シフト付き絶対位置埋め込み

清野 舜[◇,♠] 小林 颯介[♠,♡] 鈴木 潤[♠,◇] 乾 健太郎[♠,◇]
[◇]理化学研究所 [♠]東北大学 [♡]株式会社 Preferred Networks
shun.kiyono@riken.jp, sosk@preferred.jp, {jun.suzuki, inui}@tohoku.ac.jp

概要

本研究は Transformer における位置表現の改良に取り組む．既存の位置表現のうち，シフト不変性を持つ手法が高い性能を発揮することに着目し，シフト付き絶対位置埋め込み（SHAPE）の効果を検証する．SHAPE の根幹となるアイデアは，学習中に絶対位置を乱数値でシフトさせることで，シフト不変性をモデルに取り入れることである．既存のシフト不変である位置表現と比較して，SHAPE は同等の性能を達成しつつ，より高速に動作することを示す．[1)]

1 はじめに

Transformer[1] に基づく符号化復号化モデル（Encoder-Decoder）において，位置表現はモデルが系列中のトークンの順序を認識するために導入されている．位置表現は，**絶対位置埋め込み**（Absolute Position Embedding; **APE**）[2, 1] と**相対位置埋め込み**（Relative Position Embedding; **RPE**）[3] の 2 種類に大別される [4]．APE では，各位置ごとの専用の埋め込みと単語埋め込みとの和によって位置を表現する．一方で RPE は，2 つのトークン間の相対的な距離を用いて，注意機構内部で位置を表現する．

系列変換タスクにおいて，RPE は APE を上回る外挿性能[2)]を発揮することが知られている [5, 6]．その要因は，RPE の持つ性質である**シフト不変性**であることが報告されている [7]．ここでシフト不変性とは，ある関数において入力の位置シフトが関数の出力に影響しない性質を指す．しかし，RPE は注意機構に依存する形で定式化されているため，注意機構自体の改善を試みる方法論との組み合わせが困難である．そのため本研究では，APE にシフト不変性を取り入れることを試みる．

絶対位置にシフト不変性を取り入れるために，画像認識 [8] や NLP における質問応答 [9] 等では，訓練中に入力系列の位置を乱数値でシフトさせるという方法が用いられてきた．同様に APE においても，乱数値によるシフトをおこなうことで，絶対位置の代わりに相対位置を活用した学習を Transformer に強制できると期待できるが，その有効性はまだ検証されていない．そこで我々はこのシフト操作を**シフト付き絶対位置埋め込み**（Shifted Absolute Position Embedding; **SHAPE**）として定式化し，その効果を検証する．実験では，SHAPE を用いた Transformer がシフト不変性を獲得できること，また，機械翻訳タスクで SHAPE が RPE と同等の性能を達成できることを示す．

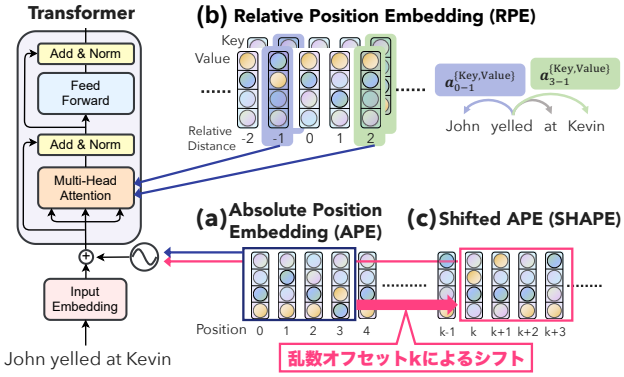

図 1 本研究で取り扱う位置表現の概要図

2 Transformer における位置表現

図 1 に本研究で取り扱う位置表現を示す．以降，入力系列 $\boldsymbol{X}$ と出力系列 $\boldsymbol{Y}$ を，それぞれ I トークンと J トークンからなる系列 $\boldsymbol{X} = (x_1, \ldots, x_I)$ と $\boldsymbol{Y} = (y_1, \ldots, y_J)$ として表す．

2.1 絶対位置埋め込み（APE）

APE では各位置に対して専用の位置埋め込みを用いる．具体的には，各トークン $x_i \in \boldsymbol{X}$ と $y_j \in \boldsymbol{Y}$ について，対応する位置埋め込みと単語埋め込みとの和によって位置を表現する（図 1a）．

Transformer において APE には Sinusoidal 位置符号化 [1] が典型的に用いられる．ここで i 番目のト一

[1)] 実装：https://github.com/butsugiri/shape
[2)] 訓練データよりも長い系列に汎化する能力のこと

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

クンの第 m 次元の値 $\mathrm{PE}(i,m)$ は

$$\mathrm{PE}(i,m)=\begin{cases}\sin\left(\frac{i}{10000^{\frac{2m}{D}}}\right) & m \text{ が偶数}\\ \cos\left(\frac{i}{10000^{\frac{2m}{D}}}\right) & m \text{ が奇数}\end{cases} \quad (1)$$

として定義され，D はモデルの次元数を表す．

2.2 相対位置埋め込み（RPE）

RPE[3] では，各トークン間の相対距離を Transformer の注意機構の特徴量として用いることで位置を表現する（図 1b）．例えば，Shaw ら [3] は i 番目と j 番目のトークン間の相対距離を，埋め込み $\boldsymbol{a}^{\mathrm{Key}}_{i-j}, \boldsymbol{a}^{\mathrm{Value}}_{i-j} \in \mathbb{R}^D$ によって表現している．これらの埋め込みは，注意機構における Key と Value 表現にそれぞれ足し合わされる．

RPE の演算はシフト不変性を持つことから，訓練データの分布外の長さの系列に対して，APE の性能を上回ることが報告されている [6, 5, 7]．しかし，注意機構を用いて位置を表現する都合上，RPE は APE よりも計算コストが大きい[3)]．また，RPE は注意機構の改変を伴うため，注意機構自体の軽量化を試みる方法論 [10, 11] との組み合わせが困難である．

2.3 シフト付き絶対位置埋め込み（SHAPE）

本研究では，Transformer においてシフト不変性を実現するための方法の一つとして，SHAPE（図 1c）の効果を検証する．RPE の課題を踏まえ，SHAPE では Transformer の注意機構の改変を回避しつつ，APE と同等の計算コストの実現を目指す．訓練中，SHAPE は入出力系列の各位置インデックスを，乱数から生成したオフセットでシフトさせる．このシフト操作は，絶対位置の代わりに相対位置を利用した学習をモデルに強制することに相当する．その結果，モデルがシフト不変性を持つ関数を学習すると期待できる．

いま，離散一様分布 $\mathcal{U}\{0,K\}$ からサンプルしたオフセットを k で表す．ここで，K は最大オフセット幅であり，k はエポックごとに各系列について独立にサンプルされる．SHAPE は $\mathrm{PE}(i,m)$（式 1 ）を次式で置き換えることで実現できる．

$$\mathrm{PE}(i+k,m) \quad (2)$$

SHAPE は APE を用いる任意のモデルに適用可能である．また，シフト操作は非常に軽量であるため，SHAPE は APE と同等の速度で動作すると期待できる．今回，k は入力系列と出力系列について独立にサンプルした．ここで，仮に $K=0$ とすると SHAPE は APE に帰着することに注意されたい[4)]．

3 実験

実験では，機械翻訳タスクを題材として，SHAPE を組み合わせた Transformer がシフト不変性を獲得することを確認する（第 3.2 節）．その後，SHAPE の性能を既存の位置表現と比較する（第 3.3 節）．

3.1 実験設定

データセット　WMT2016 英独データセットを訓練データとして用いた．トークン化やサブワード化 [12] 処理は先行研究の設定に従った [13]．開発セットと評価セットとして，それぞれ newstest2010-2013 と newstest2014-2016 を用いた．

実験には，以下の 3 つの設定を用いた：

(i) Vanilla　先行研究 [1, 13] と同等の設定である．

(ii) Extrapolate　シフト不変性を持つモデルは，外挿性能の観点で評価することが一般的である [7, 14]．我々は，先行研究 [6] に従い，Vanilla の訓練データから入力系列か出力系列のサブワード長が 50 を超えるような系列対を取り除き，新たに訓練データを作成した．また，開発セットと評価セットには Vanilla と同じものを用いた．

(iii) Interpolate　今回，我々は各モデルを内挿性能の観点でも評価することを試みる．ここで，内挿性能を，訓練中に観測した長さの系列に関して汎化する能力として定義する．本研究では，この内挿性能を長い系列を用いて評価するが，その理由は以下の通りである．

長い系列を含むようなデータセットでは，各トークンが各位置に関して疎な形で分布する（**位置のスパースネス問題**）．つまり，開発セットや評価セットにおいて，系列中のある位置に登場するトークンは，訓練データ中で同じ位置にほとんど登場しないと考えられる．このとき，絶対位置を用いる場合はトークンと位置の組み合わせに対する過学習が生じると考えられるが，シフト不変性を持つ位置表現はこの過学習を抑制できる可能性がある．

本研究では，対訳データに含まれる独立した系列を結合することで，長い系列を人工的に作成した．具体的には，Vanilla に含まれる隣り合う系列

[3)] Narang ら [5] は，RPE は APE よりも最大で 25%遅いことを報告している．

[4)] 推論時には $K=0$ としている．

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

表 1　INTERPOLATE の訓練データよりサンプリングした 1 万系列対で計測した BLEU スコア

	Original	Swapped	性能の減少幅
APE	28.81	20.74	8.07
SHAPE	28.51	27.06	1.45

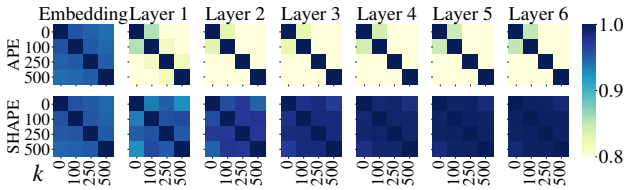

図 2　各オフセット $k \in \{0, 100, 250, 500\}$ を用いて計算した符号化器の隠れ層間の類似度

$\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{10}$ と $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{10}$ を，専用のトークン $\langle sep \rangle$ を用いて結合した．開発セットと評価セットにも同じ処理を加えてデータを作成した．

モデル　Transformer-base[1] に，APE，SHAPE と RPE を組み合わせて実験をおこなった．APE と RPE にはそれぞれ Sinusoidal 符号化と Shaw ら [3] の手法を用いた．実装には OpenNMT-py[15] を用いた．SHAPE の最大オフセット K は 500，RPE の最大相対距離は先行研究 [3, 6] に従い 16 とした．各モデルの性能は sacreBLEU[16] による detokenized BLEU で評価した[5)]．

3.2　実験 1：シフト不変性の確認

本実験では，SHAPE がシフト不変性を獲得していることを INTERPOLATE 上で訓練した APE と SHAPE を用いて，定量的・定性的な観点で検証する．

定量評価：訓練データ上の BLEU スコア　本実験では系列間の順序に対するモデルの頑健性を評価する．具体的には，INTERPOLATE の訓練データからサンプルした 1 万系列対を用いて，系列間で順序を入れ替えた場合と，入れ替えなかった場合の性能を比較する．訓練データを用いるのは，未知の系列による影響を除外し，系列間の順番の影響のみを対象とした評価をおこなうためである．

評価手順は次のとおりである．まず，元の系列 **Original**（$\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{10}$）から，先頭の系列を末尾に移動させて，系列 **Swapped**（$\boldsymbol{X}_2, \ldots, \boldsymbol{X}_{10}, \boldsymbol{X}_1$）を作成する．次に **Original** と **Swapped** を訓練済みのモデルでデコードし，それぞれ $\boldsymbol{Y}'_1, \ldots, \boldsymbol{Y}'_{10}$ と $\boldsymbol{Y}'_2, \ldots, \boldsymbol{Y}'_{10}, \boldsymbol{Y}'_1$ を得る．最後に，$\boldsymbol{Y}'_1$ の BLEU スコアを評価する．

実験結果を表 1 に示した．ここで，**Original** から **Swapped** への性能の減少幅は，SHAPE の方が APE よりも小さいことから，SHAPE がシフト不変性を獲得していることが示唆される．

[5)] ハイパーパラメータの一覧は付録 A に示した．

表 2　各位置表現の BLEU スコア：†：5 つの乱数シードの平均値．∗：訓練不能であったため，値を掲載していない．相対速度は APE からの相対速度を表す．

データセット	モデル	Valid	Test	相対速度
VANILLA	APE†	23.61	30.46	x1.00
	RPE†	23.67	30.54	x0.91
	SHAPE†	23.63	30.49	x1.01
EXTRAPOLATE	APE	22.18	29.22	x1.00
	RPE	22.97	29.86	x0.91
	SHAPE	22.96	29.80	x0.99
INTERPOLATE	APE	31.40	38.23	x1.00
	RPE∗	-	-	-
	SHAPE	32.50	39.09	x0.99

定性評価：隠れ層の類似度　本実験では，SHAPE がシフト不変性を獲得していることを定性的に確認する．図 2 はオフセット k が APE と SHAPE を用いた訓練済みモデルの隠れ層に与える影響を表す．具体的には，入力系列 $\boldsymbol{X}$ について，各オフセット $k \in \{0, 100, 250, 500\}$ を用いて訓練済みモデルの符号化器から隠れ層を計算した．その後，異なるオフセットから計算した隠れ層 $\boldsymbol{h}_i^{k_1}, \boldsymbol{h}_i^{k_2} \in \mathbb{R}^D$ について，コサイン類似度（sim）を計算し，位置方向に平均を求めた．つまり，$\frac{1}{I}\sum_{i=1}^{I} \mathrm{sim}(\boldsymbol{h}_i^{k_1}, \boldsymbol{h}_i^{k_2})$ である．

図 2 より，SHAPE の符号化器はシフト不変性を獲得できているとわかる．これは SHAPE において，オフセット k によらず，類似度がほぼ 1.0 に張り付いているためである．ここで，APE の類似度が同様の傾向を示さないことから，シフト不変性は自明に獲得可能な性質ではないと確認できる．

3.3　実験 2：位置表現間の性能比較

各位置表現の性能を評価した結果を表 2 に示した．また，APE からの性能向上幅を入力系列の長さ別に図示した結果を図 3 に示した．

VANILLA の結果　3 つのモデル全てがほとんど同等の性能を示した．APE が RPE と同等の性能を示すという結果は既存研究の知見 [3] とは一致しないものの，これは実験に使った実装の違いによるものだと考えられる．実際，Transformer の改良に関する知見は，特定の実装に依存することが多い旨が Narang らによって報告されている [5]．

EXTRAPOLATE の結果　評価セットにおいて，RPE の性能（29.86）が APE（29.22）を上回った．また，SHAPE が RPE と同等の性能を達成した（29.80）．図

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

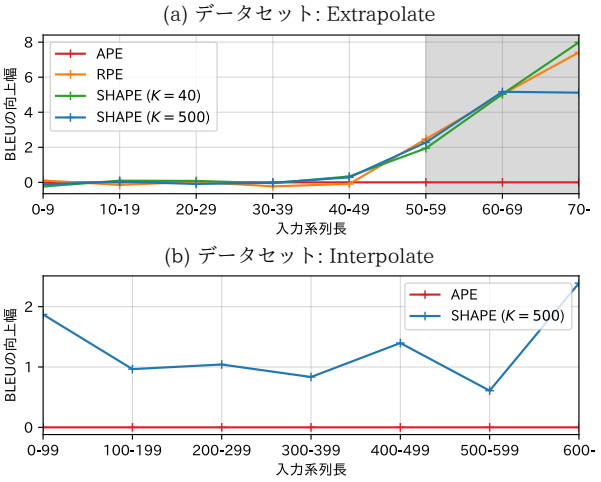

図 3　APE からの BLEU スコアの向上幅：開発セットと評価セットを用いた．灰色の背景色は対応する長さの系列が訓練データ中に存在しないことを示す．

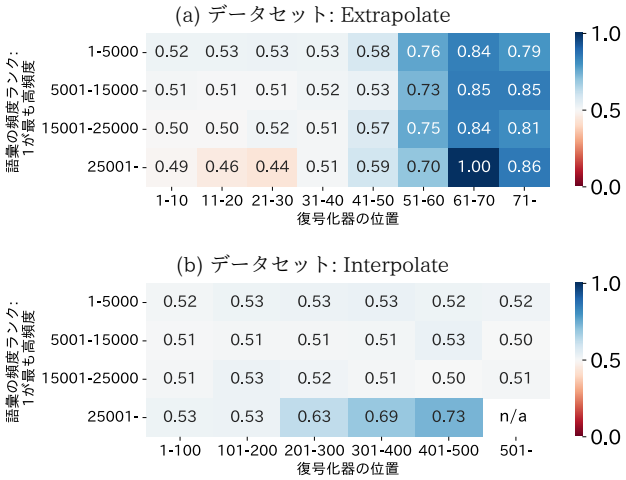

図 4　参照訳を用いたトークン単位の分析：各セルの値は，正解トークンに対して SHAPE が APE よりも高いスコア（負の対数尤度）を付与できた割合を表す．

3a より，この性能向上は，訓練データよりも長い系列（外挿）によるものだとわかる．つまり，RPE と SHAPE は APE の外挿性能を改善できている．

また，図 3a には，SHAPE の最大オフセット K を開発セット上で調整した結果を示した．$K = 40$ としたときに，開発セットと評価セットの性能はそれぞれ 23.12 と 29.86 となり，RPE を上回ったことから，SHAPE は RPE よりも良い位置表現となりえる．

INTERPOLATE の結果　RPE の学習には著しく時間がかかってしまい，現実的な時間で訓練を終えることができなかったため，値を載せていない[6]．本データにおいても，評価セット上で SHAPE（39.09）は APE（38.23）を上回る性能を示した．図 3b より，SHAPE は入力系列の長さに依存せず，一貫して性能向上を果たしている．この結果から，SHAPE は Transformer の内挿性能も改善できるとわかる．

4　分析

図 3 において，SHAPE は APE を BLEU スコアで上回った．しかし，BLEU スコアでは (1) N-gram に基づく参照訳との一致率と (2) Brevity Penalty による出力系列の長さを同時に評価しているため，出力がどの側面で改善したのかは明らかではない．そこで本節では，前者の一致率に着目するため，参照訳を用いてトークン単位のスコアを計算し，モデル間の比較をおこなった．具体的には，系列のペア $(\boldsymbol{X}, \boldsymbol{Y})$ に対して，訓練済みモデルを用いてスコア（負の対数尤度）s_j を各正解トークン y_j について計算した[7]．ここで，モデルは正解トークンにより高いスコアを付与することが望ましい．図 4 は，正解トークンに対して SHAPE が APE よりも高いスコアを付与した割合を，復号化器の位置別に図示した結果を表す．この分析には開発セットを用いた．

EXTRAPOLATE：SHAPE はトークンの一致率の向上に寄与　図 4a の右側（訓練データよりも長い系列）において，SHAPE が APE を大きく上回っている．この結果から，SHAPE が外挿における N-gram の一致率の向上に貢献していることが示唆される．

INTERPOLATE：SHAPE は低頻度語の予測に有効　図 4b に示した通り，SHAPE の性能が APE を一貫して上回った．モデル間の性能差は図の下部，つまり低頻度語に対応する箇所において特に顕著である．訓練中，SHAPE は同じ系列対に対して，エポックごとに異なる位置表現を用いて学習をおこなうが，これは一種のデータ拡張として解釈できる．先述の**位置のスパースネス問題**（第 3.1 節）は低頻度語において特に生じやすいため，SHAPE によるデータ拡張が効果的であったと考えられる．

5　おわりに

本研究では APE の簡単な亜種である SHAPE の調査をおこなった．実験では，SHAPE が RPE と同等の性能を発揮しつつ，APE と同等の速度で動作することを示した．SHAPE は数行で実装可能であるため，既存の実装に簡単に導入できる．故に，SHAPE は APE と RPE の代替手法となりえると期待できる．

[6] RPE を INTERPOLATE 上で動かした場合，パラメータの更新が APE や SHAPE と比較して 20 倍程度低速だった．

[7] 訓練時と同様に teacher-forcing アルゴリズムを用いた

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

謝辞

本研究の一部 (基礎研究) は JST ムーンショット JPMJMS2011 の助成を受けたものです.

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In **Advances in Neural Information Processing Systems 30 (NIPS 2017)**, pp. 5998–6008, 2017.

[2] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional Sequence to Sequence Learning. In **Proceedings of the 34th International Conference on Machine Learning (ICML 2017)**, pp. 1243–1252, 2017.

[3] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-Attention with Relative Position Representations. In **Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers) (NAACL 2018)**, pp. 464–468, 2018.

[4] Philipp Dufter, Martin Schmitt, and Hinrich Schütze. Position Information in Transformers: An Overview. **arXiv preprint arXiv:2102.11090**, 2021.

[5] Sharan Narang, Hyung Won Chung, Yi Tay, William Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, et al. Do Transformer Modifications Transfer Across Implementations and Applications? **arXiv preprint arXiv:2102.11972**, 2021.

[6] Masato Neishi and Naoki Yoshinaga. On the Relation between Position Information and Sentence Length in Neural Machine Translation. In **Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL 2019)**, pp. 328–338, 2019.

[7] Benyou Wang, Lifeng Shang, Christina Lioma, Xin Jiang, Hao Yang, Qun Liu, and Jakob Grue Simonsen. On Position Embeddings in BERT. In **Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)**, 2021.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. **Deep Learning**, chapter 7.4, pp. 233–234. MIT Press, 2016. http://www.deeplearningbook.org.

[9] Mor Geva, Ankit Gupta, and Jonathan Berant. Injecting Numerical Reasoning Skills into Language Models. In **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)**, pp. 946–958, 2020.

[10] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In **Proceedings of the 8th International Conference on Learning Representations (ICLR 2020)**, 2020.

[11] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient Transformers: A Survey. **arXiv preprint arXiv:2009.06732**, 2020.

[12] 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 (ACL 2016)**, pp. 1715–1725, 2016.

[13] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling Neural Machine Translation. In **Proceedings of the Third Conference on Machine Translation (WMT 2018)**, pp. 1–9, 2018.

[14] Benjamin Newman, John Hewitt, Percy Liang, and Christopher D. Manning. The EOS Decision and Length Extrapolation. In **Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP 2020)**, pp. 276–291, 2020.

[15] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In **Proceedings of ACL 2017, System Demonstrations (ACL 2017)**, pp. 67–72, 2017.

[16] Matt Post. A Call for Clarity in Reporting BLEU Scores. In **Proceedings of the Third Conference on Machine Translation: Research Papers (WMT 2018)**, pp. 186–191, 2018.

[17] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)**, pp. 2818–2826, 2016.

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).

A ハイパーパラメータ

表 3 ハイパーパラメータの一覧.

設定	値
符号化復号化モデル	*transformer-base* [1]
最適化アルゴリズム	Adam ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
学習率	"Noam" scheduler[1] を用い，係数は 2 とした
Warmup Steps	8,000
ドロップアウト	0.1
勾配クリッピング	None
ビーム探索幅	4
Label Smoothing	$\epsilon_{ls} = 0.1$ [17]
ミニバッチ	112k tokens
更新回数	200,000
Averaging	5,000 更新ごとにモデルを保存し，最終 10 モデルの平均を評価に用いた.
最大オフセット K（SHAPE）	ほとんどの実験で $K = 500$ とした．Extrapolate を用いた実験では，K の値を開発セット上で調整した． 探索に用いた K の値は{10, 20, 30, 40, 100, 500}である．最終的に，$K = 40$ と $K = 500$ の値を報告している（図 3）.
最大相対距離（RPE）	16[6]
実装	OpenNMT-py[15]

B 各 newstest の BLEU スコア

表 4 に各モデルを newstest2010-2016 を用いて評価した結果を示す[8)9)].

表 4 **newstest2010-2016 上の BLEU スコア**：平均は全 newstest のマクロ平均値を示す．†：5 つの乱数シードの平均値. ∗：訓練不能であったため，値を掲載していない．**相対速度**は APE からの相対速度を表す.

モデル	2010	2011	2012	2013	2014	2015	2016	平均	相対速度
データセット：Vanilla									
APE†	24.22	21.98	22.20	26.06	26.95	29.98	34.46	26.55	x1.00
RPE†	24.29	22.05	22.22	26.13	27.00	30.00	34.61	26.61	x0.91
SHAPE†	24.18	22.01	22.23	26.08	26.89	30.12	34.48	26.57	x1.01
データセット：Extrapolate									
APE	22.69	20.36	20.72	24.94	26.24	28.79	32.62	25.19	x1.00
RPE	23.46	21.19	21.69	25.54	26.80	29.43	33.34	25.92	x0.91
SHAPE	23.60	21.24	21.53	25.45	26.54	29.22	33.63	25.89	x0.99
データセット：Interpolate ‡									
APE	31.41	29.71	29.79	34.69	35.36	38.00	41.32	34.33	x1.00
RPE∗	-	-	-	-	-	-	-	-	-
SHAPE	32.71	30.77	30.96	35.54	35.72	39.18	42.37	35.32	x0.99

[8)] Vanilla と Extrapolate の評価における sacreBLEU のハッシュ：`BLEU+case.mixed+lang.en-de+numrefs.1+smooth.exp+test.wmt{10,11,12,13,14/full,15,16}+tok.13a+version.1.5.0`.

[9)] Interpolate の評価における sacreBLEU のハッシュ：`BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.0`（開発セットと評価セットも訓練データと同様の結合処理を加えているため，sacreBLEU 内部の参照訳の代わりに手動で参照訳を与えている．詳細については第 3.1 節の説明を参照されたい.）

This work is licensed by the author(s) under CC BY 4.0
(https://creativecommons.org/licenses/by/4.0/).