# 中日ニューラル機械翻訳における言語モデルからの転移学習

15173007　中澤 真人　　指導教員　小町 守 准教授

令和 2 年 2 月 7 日

**概 要**

近年、自然言語処理の様々なタスクにおいて、事前学習した言語モデルからの転移学習を行うと精度が向上することが報告されており、言語モデリングに関する研究が盛んである。また機械翻訳においてもこの転移学習を用いた場合に精度が向上することを報告する論文もある。しかし、この Lample ら (2019) の 論文では既存の手法との比較実験がないなど不十分なところが多い。また、アジアの言語での機械翻訳の研究は英語と比べて少ない。そのため本研究では、中日翻訳において事前学習した言語モデルから機械翻訳への転移学習を複数の実験設定で行った。

## 1　はじめに

グローバル化の進展に伴って、機械翻訳の需要は高まっている。日本でも海外からの観光客数は 2018 年には 3,000 万人を突破 [1] していて、東京オリンピックに向けてさらに訪日観光客が増加することが予想されており、訪日観光客とのコミュニケーションで用いられる翻訳ツール [2] などに用いられている技術である機械翻訳の研究は欠かせないものとなっている。

近年の機械翻訳に関する研究の主流はニューラル機械翻訳である。ニューラル機械翻訳は対訳データという、翻訳元と翻訳先で対になった教師データから学習を行うが、その前に人手で情報が付与されていない生コーパスと呼ばれているもので事前学習をするという研究がある。この事前学習にはいくつかの方法があり、単語を表現するベクトルの事前学習 [3, 4, 5] は数多く研究されているが、言語モデルを用いた事前学習 [6] に関する研究はほとんどされていない。今回の研究では、言語モデルによる事前学習に焦点を当てる。

言語モデルとは、文の品詞や構造、単語列の生成確率や、単語列どうしの関係などについて定式化して学習したもののことである。また、定式化したタスクを言語モデリングという。言語モデリングは近年、別のタスクへの転移学習を行うための事前学習として注目されており、多くの研究が行われている [7, 8, 6, 9, 10, 11, 12]。その中でも転移学習先をニューラル機械翻訳とする研究は少数であり [6]、ほとんどの研究では転移学習先として自然言語推論 [13, 14] や言い換えタスク [15]、固有表現抽出 [16]、質問応答 (SQuAD)

[17] などを用いている。これらのタスクはニューラル機械翻訳とは性質が異なり、典型的には系列を入力して離散値や実数値を出力したり、入力の一部分を出力したりする。一方、ニューラル機械翻訳は系列を入力して全く別の系列を出力する。

今回の研究では、言語モデルの転移学習先をニューラル機械翻訳とした Lample らの研究 [6] に注目した。この研究は提案手法と既存の手法との比較実験がないなど不十分な点がある。またアジアの言語対での研究は、英語が翻訳元や翻訳先になる場合と比べてかなり少ない。そこで、本研究では Lample らの手法 [6] を中日に適用し、提案手法と既存の手法を含む複数の実験設定で結果を比較する。

# 2 関連研究

## 2.1 ニューラル翻訳

### 2.1.1 RNN のみを用いたモデル

ニューラルネットワークを用いたモデルが統計的なモデルに遜色ない精度に初めてなったのは、再帰ニューラルネットワーク (Recurrent Neural Network: RNN) を用いた Sutskever らの研究である [18]。このモデルは、符号化器という入力を処理する部分と、復号器という出力を生成する部分で構成されており、それぞれ RNN が用いられている。翻訳元の言語（原言語）を 1 トークン[1] ずつ符号化器に入力してできた 1 つの隠れ状態を元に、復号器で翻訳先の言語（目的言語）を 1 トークンずつ出力する。しかし、この研究の実験結果 [18] では入力文が長いと文の始めの方の情報が失われやすくなることが分かっている。

### 2.1.2 注意機構を用いたモデル

Sutskever らの研究 [18] における、入力文が長いと文の始めの方の情報が失われやすくなる問題を解決するために、Bahdanau らの研究 [19] では、復号器で目的言語を 1 トークンずつ出力するときに、符号化器から出力された原言語の 1 トークンごとの情報に直接エッジを張っている。この入力の情報に直接張ったエッジを注意と呼び、注意を用いたモデルを注意機構と呼ぶ。また、この研究では Sutskever ら [18] と同様に符号化器と復号器に RNN を用いているが、符号化器には双方向 RNN を用いている。双方向 RNN は単方向 RNN に比べて文の始めの方の情報が失われにくい。

### 2.1.3 注意機構のみで構成されたモデル (Transformer)

Sutskever らの研究 [18] と Bahdanau らの研究 [19] のいずれにおいても RNN が用いられているが、系列長を $n$ とすると RNN は任意の入力位置と出力位置を結ぶパスの最大経路長は $\mathcal{O}(n)$ である。一方、注意機構の場合は $\mathcal{O}(1)$ となる。そのため、注意機構の方が長距離依存を考慮しやすいと Vaswani ら [20] は述べている。この主張にのっとって、

[1] 単語よりもさらに細かい単位にまで分解することがあり、それをトークンという。

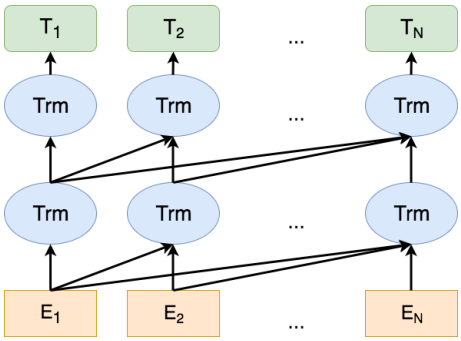


図 1: GPT の概略図

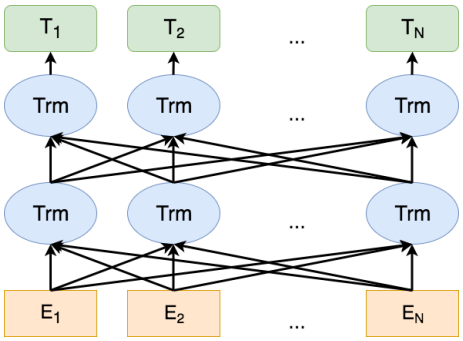


図 2: BERT の概略図

Vaswani ら [20] は Transformer と呼ばれる符号化器と復号器共に注意機構のみで構成されたニューラルネットワークを提案した。

Transformer は RNN とは異なり、構造的にトークン列の順番の情報を持たないため、それを追加する必要がある。そのため位置エンコーディング $PE$ を要素ごとに加算する。$PE$ の各成分は式 (1)、式 (2) で計算される。

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}}) \tag{1}$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}}) \tag{2}$$

式 (1)、式 (2) の $pos$ は単語の位置、$i$ は成分の次元、$d_{model}$ は隠れ層の次元数である。

## 2.2 言語モデリング

別のタスクへの転移学習を前提として言語モデリングを学習する先駆けとなった 2 つの研究と今回使用するモデルを含む XLM [6] を紹介する。

### 2.2.1 GPT

Radford らの研究 [7] (GPT) で用いられたモデルの概略図を図 1 に示す。図 1 の “$E$” は入力するトークン列、“$Trm$” は Transformer の構成要素、“$T$” は出力である。この図から分かるように、トークン列の左側から右側への接続のみがある。

GPT は入力するトークン列を $U = \{u_1, ..., u_n\}$ とすると、以下の式 (3) を最大化するように学習する。

$$L(U) = \sum_i \log P(u_i | u_{i-k}, ..., u_{i-1}; \Theta) \tag{3}$$

この式で、$k$ は考慮する文脈の幅、条件付き確率 $P$ はニューラルネットワークのパラメータ $\Theta$ によって定められる値である。

### 2.2.2 BERT

Devlin らの研究 [8] (BERT) で用いられたモデルの概略図を図 2 に示す。この図からも分かるように、BERT はトークン列の左右両側からの接続がある。BERT では GPT とは異なり、マスク言語モデルと文同士の関係を学習するための後続文予測の 2 種類を用いて学習する。

マスク言語モデルでは、以下の手順を行った後に元のトークンを予測するように学習する。

1. 15% の確率でトークンを選ぶ。そして選ばれたトークンのうち
2. 80% の確率で [MASK] トークンに置き換える。
3. 10% の確率でランダムな単語に置き換える。
4. 10% の確率で、単語を変えずにそのままにする。

後続文予測では正例として「連続する 2 文」、負例として「正例と同じ 1 文目とランダムな 2 文目の 2 文」を用い、正例と負例を分類するように学習する。図 2 の [CLS] の部分で [isNext] と予測している部分がそれである。

また、BERT では Vaswani らの研究 [20] とは異なる特徴量を用いる。図 3 に BERT の特徴量を示す。BERT では Vaswani らの研究 [20] で用いられた「位置エンコーディング」に加えて、「区切りエンコーディング」という処理を行う。これは、「位置エンコーディング」において用いられるのと同じ式 (1, 2) で行われるが、「区切りエンコーディング」では $pos$ は単語の位置ではなく、文の位置となる。そして「トークン特徴量」、「位置特徴量」、「区切りエンコーディング」で得られた「区切り特徴量」を足しあわせたものを機構に入力する特徴量とする。

### 2.2.3 XLM

Lample らの研究 [6] は、BERT と異なる特徴量を用いて、学習には BERT における「マスク言語モデル」と「後続文予測」のうちの「マスク言語モデル」のみを行ったものである。図 4 に XLM の特徴量について示す。

XLM には 3 つのモデル (CLM、MLM、TLM) があるが、XLM の全てのモデルにおいて BERT では「区切り特徴量」となっている部分が、「言語特徴量」という入力言語を識別するための特徴量となっている。

以下に XLM の 3 つのモデルの説明を箇条書きで示す。

**CLM** GPT と同じ式 (3) を最大化するように学習するため、GPT と同様に左側から右側への接続のみがある。

**MLM** BERT のマスク言語モデルと後続文予測のうち、マスク言語モデルのみを学習するモデルである。

**TLM** 学習自体は MLM と同じであるが、対訳データ（複数の言語で一方がもう一方の翻訳となっているデータ）を学習に用いる。

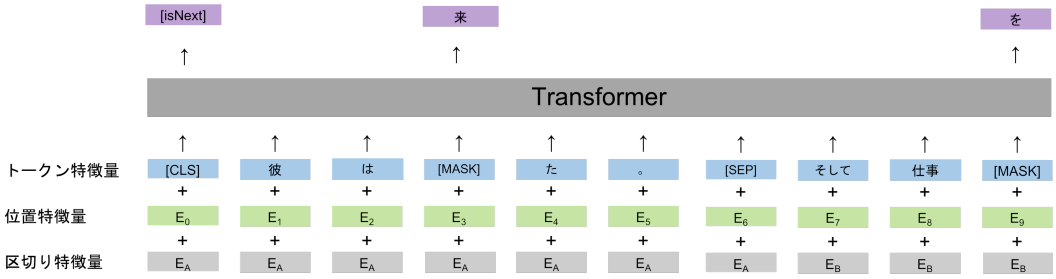


図 3: BERT の特徴量

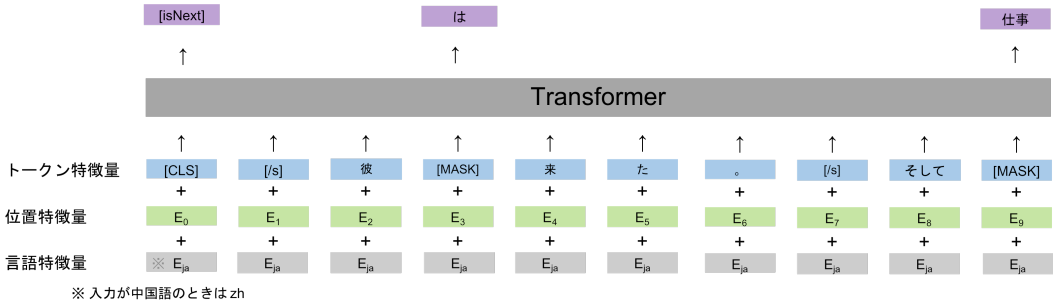


図 4: XLM の特徴量

# 3　言語モデリングから中日機械翻訳への転移学習

今回使用した言語モデリングと機械翻訳に用いたモデルは Transformer で構成されているため、2.1.3 項で述べていないモデルの詳細を 3.1 節で述べる。

## 3.1　Transformer

図 5 に Transformer の概略図を示す。

図の左側が符号化器、右側が復号器である。Transformer は翻訳をするためのモデルとして符号化器と復号器がセットで提案されていたが、言語モデリングなどにおいては、符号化器のみや復号器のみが用いられたりする。

RNN では 1 トークンずつ順番に処理していく構造になっているため、系列が長くなると計算に時間がかなりかかる。一方 Transformer はモデル自体が一般的な系列長よりも十分に長い構造となっているため、単語列を一度に処理できる。そのため、計算時間が短縮される。

Transformer を構成するものには「複数ヘッドの注意」、「マスク付き複数ヘッドの注意」、「加算 & 正規化」、「フィードフォワード」がある。「複数ヘッドの注意」は符号化器では「自己注意」、復号器では「ソースターゲット注意」というもので構成されている。

「自己注意」と「ソースターゲット注意」の出力は、入力と同じ数のノードであり、出

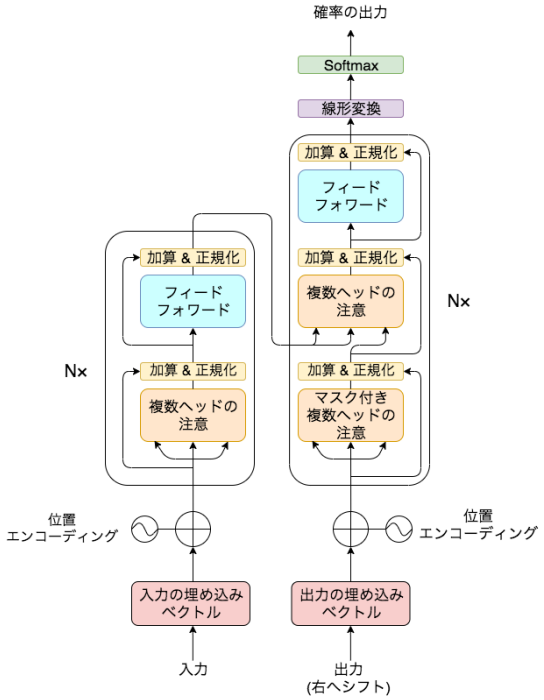


図 5: Transformer の概略図

力のそれぞれのノードは query、Key、Value という 3 つ要素を用いた式 (4) で計算する。

$$\text{softmax}(\text{query} \cdot \text{Key}^{\text{T}}) \cdot \text{Value} \tag{4}$$

この式 (4) で、query は 1 つのノードのベクトル、Key と Value は同じ層のノード列それぞれのベクトルを各列にもつ行列、$\text{Key}^{\text{T}}$ は Key の転置行列である。query と $\text{Key}^{\text{T}}$ の内積によって query と Key に含まれる各ノードとの関連度を計算し、その softmax をとって Value との内積をとることで、query との関連度に応じて Key を重みづけしてから足し合わせるという操作になる。「自己注意」の場合、query は 1 つ下のノード、Key と Value は 1 つ下のノード列全てからの出力である。「ソースターゲット注意」の場合、query は 1 つ下のノード、Key と Value は符号化器の出力ノード列全てである。

「複数ヘッドの注意」では「自己注意」または「ソースターゲット注意」の計算前により小さい次元に複数の線形写像を行う。そして、「自己注意」または「ソースターゲット注意」の計算後に元の次元に線形写像する。

「マスク付き複数ヘッドの注意」は、符号化器の「複数ヘッドの注意」においてトークン列内の後ろ側から前側への接続をなくしたものであり、復号器のみに用いられている。復号器は入力を元に翻訳を生成する部分であるため、学習時にはその答えを参照することができるが、推論時には答えは分からない。そのため、未来の情報が参照できないようにするためにこのようになっている。

「加算 & 正規化」は入力の加算と層正規化である。

「フィードフォワード」は注意が全く張られていない単純な一方向のニューラルネットワークである。

最適化には Adam [21] が用いられ、パラメータは $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$ となっている。

また、Transformer は局所解における目的関数の値が大域的な解における目的関数の値と大きく異なる場合が多いため、学習初期には学習率を下げておいて徐々に上げていくという操作を、式 (5) に従って行う。

$$l_{rate} = d_{model}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num^{-0.5} \cdot warmup\_steps^{-1.5}) \tag{5}$$

式 (5) の $step\_num$ は現在の $step$ 数、$warmup\_steps$ は学習率を徐々に上げていく操作が終わる $step$ 数で $warmup\_steps = 4000$ である。

## 3.2 言語モデリング

本研究では XLM の 3 つのモデルのうちの MLM を言語モデリングとして使用する。これを以後 XMLM (Cross-lingual Masked Language Modeling) と呼ぶ。また、XMLM から「言語特徴量」を除いたものについてのみ以後 MLM と呼ぶ。

中国語と日本語には「今日」など、文字と意味が全く同じトークンが数多くある。そのため中日で語彙を共有すると、そのようなトークンの翻訳時に単語ベクトルの変換が必要なくなる。しかし一方で、「切手」など同じ文字でも意味が全く異なるトークンも数多くある。このようなトークンでも、言語特徴量があると区別できるようになる。言語特徴量は 1 次元の情報であり、トークン特徴量より区別がしやすいため、語彙を共有しないよりも語彙を共有して同じ文字の単語を言語特徴量で区別するほうが、学習が容易であると予想した。

## 3.3 機械翻訳

言語モデリングの事前学習を行った後、そのパラメータを用いて機械翻訳を学習する。機械翻訳の概略図を図 6 に示す。なお、機械翻訳には評価手法として、BLEU [22] を用いた。

まず、3.3 節で用いる記号について説明する。入力系列中の $i$ 番目の要素を $x_i$、出力系列中の $j$ 番目の要素を $y_j$ とする。また、$x_i$ と $y_i$ は、それぞれ one-hot ベクトル（対応するトークンの要素のみ 1、それ以外は 0 の語彙数次元のベクトル）を仮定する。入力側の語彙を $V^{(s)}$、出力側の語彙を $V^{(t)}$ とすると、すべての $i$ に対して $x_i \in \mathbb{R}^{|V^{(s)}|}$、すべての $j$ に対して $y_j \in \mathbb{R}^{|V^{(t)}|}$ である。また入力文長を $I$、出力文長を $J$ とする。よって、

$$\text{（入力文）} X = (x_1, ..., x_i, ..., x_I) = (x_i)_{i=1}^{I} \tag{6}$$

$$\text{（出力文）} Y = (y_1, ..., y_j, ..., y_J) = (y_j)_{j=1}^{J} \tag{7}$$

となる。また、$y_0$ を文の開始を表す仮想トークン $BOS$ とする。

機械翻訳のモデルは以下の 5 つの構成要素から成り立っていると解釈することができる。

1. 符号化器埋め込み層

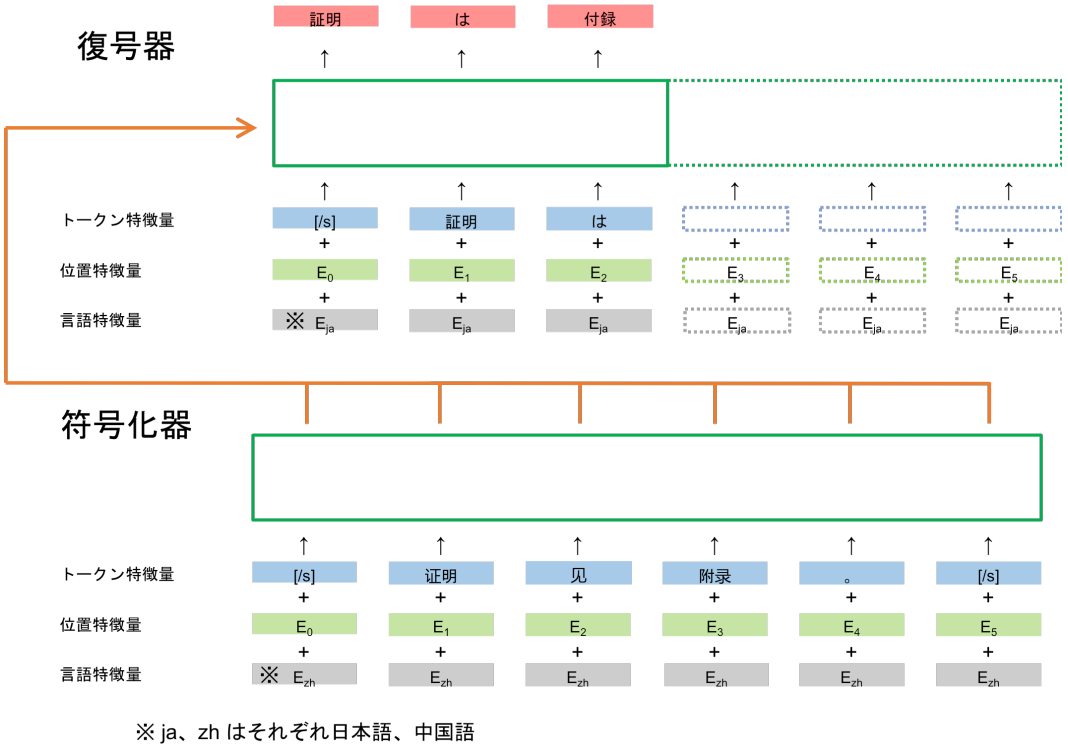


図 6: XLM を用いたニューラル中日翻訳

2. 符号化器隠れ層
3. 復号器埋め込み層
4. 復号器隠れ層
5. 復号器出力層

次項から機械翻訳の 5 つの構成要素を順番に説明する。

### 3.3.1 符号化器埋め込み層

まず、最初の処理として、入力文中の各トークンをベクトル表現に変換する処理を行う。入力文の位置 $i$ での符号化器埋め込み層の処理に対する入出力は以下のようになる。

- 入力：入力文中の $i$ 番目の単語を意味する one-hot ベクトル $x_i$
- 出力：入力文中の $i$ 番目に対応する埋め込みベクトル $\overline{x}_i$

この変換処理を、位置 $i$ に対して $i = 1$ から $i = I$ まで順番に（可能な場合は一括で）処理する。

個々の $x_i$ から $\overline{x}_i$ へ変換する処理は以下の式 (8) で表される。

$$\overline{x}_i = E^{(s)} x_i \quad \forall i \tag{8}$$

$E^{(s)} \in \mathbb{R}^{D \times |V^{(s)}|}$ は、埋め込み行列と呼ばれる。この処理によって $X$ は、$\overline{X} = (\overline{x}_1, ..., \overline{x}_I)$ に変換される。

### 3.3.2 符号化器隠れ層

符号化器隠れ層は、符号化器埋め込み層で得られた埋め込みベクトルのリストを用いて、有益な符号を生成する処理になる。入力文の位置 $i$ で符号化器隠れ層の入出力は以下のようになる。

- 入力：入力文中の $i$ 番目に対応する埋め込みベクトル $\overline{x}_i$（符号化器埋め込み層の出力に相当する）
- 出力：隠れ状態ベクトル $h_i^{(s)}$

この変換処理を、位置 $i$ に対して $i = 1$ から $i = I$ まで順番に（可能な場合は一括で）処理する。この処理によって隠れ状態ベクトル列、$H^{(s)} = (h_1^{(s)}, ..., h_I^{(s)})$ が得られる。また、どのように変換処理を行うのかはモデルに依存する。

### 3.3.3 復号器埋め込み層

復号器は、符号化器が作成したベクトル表現（符号化器の文脈では「符号」）を用いて、出力文を生成する処理過程に相当する。符号化器埋め込み層と本質的には同じ処理である。

ただし注意点として、符号化器埋め込み層の処理と微妙な違いがある。符号化器では、入力文全体が事前に与えられることを仮定するので $i$ に関して一括で処理できるのに対して、復号器では、各位置 $j$ に対して必ず $j = 1$ から、処理の終了信号を受け取るまで位置 $j$ を 1 つずつ増加しながら逐次処理を行うのが一般的である。これは、位置 $j$ の処理に、位置 $j - 1$ の処理結果（つまり出力単語）を利用する構造になっていることに起因する。ただし学習時には、$Y$ の情報は訓練データとして事前に与えられるので、符号化器と同様な一括処理が可能である。位置 $j$ での復号器埋め込み層の処理に対する入出力は以下のようになる。

- 入力：（後述する）復号器出力層で選択された出力 $y_{j-1}$
- 出力：埋め込みベクトル $\overline{y}_j$

復号器埋め込み層は、各復号器の処理位置 $j$ で、式 (9) の計算を行う。

$$\overline{y}_j = E^{(t)} y_{j-1} \tag{9}$$

符号化器埋め込み層 $E^{(s)}$ と同様に、$E^{(t)} \in \mathbb{R}^{\mathrm{D} \times |V^{(t)}|}$ は、復号器埋め込み層の埋め込み行列である。

### 3.3.4 復号器隠れ層

復号器の位置 $j$ での復号器埋め込み層の処理に対する入出力は以下のようになる。

- 入力：復号器埋め込み層の出力に対応する埋め込みベクトル $\overline{y}_j$（と、Transformer の場合は符号化器隠れ層の出力 $H^{(s)}$）
- 出力：隠れ状態ベクトル $h_j^{(t)}$

符号化器と復号器に 1 層単方向 RNN を用いた場合、$h_0^{(t)}$ を $h_I^{(s)}$ で初期化してからは復号化器隠れ層に $H^{(s)}$ を用いないが、注意機構のみで構成された Transformer の場合、全ての位置 $j$ で $H^{(s)}$ を用いる。

この変換処理を、位置 $j$ に対して $j = 1$ から $j = J$ まで順番に（可能な場合は一括で）処理する。この処理によって隠れ状態ベクトル列、$H^{(t)} = (h_1^{(t)}, ..., h_I^{(t)})$ が得られる。

### 3.3.5 復号器出力層

復号器の単語位置 $j$ での復号器出力層の処理に対する入出力は以下のようになる。

- 入力：位置 $j$ での復号器隠れ層の隠れ状態ベクトル $h_j^{(t)}$
- 出力：$y_j$ が生成される確率 $p_j$

出力層の計算は、学習時と推論時で処理が異なる。まず共通の処理として、確率計算の元となるスコアベクトル $o_j$ を式 (10) で計算する。

$$o_j = W^{(o)} h_j^{(t)} + b^{(o)} \tag{10}$$

$W^{(o)} \in \mathbb{R}^{|V^{(t)}| \times H}$ と $b^{(o)} \in \mathbb{R}^{|V^{(t)}|}$ は、出力層内の変換行列とバイアス項のベクトルである。

次に、学習の場合は、訓練データと現在のモデルが適合しているかを判断するために、確率計算の処理のほうを通常用いる。よって、$j$ 番目の単語 $y_j$ の生成確率を以下の式 (11) で計算する。

$$P_\theta(y_j | Y_{<j}) = softmax(o_j) \cdot y_j \tag{11}$$

ソフトマックス関数 $softmax(\cdot)$ は、各出力語彙のスコアをベクトル表現した $o_j$ から、各出力語彙の確率に変換する。その後 one-hot ベクトル $y_j$ との内積をとることで、$j$ 番目の単語の生成確率を得ることができる。

一方、未知の入力に対して出力系列を予測する場合は、以下の式 (12) を用いて単語を生成する。

$$\hat{y}_j = softmax_a(o_j) \tag{12}$$

ただし、

$$softmax_a(x) = \frac{1}{\exp(ax) \cdot 1} \exp(ax) \tag{13}$$

この式 (12) においてパラメータ $a$ を十分大きい値に設定した場合、ベクトル $o_j$ の中で最大の要素が 1 で、それ以外が 0 の one-hot ベクトルに近似する。よって、本質的に、単語を選択する処理と見なすことができる。$y_j$ は one-hot ベクトルとなるので、これを、次の処理位置の入力として、復号器埋め込み層の処理に戻る。このように、復号器埋め込み層、復号器隠れ層、復号器出力層の 3 種類の処理を終了信号を受け取るまで繰り返し処理を行うことになる。

#### 3.3.6 学習時の目的関数

機械翻訳の学習時の目的関数は、各対訳文対で以下の式 (14) を計算し、それらを総和したものである。

$$-\sum_{j=1}^{J}\log(P_\theta(y_j|Y_{<j})) = -\sum_{j=1}^{J}\log(softmax(o_j)\cdot y_j) \tag{14}$$

# 4 実験

## 4.1 実験データ

本研究では実験データとしてアジア学術論文抜粋コーパス (Asian Scientific Paper Excerpt Corpus: ASPEC) [23] と Wikipedia を中日それぞれで用いた。言語モデリングと機械翻訳で用いた学習データ、開発データ、テストデータの分割の文数を表 1 に示す。表 1 における言語モデリングに用いた学習データのうちの ASPEC については、機械翻訳に用いられた学習データと全く同じもので、開発データとテストデータには一切含まれていない。また、開発データは言語モデリングと機械翻訳で同じデータを用いており、言語モデリングにおけるモデル選択は perplexity [24] という指標を用いている。

日本語のデータは京都テキスト解析ツールキット KyTea [25]、中国語のデータは Stanford Word Segmenter [26] を用いてそれぞれ単語分割を行った。その後、BPE (Byte Pair Encoding) [27] という処理を行った。BPE は単語の出現頻度の辞書を元に以下の手順で行われ、得られた語彙の辞書が翻訳における語彙として用いられる。

1. 単語の出現頻度の辞書におけるそれぞれの単語を文字単位まで分解し、文字とその頻度を初期の語彙の辞書にする
2. 語彙の辞書に登録されているものが連続する頻度を単語の出現頻度の辞書を使ってカウントし、最も頻度の大きい連続する 2 つの語彙を新たに語彙の辞書に追加
3. 指定語彙数までそれを繰り返す

| | 言語モデリング | 機械翻訳 |
|---|---|---|
| 学習 | 5,000,000 文<br>（Wikipedia: 4,327,685 文+ASPEC: 672,315 文） | 672,315 文 (ASPEC) |
| 開発 | 2,090 文 (ASPEC) | 2,090 文 (ASPEC) |
| テスト | | 2,107 文 (ASPEC) |

表 1: 中日それぞれの実験データの文数。括弧内は用いたデータの種類を示す。

## 4.2 実験設定

今回は 6 種類のモデルを実験して比較した。まず、中国語と日本語の語彙を符号化器と復号器で共有する場合と、しない場合がある。前者に関する実験を 4 種類、後者に関する実験を 2 種類行った。

共有語彙の実験では、単純に翻訳だけをするモデルを Baseline とした。そして、言語モデリングを用いた学習後に翻訳器へ転移学習をする実験では 3 種類のモデルを用いている。1 つ目は XMLM を学習したパラメータで、翻訳器の符号化器と復号器（共通部分のみ） を初期化するもので XMLM と呼ぶことにする。2 つめは MLM の符号化器を学習したパラメータで、翻訳器の符号化器と復号器（共通部分のみ） を初期化するもので MLM_shared と呼ぶことにする。3 つめは MLM の符号化器を 中国語のデータと日本語のデータそれぞれで学習したパラメータで、翻訳器の符号化器と復号器（共通部分のみ）をそれぞれ初期化するもので MLM_separate と呼ぶことにする。

中日の語彙を共有しない実験では、単純に翻訳だけをするモデルを Baseline とした。そして、言語モデリングの学習では、中日のデータを混ぜて学習することができないため、MLM_separate のみを行っている。

使用したハイパーパラメータは 6 種類のモデルで共通である。Lample ら [6] が使用したものと同じものを使った。埋め込み層と隠れ層は 1,024 次元で、Baseline のパラメータは 0.5 から $-0.5$ の間のランダムな値で初期化した。また、dropout の係数は 0.1 で attention dropout の係数も 0.1 とした。レイヤの数は 6、ヘッドの数は 8、活性化関数は GELU [28]、最大文長は 256、学習率は 0.0001、1 エポックあたりの文数は 200,000 である。言語モデリングの学習時はバッチサイズを 32、翻訳の学習時は 1 バッチあたりのトークン数を 2,000 としている。

言語モデリングでは perplexity [24]、機械翻訳では BLEU [22] の値が開発データにおいて 10 エポック連続で改善しない場合には学習を終了し、10 エポック前のモデルで評価を行う。

## 4.3 実験結果

4.2 節で述べた 6 つのモデルについての実験結果 (BLEU) を表 2 に示す。まず、語彙は中日で共有したほうがいいことが分かる。次に中日共有語彙の XMLM と MLM を比較すると MLM の方がスコアが高く、「言語特徴量」はない方がいいことが分かる。また、MLM_separate の方が MLM_shared よりもスコアが高いことから、符号化器は中国語の

| | Baseline | XMLM | MLM_shared | MLM_separate |
|---|---|---|---|---|
| 中日共有語彙 | 43.65 | 45.03 | 46.75 | **47.26** |
| 中日別の語彙 | 42.31 | N/A | N/A | 45.96 |

表 2: 言語モデルから中日機械翻訳への転移学習の実験結果 (BLEU)

データのみ（語彙は中日共有）、復号器は日本語のデータのみ（語彙は中日共有）で学習した方がいいことが分かる。

中日共有語彙でも中日別の語彙でも、Baseline よりもそれ以外の方がスコアが高いことから、言語モデリングによる転移学習は機械翻訳においても有効なことが分かる。

# 5 考察

## 5.1 出力文の分析

中日共有語彙における Baseline と XMLM の出力文を比較する。機械翻訳特有の単語の繰り返しが Baseline では散見されるが、XMLM では見られなくなった。例文を 1 つ挙げておく。

**Baseline** ゆえに，ゆえに，植物多様性保護について，保護条件，保護用語，保護範囲，ゆえに，ゆえに，ゆえに，ゆえに，ゆえに，ゆえに，ゆえに，ゆえに，目標となる保護用語について議論した．

**XMLM** ＰＶＰ（植物多様性保全）については，保全条件，保全用語，保全範囲，ＰＶＰ の軽減について検討した．

**参照文** ＰＶＰ（植物多様性保護）に関しては，保護条件，保護用語，保護範囲，ＰＶＰ に対する免除，などについて議論を行った．

また、XMLM は言語モデリングによって学習したパラメータから翻訳の学習を開始しているからか、より参照訳に近い翻訳をする場合が多かった。以下に例を挙げる。

**Baseline** そして，その後，メモリ・プロセッサにコンパイルし，ＭＡ，ＳＡ を実行する．

**XMLM** そして，Ａｓｐｅｃｔ コンパイラでコンパイルし，ＭＡ，ＳＡ を実行する．

**参照文** そして ＡｓｐｅｃｔＪ コンパイラでコンパイルして，ＭＡ，ＳＡ を実行するという手順になる．

| | Baseline | XMLM | MLM_shared | MLM_separate |
|---|---|---|---|---|
| 中日共有語彙 | 1 日と 20 時間 | 15 日と 5 時間 | 13 日と 5 時間 | 23 日と 0 時間 |
| | 1 日と 20 時間 | 1 日と 6 時間 | 3 日と 4 時間 | 4 日と 3 時間 |
| 中日別の語彙 | 1 日と 12 時間 | N/A | N/A | 24 日と 14 時間 |
| | 1 日と 12 時間 | N/A | N/A | 3 日と 2 時間 |

表 3: 計算時間

### 5.2 計算時間

表 3 に計算時間を示す。中日共有語彙と中日別の語彙のそれぞれにおいて、1 行目が言語モデリングの学習時間を含めた場合、2 行目が言語モデリングの学習時間を含めない場合である。これらの実験は、MLM_shared における日本語データでの事前学習を除く[2] と TITAN X 1 枚で行った。そのため表 3 における MLM_shared はこれよりも時間がかかると予測される。

表 3 のそれぞれの語彙の 1 行目を見ると言語モデリングで事前学習する場合、計算時間が最大で 16 倍程度増えることが分かる。また MLM_separate は事前学習に中国語のみで学習するモデルと、日本語のみで学習するモデルの 2 つが必要なため、計算時間が XMLM と MLM_shared と比べて 2 倍近くになっている。

表 3 のそれぞれの語彙の 2 行目を見ると、XMLM を除くと学習した言語モデルから転移学習を行う場合、翻訳の学習時間もかなり長くなることが分かる。

## 6 おわりに

本研究では、言語モデリングから機械翻訳への転移学習において複数の実験設定での比較を行った。そして、言語モデリングでの事前学習を行うと翻訳の精度が上がることが分かった。また、中国語と日本語には共通する語彙が多く、語彙は共有した方がいいことも分かった。

言語モデリングによる事前学習は汎用性が高いが、大量のデータが必要となるのと計算量が増加する。今回の研究で機械翻訳においても言語モデリングによる事前学習は有用なことが分かったが、他の用途で言語モデルを学習する必要がない場合、言語モデリングの計算量がそのまま機械翻訳全体の計算量の増加になってしまう。そのため転移学習先を機械翻訳とする場合に、最適な言語モデリングに関する研究も必要であると考えている。

## 謝辞

卒業論文に限らず、何度もお世話になった指導教員の小町先生に感謝の意を表します。また、RA として研究の方向性や実験設定などについて、何度も議論してくださった嶋中

[2] メモリが足りなかったため、Tesla P40 で行った

宏希さんと黒澤道希さんにも謝意を表します。また、研究室の同期・先輩方には困ったときに相談にのってもらったりしていただきました。
皆様への感謝の意を表して、謝辞とさせていただきます。

# 参考文献

[1] 国籍/月別　訪日外客数 (2003 年〜2019 年). `https://www.jnto.go.jp/jpn/statistics/since2003_visitor_arrivals.pdf`.

[2] ポケトーク. `https://pocketalk.jp/`.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc., 2013.

[4] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[5] 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.

[6] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *CoRR*, 2019.

[7] Alec Radford and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, OpenAI, 2018.

[8] 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.

[9] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. *CoRR*, 2019.

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly optimized BERT pretraining approach. *CoRR*, 2019.

[11] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, 2019.

[12] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism. *CoRR*, 2019.

[13] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[14] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[15] William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, pp. 9–16, 2005.

[16] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning*, pp. 142–147, 2003.

[17] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[18] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pp. 3104–3112. Curran Associates, Inc., 2014.

[19] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2014. cite arxiv:1409.0473 ICLR 2015.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan,

and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980 ICLR 2015.

[22] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[23] Toshiaki Nakazawa, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. ASPEC: Asian scientific paper excerpt corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 2204–2208, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[24] Language Modeling; Introducton to N-grams. `https://web.stanford.edu/class/cs124/lec/languagemodeling.pdf`.

[25] Graham Neubig and Shinsuke Mori. Word-based partial annotation for efficient corpus construction. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, pp. 2723–2727, Valletta, Malta, May 2010. European Language Resources Association (ELRA).

[26] Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pp. 224–232, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[27] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[28] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *CoRR*, 2016.