同期注意制約を与えたTransformerによるニューラル機械翻訳

出口 祥之　　田村 晃裕　　二宮 崇
愛媛大学 大学院理工学研究科
{deguchi@ai., tamura@, ninomiya@}cs.ehime-u.ac.jp

1 はじめに

近年，自然言語処理の多くのタスクにおいて，ニューラルネットワークが活用されている．機械翻訳の分野においてもその有効性が示されており，その中でも，Transformer モデル [10] が再帰型ニューラルネットワーク（Recurrent Neural Network; RNN）や畳み込みニューラルネットワーク（Convolutional Neural Network; CNN）ベースのモデルの翻訳性能を上回り，注目を浴びている．特に，Transformer の特徴の一つである自己注意は文内における単語間の関連の強さを捉えることができ，係り受け関係などの文構造を捉えられていることを示す解析結果も報告されている [11]．さらに，係り受け解析結果を自己注意に対する制約として与えてモデルを訓練することで翻訳性能をさらに改善するモデルも提案されている [2]．

本稿で提案する**同期注意制約**は同期文法（Synchronous Grammar）から着想を得た手法であり，Transformer モデルに対してこの制約を与えて訓練することで翻訳性能の改善を狙う．同期文法は，2 言語で定義される文法であり，2 言語の文構造を同時に生成することができる．これまで，統計的機械翻訳では原言語文と目的言語文間の同期文法を考慮することで翻訳性能を改善してきた [5, 3]．本稿では，Transformer モデルの原言語側と目的言語側の自己注意間で，両側の注意の整合性を保つ制約を与えて訓練する．図 1 は対訳文に対するアライメント及び各文の係り受け構造の例を表している．提案する制約では，図 1 において，「I」が「like」を係っていて，「I」が「私は」に，「like」が「好き」に対応している時，「私は」が「好き」に係るような制約を与える．自己注意の言語間における対応関係は，Transformer モデルの言語間注意によって求める．言語間注意は，目的言語側から見た原言語側の各単語との関連を計算する．この言語間注意は，図 1 の青矢印のような単語のアライメント関係を捉えているという実験結果も報告されている [4]．

Transformer モデルの注意機構に文構造に関する制約を与えるという点で文献 [2] の手法が関連する．しかし，この手法は訓練時に制約の教師データを必要とするのに対して，提案手法は，訓練時にも推論時にも係り受け解析等の教師データを必要とせず，モデル内で計算された値のみから制約を与えられる点が異なる．

提案手法である制約付き Transformer モデルと従来の Transformer モデルの翻訳性能を BLEU [7] を用いた評価により比較したところ，WMT14 英独・独英翻訳タスクではそれぞれ 0.19 ポイント，0.14 ポイントの上昇，WAT Asian Scientific Paper Excerpt Corpus（ASPEC）日英・英日翻訳タスクでは 0.11 ポイント，0.25 ポイントの上昇を確認した．

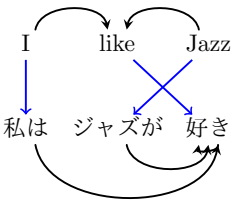

図 1: 対訳文の係り受け構造とアライメントの例

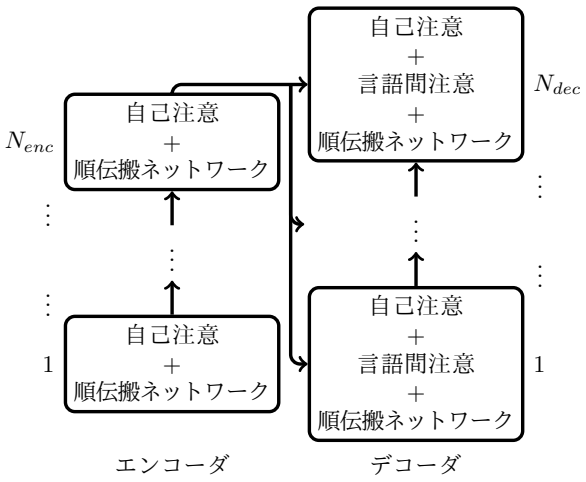

図 2: Transformer モデルの概要

2 Transformer モデル

本節では，提案モデルの基礎となる Transformer モデル [10] を説明する．Transformer モデルの概要を図 2 に示す．

Transformer モデルは，入力された原言語の単語列 $\boldsymbol{f} = (f_1, f_2, \ldots, f_I)^{\mathsf{T}}$ を符号化する Transformer エンコーダ（以下，エンコーダ）と，エンコーダの出力を受け取り目的言語の単語列 $\boldsymbol{e} = (e_1, e_2, \ldots, e_J)^{\mathsf{T}}$ に復号する Transformer デコーダ（以下，デコーダ）を組み合わせたエンコーダ・デコーダモデルである．エンコーダとデコーダはそれぞれ，図 2 のように N_{enc} 層のエンコーダ層と，N_{dec} 層のデコーダ層から構成される．

Transformer は RNN のような再帰構造を持たないため，入力文の各単語に対して並列に処理することができるが，そのままでは入力単語の位置関係を捉えることができない．そのため，Transformer は単語の位置情報を符号化する位置符号（Positional Encoding）を用いる．具体的には，単語埋め込みの次元数を d_{emb} とすると，単語の位置情報を符号化するための行列 PE は

$$PE_{(pos,2d)} = \sin(pos/10000^{2d/d_{emb}}), \quad (1)$$

$$PE_{(pos,2d+1)} = \cos(pos/10000^{2d/d_{emb}}), \quad (2)$$

と表される．なお，pos は単語の位置，d は埋め込み次元の各成分である．式（1），(2）によって計算された行列 PE を単語の埋め込み行列に加算した行列 Z を，エンコーダ及びデコーダの入力とする．

各エンコーダ層や各デコーダ層の入力・出力をそれぞれ X，Y，層正規化（Layer Normalization）[1] を $\mathrm{LN}(\cdot)$ とすると，各エンコーダ層は自己注意（$\mathrm{SelfAttn}(\cdot)$）と順伝搬ネットワーク（$\mathrm{FFN}(\cdot)$）により以下の通りの演算を行う．

$$H_{sa} = \mathrm{LN}(X + \mathrm{SelfAttn}(X)), \quad (3)$$

$$Y = \mathrm{LN}(H_{sa} + \mathrm{FFN}(H_{sa})). \quad (4)$$

ここで，各エンコーダ層への入力は直前のエンコーダ層の出力 Y であり，第 1 層目のエンコーダ層の入力は原言語文の埋め込み行列 Z である．H_{sa} は自己注意の出力である．

各デコーダ層は自己注意（$\mathrm{SelfAttn}(\cdot)$）と言語間注意（$\mathrm{CrossAttn}(\cdot,\cdot)$），順伝搬ネットワーク（$\mathrm{FFN}(\cdot)$）により以下の演算を行う．ただし，$Y_{enc}$ はエンコーダの最終層の出力とする．

$$H_{sa} = \mathrm{LN}(X + \mathrm{SelfAttn}(X)), \quad (5)$$

$$H_{ea} = \mathrm{LN}(H_{sa} + \mathrm{CrossAttn}(H_{sa}, Y_{enc})), \quad (6)$$

$$Y = \mathrm{LN}(H_{ea} + \mathrm{FFN}(H_{ea})). \quad (7)$$

ここで，各デコーダ層への入力は直前のデコーダ層の出力 Y であり，第 1 層目のデコーダ層の入力は目的言語文の埋め込み行列 Z である．H_{sa} は自己注意の出力，H_{ea} は言語間注意の出力である．

Y_{dec} をデコーダの最終層の出力とすると，Y_{dec} は V 次元の行列に線形変換される．ここで，V は出力の語彙数である．最後に，この V 次元行列に softmax 関数を適用することで $P(\boldsymbol{e} \mid \boldsymbol{f})$ が計算され，これに基づいて出力列 $\boldsymbol{e}$ が生成される．

自己注意は同一文中の単語間（原言語文中の単語間あるいは目的言語文中の単語間）の関連の強さを計算し，言語間注意は原言語文中の単語と目的言語文中の単語間の関連の強さを計算する．自己注意や言語間注意は複数ヘッド注意（$\mathrm{MHAttn}(Q, K, V)$）を用いて計算される．

$$\mathrm{SelfAttn}(Y) = \mathrm{MHAttn}(Y, Y, Y), \quad (8)$$

$$\mathrm{CrossAttn}(H_{sa}, Y_{enc}) = \mathrm{MHAttn}(H_{sa}, Y_{enc}, Y_{enc}). \quad (9)$$

複数ヘッド注意では，単語の埋め込み空間を N_{head} 個の $d_{head} = \frac{d_{emb}}{N_{head}}$ 次元部分空間に射影し，それぞれの部分空間で注意を計算する（Attn）．なお，$1 \leq h \leq N_{head}$ である．

$$\mathrm{MHAttn}(Q, K, V) = W^M[M_1; \ldots; M_{N_{head}}], \quad (10)$$

$$M_h = \mathrm{Attn}(W_h^Q Q, W_h^K K, W_h^V V) \quad (11)$$

具体的には，自己注意は直前の層の出力 Y を，$W_h^Q \in \mathbb{R}^{d_{emb} \times d_{head}}$，$W_h^K \in \mathbb{R}^{d_{emb} \times d_{head}}$，$W_h^V \in \mathbb{R}^{d_{emb} \times d_{head}}$ を用いて d_{head} 次元の部分空間 Q_h，K_h，V_h に射影する．デコーダで用いられる言語間注意では，直前のデコーダ層の出力 H_{sa} を部分空間 Q_h に，エンコーダの最終層の出力 Y_{enc} を部分空間 K_h，V_h に射影する．射影後，次の式によって各部分空間で単語間の関連の強さを表す分布行列 A_h を算出する．

$$A_h = \mathrm{softmax}(\frac{Q_h K_h^\intercal}{\sqrt{d_{head}}}) \quad (12)$$

この分布行列 A_h に対して V_h を掛け合わせることで，単語間の関連の強さを重みとする荷重和による表現 M_h を得ることができる．

$$M_h = A_h V_h \quad (13)$$

最後に，式（10）により，各部分空間の M_1，…，$M_{N_{head}}$ を結合し，単語の埋め込み次元に線形変換する．ただし，$W^M \in \mathbb{R}^{d_{emb} \times d_{emb}}$ はパラメータ行列である．自己注意の分布行列 A_h は原言語文あるいは目的言語文内の全ての単語間の関連の強さを含んでおり，言語間注意の分布行列 A_h は原言語文の単語と目的言語文の単語の間の全ての関連の強さを含んでいる．また，デコーダの自己注意は，推論時に予測していない単語に関する関連を求めないように未来の単語情報をマスクして訓練する．

3 提案手法

本稿では，Transformer のエンコーダ側とデコーダ側の自己注意に対し，両側の注意の整合性を保つような制約を与えて訓練する同期注意制約を提案する．

従来の Transformer モデルの目的関数 $\mathcal{L}$ を

$$\mathcal{L} = \mathcal{L}_{translation} \quad (14)$$

とすると，提案手法の同期注意制約を与えたモデルの目的関数は

$$\mathcal{L} = \mathcal{L}_{translation} + \lambda \mathcal{L}_{sync} \quad (15)$$

となる．なお，λ は目的関数に対して $\mathcal{L}_{sync}$ を考慮する度合いをコントロールするためのハイパーパラメータである．本節では以降，提案手法の同期注意制約 $\mathcal{L}_{sync}$ について述べる．

同期注意制約 $\mathcal{L}_{sync}$ は，エンコーダ・デコーダ間で，対応関係にある単語が指す自己注意の分布を近づける

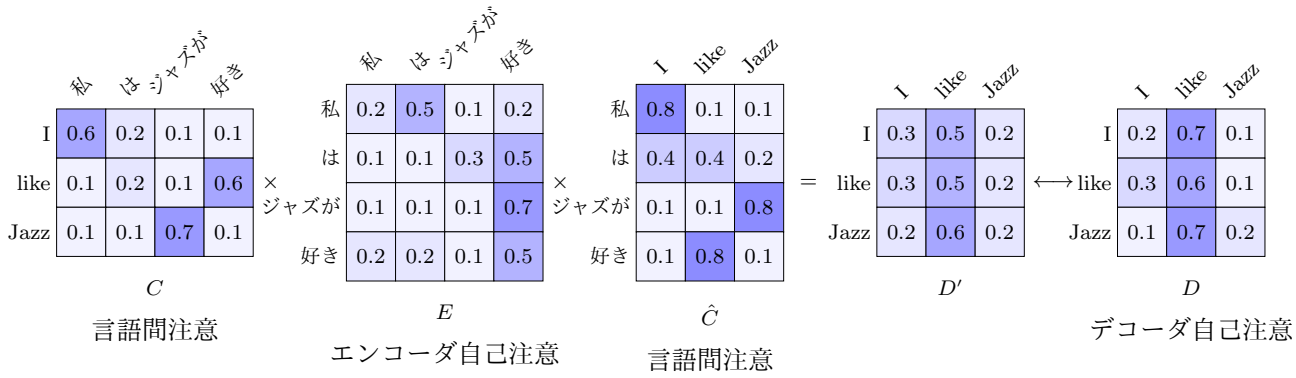

図 3: 同期注意の例

制約である．単語の対応関係は Transformer の言語間注意によって得る．

第 p 層目のエンコーダ・デコーダそれぞれの自己注意の分布行列 A_h を E，D とする．このとき，次式により E から D' を求める．

$$D' = CE\hat{C} \tag{16}$$

ただし，C と $\hat{C}$ は，次式により求められる行列である．Q，K はそれぞれ，デコーダ第 q 層目の言語間注意の Q_h，K_h を表す．

$$C = \mathrm{softmax}(\frac{Q_h K_h^\mathsf{T}}{\sqrt{d_{head}}}) \tag{17}$$

$$\hat{C} = \mathrm{softmax}(\frac{(Q_h K_h^\mathsf{T})^\mathsf{T}}{\sqrt{d_{head}}}) = \mathrm{softmax}(\frac{K_h Q_h^\mathsf{T}}{\sqrt{d_{head}}}) \tag{18}$$

求めた D' とデコーダ自己注意の D を用い，最小自乗法により同期注意制約 $\mathcal{L}_{sync}$ を算出する．

$$\mathcal{L}_{sync} = \sum_{i,j}(D'_{i,j} - D_{i,j})^2 \tag{19}$$

なお，Transformer モデルの訓練時は D に対して未来の単語情報をマスクするため，D' に対しても同様に未来の単語情報をマスクしてから $\mathcal{L}_{sync}$ を計算する．

図 3 に同期注意の例を示す．各セルに書かれている数値は分布行列の成分を表しており，各セルの色は濃いほど大きい値であることを表している．全ての行列は，各行が各単語における注意を表しており，行ごとに正規化された尤度となっている．同期注意制約は図 3 のように D' を計算し，D' と D の分布を近づける．

4 実験

4.1 実験設定

本実験では，提案手法の有効性を確認するため，提案手法を適用したモデルと従来の Transformer モデルの翻訳性能を比較する．ベースラインの Transformer モデルには `base` [10] モデルを使用した．

コーパス	言語対	文対数 訓練	開発	評価
WMT14	En↔De	3,805,838	3,000	3,003
ASPEC	Ja↔En	1,255,407	1,790	1,812

表 1: 実験データの対訳文対数

翻訳性能の評価実験は，WMT14 英独・独英翻訳タスク，WAT Asian Scientific Paper Excerpt Corpus (ASPEC)[1] 日英・英日翻訳タスクを用いた．

WMT14 英独・独英翻訳の実験データは，英語，独語ともに Moses トークナイザ[2]によって単語分割[3]した後，訓練データから学習したバイトペア符号化（Byte Pair Encoding; BPE）によりサブワード単位に分割した．BPE の語彙数は 37,000 とした．また，訓練データに対して，英語側・独語側ともにデータの正規化を行った[4] [5]．さらに，訓練データ中のノイズデータを除去するため `langid`[6]を用いてフィルタリング [6] した．モデル訓練時のミニバッチの大きさは約 25,000 トークンになるように設定した．

ASPEC 日英・英日翻訳の実験データは，WAT のベースラインシステムの構築方法[7] に従い単語分割した[3]が，日本語文の単語分割には `KyTea` を用いた．訓練データは `train-1.txt` と `train-2.txt` から上位 150 万を抽出して用いた．単語分割した後，訓練データから学習した BPE によりサブワード単位に分割した．ASPEC を用いた実験では BPE の学習を原言語側・目的言語側で独立して行い，BPE の語彙数は 16,000 とした．また，訓練データに対して，英語側のデータの正規化を行った[4]．モデル訓練時のミニバッチの大き

[1] ASPEC: `http://lotus.kuee.kyoto-u.ac.jp/ASPEC/`
[2] `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl`
[3] aggressive hyphen splitting オプションを用いた．
[4] `normalize-punctuation.perl`: `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/normalize-punctuation.perl`
[5] `remove-non-printing-char.perl`: `https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/remove-non-printing-char.perl`
[6] langid: `https://github.com/saffsd/langid.c`
[7] WAT19 ベースラインシステム: `http://lotus.kuee.kyoto-u.ac.jp/WAT/WAT2019/baseline/dataPreparationJE.html`

コーパス	言語対	モデル	BLEU(%)
WMT14	En→De	Transformer	27.18
		+同期注意	**27.37**
	De→En	Transformer	31.60
		+同期注意	**31.74**
ASPEC	Ja→En	Transformer	29.20
		+同期注意	**29.31**
	En→Ja	Transformer	42.95
		+同期注意	**43.20**

表 2: 実験結果

さは約 10,000 トークンになるように設定した．

全ての翻訳実験において，訓練データにはサブワード分割した後の原言語文・目的言語文のトークン数がともに 250 以下かつトークン数の比が 1.5 以内の対訳文対のみを用いた．なお，訓練・開発・評価データの対訳文対の数は表 1 に示す通りである．

目的関数の $\mathcal{L}_{translation}$ はラベル平滑化交差エントロピーを用い，平滑化のための ϵ [9] は 0.1 とした．同期注意制約を考慮する度合いをコントロールするハイパーパラメータの λ は 1.0 とした．同期注意制約を与える層 p は Transformer `base` モデルの最終層である第 6 層目とし，D' を求める際に必要な言語間注意の層 q は第 5 層目とした．全ての実験において，モデルのパラメータ更新回数は 100,000 回とした．また，学習率は 4000 回更新時で 7e-4 となるように線形的に増加させ，以降は更新回数の平方根の逆数に比例して減衰させた [10]．

モデル性能の評価時は，全ての実験において訓練終了時から前 1250 更新ずつ 5 つ分までのモデルパラメータを平均化したモデルを用いている [10]．翻訳文の生成にはビーム探索を用い，ビーム幅は 4，文長正則化パラメータは 0.6 [12] とした．

4.2 実験結果

表 2 に実験結果を示す．モデルの翻訳性能は BLEU [7] によって評価した．WMT14 は `sacreBLEU` [8] を用い,[8] [9]，ASPEC は WAT の評価方法[10]に従ってスコアを算出した．表 2 より，提案手法の同期注意制約を与えて Transformer モデルを訓練することにより，WMT14 の英独翻訳実験では 0.19 ポイント，独英翻訳実験では 0.14 ポイントの上昇が得られた．ASPEC の日英翻訳では 0.11 ポイント，英日翻訳では 0.25 ポイントの上昇が得られた．実験結果より，提案手法の同期注意制約を与えたモデルである “Transformer+同期注意” がベースラインモデルである “Transformer” より高い翻訳性能であることが分かり，提案手法の有効性が確認できる．

[8]En→De: `BLEU+case.mixed+lang.en-de+numrefs.1+smooth.exp+test.wmt14+tok.intl+version.1.3.7`

[9]De→En: `BLEU+case.mixed+lang.de-en+numrefs.1+smooth.exp+test.wmt14+tok.intl+version.1.3.7`

[10]`http://lotus.kuee.kyoto-u.ac.jp/WAT/evaluation/index.html#automatic_evaluation_systems.html`

5 おわりに

本稿では，原言語側と目的言語側の自己注意の整合性を保つ同期注意制約を与えて訓練する新たな Transformer モデルを提案した．同期注意制約は原言語側と目的言語側の自己注意を言語間注意によって対応付けることで，言語間で相互的に文構造を考慮する自己注意を獲得する．また，同期注意制約は Transformer モデル内の自己注意と言語間注意から作られるため，制約を与えるための教師データやモデルパラメータ量の増加がない．実験結果より，ベースラインの Trasnformer モデルと提案モデルの BLEU を比較したところ，WMT14 英独翻訳・独英翻訳タスクにおいてそれぞれ 0.19 ポイント，0.14 ポイントのスコア上昇，WAT ASPEC 日英翻訳・英日翻訳タスクにおいてそれぞれ 0.11 ポイント，0.25 ポイントのスコア上昇を確認できた．今後は，他の言語対においても提案手法の有効性を確認したい．

謝辞

本研究成果は，国立研究開発法人情報通信研究機構の委託研究により得られたものである．また，本研究の一部は JSPS 科研費 18K18110 の助成を受けたものである．ここに謝意を表する．

参考文献

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] H. Deguchi, A. Tamura, and T. Ninomiya. Dependency-based self-attention for transformer nmt. In *Proc. of RANLP 2019*, pp. 239–246, 2019.

[3] Y. Ding and M. Palmer. Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars. In *Proc. of ACL 2005*, pp. 541–548, 2005.

[4] S. Garg, S. Peitz, U. Nallasamy, and M. Paulik. Jointly learning to align and translate with transformer models. In *Proc. of EMNLP-IJCNLP 2019*, pp. 4452–4461, 2019.

[5] H. Jiang, M. Yang, T. Zhao, S. Li, and B. Wang. A Statistical Machine Translation Model Based on a Synthetic Synchronous Grammar. In *Proc. of ACL-IJCNLP 2009 Short Papers*, pp. 125–128, 2009.

[6] N. Ng, K. Yee, A. Baevski, M. Ott, M. Auli, and S. Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proc. of WMT 2019 (Volume 2: Shared Task Papers, Day 1)*, pp. 314–319, 2019.

[7] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL 2002*, pp. 311–318, 2002.

[8] M. Post. A call for clarity in reporting BLEU scores. In *Proc. of WMT 2018*, pp. 186–191, 2018.

[9] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *The IEEE Conference on CVPR 2016*, 2016.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in NIPS 30*, pp. 5998–6008. 2017.

[11] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proc. of ACL 2019*, pp. 5797–5808, 2019.

[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.