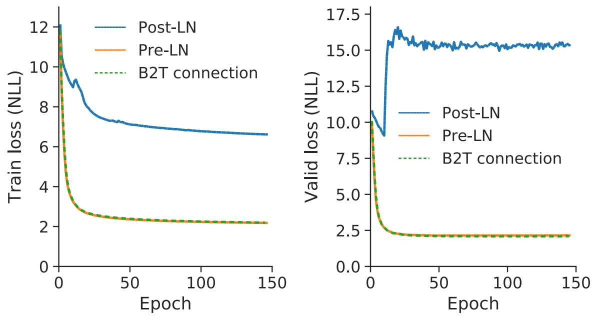
B2T Connection: Serving Stability and Performance in Deep Transformers

Sho Takase[†*] Shun Kiyono[†] Sosuke Kobayashi[‡] Jun Suzuki[‡]
[†]LINE Corporation [‡]Tohoku University
{sho.takase, shun.kiyono}@linecorp.com
sosk@preferred.jp
jun.suzuki@tohoku.ac.jp

Abstract

From the perspective of the layer normalization (LN) positions, the architectures of Transformers can be categorized into two types: Post-LN and Pre-LN. Recent Transformers tend to be Pre-LN because, in Post-LN with deep Transformers (e.g., those with ten or more layers), the training is often unstable, resulting in useless models. However, Post-LN has consistently achieved better performance than Pre-LN in relatively shallow Transformers (e.g., those with six or fewer layers). This study first investigates the reason for these discrepant observations empirically and theoretically and made the following discoveries: 1, the LN in Post-LN is the main source of the vanishing gradient problem that leads to unstable training, whereas Pre-LN prevents it, and 2, Post-LN tends to preserve larger gradient norms in higher layers during the back-propagation, which may lead to effective training. Exploiting the new findings, we propose a method that can provide both high stability and effective training by a simple modification of Post-LN. We conduct experiments on a wide range of text generation tasks. The experimental results demonstrate that our method outperforms Pre-LN, and enables stable training regardless of the shallow or deep layer settings. Our code is publicly available at https://github.com/takase/b2t_connection.



(a) Train loss (b) Valid loss

Figure 1: Loss values of 18 layered Transformer-based encoder-decoder architectures.

1 Introduction

To prevent the vanishing (or exploding) gradient problem in the training of a deep neural network (DNN), various techniques, such as batch normalization (Ioffe and Szegedy, 2015) and residual connection (Srivastava et al., 2015; He et al., 2016a), have been proposed and widely used in almost all recent DNNs. Transformer (Vaswani et al., 2017) employs the layer normalization (Ba et al., 2016) for this purpose. Transformer is currently the most successful model architecture in DNNs. It was firstly developed for applying sequence-to-sequence tasks, such as machine translation (Vaswani et al., 2017), summarization (Takase and Okazaki, 2019), and automatic speech recognition (ASR) (Wang et al., 2020), and is currently used in speech, vision, and many other information processing research fields.

As reported in the batch normalization literature (He et al., 2016b), the position of the normalization layers primarily affects both the stability and resultant performance of a trained model. In Transformers, some previous studies have investigated the impact of the layer normalization positions (Wang et al., 2019; Xiong et al., 2020). There are currently two major layer normalization positions in Transformers: Pre-Layer Normalization (Pre-LN) and Post-Layer Normalization (Post-LN). Pre-LN applies the layer normalization to an input for each sub-layer, and Post-LN places the layer normalization after each residual connection. The original Transformer (Vaswani et al., 2017) employs Post-LN. However, many recent studies have suggested using Pre-LN (Wang et al., 2019; Baevski and Auli, 2019; Brown et al., 2020) because the training of deep Transformers (e.g., those with ten or more layers) using Post-LN is often unstable, resulting in useless models. Figure 1

[*] A part of this work was done when the author was at Tokyo Institute of Technology.

shows loss curves for an actual example; the training of 18 layered Transformer encoder-decoders (18L-18L) on a widely used WMT English-to-German machine translation dataset. These figures clearly show that the Post-LN Transformer encoder-decoders fail to train the model. However, in contrast, Liu et al. (2020) reported that Post-LN consistently achieved better performance than Pre-LN on a machine translation task when they used 6 layered (relatively shallow, 6L-6L) Transformers.

This paper focuses specifically on such discrepancies between Pre-LN and Post-LN in configurations with various number of layers. We investigate the sources of the instability of training in deep configurations and the superior performance in shallow configurations for Post-LN, compared with that for Pre-LN, to understand the essentials of the differences between Pre-LN and Post-LN. We discover that the layer normalization in Post-LN is the main source of the vanishing gradient problem that leads to unstable training, whereas Pre-LN prevents it, as shown in Figure 1. In particular, we clarify that the layer normalization is a significant factor of the vanishing gradient problem by comparing the input/output vector norms of gradient flows for each layer normalization during back-propagation. These analyses bring us a novel idea that can satisfy higher stability by skipping over layer normalizations and provide better performance than Pre-LN regardless of their layer sizes. Consequently, we propose a method that is based on Post-LN Transformers but has additional residual connections to enable stable training.

We conduct experiments on a wide range of text generation tasks, namely machine translation, summarization, language modeling, and ASR. The experimental results lead to the following three new major findings:

1. Post-LN Transformers achieve better performance than Pre-LN Transformers on text generation tasks (not only machine translation (Liu et al., 2020) but also other tasks). Thus, Post-LN is superior to Pre-LN if the problem of its unstable training can be solved.

2. Our modification enables Post-LN Transformers to stack many layers.

3. Our method can maintain the performance advantage of Post-LN and mitigate its unstable training property, thus providing better performance than Pre-LN.

2 Post-LN and Pre-LN Transformers

We briefly describe Post-LN and Pre-LN Transformers in this section. The original Transformer (Vaswani et al., 2017) uses Post-LN, in which layer normalizations are located after each residual connection. Let x be an input of a sub-layer, and $\mathcal{F}(\cdot)$ be a sub-layer of a Transformer, such as a feed-forward network or multi-head attention. Post-LN is defined as follows:

$$\mathrm{PostLN}(x) = \mathrm{LN}(x + \mathcal{F}(x)), \tag{1}$$

where $\mathrm{LN}(\cdot)$ is the layer normalization function.

In contrast, Pre-LN places the layer normalization before an input of each sub-layer:

$$\mathrm{PreLN}(x) = x + \mathcal{F}(\mathrm{LN}(x)). \tag{2}$$

Figure 2 (a) and (b) illustrate Post-LN and Pre-LN Transformer architectures, respectively.

3 Gradients of Transformer Layers

As described in Liu et al. (2020), the vanishing gradient problem often occurs in Post-LN Transformers. Figure 3 shows the gradient norms of each layer for the (a) encoder-side and (b) decoder-side at the beginning of training, when 18L-18L Transformer encoder-decoders are trained on a widely used machine translation dataset (the WMT English-to-German dataset). Focus on the decoder-side of Post-LN as illustrated in Figure 3 (b). This figure shows that shallower layers have smaller gradient norms. In other words, the vanishing gradient occurs in the decoder-side of Post-LN because its gradient norms exponentially decay as they are back-propagated to shallower layers. This result is consistent with the previous study (Liu et al., 2020). We consider that this vanishing gradient causes the difficulty of stacking many layers with the Post-LN setting, as shown in Figure 1.

To investigate the vanishing gradient empirically in more detail, we measure the gradient norms of parts (1) - (5) of Figure 2 (a). Figure 4 shows the gradient norms of each part in the 18th layer[1]. This figure shows that the gradient norms decrease drastically from (4) to (3) and (2) to (1). These parts correspond to layer normalizations, as shown in Figure 4. This suggests that layer normalizations in Post-LN Transformers are probably the cause of the vanishing gradient problem.

[1]Appendix B shows the gradient norms of each part in the 1st and 9th decoders as additional examples.

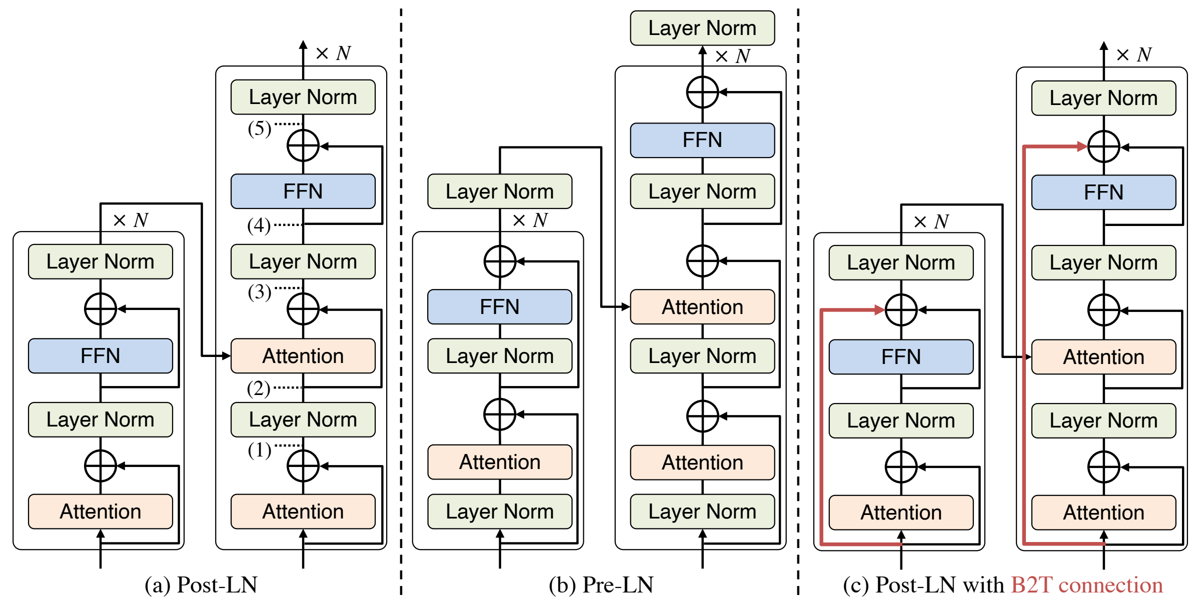


Figure 2: Transformer-based encoder-decoder architectures for (a) Post-LN, (b) Pre-LN, and (c) Post-LN with our proposed B2T connection.

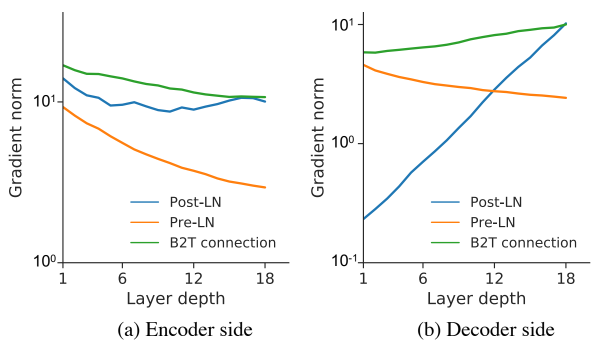


Figure 3: Gradient norms of 18 layered Transformer-based encoder-decoder architectures.

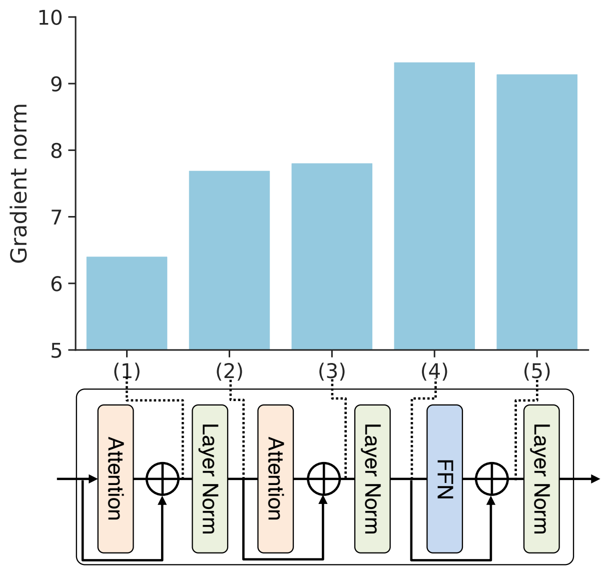


Figure 4: Gradient norms of each location in the 18th decoder for the 18 layered Post-LN Transformer encoder-decoder on WMT English-to-German translation training data.

To investigate the difference between the gradient flows of Post-LN and those of Pre-LN theoretically, we calculate the derivatives of Equations (1) and (2), as follows:

$$\frac{\partial \mathrm{PostLN}(x)}{\partial x} = \frac{\partial \mathrm{LN}(x+\mathcal{F}(x))}{\partial (x+\mathcal{F}(x))}\left(I + \frac{\partial \mathcal{F}(x)}{\partial x}\right), \tag{3}$$

$$\frac{\partial \mathrm{PreLN}(x)}{\partial x} = I + \frac{\partial \mathcal{F}(\mathrm{LN}(x))}{\partial \mathrm{LN}(x)}\frac{\partial \mathrm{LN}(x)}{\partial x}, \tag{4}$$

where I is the identity matrix. As Equation (3), the derivative of Post-LN is equal to the product of two derivatives: one is the layer normalization, and the other consists of the residual connection and sub-layer $\mathcal{F}$. In contrast, in Pre-LN, the derivative of the residual connection is isolated from the term related to the derivative of the layer normalization. The difference between these equations implies that the residual connection in Pre-LN prevents the vanishing gradient because it retains the gradients of upper layers even if the derivative of the layer normalization decreases gradients drastically.

4 Transformations by Each Layer

As described, it is difficult to stack many layers in Post-LN Transformers because the vanishing gra-

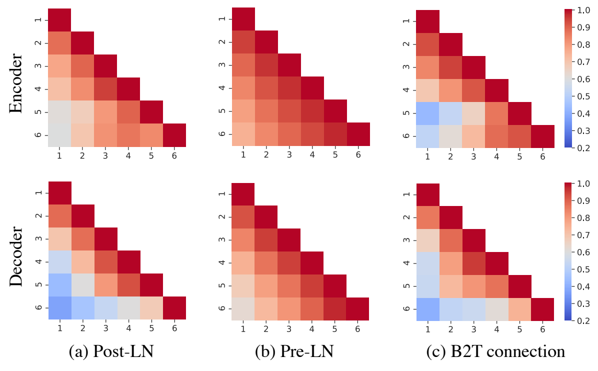


Figure 5: Cosine similarities between the outputs of each pair of layers.

dient problem occurs. Although Pre-LN is more stable in training, Post-LN can achieve better performance if training succeeds (see Section 6). In this section, we explore the reason for this difference in performance.

Focus Pre-LN in Figure 3. In contrast to Post-LN, in Pre-LN, a deeper (higher) layer has a smaller gradient norm. Thus, the parameters of higher layers are not required to change dramatically from their initial values. This implies that higher layers in Pre-LN are not sufficiently effective.

To investigate the effectiveness of higher layers, we focus on the transformations by each layer. Figure 5 shows the average cosine similarities between the outputs of each pair of layers for 6L-6L Transformer encoder-decoders trained on the WMT dataset when several sequences are input. This figure indicates that the lower-left similarities of Pre-LN are higher than those of Post-LN. This result means that the outputs of shallow layers are similar to the output of the final layer in Pre-LN, but not in Post-LN. Consequently, higher layers in Pre-LN are less effective than those in Post-LN if training succeeds.

We consider that the residual connection in Pre-LN causes this phenomenon. As Equation (2) shows, in Pre-LN, an input x skips over the sub-layer $\mathcal{F}(\cdot)$ by the residual connection. Thus, the input x is directly connected to the final layer output. This property makes the training stable, as described in Section 3, but causes high similarities between the outputs of the various layers. Therefore, we consider that Pre-LN underperforms Post-LN because the residual connection in Pre-LN reduces the effectiveness of its higher layers. In contrast, in Post-LN, larger gradient norms in higher layers (as shown in Figure 3) make higher layers more effective (as shown in Figure 5) but it is necessary to prevent the vanishing gradient problem in shallow layers when we stack many layers.

5 Modification for Stable Training in Post-LN: Bottom-to-Top Connection

This section introduces a modification that makes the training of Post-LN more stable while preserving its high performance. This modification comprises an additional residual connection to mitigate the vanishing gradient in Post-LN by enabling many layers to be stacked.

As discussed in the previous sections, we need a term that retains gradients in the derivatives, as in Equation (4), to prevent the vanishing gradient. To satisfy this requirement, we propose a residual connection that skips over all layer normalizations except the final one in each layer. Our introduced connection ties an input of a layer to the result of the feed-forward network (FFN), as illustrated by the red arrows in Figure 2 (c). We call this connection **Bottom-to-Top (B2T)** connection, which is formalized in the following equation:

$$x_{inp} + x_{ffn} + \mathrm{FFN}(x_{ffn}), \tag{5}$$

where x_{inp} is an input of a layer, $\mathrm{FFN}(\cdot)$ is an FFN, and x_{ffn} is an input of the FFN. In short, x_{inp} skips the layer normalizations after the self-attention and encoder-decoder cross-attention. Because the derivative of x_{inp} is isolated from the terms related to the derivatives of the layer normalizations just behind the attention sub-layers, it retains gradients, as in Pre-LN. For example, in an encoder-side, x_{ffn} is as follows:

$$x_{ffn} = \mathrm{LN}(\mathrm{SelfAttn}(x_{inp}) + x_{inp}), \tag{6}$$

where $\mathrm{SelfAttn}(\cdot)$ is a self-attention network. Thus, Equation (5) can be written as follows:

$$\begin{aligned} &x_{inp} + \mathrm{LN}(\mathrm{SelfAttn}(x_{inp}) + x_{inp}) \\ &\quad + \mathrm{FFN}(\mathrm{LN}(\mathrm{SelfAttn}(x_{inp}) + x_{inp})), \end{aligned} \tag{7}$$

The derivative of this equation is the following equation:

$$\begin{aligned} &I + \frac{\partial(\mathrm{LN}(\mathrm{SelfAttn}(x_{inp}) + x_{inp}))}{\partial x_{inp}} \\ &+ \frac{\partial(\mathrm{FFN}(\mathrm{LN}(\mathrm{SelfAttn}(x_{inp}) + x_{inp})))}{\partial x_{inp}}, \end{aligned} \tag{8}$$

Because this derivative contains I, which is unrelated to the derivatives of internal layer normalizations, our B2T connection (i.e., x_{inp}) helps to propagate gradients. For a decoder-side, we can prove this property in the same manner.

Figure 3 (b) indicates that B2T connection mitigates the vanishing gradient of 18L-18L encoder-decoders. Moreover, we locate B2T connection before the final layer normalization in each layer to avoid a direct connection to the final layer output based on the discussion in Section 4. Thus, B2T connection preserves the property of Post-LN with respect to the transformations performed by each layer, as illustrated in Figure 5 (c)[2].

6 Experiments

Through experiments, we indicate following three findings.

- Post-LN Transformers achieve better performance than Pre-LN Transformers if their training succeeds.
- B2T connection enables the training of deep Transformers with the Post-LN configuration.
- Our modification preserves the performance advantage of Post-LN Transformers, which therefore outperform Pre-LN Transformers.

We describe the essential experimental configurations in this section. Appendix A presents more details, such as the hyper-parameters and computational budgets.

6.1 Machine Translation

6.1.1 Dataset

The machine translation task has been widely used to investigate the performance of Transformer-based methods since the original Transformer (Vaswani et al., 2017; Ott et al., 2018; Wang et al., 2019; Xiong et al., 2020; Liu et al., 2020). We adopted the widely used WMT English-to-German training dataset (Vaswani et al., 2017; Ott et al., 2018), which contains 4.5M sentence pairs. We applied the byte-pair-encoding (BPE) algorithm (Sennrich et al., 2016) to construct a vocabulary set in the same manner as previous studies. We set the number of BPE merge operations to 32K and shared the vocabulary between the source and target languages. We used newstest2010-2016 to investigate the performance, following Takase and Kiyono (2021).

6.1.2 Methods

We compare **Post-LN**, **Pre-LN**, and Post-LN with our B2T connection (**B2T connection**) Transformers. We used `fairseq`[3] (Ott et al., 2019) as an implementation of Transformers. We stacked 6 and 18 layers for the encoders and decoders (6L-6L and 18L-18L) as the widely used configuration and deep configuration, respectively. We used the Transformer (base) setting for dimension sizes of internal layers. In addition to the above methods, we evaluate the following five methods, which are recent approaches that enable the training of deep Transformers. We used the same hyper-parameters for all methods except T-Fixup. For T-Fixup, we used the hyper-parameters reported in Huang et al. (2020) to prevent divergence.

DLCL To make Transformers deep, Wang et al. (2019) proposed dynamic linear combination of layers (DLCL), which uses the weighted sum of the lower layers as an input of a layer. In contrast to our B2T connection, which is an additional connection within each layer, DLCL uses a connection among layers. We apply DLCL to Post-LN Transformers. We used the official implementation[4].

Admin Liu et al. (2020) proposed adaptive model initialization (Admin), which uses additional parameters to stabilize the training of Post-LN Transformers. This method requires the variances of internal layers to initialize the additional parameters. Thus, this method first processes several forward steps for the initialization, and then conducts the actual training. In a nutshell, this method incurs additional computational costs. We used the official implementation[5].

T-Fixup Huang et al. (2020) proposed an initialization scheme for Transformers, T-Fixup, to perform stable training without the learning rate warm-up and layer normalizations. Because this method can remove the cause of the vanishing gradient, we can

[2]We also tried a connection that skips over all layer normalizations including the final one in each layer but it significantly impaired the performance. When we prepare such a connection, the connection ties an input to the output directly. Because this connection inhibits transformations performed by each layer as described in Section 4, it is reasonable that the performance is impaired. Therefore, we avoid skipping the final layer normalization in each layer to take the advantage of Post-LN.

[3]https://github.com/pytorch/fairseq

[4]https://github.com/wangqiangneu/dlcl

[5]https://github.com/LiyuanLucasLiu/Transformer-Clinic

Method	2010	2011	2012	2013	2014	2015	2016	Average
				Enc-Dec: 6L-6L				
Post-LN	24.27	**22.06**	**22.43**	26.11	27.13	29.70	34.40	**26.59**
Pre-LN	24.03	21.77	22.08	25.63	26.27	29.07	33.84	26.10
DLCL (Wang et al., 2019)	23.94	22.00	22.24	26.11	**27.37**	**29.71**	34.26	26.52
Admin (Liu et al., 2020)	**24.32**	21.79	22.17	26.26	27.14	29.61	34.12	26.49
T-Fixup (Huang et al., 2020)	24.09	21.98	22.04	25.96	26.92	29.45	34.56	26.43
RealFormer (He et al., 2021)	24.18	22.02	22.17	26.02	26.98	29.36	34.15	26.41
DeepNet (Wang et al., 2022)	24.08	21.76	22.09	25.90	26.85	29.62	34.39	26.38
B2T connection	24.12	21.93	22.29	**26.31**	26.84	29.48	**34.73**	26.53
				Enc-Dec: 18L-18L				
Post-LN				Training failed (See Figure 1)				N/A
Pre-LN	24.07	21.98	22.40	26.28	27.36	29.74	34.16	26.57
DLCL (Wang et al., 2019)	24.20	**22.51**	22.83	26.59	27.97	30.24	33.98	26.90
Admin (Liu et al., 2020)	24.56	22.17	22.62	26.48	27.99	30.35	33.88	26.86
T-Fixup (Huang et al., 2020)	24.45	22.29	22.76	26.57	27.71	30.13	34.69	26.94
RealFormer (He et al., 2021)	24.32	22.42	22.68	26.59	**28.58**	30.36	33.71	26.95
DeepNet (Wang et al., 2022)	**24.70**	22.40	**22.92**	**26.85**	28.21	30.60	34.25	27.13
B2T connection	24.62	**22.51**	22.86	26.74	28.48	**30.99**	**34.93**	**27.30**

Table 1: BLEU scores of each method on WMT newstest2010-2016 and their averages.

stack many layers. We used the official implementation[6].

RealFormer To improve the performance of Transformers, He et al. (2021) proposed RealFormer, which introduces additional connections into attention sub-layers. Although their motivation is not addressing the vanishing gradient problem, their method is similar to ours with respect to the use of additional connections.

DeepNet Wang et al. (2022) proposed DeepNorm, which uses a weight that corresponds to the number of layers in a residual connection before layer normalizations to stabilize Post-LN based Transformers. They also provided the combination of the initialization scheme and DeepNorm as DeepNet.

6.1.3 Results

We measured case-sensitive detokenized BLEU scores with SacreBLEU (Post, 2018)[7]. Table 1 shows BLEU scores[8] of each method on newstest2010-2016 and the averaged scores of them. Since the BLEU score is precision-based n-gram overlapping between the model output and correct examples, a higher score represents better performance.

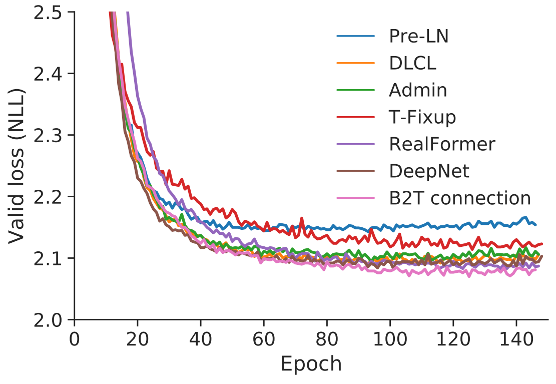


Figure 6: Negative Log-Likelihood (NLL) on validation data (newstest2013) when we stack 18 layers.

The upper part of Table 1 shows results in the 6L-6L configuration. This part indicates that Post-LN achieved better scores than Pre-LN on all test sets. In addition, B2T connection outperformed Pre-LN on all test sets. Thus, these methods are superior to Pre-LN when the total number of layers is small.

The lower part of Table 1 shows results in the 18L-18L configuration. This part shows that the training of Post-LN failed, and thus we cannot successfully stack 18L-18L in the vanilla Post-LN. With the B2T connection, its training succeeded and it outperformed Pre-LN in the 18L-18L configuration. Figure 6 shows the negative log-likelihood (NLL) values of all methods when we regard newstest2013 as validation data. This figure indicates that the NLLs of Pre-LN are worse than those of the other methods. These results demonstrate that our modification enabled the stacking of many layers

[6]https://github.com/layer6ai-labs/T-Fixup

[7]The BLEU scores calculated by SacreBLEU are often lower than those calculated by the procedure of Vaswani et al. (2017) as reported in Ott et al. (2018). In fact, Pre-LN and B2T connection in the 18L-18L configuration achieved scores of 28.94 and 29.91, respectively, on newstest2014 when we used the same procedure of Vaswani et al. (2017). However, we used SacreBLEU to ensure the compatibility of results, as described in Post (2018).

[8]The signature of SacreBLEU is BLEU+nrefs:1+case:mixed+eff:no+tok:13a+smooth:exp+version:2.0.0.

Method	2010	2011	2012	2013	2014	2015	2016	Average
Enc-Dec: 100L-100L								
Post-LN	Training failed							N/A
Pre-LN	24.81	22.67	23.15	26.98	28.42	30.50	34.53	27.29
B2T connection	**25.26**	**23.27**	**23.72**	**27.50**	**29.33**	**31.57**	**35.37**	**28.00**

Table 2: BLEU scores on WMT newstest2010-2016 and their averages in the 100L-100L configuration.

without harm to its performance such as Pre-LN.

In the comparison with the recent methods, B2T connection outperformed them with respect to the averaged BLEU score. This result implies that our modification is superior to the recent methods. To make our findings more reliable, we also conduct a comparison with the recent methods on the summarization task.

Table 2 shows results in a much deeper configuration: 100L-100L. This table also indicates that B2T connection stabilized the training and outperformed Pre-LN. Appendix C describes the details of this 100L-100L configuration and shows a comparison with the latest method, DeepNet (Wang et al., 2022).

6.2 Abstractive Summarization

6.2.1 Dataset

The abstractive summarization task is one of the most famous sequence-to-sequence problems in NLP. In this study, we conduct the experiment on the headline generation task, which is the task of generating a headline from a given sentence (Rush et al., 2015). We used headline-sentence pairs extracted from Annotated English Gigaword (Napoles et al., 2012) by Rush et al. (2015). This dataset contains 3.8M headline-sentence pairs as the training set and 1951 pairs as the test set. In addition, we used 13M additional headline-sentence pairs extracted from REALNEWS (Zellers et al., 2019) and NewsCrawl (Barrault et al., 2019) for training deep Transformers, following Takase and Kiyono (2021). We applied BPE (Sennrich et al., 2016) to construct a vocabulary set. As in the machine translation experiments, we set the number of BPE merge operations to 32K and shared the vocabulary between the encoder and decoder sides.

6.2.2 Methods

We compare **Post-LN**, **Pre-LN**, and **B2T connection** Transformers in the same manner as in Section 6.1. In addition, we compare **DLCL**, **Admin**, **T-Fixup**, **RealFormer**, and **DeepNet** because it would be premature to conclude that our modification is more effective than those methods from the results of experiments on the machine translation task alone. We set the numbers of layers of encoders and decoders to 6L-6L and 18L-18L as the base and deep configurations, respectively.

Method	R-1	R-2	R-L
Enc-Dec: 6L-6L			
Post-LN	**38.57**	**19.37**	**35.79**
Pre-LN	38.27	19.29	35.39
DLCL (Wang et al., 2019)	38.13	18.49	35.00
Admin (Liu et al., 2020)	37.96	18.93	35.05
T-Fixup (Huang et al., 2020)	38.11	19.13	35.32
RealFormer (He et al., 2021)	38.30	19.32	35.46
DeepNet (Wang et al., 2022)	38.27	18.89	35.34
B2T connection	38.43	**19.37**	35.72
Enc-Dec: 18L-18L			
Post-LN	Training failed		
Pre-LN	38.97	19.94	35.99
DLCL (Wang et al., 2019)	38.25	19.44	35.57
Admin (Liu et al., 2020)	39.10	20.08	36.30
T-Fixup (Huang et al., 2020)	39.15	19.97	36.34
RealFormer (He et al., 2021)	39.22	20.12	36.49
DeepNet (Wang et al., 2022)	39.27	19.97	36.41
B2T connection	**39.61**	**20.28**	**36.66**

Table 3: F1 based ROUGE-1, 2, and L scores (columns headed R-1, R-2, and R-L, respectively) on headline generation (Rush et al., 2015).

6.2.3 Results

Table 3 shows the ROUGE-1, 2, and L scores achieved by each method on the test set. Since these scores are computed by n-gram overlapping between the generated and correct headlines, a higher score represents better performance.

In the 6L-6L configuration, Post-LN achieved better performance than Pre-LN. Thus, Post-LN outperformed Pre-LN on the headline generation task if training succeeded. Moreover, B2T connection achieved scores comparable to those of Post-LN.

In the 18L-18L configuration, the training of Post-LN failed. In contrast, the training of B2T connection succeeded, and this method outperformed Pre-LN. Thus, our modification is more suitable than Pre-LN for training deep Transformers to perform the headline generation task.

B2T connection outperformed the recent methods in the 6L-6L configuration and achieved the best ROUGE scores in the 18L-18L configuration. According to the results on both the machine translation and headline generation tasks, B2T connection achieved performance that was better than, or comparable to, that of previous methods. It is worth emphasizing that, in addition to the performance, our modification does not incur additional computational costs, such as those incurred by DLCL and Admin.

6.3 Language Model

In addition to encoder-decoders, we investigate the effect of our B2T connection when used in the decoder side only, i.e., a neural language model. Because recent pre-trained models, such as the GPT series, are language models trained on a large amount of training data, experimental results in this section give an insight for pre-trained models.

6.3.1 Dataset

We used WikiText-103 (Merity et al., 2017), which consists of a large number of tokens. The training, validation, and test sets contain 103M, 0.2M, and 0.2M tokens, respectively. The vocabulary set contains 0.3M words.

6.3.2 Methods

We used a Transformer with adaptive input representations (Baevski and Auli, 2019), which is implemented in `fairseq`, as the base architecture in this experiment. For the base configuration, we stacked 6 layers, in the same manner as in the machine translation and summarization experiments. For the deep configuration, we used 16 layers, following Baevski and Auli (2019). For the dimensions of internal layers, we used the same values as those used by Baevski and Auli (2019). We compare Post-LN, Pre-LN, and B2T connection.

6.3.3 Results

Table 4 shows perplexities of each method on the validation and test sets of WikiText-103. Since the perplexity is computed based on the negative log-likelihood, a smaller value corresponds to better performance. The upper part of this table indicates that, with 6 layers, Post-LN and our B2T connection outperformed Pre-LN. When we stacked 16 layers, the training of Post-LN failed, but B2T connection achieved better performance than Pre-LN. These results are consistent with results on the machine translation and summarization tasks. Thus, our modification enables the training of deep Transformers for language modeling, and it is more effective than Transformers with Pre-LN.

Method	Valid	Test
Dec: 6L		
Post-LN	**20.24**	**21.22**
Pre-LN	20.98	21.93
B2T connection	20.50	21.47
Dec: 16L		
Post-LN	Training failed	
Pre-LN	18.53	19.24
B2T connection	**18.38**	**19.20**

Table 4: Perplexities on WikiText-103 (Merity et al., 2017).

	Dev		Test	
Method	Clean	Other	Clean	Other
Enc-Dec: 6L-6L				
Post-LN	3.78	**8.76**	4.19	**8.74**
Pre-LN	3.89	9.69	4.22	9.65
B2T connection	**3.69**	8.97	**3.86**	8.94
Enc-Dec: 12L-6L				
Post-LN	Training failed			
Pre-LN	**3.21**	7.91	3.49	8.22
B2T connection	3.26	**7.74**	**3.48**	**7.68**

Table 5: Word error rates of each method on LibriSpeech.

6.4 Automatic Speech Recognition

In addition to experiments on natural language processing tasks, we conduct an experiment on another modality, ASR.

6.4.1 Dataset

We used LibriSpeech (Panayotov et al., 2015), which is the standard English ASR benchmark dataset. The dataset contains 1,000 hours of English speech extracted from audiobooks. We used the standard splits of LibriSpeech: we used all available training data for training and two configurations ('clean' and 'other') of development sets and test sets for evaluation. We applied the same pre-processing as that used by Wang et al. (2020). We constructed a vocabulary set for the decoder-side with SentencePiece (Kudo and Richardson, 2018) by setting the vocabulary size to 10,000. To obtain speech features, we used torchaudio[9].

6.4.2 Methods

We used the Transformer-based speech-to-text model described in Wang et al. (2020) as the base

[9]https://github.com/pytorch/audio

architecture in this experiment. This model contains a convolutional layer to construct an embedding for the encoder-side but the other parts are identical to the Transformers used on the machine translation and summarization tasks. We used the same dimensions as those of T-Md, described in Wang et al. (2020). We set the numbers of layers to 6L-6L and 12L-6L as the base and deep configurations, respectively, because Wang et al. (2020) stacked many layers on the encoder-side only. We compare Post-LN, Pre-LN, and B2T connection.

6.4.3 Results

Table 5 shows the word error rates (WERs) of each method on each set. A smaller value of WER corresponds to better performance. The upper part of this table indicates that Post-LN and B2T connection outperformed Pre-LN on all sets in the 6L-6L configuration. The lower part of the table shows that B2T connection succeeded in training and achieved performance that was better than (or comparable to) that of Pre-LN in the 12L-6L configuration[10]. These results are consistent with those of the other experiments in this study.

7 Related Work

Layer normalization (Ba et al., 2016) is a useful technique for training neural networks but its mechanism has been unclear (Xu et al., 2019). The Transformer, which is the standard architecture for various tasks, also contains layer normalizations. The original Transformer architecture adopted the Post-LN configuration (Vaswani et al., 2017). However, recent Transformer implementations have adopted Pre-LN configurations (Klein et al., 2017; Vaswani et al., 2018; Ott et al., 2019; Baevski and Auli, 2019).

To construct deep Transformers that achieve better performance, recent studies have focused on the behavior of layer normalizations. Wang et al. (2019) indicated the difficulty of training deep Transformers with Post-LN due to the vanishing gradient problem, and demonstrated that Pre-LN enables the stacking of many layers through machine translation experiments. In addition, they proposed a method to connect all layers to increase the effectiveness of deep Transformers. Bapna et al. (2018) and Dou et al. (2018) also proposed such connection methods to stack many layers. He et al. (2021) introduced additional connections into attention sub-layers to improve the performance. Xiong et al. (2020) explored the relation between the warm-up strategy and layer normalizations in Transformers. Through theoretical and empirical analyses, they indicated that Post-LN requires the warm-up strategy to stabilize the training.

Liu et al. (2020) analyzed the training dynamics of Post-LN and Pre-LN Transformers. They then proposed Admin, which consists of additional weight parameters to control the variances of outputs from each sub-layer. In contrast, we indicated that we can stabilize the training of Post-LN Transformers by adding only a residual connection that skips over layer normalizations that cause the vanishing gradient.

Some studies have proposed initialization methods to make the training of deep neural networks stable (Zhang et al., 2019a,b; Huang et al., 2020). Zhang et al. (2019a) proposed the depth-scaled initialization to prevent the vanishing gradient problem in Transformers. Zhang et al. (2019b) proposed the fixed-update initialization to remove normalizations in neural networks. Inspired by these studies, Huang et al. (2020) proposed T-Fixup, which enables both warm-up and layer normalizations to be removed from Transformers. In addition to the initialization scheme, Wang et al. (2022) introduced weights into residual connections before layer normalizations, following Liu et al. (2020).

8 Conclusion

In this study, we addressed the stability of training Post-LN Transformers. Through theoretical and empirical analyses, we indicated that layer normalizations cause the unstable training when many layers are stacked. In addition, we investigated the reason for the different performance of Pre-LN and Post-LN by transformations of each layer. We introduced B2T connection to prevent the vanishing gradient while preserving the advantage of Post-LN. We conducted experiments on various tasks. The experimental results led to the following three findings; 1, Post-LN achieved better performance than Pre-LN if its training succeeded. 2, Our modification enabled the training of deep Transformers (e.g., those with ten or more layers). 3, Our modification preserved the benefit of Post-LN, and therefore outperformed Pre-LN.

[10] Wang et al. (2020) reported that the improvement was small even if they increased the number of parameters. Thus, we emphasize that B2T connection achieved better WERs on dev-other and test-other even though the number of parameters of B2T connection is (almost) equal to that of Pre-LN.

Limitations

In this paper, we indicated that the vanishing gradient problem, caused by layer normalizations, makes the training of deep Post-LN Transformers unstable. We proposed the B2T connection to mitigate this vanishing gradient problem. However, the proposed B2T connection does not perfectly prevent the vanishing gradient, as shown in Figure 3. Therefore, the vanishing gradient might harm the training in extremely deep Transformers even if our B2T connection is used.

In addition, this study depends on empirical observations. In particular, we provided little theoretical justification of the reason for Post-LN outperforming Pre-LN when training succeeds. However, as discussed in Appendix C, the method with a theoretical justification often collapses in some situations. Because the behavior of deep Transformers in various situations is not fully understood, we believe that it is important to provide empirical findings for our research field to progress.

Although Appendix C includes a comparison between our B2T connection and the latest method, DeepNet (Wang et al., 2022), we could not investigate the behavior of all methods in the 100L-100L configuration because of our limited computational budgets. However, we are confident that we conducted sufficient experiments to verify our contributions.

Ethics Statement

The proposed method helps to construct deep Transformers. As discussed in Strubell et al. (2019) and Schwartz et al. (2019), such deep neural networks consume substantial amounts of energy. In fact, as discussed in Appendix A.2, we spent a large amount of computational resources on our experiments. Therefore, we also need to explore methods of improving energy efficiency while maintaining the good performance achieved by stacking many layers.

With respect to ethical considerations, the datasets used in our experiments are publicly available. LibriSpeech (Panayotov et al., 2015) is derived from audiobooks. The other datasets are mainly constructed from newswire texts and Wikipedia. Thus, in our understanding, our used datasets do not contain any personally identifiable information or offensive contents.

Acknowledgements

We thank the anonymous reviewers for their useful suggestions. A part of this work was supported by JSPS KAKENHI Grant Number JP21K17800 and JST ACT-X Grant Number JPMJAX200I. The work of Jun Suzuki was partly supported by JST Moonshot R&D Grant Number JPMJMS2011 (fundamental research). We thank Edanz for editing a draft of this manuscript.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*.

Ankur Bapna, Mia Chen, Orhan Firat, Yuan Cao, and Yonghui Wu. 2018. Training deeper neural machine translation models with transparent attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3028–3033.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (WMT)*, pages 1–61.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, pages 1877–1901.

Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Shuming Shi, and Tong Zhang. 2018. Exploiting deep representations for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4253–4262.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Identity mappings in deep residual networks. In *14th European Conference on Computer Vision*, pages 630–645.

Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. 2021. RealFormer: Transformer likes residual attention. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 929–943.

Xiao Shi Huang, Felipe Perez, Jimmy Ba, and Maksims Volkovs. 2020. Improving transformer optimization through better initialization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 4475–4483.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37, pages 448–456.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 67–72.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 66–71.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer Sentinel Mixture Models. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 95–100.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 48–53.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation (WMT)*, pages 1–9.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation (WMT)*, pages 186–191.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 379–389.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. *CoRR*, abs/1907.10597.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1715–1725.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 2377—-2385.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3645–3650.

Sho Takase and Shun Kiyono. 2021. Rethinking perturbations in encoder-decoders for fast training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 5767–5780.

Sho Takase and Naoaki Okazaki. 2019. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*, pages 3999–4004.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2Tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas*, pages 193–199.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5998–6008.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. Fairseq S2T: Fast speech-to-text modeling with fairseq. 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 (AACL-IJCNLP)*, pages 33–39.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. 2022. Deepnet: Scaling transformers to 1,000 layers.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1810–1822.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. 2020. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 10524–10533.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. 2019. Understanding and improving layer normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 9054–9065.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019a. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 898–909.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. 2019b. Fixup initialization: Residual learning without normalization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*.

A Details of Experimental Settings

A.1 Hyper-parameters

As described in Section 6, our hyper-parameters follow those used in previous studies. Table 6 shows hyper-parameters used for each experiment. For fair comparisons, we used the same hyper-parameters for all methods except T-Fixup. For T-Fixup, we used hyper-parameters reported in Huang et al. (2020) to prevent divergence.

A.2 Computational Resources

We mainly used NVIDIA Tesla P100 GPUs for most of our experiments. Table 7 shows the number of GPUs and the computational time used to construct one model in our experiments. For the 100L-100L configuration, described in Section 6.1, we used 24 Tesla V100 GPUs and spent approximately 120 hours to train one model.

B Supplementary of Gradient Norms of Each Location

For gradient norms of each part in a layer, we check 1st and 9th decoders in addition to the 18th decoder for the 18L-18L Post-LN Transformer encoder-decoder as shown in Figure 4. Figure 7 shows the gradient norms of each part. This figure shows that the gradient norms decrease drastically through layer normalizations in the same manner as they do in the 18th decoder (Figure 4). Therefore, the vanishing gradient problem in Post-LN Transformers is probably caused by layer normalizations.

C Details of the 100L-100L Configuration

C.1 Regularizations during the Training

As reported in Section 1, we constructed 100L-100L Transformers with widely-used WMT English-to-German dataset. In the preliminary experiments, we found that regularization is the key to preventing overfitting and achieving high performance in this situation. Figure 8 shows the NLL values of Pre-LN and B2T connection on validation data in the 36L-36L configuration when we used the same hyper-parameters as those used in 6L-6L and 18L-18L configurations. As this figure shows, the NLL values began to increase from the middle of training, and thus the overfitting occurred. In addition, the use of the same hyper-parameters as 6L-6L and 18L-18L makes it difficult to improve the performance of deeper configurations. Figure 9 shows the best NLL values on validation data when we varied the number of layers: 6L-6L, 12L-12L, 18L-18L, 36L-36L, and 50L-50L[11]. This figure indicates that adding more layers to the 18L-18L configuration did not improve the performance.

To prevent overfitting during the training of 100L-100L Transformers, we increased the dropout rate from 0.3 to 0.5. In addition, we used word dropout, as described in Takase and Kiyono (2021). We set the word dropout rate to 0.1 for the encoder and decoder. We multiplied the initial parameter values, except those for embeddings, by 0.1. We set the gradient clipping to 0.1. Finally, we decreased the number of updates from 50K to 25K. These regularization techniques prevented overfitting and achieved better performance than 18L-18L, as described in Section 6.1.

C.2 Comparison with DeepNet

As described in Section 7, various studies have attempted to stabilize the training of deep Transformers. Each study indicated the effectiveness of their proposed method empirically, and some have provided theoretical justifications. However, Wang et al. (2022) demonstrated that the training of previous methods except DeepNet failed in a much deeper configuration than normally used, i.e., 100L-100L. Then, can we conclude that DeepNet is a silver bullet for deep Transformers? It is difficult to reach this conclusion because the training of DeepNet also fails in some configurations. For example, when we train deep Transformers, we might decrease the batch size because the trainable parameters occupy most of the GPU memories. When we tried this, the NLL value of DeepNet on validation data diverged, as shown in Figure 10. In other words, the training of DeepNet failed. In contrast, the training of our B2T connection succeeded in this situation. This result implies that there are problems in the training of deep Transformers that have not been solved in previous studies. Therefore, we believe that we should continue to add the empirical findings about new techniques, including B2T connection, to those of previous studies.

D B2T Connection without Layer Normalization

In addition to B2T connection, we also consider a further modification to prevent the vanishing gra-

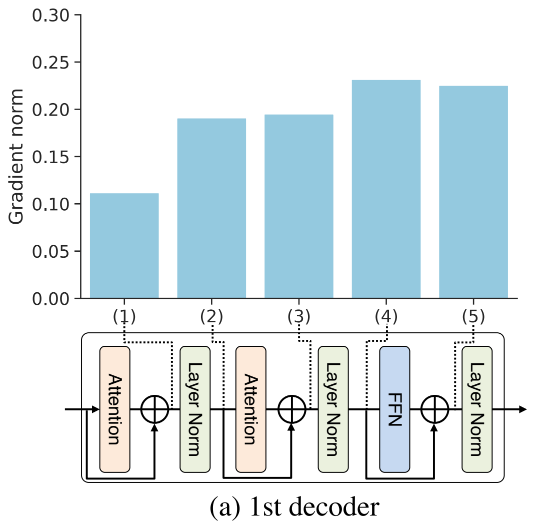
[11]The horizontal axis of Figure 9 represents the total number of layers, which are divided equally between the encoder and decoder. For example, 100 on the horizontal axis represents 50L-50L Transformers.

Params	Machine Translation	Abstractive Summarization	Language Model	ASR
Hidden dim size	512	512	1024	512
FFN dim size	2048	2048	4096	2048
Attention heads	8	8	8	8
Learning rate	0.001	0.001	0.001	0.001
Scheduler	inverse sqrt	inverse sqrt	inverse sqrt	inverse sqrt
Adam β	(0.9, 0.98)	(0.9, 0.98)	(0.9, 0.98)	(0.9, 0.98)
Warmup updates	4K	4K	2K	4K
Max updates	50K	50K	50K	150K
Max tokens / GPU	3584	3584	1024	40K

Table 6: Hyper-parameters used in our experiments.

	Machine Translation		Abstractive Summarization		Language Model		ASR	
	6L-6L	18L-18L	6L-6L	18L-18L	6L	16L	6L-6L	12L-6L
#GPU	128	128	64	144	128	192	32	32
Time (hour)	5	13	4	17	4	7	22	34

Table 7: The number of GPUs and computational time used to construct one model in our experiments.



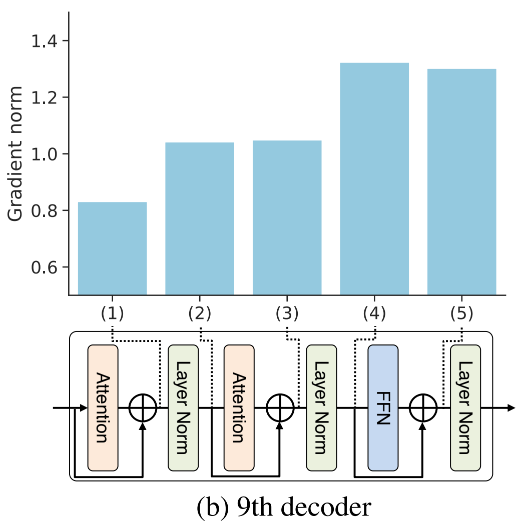


Figure 7: Gradient norms of each part in the (a) 1st decoder and (b) 9th decoder of the 18L-18L Post-LN Transformer encoder-decoder on WMT English-to-German translation training data.

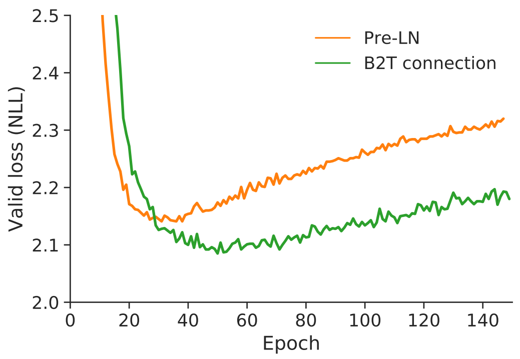


Figure 8: Negative Log-Likelihood (NLL) values of Pre-LN and our proposed B2T connection on validation data (newstest2013) in 36L-36L. We used the same hyper-parameters as those used in 6L-6L and 18L-18L.

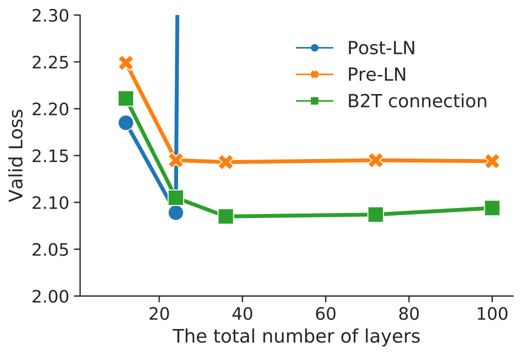


Figure 9: The best Negative Log-Likelihood (NLL) values on validation data (newstest2013) when the total number of layers is varied. The total number of layers is divided equally between the encoder and decoder.

dient problem. Because layer normalizations decrease gradients drastically, as described in Section 3, removing layer normalizations may provide stable gradients during back-propagation. However,

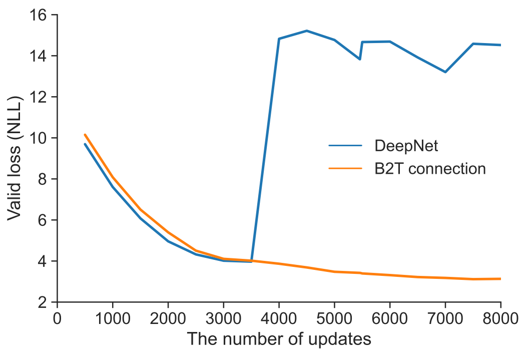


Figure 10: Negative Log-Likelihood (NLL) values of DeepNet and our proposed B2T connection on validation data (newstest2013) in 100L-100L with a small batch size.

the values in the forward pass increase exponentially if layer normalizations are removed. Therefore, we introduce weights that prevent the explosive increase in the forward pass while mitigating the decreasing gradients in back-propagation, as an alternative to the layer normalization. To use this alternative, we replace Equation (5) with the following equation:

$$\alpha x_{inp} + \beta \left(x_{ffn} + \mathrm{FFN}(x_{ffn}) \right). \tag{9}$$

Through several experiments[12], we found that the following values of α and β are suitable:

$$\alpha = \min\left(\frac{N}{12}, N^{-0.15} \right), \tag{10}$$

$$\beta = d^{-0.2}, \tag{11}$$

where N is the number of layers and d is the dimension of the input vectors x_{inp}. For example, N is set to 12 and 6 in the encoder and decoder, respectively, in the 12L-6L configuration. Therefore, as the number of layers increases, the value of α increases while N remains small (until $N = 9$), and then α starts to decrease. In short, α prevents an explosive increase in the forward pass when we stack many layers. β decreases as the dimension d increases, and thus it prevents an explosive increase when a large dimension is used. By using Equation (9), we can remove all layer normalizations in internal layers. This solves the vanishing gradient problem caused by layer normalizations.

Tables 8 and 9 shows the results of B2T connection without layer normalizations ("w/o LN") on the machine translation and summarization tasks. These results indicate that B2T connection without layer normalizations achieved scores comparable to those of B2T connection with layer normalizations. However, because the results of B2T connection without layer normalizations are slightly worse than those with layer normalizations, we recommend the use of B2T connection with layer normalizations.

[12]We could instead tune α and β to improve performance on each task but here we define values that are useful for various tasks.

Method	2010	2011	2012	2013	2014	2015	2016	Average
Enc-Dec: 6L-6L								
B2T connection	24.12	21.93	**22.29**	**26.31**	26.84	29.48	**34.73**	**26.53**
+ w/o LN	**24.17**	**22.07**	22.24	25.83	**26.96**	**29.70**	34.42	26.48
Enc-Dec: 18L-18L								
B2T connection	**24.75**	**22.88**	**23.09**	**27.12**	**28.82**	**30.99**	**33.64**	**27.33**
+ w/o LN	24.47	22.37	22.58	27.04	28.34	30.49	**34.38**	27.10

Table 8: BLEU scores of our modifications on WMT newstest2010-2016 and their averages.

Method	R-1	R-2	R-L
Enc-Dec: 6L-6L			
B2T connection	38.43	19.37	35.72
+ w/o LN	**38.63**	**19.75**	**35.77**
Enc-Dec: 18L-18L			
B2T connection	**39.61**	**20.28**	**36.66**
+ w/o LN	39.29	20.01	36.48

Table 9: F1 based ROUGE scores of our modifications on headline generation.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations section, after Conclusion section

☑ A2. Did you discuss any potential risks of your work?
Ethics Statement section, after Conclusion section

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Sections 1 and 8

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 5

☑ B1. Did you cite the creators of artifacts you used?
Sections 6 and 7

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Ethics Statement section

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Ethics Statement section

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 6

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 6

C ☑ Did you run computational experiments?

Section 6 and Appendices A, B, C, and D

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendices A and C

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Not applicable. Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 6 and Appendices A

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.