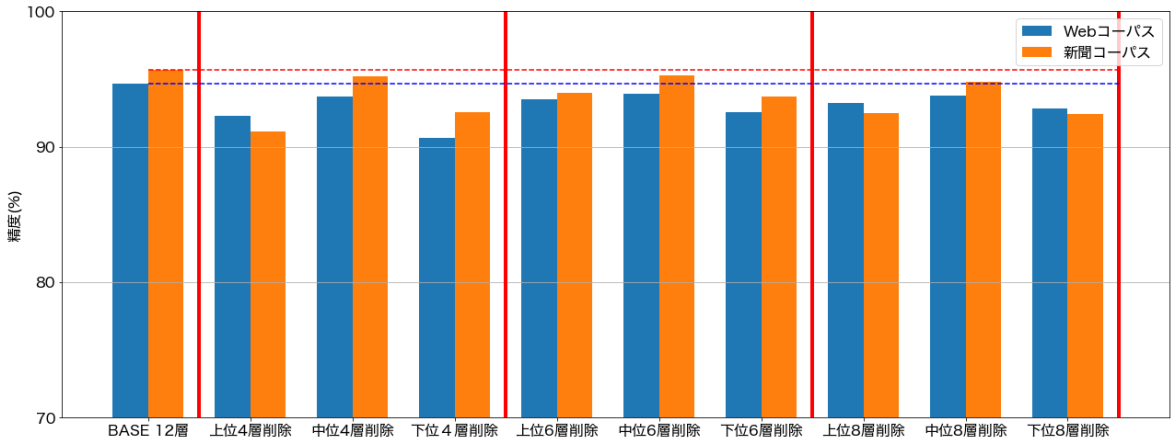

図 4.1: 各 BERT モデルのテスト結果

表 4.3 に各モデルのテスト結果の精度を数値で示す．

3~10 層目を削除した合計 4 層のモデルが，Web コーパスで 93.80%，新聞コーパスで 94.77% であり，$\text{BERT}_{\text{BASE}}$ からの精度劣化を Web コーパスで 0.84%，新聞コーパスで 0.91% に押さえる結果となった．

簡易化した BERT との比較だけでなく，KNP と word2vec との比較結果を図 4.2 に示す．図 4.2 の「下位 4 層削除」は先程の簡易化した BERT で Web コーパスの精度が一番低かったもの，「上位 4 層削除」は簡易化した BERT で新聞コーパスの精度が一番低かったものである．

word2vec は日本語 Wikipedia を基本区切りにして学習したモデルを利用した．学習には gensim を用いた．学習に利用した各パラメータを表 4.4 に示す．

KNP に比べ，$\text{BERT}_{\text{BASE}}$ の方が Web コーパスで約 5%，新聞コーパスで約 3% 精度が良い結果となった．BERT 使った構文解析では，v_i と v_j を BERT からの出力ベク

表 4.3: 各 BERT モデルのテスト結果

モデル名	テスト精度 (上段:Web コーパス 下段:新聞コーパス)
$\mathrm{BERT}_{\mathrm{BASE}}$	94.64% 95.68%
上位 4 層削除	94.64% 95.68%
中位 4 層削除	94.64% 95.68%
下位 4 層削除	94.64% 95.68%
上位 6 層削除	94.64% 95.68%
中位 6 層削除	94.64% 95.68%
下位 6 層削除	94.64% 95.68%
上位 8 層削除	94.64% 95.68%
中位 8 層削除	94.64% 95.68%
下位 8 層削除	94.64% 95.68%

トルとして構文解析を行ったが，この v_i と v_j を BERT からの出力ベクトルではなく，word2vec で得たベクトルを利用した場合，$\mathrm{BERT}_{\mathrm{BASE}}$ よりも精度がかなり落ちていることが分かる．

表 4.4: word2vec の学習パラメータ

パラメータ名	設定値
sg	1(skip-gram 利用)
vector_size	200
min_count	3
window	10

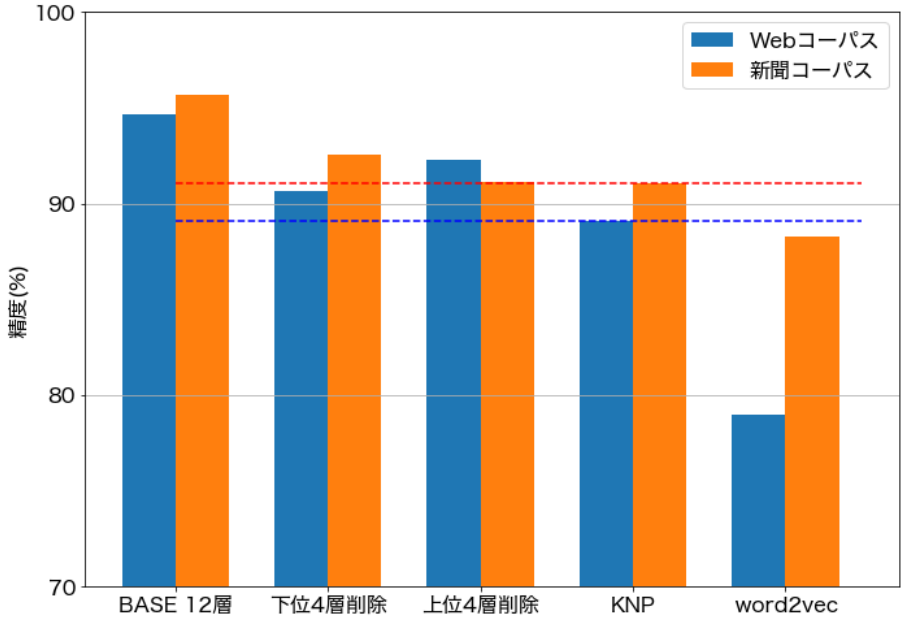

図 4.2: word2vec，KNP との比較結果

4.4　学習・推論時間の結果

次に学習・推論時間の結果について述べる．学習時間の結果は表 4.5 の通りである．EarlyStopping を採用したため，1 エポックの平均学習時間となっている．結果より削除する層数が増えるほど，学習時間は短くなっている．

$\text{BERT}_{\text{BASE}}$ は 1 エポックの平均学習時間が 3 分 56 秒，8 層削除の合計 4 層モデルでは学習時間は 3 分 16 秒となり，学習時間は 83% まで削減できた．

学習時間を折れ線グラフで表したものを図 4.3 に示す．$\text{BERT}_{\text{BASE}}$ から 4 層削除したときの削減時間は一番長く，それ以降 2 層ずつ削除すると約 10 秒ほど短くなっている．

表 4.5: 学習時間

	1ep の平均学習時間	$\text{BERT}_{\text{BASE}}$ との比較
$\text{BERT}_{\text{BASE}}$	3 分 56 秒	-
4 層削除	3 分 35 秒	-21 秒
6 層削除	3 分 27 秒	-32 秒
8 層削除	3 分 16 秒	-43 秒

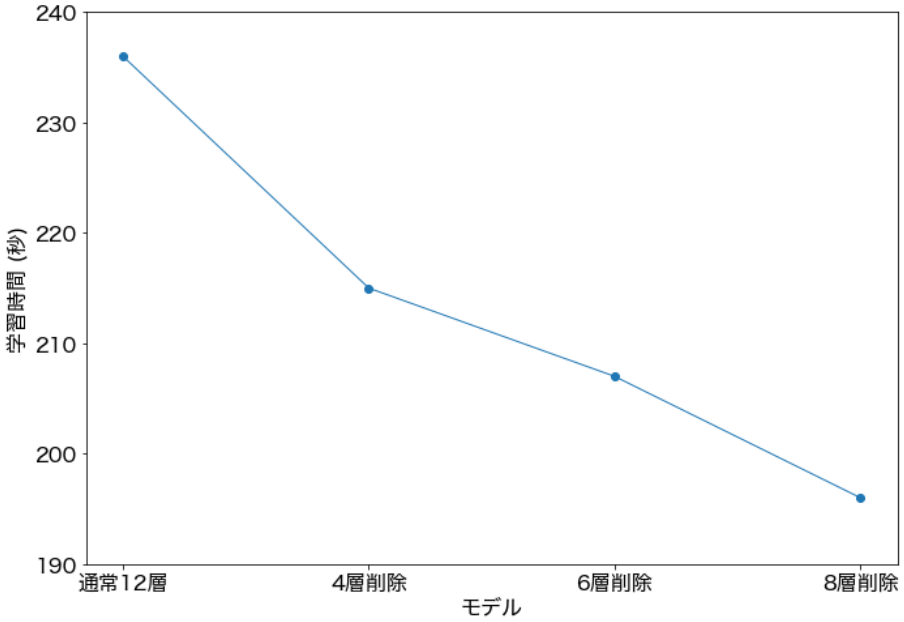

図 4.3: モデルごとの学習時間

推論時間の結果は表 4.6 の通りである．学習時間と同じく，推論時間に関しても削除する層数が増えるほど，推論時間は短くなっていることが分かる．この時，新聞コーパスが Web コーパスよりも推論時間が長いのは，新聞コーパスの方が 1 文の平均トークン数が多いためである．また，層を削除した場合，上位・中位・下位の 3 つのモデルの平均推論時間をとっている．

推論時間は $\text{BERT}_{\text{BASE}}$ のとき，Web コーパスで 37 秒，新聞コーパスで 79 秒となった．また 8 層削除モデルで Web コーパスで 24 秒，新聞コーパスで 67 秒となり，Web コーパスで 64%，新聞コーパスで 84% まで推論時間を削減することができた．

図 4.4 に各モデルの推論時間を折れ線グラフに示す．4 層削除，6 層削除，8 層削除と 2 層ずつ層を削除するとき，両コーパスともに約 2~4 秒ほどの削減されている．

表 4.6: 推論時間

	推論時間 (Web/新聞)	$\text{BERT}_{\text{BASE}}$ との比較
$\text{BERT}_{\text{BASE}}$	37 秒 / 79 秒	-
4 層削除	30 秒 / 73 秒	-7 秒 / -6 秒
6 層削除	27 秒 / 71 秒	-9 秒 / -8 秒
8 層削除	24 秒 / 67 秒	-12 秒 / -12 秒

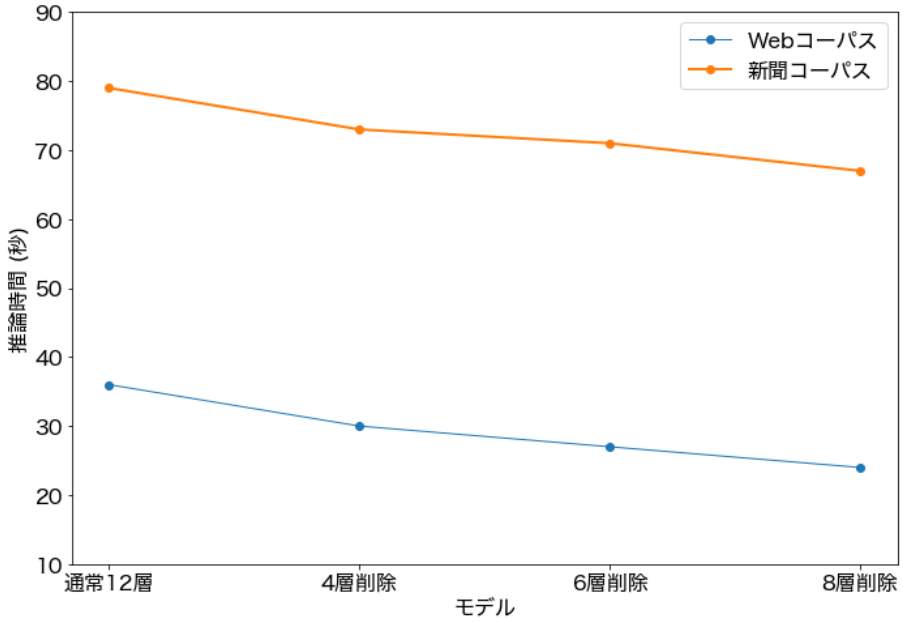

図 4.4: モデルごとの推論時間

第 5 章

考察

図 4.1 の各モデルのテスト結果より，上位層と下位層を利用したとき，構文解析において高い精度を維持しているため，上位層と下位層が構文情報を捉えていると考えられる．

関連研究では BERT において，Hassan ら [9]，Tenney ら [12] は構文情報は下位層を含んでいると述べているが，Jawahar ら [13]，Hewitt ら [14] は中間層が構文情報を捉えていると述べている．関連研究の章においても記述したが，研究によって BERT の構文情報の捉え方が違うのは，計算手法やタスクの違いのためだと考えられる．

5.1 各層ごとの精度差

本研究において，BERT のどの層が構文情報を捉えているのか，より細かく調査するため，$\mathrm{BERT}_{\mathrm{BASE}}$ の中で 11 層を削除し，1 層目から 12 層目まで 1 層ずつ fine-tuning し実験を行った．新聞コーパスでのテスト結果を図 5.1，Web コーパスでのテスト結果を図 5.2 に示す．図の縦軸が BERT で何番目の層を利用したかを表しており，縦軸一番下が BERT の 1 層目のみ使用，縦軸一番上が BERT の 12 層目のみを使用した結果である．横軸はテスト精度である．

新聞コーパスは下位の層，上位の層が，5 層から 8 層目の中間の層よりも精度がいいことがわかった．一方で Web コーパスに関してはどの位置の層を同程度の精度であった．

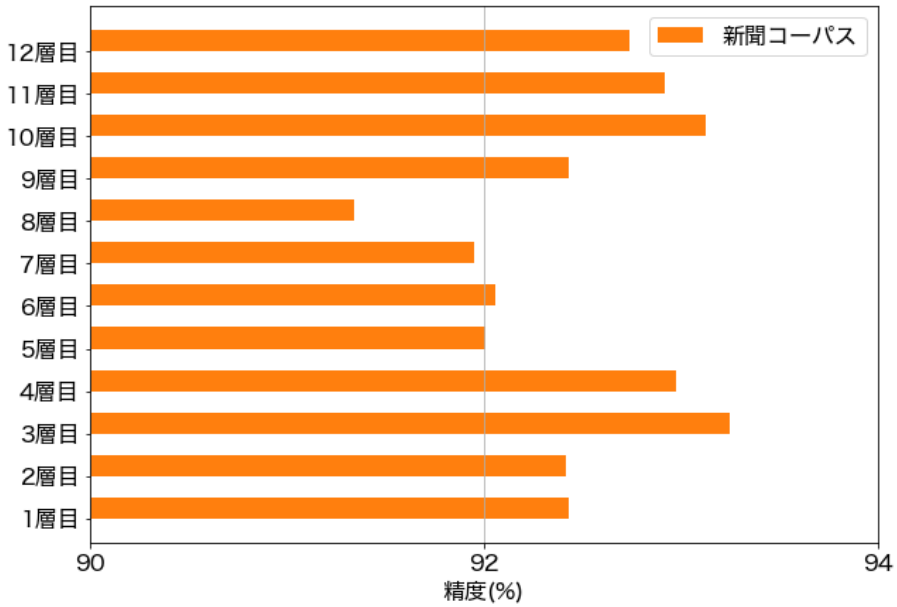

図 5.1: 新聞コーパスを 1 層ずつテスト

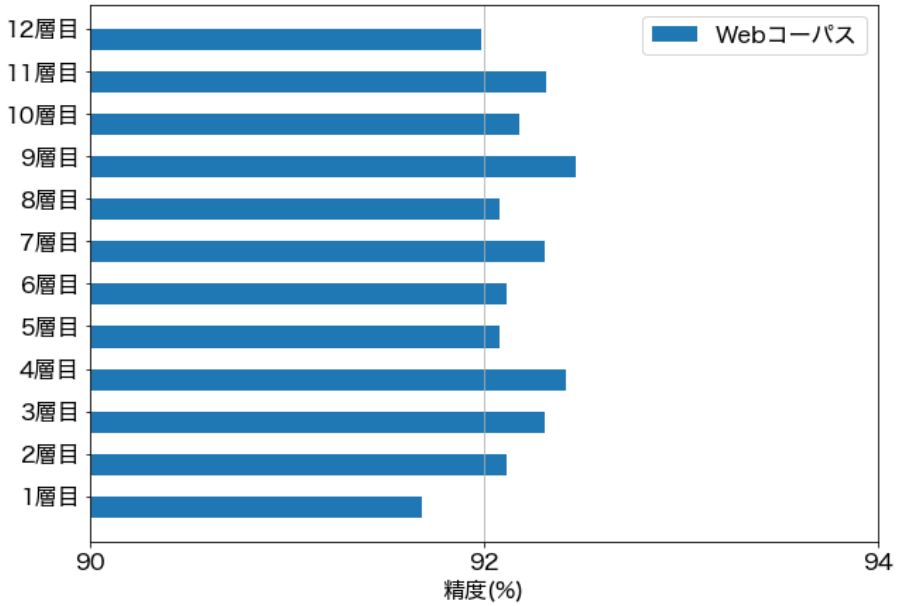

図 5.2: Web コーパスを 1 層ずつテスト

今回利用したコーパスの平均トークン数と 1 文の平均未知語数は表 5.1 に示す．新聞コーパスの方が Web コーパスに比べ，1 文の平均トークン数が長く，それに伴い平均未知語数を多くなっている．このことから，コーパスの種類や，1 文のトークン数と未知語数が多いことが，BERT が構文情報を捉える位置に影響を与えているのではないかと考えられる．

また，新聞は Web 上の日本語に比べ，大多数の読者に理解できるよう，比較的安易な文法構造を持って書かれているのではないかと予測できる．この推測も，新聞コーパスと Web コーパスで違いが出た 1 つの原因なのではないかと考えている．

表 5.1: 各テストコーパスの特徴

	Web コーパス	新聞コーパス
平均トークン数	14.85	23.55
1 文の平均未知語数	1.7	2.7

5.2　未知語と格による影響

コーパスによって精度の違いが出ることが分かったが，未知語の影響や層ごとによる構文情報の捉え方の違いを見つけるため，上位・中位・下位でそれぞれ 4 層ずつ層を削除したモデルで調査を行った．

5.2.1　未知語に対する扱い

層を削除することによって，BERT が未知語をどのように扱うのかの変化を調査するため，BERT が正解，不正解とみなしたペアの内訳を調べた．

図 5.3 は，テストデータにおいて係り受けラベルの 1 を付与したペアを，モデルが正解だと予測したものにおいて，そのペア中に未知語がどのぐらい含まれているのか割合を調査した結果である．

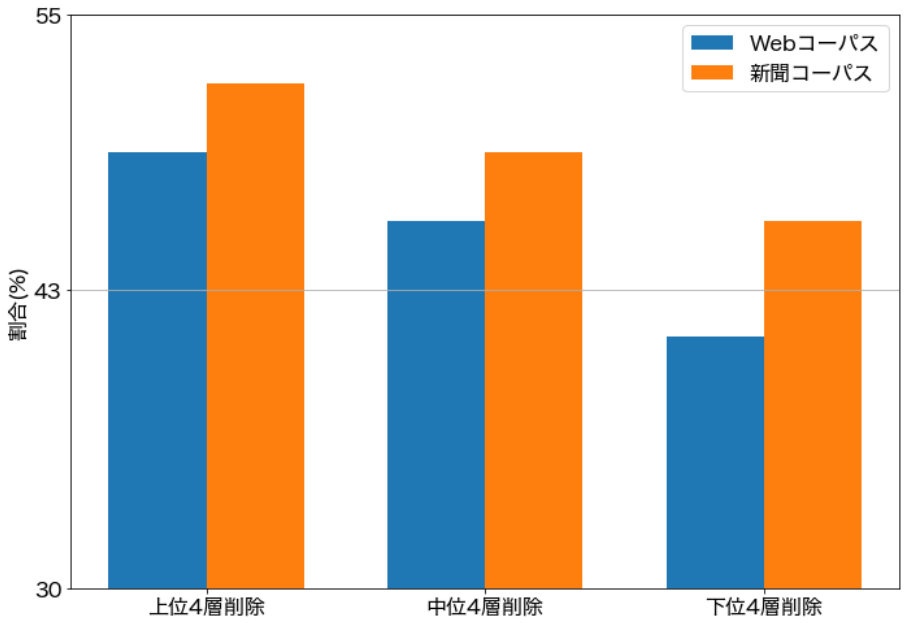

図 5.3: 正解ペアに含まれる未知語の割合

一方で，図 5.4 は，テストデータにおいて係り受けラベルの 1 を付与したペアを，モデルが不正解とみなしたものに対して，そのペア中に未知語がどのぐらい含まれているのか割合を調査した結果である．

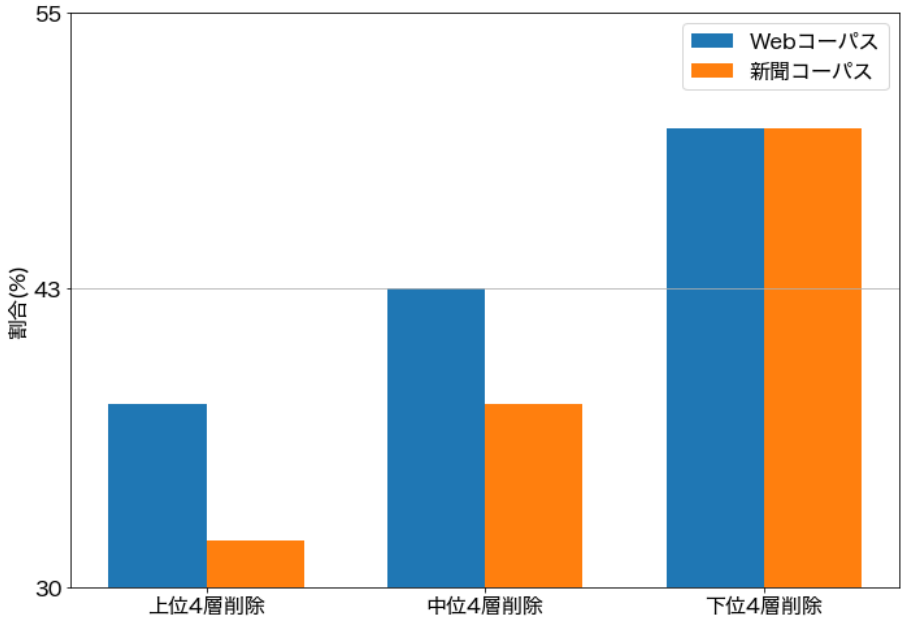

図 5.4: 不正解ペアに含まれる未知語の割合

図 5.3，5.4 より，上位層を削除するよりも，下位層を削除する方が，正解とみなしたペアの中での未知語の割合が低いことがわかる．つまり，BERT は下位層より上位層の

方が，語彙を捉えて構文解析を行っているといえる．

5.2.2　層ごとによる格パターン

次に各層を削除したことによる正解・不正解ペアの変化を調べた．まず表 5.2 では各モデルにおいて，正解したペアに含まれる格パターンの出現割合が多い順に上位 3 件を示す．これは，テストを行ったとき正解したペアを KNP を使って格解析した結果である．例えば「自然は」が「美しい.」に係っているペアを，モデルが正解とみなしたとき，「自然は美しい.」を KNP で解析し，述語側に表示される格解析の結果を使用する．

結果，どのモデルにおいても，「ガ格」「ヲ格」「修飾」が上位を占めていた．

表 5.2: 正解したペアに含まれる格パターン (上位 3 件)

上位 4 層削除		中位 4 層削除		下位 4 層削除	
Web	新聞	Web	新聞	Web	新聞
ガ格	ガ格	ガ格	ガ格	ガ格	ガ格
修飾	ヲ格	修飾	ヲ格	修飾	ヲ格
ヲ格	修飾	ヲ格	修飾	ヲ格	修飾

一方で，表 5.3 はテストの際，不正解とみなされたペアに含まれる格パターンである．表 5.3 でも先程の表 5.2 と同じように，「ガ格」「ヲ格」「修飾」が上位を占める結果となった．「ガ格」「ヲ格」「修飾」に分類された例は表 5.4 に示している．

表 5.3: 不正解だったペアに含まれる格パターン (上位 3 件)

上位 4 層削除		中位 4 層削除		下位 4 層削除	
Web	新聞	Web	新聞	Web	新聞
ガ格	ガ格	ガ格	ガ格	ガ格	ガ格
修飾	ヲ格	修飾	修飾	修飾	ヲ格
ヲ格	修飾	ヲ格	ヲ格	ヲ格	修飾

表 5.2，5.3 より，正解・不正解のどちらにおいても，層ごとに格に関しては差が出な

表 5.4: 格の例

格	例
ガ格	自然は美しい, 私は学生です
ヲ格	水買ってきて, 日記書く
修飾	とても寒い, 一番少ない

かった.

以上のことから，BERT は層ごとで格について大きな違いがなかったが，下位層よりも上位層の方が，既知の語彙を考慮した構文解析を行っているのではないかと考えられる.

5.3　今後の課題

今後の研究課題としては，以下の点が挙げられる.

- コーパスを基本句区切りではなく，形態素の区切りを使うことや，MeCab の区切りを利用した事前学習済み BERT を使うことなど，それぞれの区切りごとによって，構文解析の精度差があるのかを調べる．柴田ら [3] は BERT で構文解析した際，基本句よりも形態素区切りを利用した方が 2,3% 精度がよくなっていると述べている．BERT の層を削除した場合にも，形態素区切りが基本句区切りよりも有効に働くのか調査する.
- $\mathrm{BERT_{BASE}}$ では上位と下位を利用することが今回の構文解析では有効であったが，$\mathrm{BERT_{LARGE}}$ においても違いが出るのか.
- 本研究での学習は fine-tuning を 1 回行い，テストをしている．しかし，11 層削除モデルでもテスト精度の揺れが大きかった．そのため，fine-tuning を数回行った後にテストをすることや，交差検証を行うことで，より正確な結果が出るのではないかと考える.
- 本研究では BERT が正解/不正解とみなしたペアの格パターンを調査したが，変化が出なかった．そこで，格パターンだけでなく，BCCWJ で利用されている品詞パターンなどを用いることで，詳細な構文情報を調べる.
  例えば「警察 + メディア」というペアがあったとき，BCCWJ に付与されている品詞パターンは「名詞-普通名詞-一般」と「名詞-普通名詞-一般」がそれ

ぞれのトークンに付与されている．このパターンを集計し，BERT の層ごとの違いを見つける．

第 6 章

結論

本研究では BERT の一部層を削除し，簡易化した BERT を用いることによる，日本語構文解析の精度の差を調べた．結果，$\text{BERT}_{\text{BASE}}$ が両コーパスともに精度が最も高いが，削除する層数が増えても，精度にあまり差がでなかった．このとき上位層と下位層を両方利用するモデルが高い精度を維持していることがわかった．また，削除する層数を増やすほど，学習・推論にかかる時間は短くなった．

BERT は格パターンについて，層ごとによって捉え方の変化はなかったが，未知語と語彙による捉え方は層ごとに変化が出た．BERT の上位層を削除するよりも，下位層を削除する方が，正解とみなしたペアの中での未知語の割合が低かった．よって，BERT は下位層より上位層の方が，語彙を捉えて構文解析を行っている結果となった．

今後は BERT による日本語構文情報の捉え方を調査し，より高精度な日本語構文解析を目指す．

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．本研究に携わっていただいた方々に，感謝申し上げます．

指導教員である新納浩幸教授には研究に関する知見や先行研究のご紹介など，研究を進めていくにあたって多大なご指導をいただきました．私は大学院 2 年生から新納先生にお世話になりましたが，この 1 年間で研究に関するプログラミングの知識習得や学外発表など，大学院 2 年生になっても多くの経験を積むことができました．

東京農工大の古宮嘉那子准教授には，私が大学院 1 年生まで研究のご指導をいただきました．今年度から研究テーマが変わっても，本研究に関するアドバイスをいただき，研究を進める上で大変参考になりました．

新納研究室の同期である田中君とは，今年度同じ研究室の部屋になり，サーバーの構築や，研究が詰まったときにアドバイスをもらい，大変助かりました．

12 月の自然言語処理研究会で若手奨励賞を受賞することができたのは，新納先生だけでなく，古宮先生，田中君からのご指摘もあってこそ受賞できたと感じており，大変感謝しております．

そして，卒業後も日立まで遊びに来てくれた元古宮研同期の高久君と、芝軒研の先輩方に感謝いたします．私が学部 4 年生の時に，802 で夜遅くまでお酒を飲みながら過ごしたあの一年は，研究以外にも多くの事を学ぶことができ，深く記憶に残っています．

最後に改めて，本研究に携わっていただいた方々に，感謝申し上げます．

参考文献

[1] 河原大輔, 黒橋禎夫. 自動構築した大規模格フレームに基づく構文・格解析の統合的確率モデル. 自然言語処理, Vol. 14, No. 4, pp. 67–81, 2007.

[2] 笹野遼平, 黒橋禎夫. 大規模格フレームを用いた識別モデルに基づく日本語ゼロ照応解析. 情報処理学会論文誌, Vol. 52, No. 12, pp. 3328–3337, 2011.

[3] 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, pp. 205–208, 3 2019.

[4] 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.

[5] 萩行正嗣, 河原大輔, 黒橋禎夫. 多様な文書の書き始めに対する意味関係タグ付きコーパスの構築とその分析. 研究報告音声言語情報処理, Vol. 7, pp. 1–8, 2012.

[6] S. KUROHASHI. Building a japanese parsed corpus while improving the parsing system. *Proceedings of the Natural Language Processing Pacific Rim Symposium, 1997*, 1997.

[7] Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 665–676, Valencia, Spain, April 2017. Association for Computational Linguistics.

[8] 宇田川忠朋, 久保大亮, 松崎拓也. BERT を用いた日本語係り受け解析の精度向上要因の分析. 人工知能学会第 35 回全国大会論文集, 6 2021.

[9] Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. On the effect of dropping layers of pre-trained transformer models, 2021.

[10] Mitchell A. Gordon, Kevin Duh, and Nicholas Andrews. Compressing bert: Studying the effects of weight pruning on transfer learning, 2020.

[11] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[12] Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline, 2019.

[13] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3651–3657, Florence, Italy, July 2019. Association for Computational Linguistics.

[14] John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. 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. 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[15] Leo Horne, Matthias Matti, Pouya Pourjafar, and Zuowen Wang. GRUBERT: A GRU-based method to fuse BERT hidden layers for Twitter sentiment analysis. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pp. 130–138, Suzhou, China, December 2020. Association for Computational Linguistics.

[16] Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. Frequency effects on syntactic rule learning in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 932–948, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[17] 白静, 田中裕隆, 曹鋭, 馬ブン, 新納浩幸. BERT の下位階層の単語埋め込み表現列を用いた感情分析の教師なし領域適応. 情報処理学会自然言語処理研究会, No. 17, jun

2019.

[18] Yoav Goldberg. Assessing bert's syntactic abilities, 2019.

[19] 新納浩幸. PyTorch 自然言語処理プログラミング. インプレス, 2021.

[20] 柴田知秀. ヤフーにおける自然言語処理モデル BERT の利用, 2021. https://techblog.yahoo.co.jp/entry/2021122030233811/.

[21] 芝山直希. ラベル付き文集合を用いた事前学習済み BERT モデルの評価. 茨城大学大学院 理工学研究科 情報工学専攻 (未公刊), 2 2021.

[22] 田中裕隆, 曹鋭, 白静, 馬ブン, 新納浩幸. BERT を利用した文書の特徴ベクトルの作成. 情報処理学会自然言語処理研究会, No. 8, nov 2019.