KOREA UNIVERSITY

DSBA Transformer survey paper study

A Survey of Transformers

#5: Other Module-Level Modifications

arXiv preprint

DSBA
Data Science & Business Analytics

고려대학교 산업경영공학과

Data Science & Business Analytics Lab

이유경, 김명섭, 윤훈상, 김지나, 허재혁, 김수빈

발표자 : 허재혁

05

발표 목차

Contents

05 Position Representations

Overview

- **2017**
  - Vanilla Transformer, NIPS (24,273 citations)
  - Convolutional Sequence to Sequence Learning, ICML (2,282 citations)
- **2018**
  - Self-Attention with Relative Position Representations, NAACL (594 citations)
- **2019**
  - BERT, ACL (21,997 citaitons)
  - InDIGO, TACL (67 citations)
  - Music Transformer, ICLR (234 citations)
  - Transformer-XL, ACL (1,190 citations)
  - R-Transformer, arXiv (24 citations)
  - Transformer Dissection- An Unified Understanding for Transformer's Attention via the Lens of Kernel, EMNLP (34 citations)
  - Language Modeling with Deep Transformers, INTERSPEECH (77 citations)
- **2020**
  - FLOATER, ICML (18 citations)
  - Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, JMLR (1,423 citations)
  - Encoding word order in complex embeddings, ICLR (23 citations)
- **2021**
  - On Position Embeddings in BERT, ICLR (7 citations)
  - DeBERTa, ICLR (70 citations)
  - TUPE, ICLR (21 citations)
  - RoFormer, arXiv (4 citations)
  - Conditional Positional Encodings for Vision Transformers, arXiv (19 citations)
  - Linear Transformers Are Secretly Fast Weight Memory Systems, ICML (5 citations)

Absolute Position Representations

Relative Position Representations

Other Representations

Position Representations without Explicit Encoding

Position Representation on Transformer Decoders

05 Position Representations

Overview

Absolute Position Representations

- 입력 문장에 대해 token별 위치 정보를 제공하는 방법

Relative Position Representations

- Token 간의 상대적인 위치 정보를 제공하는 방법

Other Representations

- Absolute & Relative Position을 모두 활용하는 등 다른 방식의 위치 정보 제공하는 방법

Position Representations without Explicit Encoding

- 위치 정보를 별도로 제공하는 것이 아닌 내재적인 위치 정보를 받는 방법

Position Representation on Transformer Decoders

- Transformer decoder에서의 위치 정보 사용하는 방법

05 Position Representations

Absolute Position Representations

Absolute Position Representations

- 입력 문장에 대해 token별 위치 정보를 제공하는 방법

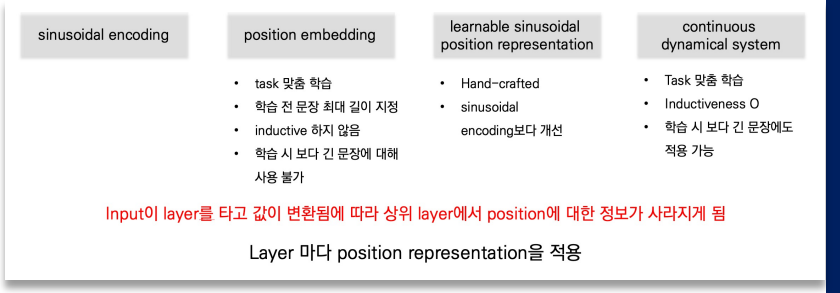
sinusoidal encoding	position embedding	learnable sinusoidal position representation	continuous dynamical system
	• task 맞춤 학습 • 학습 전 문장 최대 길이 지정 • inductive 하지 않음 • 학습 시 보다 긴 문장에 대해 사용 불가	• Hand-crafted • sinusoidal encoding보다 개선	• Task 맞춤 학습 • Inductiveness O • 학습 시 보다 긴 문장에도 적용 가능

Input이 layer를 타고 값이 변환됨에 따라 상위 layer에서 position에 대한 정보가 사라지게 됨

Layer 마다 position representation을 적용

05 Position Representations

Absolute Position Representations



Positional Encodings

방법 1

- 입력 문장의 token에 0~1사이 값으로 위치 정보를 반영

I	will	be	the	pirate	king
0	0.2	0.4	0.6	0.8	1

한계점

- 문장이 얼마나 많은 token을 내포하고 있는지 알 수 없음
- 특정 문장에서의 단어 간 거리가 다른 문장에서 다를 수 있음

05 Position Representations

Absolute Position Representations

sinusoidal encoding	position embedding	learnable sinusoidal position representation	continuous dynamical system
	• task 맞춤 학습 • 학습 전 문장 최대 길이 지정 • inductive 하지 않음 • 학습 시 보다 긴 문장에 대해 사용 불가	• Hand-crafted • sinusoidal encoding보다 개선	• Task 맞춤 학습 • Inductiveness O • 학습 시 보다 긴 문장에도 적용 가능

Input이 layer를 타고 값이 변환됨에 따라 상위 layer에서 position에 대한 정보가 사라지게 됨

Layer 마다 position representation을 적용

Positional Encodings

방법 2

- 문장 내 Token마다 1부터 N(문장 내 token 수)까지 정수 값을 순차적으로 부여

I	will	be	the	pirate	king
1	2	3	4	5	6

한계점

- 문장의 길이가 길어지면 큰 값이 입력 값으로 주어 질 수 있음
- 평가 시 학습데이터 보다 긴 문장이 들어오는 경우 일반화 성능이 떨어질 수 있음

05 Position Representations

Absolute Position Representations

sinusoidal encoding	position embedding	learnable sinusoidal position representation	continuous dynamical system
	• task 맞춤 학습 • 학습 전 문장 최대 길이 지정 • inductive 하지 않음 • 학습 시 보다 긴 문장에 대해 사용 불가	• Hand-crafted • sinusoidal encoding보다 개선	• Task 맞춤 학습 • Inductiveness O • 학습 시 보다 긴 문장에도 적용 가능

Input이 layer를 타고 값이 변환됨에 따라 상위 layer에서 position에 대한 정보가 사라지게 됨

Layer 마다 position representation을 적용

Sinusoidal Encodings

Attention is All You Need, 2017, NIPS (24,273 citations)

- sine & cosine 사용하여 앞선 문제 해결

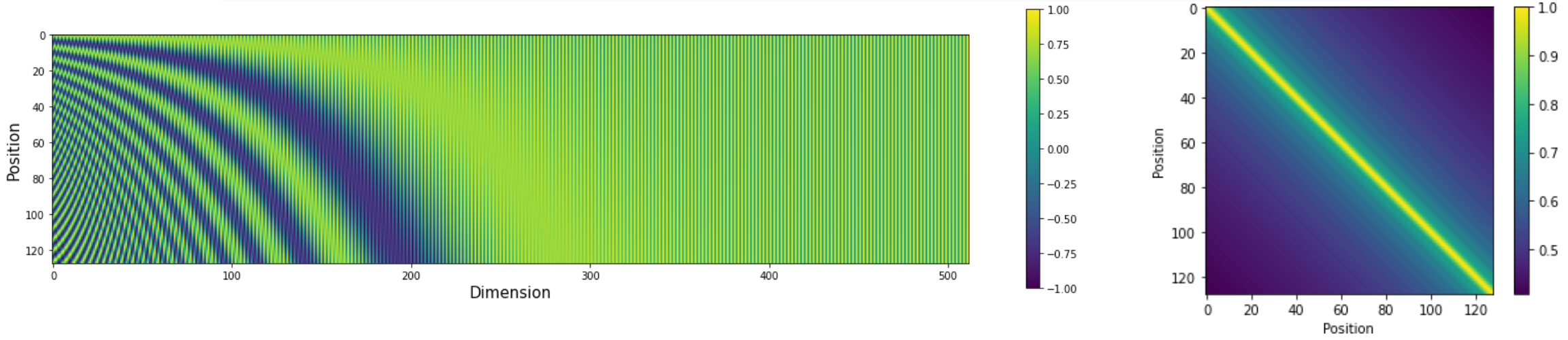
$$\mathrm{PE}(t)_i = \begin{cases} \sin(\omega_i t) & \text{if } i \text{ is even} \\ \cos(\omega_i t) & \text{if } i \text{ is odd} \end{cases}$$

A Survey of Transformers

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

Attention is All You Need

- t = position
- $\omega_i = 1/10000^{2i/d_{model}}$



05 Position Representations

Absolute Position Representations

Positional Encodings

Attention is All You Need, 2017, NIPS (24,273 citations)

- sine & cosine 사용하여 앞선 문제 해결

  - ✓ Position 마다 고유값을 가짐
  - ✓ 서로 다른 길이의 문장에도 token간 거리가 일정
  - ✓ 학습 시 등장하지 않은 문장 길이도 반영 가능
  - ✓ 항상 동일한 값으로 생성

Encoding이라서 가능

sinusoidal encoding	position embedding	learnable sinusoidal position representation	continuous dynamical system
	• task 맞춤 학습 • 학습 전 문장 최대 길이 지정 • inductive 하지 않음 • 학습 시 보다 긴 문장에 대해 사용 불가	• Hand-crafted • sinusoidal encoding보다 개선	• Task 맞춤 학습 • Inductiveness O • 학습 시 보다 긴 문장에도 적용 가능

Input이 layer를 타고 값이 변환됨에 따라 상위 layer에서 position에 대한 정보가 사라지게 됨

Layer 마다 position representation을 적용

We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, $\boldsymbol{PE_{pos+k}}$ can be represented as a linear function of $\boldsymbol{PE_{pos}}$

$$M.\begin{bmatrix}\sin(\omega_k.t)\\ \cos(\omega_k.t)\end{bmatrix}=\begin{bmatrix}\sin(\omega_k.(t+\phi))\\ \cos(\omega_k.(t+\phi))\end{bmatrix}$$

Source: https://timodenk.com/blog/linear-relationships-in-the-transformers-positional-encoding/

05 Position Representations

Absolute Position Representations

sinusoidal encoding	position embedding	learnable sinusoidal position representation	continuous dynamical system
	• task 맞춤 학습 • 학습 전 문장 최대 길이 지정 • inductive 하지 않음 • 학습 시 보다 긴 문장에 대해 사용 불가	• Hand-crafted • sinusoidal encoding보다 개선	• Task 맞춤 학습 • Inductiveness O • 학습 시 보다 긴 문장에도 적용 가능

Input이 layer를 타고 값이 변환됨에 따라 상위 layer에서 position에 대한 정보가 사라지게 됨

Layer 마다 position representation을 적용

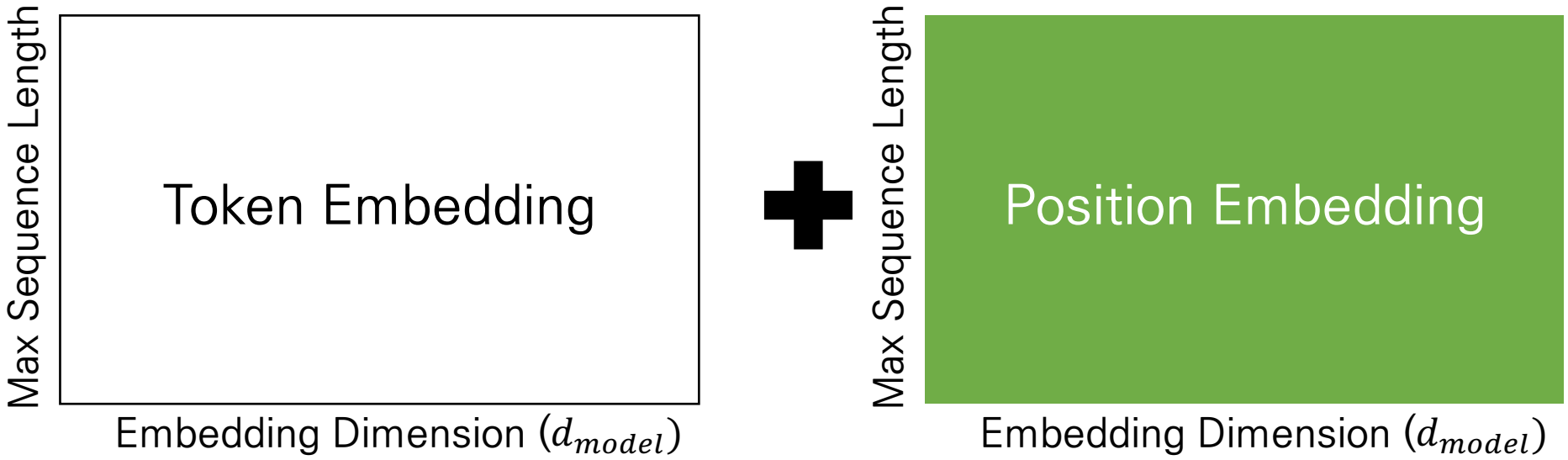
Positional Embedding

Attention is All You Need, 2017, NIPS (24,273 citations)

Convolutional Sequence to Sequence Learning, 2017, ICML (2,282 citations)

BERT, 2019, ACL (21,997 citaitons)

- 입력 token의 Position 정보를 학습하여 제공
- task에 맞게 학습하여 hand-crafted position representation에 비해 유연성이 있음



05 Position Representations

Relative Position Representations

Relative Position Representations

- Token 간의 상대적인 위치 정보를 제공하는 방법

Relative Position Embedding

최대 거리(k)를 정함

- 학습 시 없던 길이의 문장에도 적용 가능

최대 거리(k)를 3으로 사용

- 유연한 문장 생성 가능

Relative position 연산량 개선

Relative position bias만 사용

- 파라미터 수 절감

sinusoidal matrix를 활용하여 contents와 섞어서 사용

position embedding으로 대체 disentangle 형태

05 Position Representations

Relative Position Representations

Relative Position Embedding	Relative position 연산량 개선	sinusoidal matrix를 활용하여 contents와 섞어서 사용
최대 거리(k)를 정함 • 학습 시 없던 길이의 문장에도 적용 가능	최대 거리(k)를 3으로 사용 • 유연한 문장 생성 가능	Relative position bias만 사용 • 파라미터 수 절감	position embedding으로 대체 disentangle 형태

Relative Position Emebedding

Self-Attention with Relative Position Representations, 2018, NAACL (594 citations)

- Key와 Query에 relative position embedding 추가

$$e_{ij} = \frac{x_i W^Q (x_j W^K + a_{ij}^K)^T}{\sqrt{d_z}} \qquad \alpha_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{n} \exp e_{ik}} \qquad z_i = \sum_{j=1}^{n} \alpha_{ij} (x_j W^V + a_{ij}^V)$$

d_z: Attention Head 차원

$$a_{ij}^K = w_{\mathrm{clip}(j-i,k)}^K$$
$$a_{ij}^V = w_{\mathrm{clip}(j-i,k)}^V$$
$$\mathrm{clip}(x,k) = \max(-k, \min(k,x))$$

We then learn relative position representations $w^K = (w_{-k}^K, \ldots, w_k^K)$ and $w^V = (w_{-k}^V, \ldots, w_k^V)$ where $w_i^K, w_i^V \in \mathbb{R}^{d_a}$.

05 Position Representations

Relative Position Representations

Relative Position Embedding		Relative position 연산량 개선	sinusoidal matrix를 활용하여 contents와 섞어서 사용
최대 거리(k)를 정함 • 학습 시 없던 길이의 문장에도 적용 가능	최대 거리(k)를 3으로 사용 • 유연한 문장 생성 가능	Relative position bias만 사용 • 파라미터 수 절감	position embedding으로 대체 disentangle 형태

Relative Position Emebedding

InDIGO, 2019, TACL (67 citations)

- 매 스텝마다 position이 바뀜

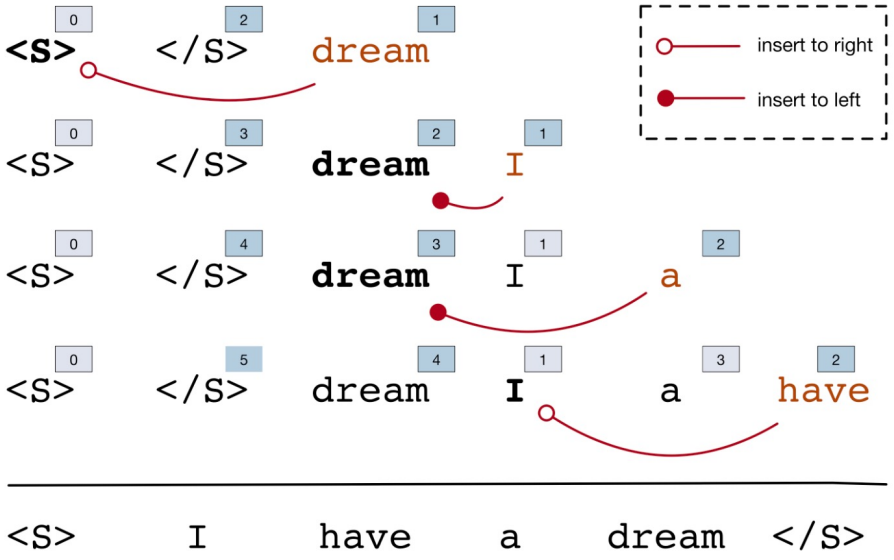


Figure 1: An example of InDIGO. At each step, we simultaneously predict the next token and its (relative) position to be inserted. The final output sequence is obtained by mapping the words based on their positions.

05 Position Representations

Relative Position Representations

Relative Position Embedding		Relative position 연산량 개선	sinusoidal matrix를 활용하여 contents와 섞어서 사용
최대 거리(k)를 정함 • 학습 시 없던 길이의 문장에도 적용 가능	최대 거리(k)를 3으로 사용 • 유연한 문장 생성 가능	Relative position bias만 사용 • 파라미터 수 절감	position embedding으로 대체 disentangle 형태

Relative Position Emebedding

InDIGO, 2019, TACL (67 citations)

- Ternary vector 사용하여 R 행렬을 생성한 후 고정해서 사용

$$\boldsymbol{r}_{i,j}^{t} = \begin{cases} -1 & z_j^t > z_i^t \ \text{(left)} \\ 0 & z_j^t = z_i^t \ \text{(middle)} \\ 1 & z_j^t < z_i^t \ \text{(right)} \end{cases}$$

$$z_i^t = \sum_{j=0}^{t} \max(0, \boldsymbol{r}_{i,j}^{t})$$

t: token position

$$R^{t+1} = \left[\begin{array}{ccc|c} & R^t & & \boldsymbol{r}_{t+1,0}^{t+1} \\ & & & \vdots \\ & & & \boldsymbol{r}_{t+1,t}^{t+1} \\ \hline -\boldsymbol{r}_{t+1,0}^{t+1} & \cdots & -\boldsymbol{r}_{t+1,t}^{t+1} & 0 \end{array}\right]$$

05 Position Representations

Relative Position Representations

Relative Position Embedding
- 최대 거리(k)를 정함
  - 학습 시 없던 길이의 문장에도 적용 가능
- 최대 거리(k)를 3으로 사용
  - 유연한 문장 생성 가능

Relative position 연산량 개선
- Relative position bias만 사용
  - 파라미터 수 절감

sinusoidal matrix를 활용하여 contents와 섞어서 사용
- position embedding으로 대체 disentangle 형태

Relative Position 연산량 개선

Music Transformer, 2019, ICLR (234 citations)

- 이전과 같은 방식으로 relative position embedding 학습

$$\text{RelativeAttention} = \text{Softmax}\left(\frac{QK^\top + S^{rel}}{\sqrt{D_h}}\right)V.$$

$$S^{rel} = QR^\top$$

05 Position Representations

Relative Position Representations

Relative Position Embedding

최대 거리(k)를 정함
- 학습 시 없던 길이의 문장에도 적용 가능

최대 거리(k)를 3으로 사용
- 유연한 문장 생성 가능

Relative position 연산량 개선

Relative position bias만 사용
- 파라미터 수 절감

sinusoidal matrix를 활용하여 contents와 섞어서 사용

position embedding으로 대체 disentangle 형태

Relative Position 연산량 개선

Music Transformer, 2019, ICLR (234 citations)

- "Skewing" 과정을 통해 연산량 감소 $O(L^2D) \rightarrow O(LD)$

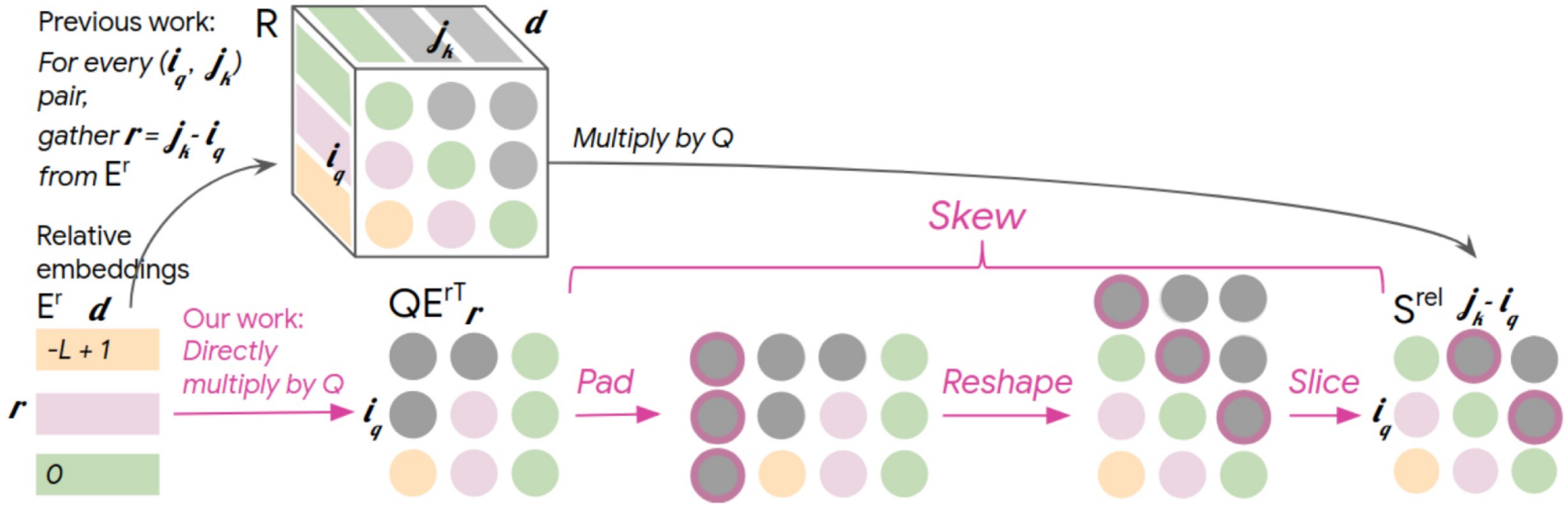


Figure 1: Relative global attention: the bottom row describes our memory-efficient "skewing" algorithm, which does not require instantiating R (top row, which is $O(L^2D)$). Gray indicates masked or padded positions. Each color corresponds to a different relative distance.

05 Position Representations

Relative Position Representations

Relative Position Embedding

최대 거리(k)를 정함
- 학습 시 없던 길이의 문장에도 적용 가능

최대 거리(k)를 3으로 사용
- 유연한 문장 생성 가능

Relative position 연산량 개선

Relative position bias만 사용
- 파라미터 수 절감

sinusoidal matrix를 활용하여 contents와 섞어서 사용

position embedding으로 대체 disentangle 형태

Relative Position Bias만 사용

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, 2020, JMLR (1,423 citations)

- 각 head 마다 개별적으로 생성하여 모든 layer에서 sharing하여 사용

$$\alpha_{ij}^{T5} = \frac{1}{\sqrt{d}}(x_i^l W^{Q,l})(x_j^l W^{K,l})^T + b_{j-i}.$$

05 Position Representations

Relative Position Representations

Relative Position Embedding | Relative position 연산량 개선 | sinusoidal matrix를 활용하여 contents와 섞어서 사용

최대 거리(k)를 정함
- 학습 시 없던 길이의 문장에도 적용 가능

최대 거리(k)를 3으로 사용
- 유연한 문장 생성 가능

Relative position bias만 사용
- 파라미터 수 절감

position embedding으로 대체 disentangle 형태

Sinusoidal matrix를 활용하여 contents와 섞어서 사용

Transformer-XL, 2019, ACL (1,190 citations)

- Position content와 별개로 적용

$$\mathbf{A}^{\text{rel}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(b)} + \underbrace{u^\top \mathbf{W}_{k,E} \mathbf{E}_{x_j}}_{(c)} + \underbrace{v^\top \mathbf{W}_{k,R} \mathbf{R}_{i-j}}_{(d)}.$$

(a) Query – Key

(b) Query – Position

(c) Global Content Bias

(d) Global Position Bias

$\mathbf{R}$는 sinusoidal matrix

u, v는 학습 파라미터

05 Position Representations

Relative Position Representations

Relative Position Embedding	Relative position 연산량 개선	sinusoidal matrix를 활용하여 contents와 섞어서 사용
최대 거리(k)를 정함 • 학습 시 없던 길이의 문장에도 적용 가능 최대 거리(k)를 3으로 사용 • 유연한 문장 생성 가능	Relative position bias만 사용 • 파라미터 수 절감	position embedding으로 대체 disentangle 형태

Position embedding으로 대체 disentangle 형태

DeBERTa, 2021, ICLR (70 citations)

$$\delta(i,j) = \begin{cases} 0 & \text{for} \quad i-j \leqslant -k \\ 2k-1 & \text{for} \quad i-j \geqslant k \\ i-j+k & \text{others.} \end{cases}$$

$$\boldsymbol{Q_c} = \boldsymbol{HW_{q,c}}, \boldsymbol{K_c} = \boldsymbol{HW_{k,c}}, \boldsymbol{V_c} = \boldsymbol{HW_{v,c}}, \boldsymbol{Q_r} = \boldsymbol{PW_{q,r}}, \boldsymbol{K_r} = \boldsymbol{PW_{k,r}}$$

$$\tilde{A}_{i,j} = \underbrace{\boldsymbol{Q_i^c K_j^{cT}}}_{\text{(a) content-to-content}} + \underbrace{\boldsymbol{Q_i^c K_{\delta(i,j)}^{rT}}}_{\text{(b) content-to-position}} + \underbrace{\boldsymbol{K_j^c Q_{\delta(j,i)}^{rT}}}_{\text{(c) position-to-content}}$$

$$\boldsymbol{H_o} = \text{softmax}\left(\frac{\tilde{\boldsymbol{A}}}{\sqrt{3d}}\right)\boldsymbol{V_c}$$

[Paper Review] DeBERTa (DSBA 석사과정 김수빈)

05 Position Representations

Other Representations

Other Representations

- Absolute & Relative Position을 모두 활용하는 등 다른 방식의 위치 정보 제공하는 방법

absolute와 relative position을 함께 적용

Rotation matrix를 적용하여 사용

05 Position Representations

Other Representations

absolute와 relative position을 함께 적용

Rotation matrix를 적용하여 사용

Absolute와 relative position을 함께 적용

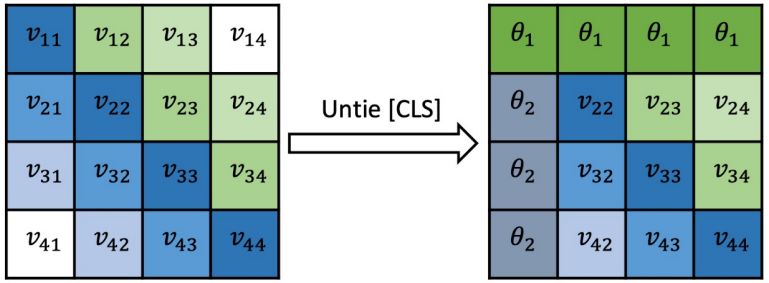
TUPE, 2021, ICLR (21 citations)

$$\alpha_{ij}^{\text{TUPE-A}} = \frac{1}{\sqrt{2d}}(x_i^l W^{Q,l})(x_j^l W^{K,l})^T + \text{reset}_\theta\left(\frac{1}{\sqrt{2d}}(p_i U^Q)(p_j U^K)^T, i, j\right)$$

$$\alpha_{ij}^{\text{TUPE-R}} = \frac{1}{\sqrt{2d}}(x_i^l W^{Q,l})(x_j^l W^{K,l})^T + \text{reset}_\theta\left(\frac{1}{\sqrt{2d}}(p_i U^Q)(p_j U^K)^T + b_{j-i}, i, j\right)$$

$$\text{reset}_\theta(v, i, j) = \begin{cases} v_{ij} & i \neq 1, j \neq 1, \text{(not related to [CLS])} \\ \theta_1 & i = 1, \text{(from [CLS] to others)} \\ \theta_2 & i \neq 1, j = 1, \text{(from others to [CLS])} \end{cases},$$

where $\theta = \{\theta_1, \theta_2\}$ is a learnable parameter. A visualization is put in Figure 4.



05 Position Representations

Other Representations

absolute와 relative position을 함께 적용

Rotation matrix를 적용하여 사용

Absolute와 relative position을 함께 적용

TUPE, 2021, ICLR (21 citations)

Table 1: GLUE scores on dev set. All settings are pre-trained by BERT-Base (110M) model with 16GB data. TUPE-A$^{\text{mid}}$ (TUPE-R$^{\text{mid}}$) is the intermediate $300k$-step checkpoint of TUPE-A (TUPE-R). TUPE-A$^{\text{tie-cls}}$ removes the reset function from TUPE-A. BERT-A^{d} uses different projection matrices for words and positions, based on BERT-A.

	Steps	MNLI-m/mm	QNLI	QQP	SST	CoLA	MRPC	RTE	STS	Avg.
BERT-A	$1M$	84.93/84.91	91.34	91.04	92.88	55.19	88.29	68.61	**89.43**	82.96
BERT-R	$1M$	85.81/85.84	92.16	91.12	92.90	55.43	89.26	71.46	88.94	83.66
TUPE-A	$1M$	86.05/85.99	91.92	91.16	93.19	63.09	88.37	71.61	88.88	84.47
TUPE-R	$1M$	**86.21/86.19**	**92.17**	**91.30**	**93.26**	**63.56**	**89.89**	**73.56**	89.23	**85.04**
TUPE-A$^{\text{mid}}$	$300k$	84.76/84.83	90.96	91.00	92.25	62.13	87.1	68.79	88.16	83.33
TUPE-R$^{\text{mid}}$	$300k$	84.86/85.21	91.23	91.14	92.41	62.47	87.29	69.85	88.63	83.68
TUPE-A$^{\text{tie-cls}}$	$1M$	85.91/85.73	91.90	91.05	93.17	59.46	88.53	69.54	88.97	83.81
BERT-A^{d}	$1M$	85.26/85.28	91.56	91.02	92.70	59.73	88.46	71.31	87.47	83.64

05 Position Representations

Other Representations

absolute와 relative position을 함께 적용

Rotation matrix를 적용하여 사용

Rotation matrix를 적용하여 사용

RoFormer, 2021, arXiv (4 citations)

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_{\{q,k\}}\boldsymbol{x}_m$$

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix}$$

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = (\boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_q\boldsymbol{x}_m)^\intercal(\boldsymbol{R}^d_{\Theta,n}\boldsymbol{W}_k\boldsymbol{x}_n) = \boldsymbol{x}^\intercal\boldsymbol{W}_q R^d_{\Theta,n-m}\boldsymbol{W}_k\boldsymbol{x}_n$$

$$\boldsymbol{R}^d_{\Theta,n-m} = (\boldsymbol{R}^d_{\Theta,m})^\intercal \boldsymbol{R}^d_{\Theta,n}$$

Dimension = 2

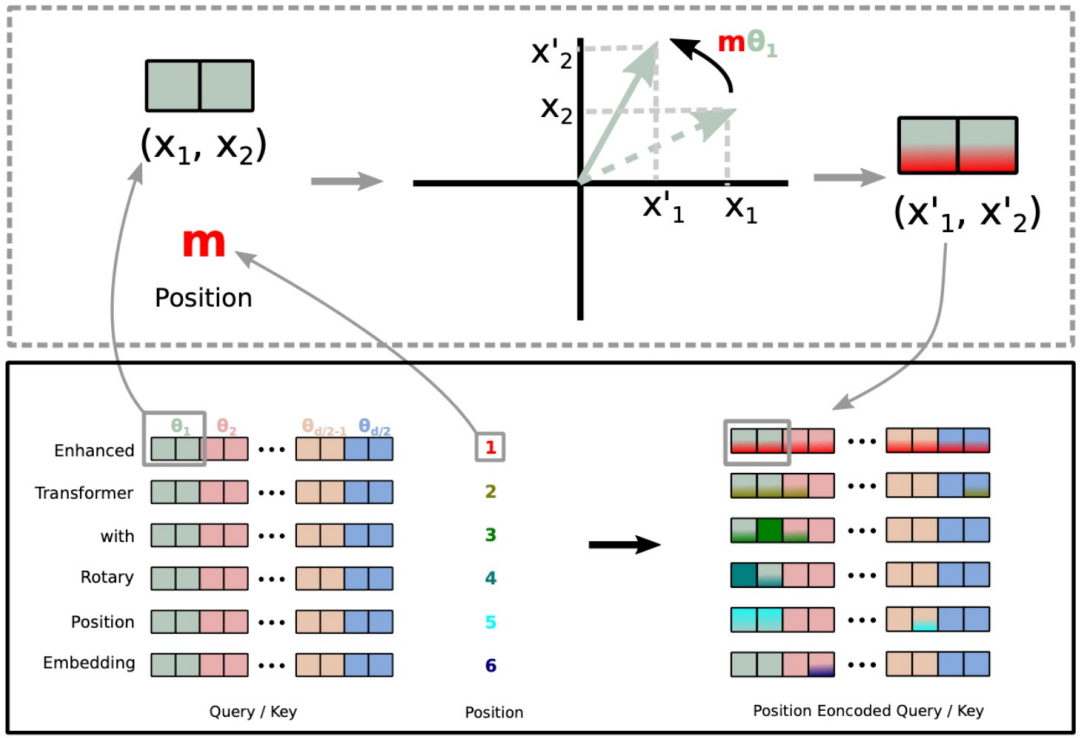


Figure 1: Implementation of Rotary Position Embedding(RoPE).

05 Position Representations

Position Representations without Explicit Encoding

Position Representations without Explicit Encoding

- 위치 정보를 별도로 제공하는 것이 아닌 내재적인 위치 정보를 받는 방법

continuous function으로 position을 변환	RNN 적용	CNN 적용
	• Sequential 정보 반영	• Local 정보 반영

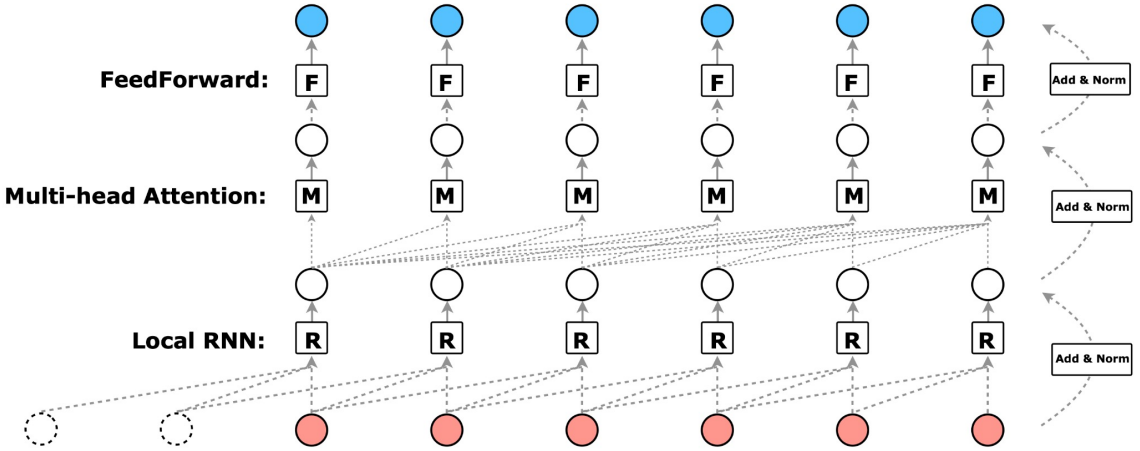
05 Position Representations

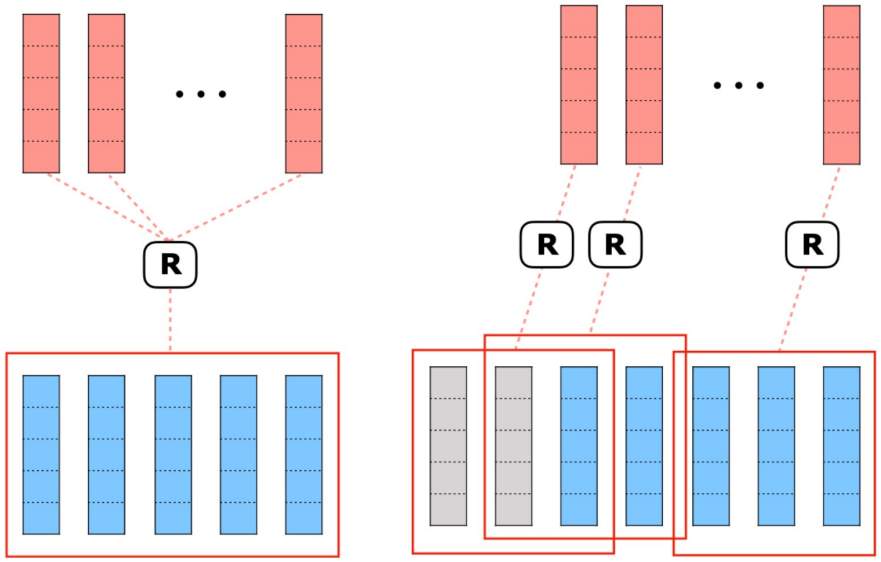
Position Representations without Explicit Encoding

continuous function으로 position을 변환	RNN 적용	CNN 적용
	• Sequential 정보 반영	• Local 정보 반영

RNN 적용

R-Transformer, 2019, arXiv (24 citations)





(a) Original RNN. (b) Local RNN.

05 Position Representations

Position Representations without Explicit Encoding

continuous function으로 position을 변환	RNN 적용	CNN 적용
	• Sequential 정보 반영	• Local 정보 반영

RNN 적용

R-Transformer, 2019, arXiv (24 citations)

Table 4: Word-level language modeling. Italic numbers denote that the results are directly copied from other papers that have the same settings.

Model	# of layers / hidden size	Perplexity
RNN (Bai et al., 2018)	-	*114.50*
GRU (Bai et al., 2018)	-	*92.48*
LSTM (Bai et al., 2018)	*3 /700*	*78.93*
TCN (Bai et al., 2018)	*4 /600*	*88.68*
Transformer	3/128	122.37
R-Transformer	3/128	84.38

05 Position Representations

Position Representations without Explicit Encoding

continuous function으로 position을 변환	RNN 적용	CNN 적용
	• Sequential 정보 반영	• Local 정보 반영

CNN 적용

Conditional Positional Encodings for Vision Transformers, 2021, arXiv (19 citations)

Computer Vision task에서 position encoding이 지켜야하는 요소

1. Permutation variant 이지만 translation invariant여야 한다.
2. Inductive를 주어야하고 학습 시 보다 긴 길이를 처리해야 한다.
3. Absolute positio에 대한 정보가 반영 되어야 한다.

05 Position Representations

Position Representations without Explicit Encoding

continuous function으로 position을 변환	RNN 적용	CNN 적용
	• Sequential 정보 반영	• Local 정보 반영

CNN 적용

Conditional Positional Encodings for Vision Transformers, 2021, arXiv (19 citations)

Computer Vision task에서 position encoding이 지켜야하는 요소

1. Permutation variant 이지만 translation invariant여야 한다.
2. Inductive를 주어야하고 학습 시 보다 긴 길이를 처리해야 한다.
3. Absolute position에 대한 정보가 반영 되어야 한다.

입력 값이 변하면 출력 값이 변하는 것

입력 위치가 변해도 결과는 동일 한 것

05 Position Representations

Position Representations without Explicit Encoding

continuous function으로 position을 변환

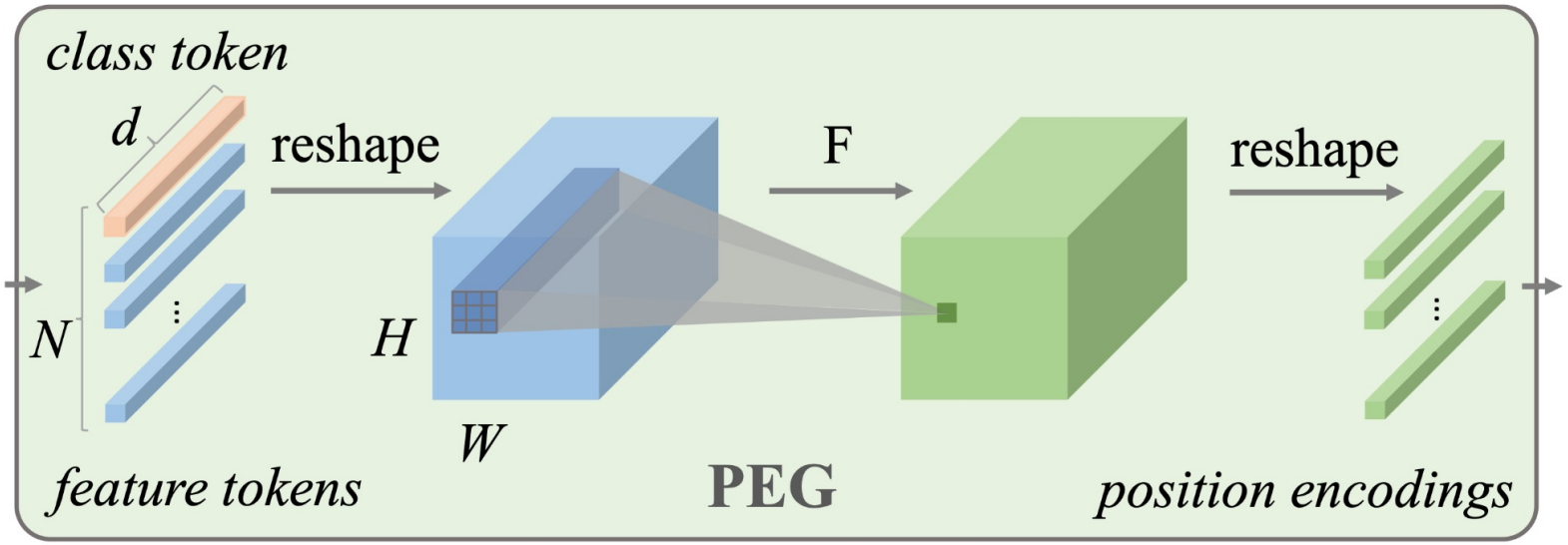
RNN 적용
- Sequential 정보 반영

CNN 적용
- Local 정보 반영

CNN 적용

Conditional Positional Encodings for Vision Transformers, 2021, arXiv (19 citations)

Position Encoding Generator (PEG)



Zero padding이 alsolute position을 모델이 학습하도록 도움

05 Position Representations

Overview

Position Representation on Transformer Decoders

- Transformer decoder에서의 위치 정보 사용하는 방법

Decoder의 masked self-attention을 permutation equivariant가 아님

LM에서는 position encoding 필요 없음

05 Position Representations

Position Representation on Transformer Decoders

Position information 제거

Linear Transformers Are Secretly Fast Weight Memory Systems, 2021, ICML (5 citations)

Table 3. WikiText-103 language model perplexities for Linear Transformers (*medium* configuration) with our update rule.

Position Encoding	Attn. Normalisation	Valid	Test
Yes	Yes	30.4	32.1
No	Yes	29.2	31.2
Yes	**No**	29.7	31.5
No	**No**	**28.1**	**31.1**

05 Layer Normalization

Overview

2017

Vanilla Transformer, NIPS (24,273 citations)

2019

Learning Deep Transformer Models for Machine Translation, ACL (149 citations)

Understanding and Improving Layer Normalization, NIPS (35 citaitons)

Transformers without Tears- Improving the Normalization of Self-Attention, arXiv (72 citations)

2020

On Layer Normalization in the Transformer Architecture, ICML (71 citations)

Understanding the Difficulty of Training Transformers, EMNLP (35 citations)

PowerNorm- Rethinking Batch Normalization in Transformers, ICML (15 citations)

ReZero, arXiv (47 citations)

Placement of Layer Normalization

Substitutes of Layer Normalization

Normalization-free Transformer

05 Layer Normalization

Overview

Placement of Layer Normalization

- Layer Normalization의 위치에 따른 연구

Substitutes of Layer Normalization

- Layer Normalization를 다른 방법으로 대체한 연구

Normalization-free Transformer

- Normalization 없는 transformer 연구

05 Layer Normalization

Overview

Placement of Layer Normalization

- Layer Normalization의 위치에 따른 연구

Post-LN은 Pre-LN에 비해 학습 불안정

Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

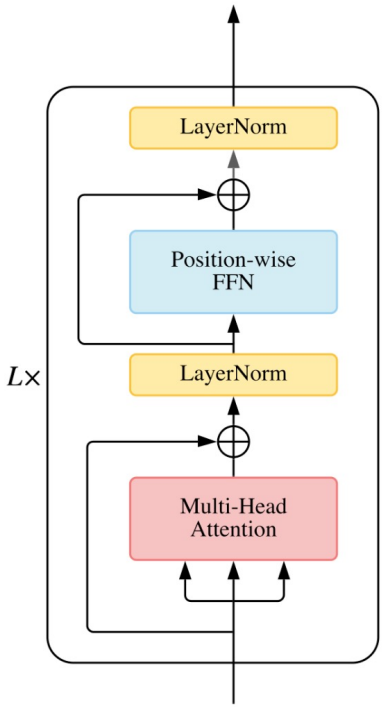
Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

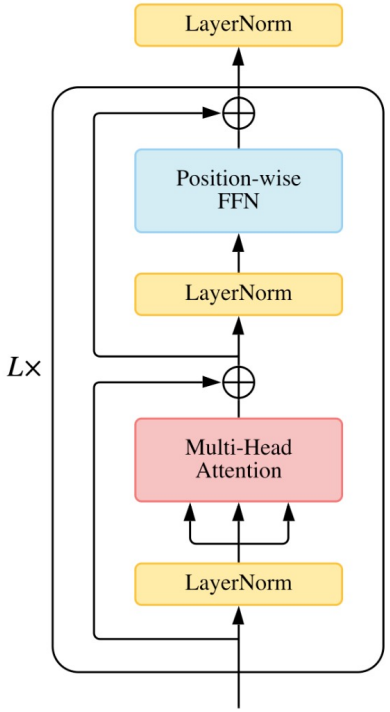
Layer가 많아질 수록 gradient vanishing 또는 exploring 발생

Learning Deep Transformer Models for Machine Translation, ACL (149 citations)

On Layer Normalization in the Transformer Architecture, 2020, ICML (71 citations)



(a) post-LN



(b) pre-LN

Post Norm $x_{l+1} = \mathrm{LN}(x_l + \mathcal{F}(x_l; \theta_l))$

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L} \times \prod_{k=l}^{L-1} \frac{\partial \mathrm{LN}(y_k)}{\partial y_k} \times$$

Pre Norm $x_{l+1} = x_l + \mathcal{F}(\mathrm{LN}(x_l); \theta_l)$

$$\frac{\partial \mathcal{E}}{\partial x_l} = \frac{\partial \mathcal{E}}{\partial x_L} \times \left(1 + \sum_{k=l}^{L-1} \frac{\partial \mathcal{F}(\mathrm{LN}(x_k); \theta_k)}{\partial x_l}\right)$$

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Layer가 많아질 수록 gradient vanishing 또는 exploring 발생

Learning Deep Transformer Models for Machine Translation, ACL (149 citations)

On Layer Normalization in the Transformer Architecture, 2020, ICML (71 citations)

Model (Base, 16L)		BLEU
post-norm	Bapna et al. (2018)	28.0
	Transformer	failed
	DLCL	28.4
pre-norm	Transformer	28.0
	DLCL	28.2

Table 2: Compare with Bapna et al. (2018) on WMT'16 English-German translation under a 16-layer encoder.

Model		Param.	Batch (×4096)	Updates (×100k)	†Times	BLEU	Δ
Vaswani et al. (2017) (Base)		65M	1	1	reference	27.3	-
Bapna et al. (2018)-deep (Base, 16L)		137M	-	-	-	28.0	-
Vaswani et al. (2017) (Big)		213M	1	3	3x	28.4	-
Chen et al. (2018a) (Big)		379M	16	†0.075	1.2x	28.5	-
He et al. (2018) (Big)		†210M	1	-	-	29.0	-
Shaw et al. (2018) (Big)		†210M	1	3	3x	29.2	-
Dou et al. (2018) (Big)		356M	1	-	-	29.2	-
Ott et al. (2018) (Big)		210M	14	0.25	3.5x	29.3	-
post-norm	Transformer (Base)	62M	1	1	1x	27.5	reference
	Transformer (Big)	211M	1	3	3x	28.8	+1.3
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	failed	failed
	DLCL (Base)	62M	1	1	1x	27.6	+0.1
	DLCL-deep (Base, 25L)	121M	2	0.5	1x	29.2	+1.7
pre-norm	Transformer (Base)	62M	1	1	1x	27.1	reference
	Transformer (Big)	211M	1	3	3x	28.7	+1.6
	Transformer-deep (Base, 20L)	106M	2	0.5	1x	28.9	+1.8
	DLCL (Base)	62M	1	1	1x	27.3	+0.2
	DLCL-deep (Base, 30L)	137M	2	0.5	1x	**29.3**	**+2.2**

Table 1: BLEU scores [%] on English-German translation. `Batch` indicates the corresponding batch size if running on 8 GPUs. `Times` $\propto$ `Batch×Updates`, which can be used to approximately measure the required training time. † denotes an estimate value. Note that "-deep" represents the best-achieved result as depth changes.

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

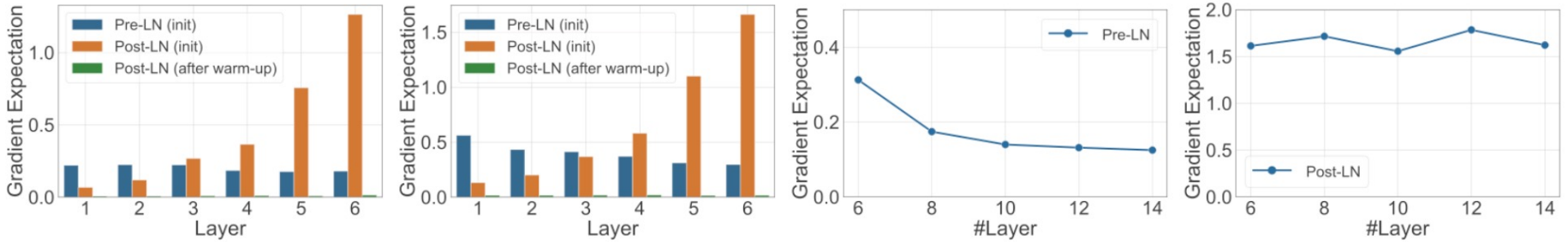
Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Layer가 많아질 수록 gradient vanishing 또는 exploring 발생

Learning Deep Transformer Models for Machine Translation, ACL (149 citations)

On Layer Normalization in the Transformer Architecture, 2020, ICML (71 citations)



(a) W^1 in the FFN sub-layers (b) W^2 in the FFN sub-layers (c) Pre-LN Transformer (d) Post-LN Transformer

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

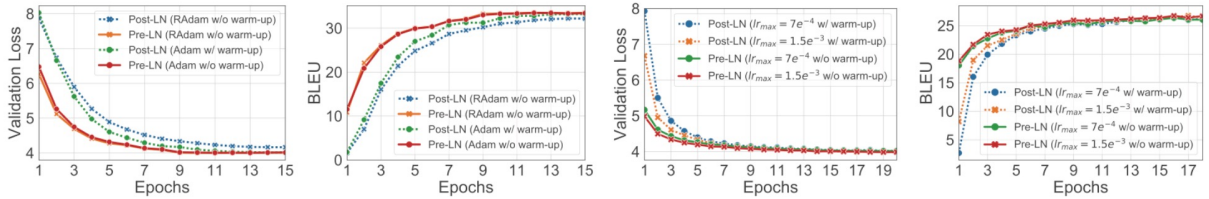
Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Layer가 많아질 수록 gradient vanishing 또는 exploring 발생

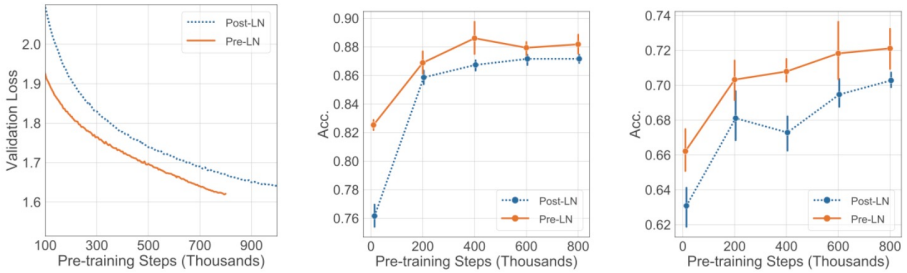
Learning Deep Transformer Models for Machine Translation, ACL (149 citations)

On Layer Normalization in the Transformer Architecture, 2020, ICML (71 citations)



(a) Validation Loss (IWSLT) (b) BLEU (IWSLT) (c) Validation Loss (WMT) (d) BLEU (WMT)

Figure 4. Performances of the models on the IWSLT14 De-En task and WMT14 En-De task



(a) Validation Loss on BERT (b) Accuracy on MRPC (c) Accuracy on RTE

Figure 5. Performances of the models on unsupervised pre-training (BERT) and downstream tasks

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

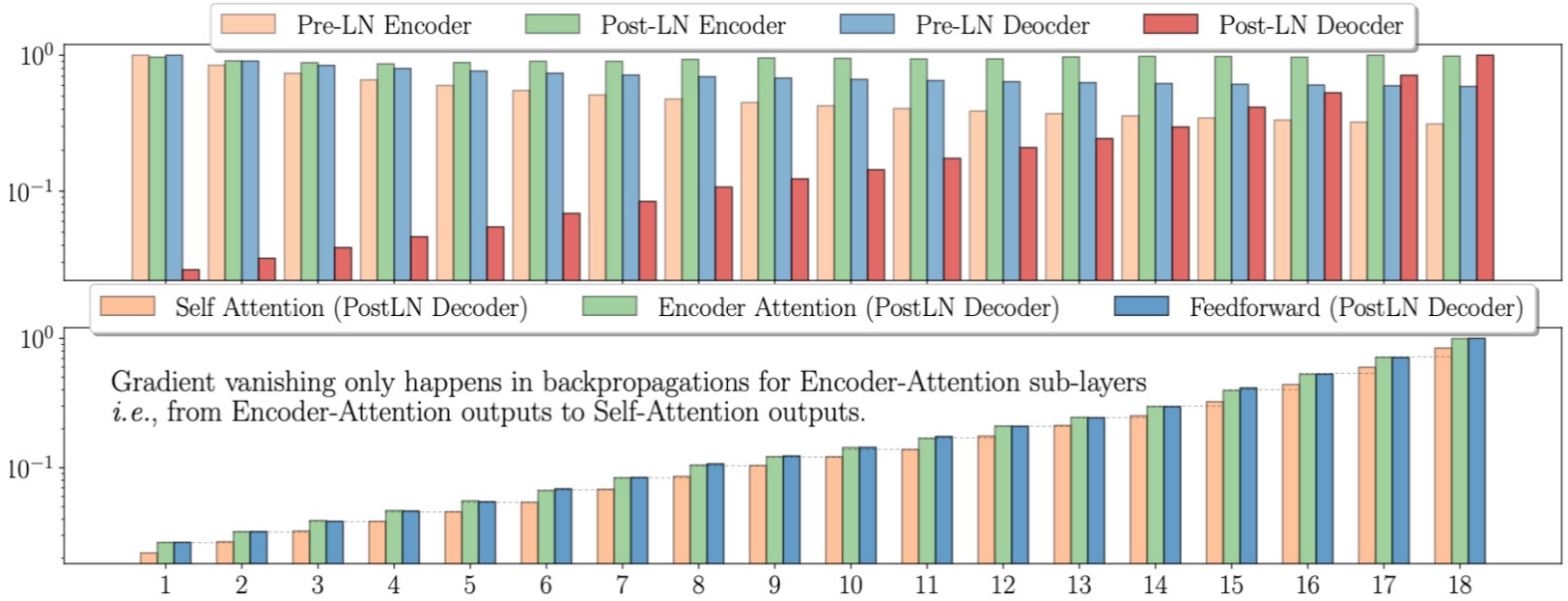
Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Residual branch로 인한 amplification effect 발생

Understanding the Difficulty of Training Transformers, 2020, EMNLP (35 citations)

Gradient에 의한 발생은 decoder에서만 발생



05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Residual branch로 인한 amplification effect 발생

Understanding the Difficulty of Training Transformers, 2020, EMNLP (35 citations)

Gradient에 의한 발생은 decoder에서만 발생

Encoder	Decoder	Gradient	Training
Post-LN	Post-LN	Varnishing	Diverged
Post-LN	Pre-LN	~~Varnishing~~	Diverged
Pre-LN	Pre-LN	~~Varnishing~~	Converged

Table 1: Changing decoders from Post-LN to Pre-LN fixes gradient vanishing, but does not stabilize model training successfully. Encoder/Decoder have 18 layers.

05 Layer Normalization

Placement of Layer Normalization

Post-LN은 Pre-LN에 비해 학습 불안정

Layer가 많아질 수록
gradient vanishing 또는 exploring 발생

Residual branch로 인한
amplification effect 발생

Residual branch로 인한 amplification effect 발생

Understanding the Difficulty of Training Transformers, 2020, EMNLP (35 citations)

Gradient에 의한 발생은 decoder에서만 발생

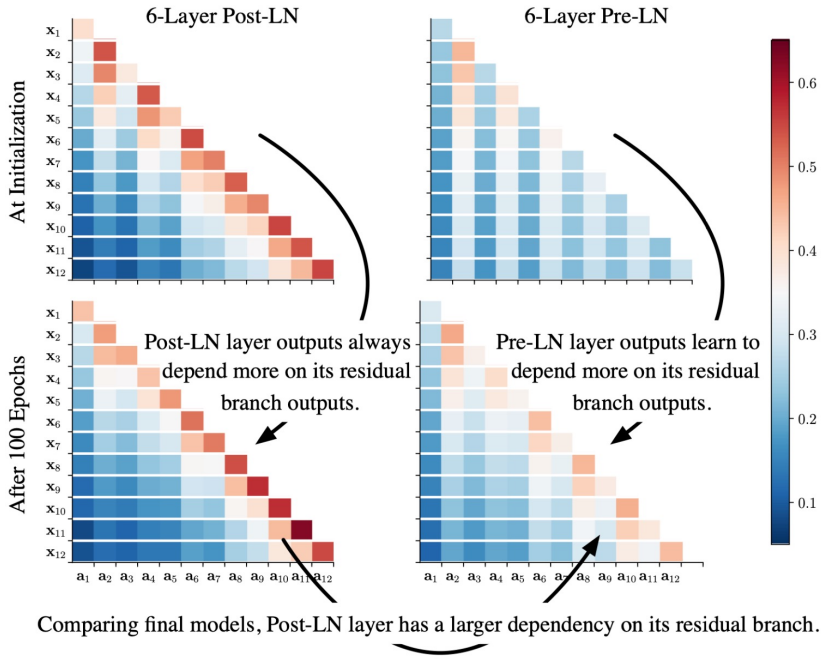


Figure 7: $\beta_{i,j}$ in 6-Layer Post-LN and Pre-LN on the WMT-14 En-De dataset (contains 12 sub-layers).

05 Layer Normalization

Overview

Substitutes of Layer Normalization

- Layer Normalization를 다른 방법으로 대체한 연구

LN은 forward normalization 때문 X
Backward normalization 때문 O

- LN의 파라미터가 과적합 야기
- 파라미터 없는게 더 나음

L2 normalization으로 대체 가능

Power Norm으로 대체 가능

05 Layer Normalization

Substitutes of Layer Normalization

LN은 forward normalization 때문 X
Backward normalization 때문 O

- LN의 파라미터가 과적합 야기
- 파라미터 없는게 더 나음

L2 normalization으로 대체 가능

Power Norm으로 대체 가능

LN은 backward normalization 때문 O

Understanding and Improving Layer Normalization, 2019, NIPS (35 citaitons)

Table 1: The bias and gain do not work on six out of eight datasets. "w/o Norm" is a naive model without LayerNorm. "LayerNorm-simple" is a variant of LayerNorm that drops the bias and gain. "(+)" means higher is better. "(-)" means lower is better.

Models	Machine Translation			Language Modeling	Classification			Parsing
	En-De(+)	De-En(+)	En-Vi(+)	Enwiki8(-)	RT(+)	SST5(+)	MNIST(+)	PTB(+)
Model Layers	12	12	12	12	4	4	3	3
w/o Norm	Diverge	34.0	28.4	**1.04**	76.85	38.55	**99.14**	88.31
LayerNorm	28.3	**35.5**	31.2	1.07	**77.21**	39.23	99.13	89.12
LayerNorm-simple	**28.4**	**35.5**	**31.6**	1.07	76.66	**40.54**	99.09	**89.19**

05 Layer Normalization

Substitutes of Layer Normalization

LN은 forward normalization 때문 X
Backward normalization 때문 O

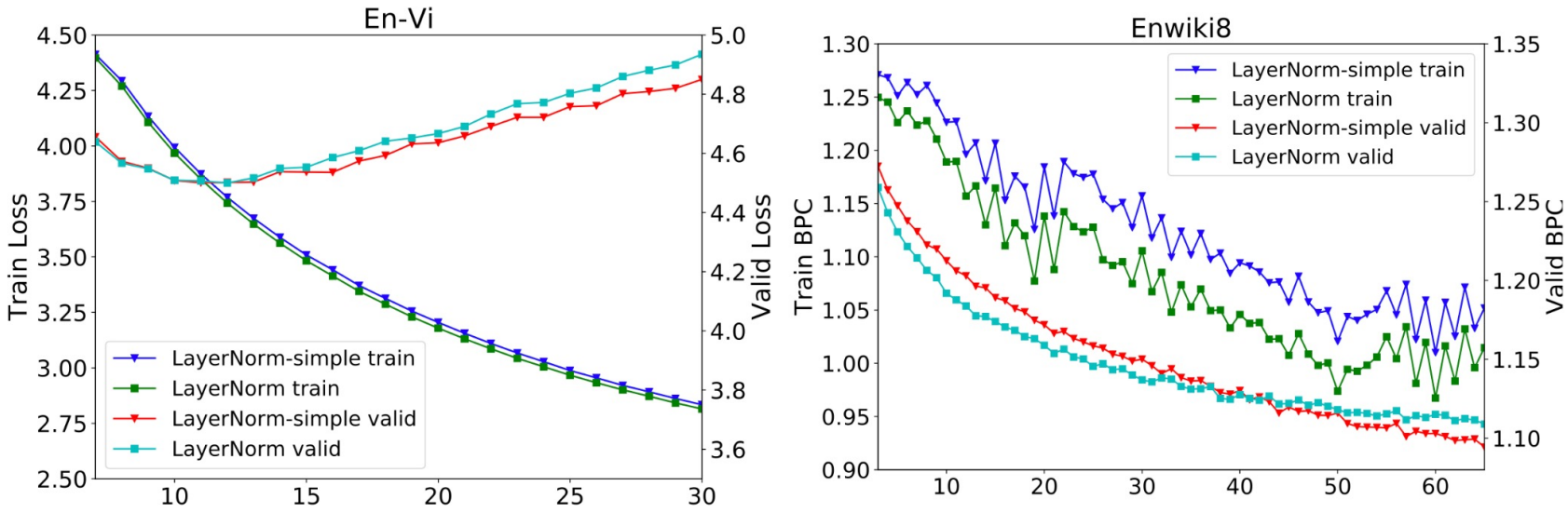
- LN의 파라미터가 과적합 야기
- 파라미터 없는게 더 나음

L2 normalization으로 대체 가능

Power Norm으로 대체 가능

LN은 backward normalization 때문 O

Understanding and Improving Layer Normalization, 2019, NIPS (35 citaitons)



05 Layer Normalization

Overview

Normalization-free Transformer

- Normalization 없는 transformer 연구

Normalization 버리고 Residual Connection을 모두 받지 말고 학습 가능한 가중치를 만들어 받자

05 Layer Normalization

Normalization-free Transformer

ReZero

ReZero, 2020, arXiv (47 citations)

- Normalization 버리고 Residual Connection을 모두 받지 말고 학습 가능한 가중치를 만들어 받자

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i F(\boldsymbol{x}_i)$$

α_i는 학습 파라미터

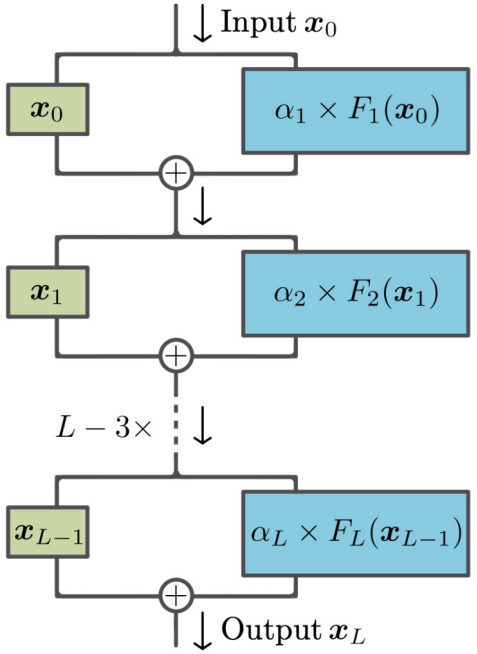


Figure 1: ReZero

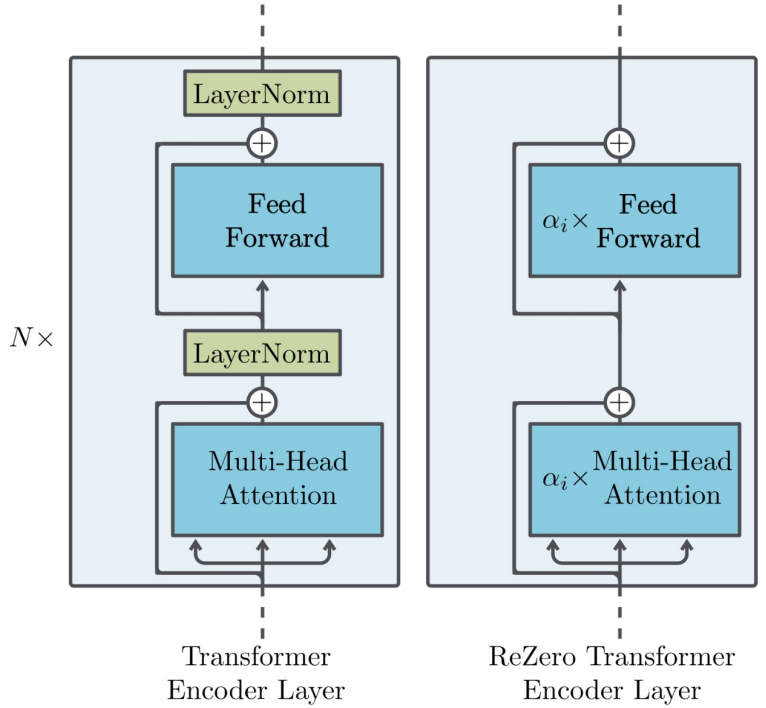


Figure 4: ReZero for Transformers

05 Layer Normalization

Normalization-free Transformer

ReZero

ReZero, 2020, arXiv (47 citations)

- Normalization 버리고 Residual Connection을 모두 받지 말고 학습 가능한 가중치를 만들어 받자

Table 1: Various forms of normalization and residual connections. F represents the transformation of an arbitrary layer and "Norm" is a normalization (e.g. LayerNorm or BatchNorm).

(1) Deep Network	$\boldsymbol{x}_{i+1} = F(\boldsymbol{x}_i)$
(2) Residual Network	$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + F(\boldsymbol{x}_i)$
(3) Deep Network + Norm	$\boldsymbol{x}_{i+1} = \text{Norm}\big(F(\boldsymbol{x}_i)\big)$
(4) Residual Network + Pre-Norm	$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + F(\text{Norm}\big(\boldsymbol{x}_i\big))$
(5) Residual Network + Post-Norm	$\boldsymbol{x}_{i+1} = \text{Norm}\big(\boldsymbol{x}_i + F(\boldsymbol{x}_i)\big)$
(6) **ReZero**	$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i F(\boldsymbol{x}_i)$

05 Layer Normalization

Normalization-free Transformer

ReZero

ReZero, 2020, arXiv (47 citations)

- Normalization 버리고 Residual Connection을 모두 받지 말고 학습 가능한 가중치를 만들어 받자

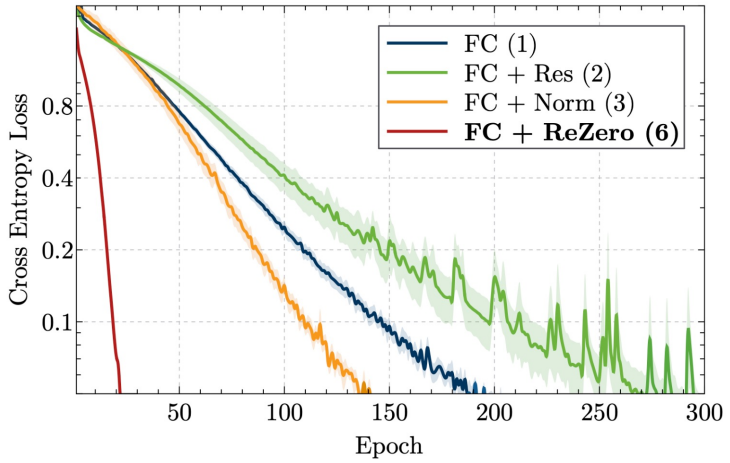


Figure 3: Cross entropy loss during training of four variants of 32 layer fully-connected networks with width 256 and ReLU activations. Numbers in parentheses refer to the architectures in the corresponding rows of Table 1. We average over five runs each and show 1σ error bands. We train using Adagrad [28] with learning rate 0.01.

Table 4: Comparison of Transformers (TX) on the `enwiki8` test set. Char-TX refers to the Character Transformer [14] and uses additional auxiliary losses to achieve its performance.

Model	Layers	Parameters	BPB
Char-TX [14]	12	41M	1.11
TX + Warm-up	12	38M	1.17
TX + ReZero $\alpha = 1$	12	34M	1.17
TX + ReZero $\alpha = 0$	12	34M	1.17
Char-TX [14]	64	219M	1.06
TX	64	51M	Diverged
TX + Warm-up	64	51M	Diverged
TX + ReZero $\alpha = 1$	64	51M	Diverged
TX + ReZero $\alpha = 0$	64	51M	1.11
TX + ReZero	128	101M	1.08

05 Layer Normalization

Normalization-free Transformer

ReZero

ReZero, 2020, arXiv (47 citations)

- Residual Connection을 거의 안받게 됨

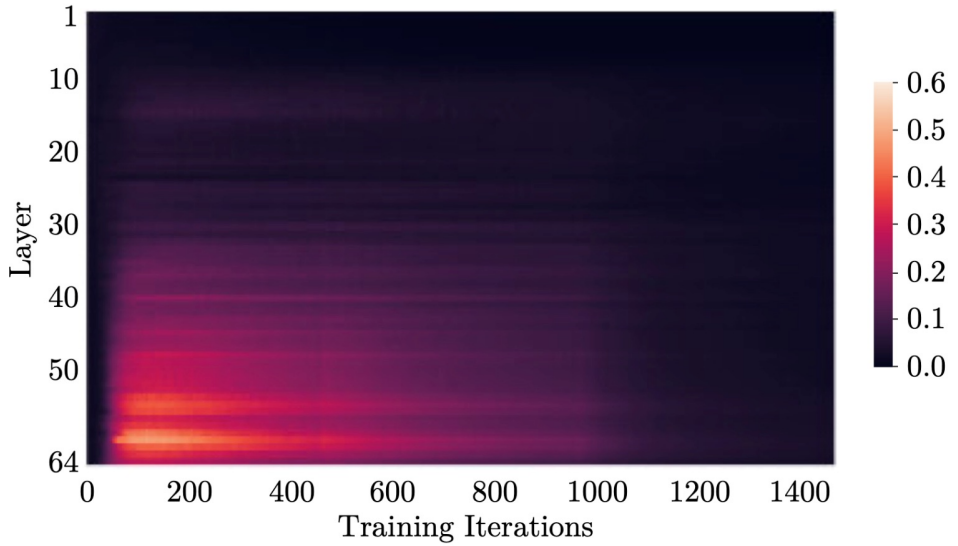


Figure 6: Heat map for residual weight $|\alpha_i|$ evolution during training for 64L ReZero Transformer.

05 Position-wise FFN

Overview

2017

- Vanilla Transformer, NIPS (24,273 citations)
- MoE, ICLR (679 citations)

2018

- Searching for Activation Functions, ICLR (1,177 citations)
- GPT, OpenAI (2,202 citaitons)

2019

- Large Memory Layers with Product Keys, NIPS (59 citations)
- Augmenting Self-attention with Persistent Memory, arXiv (30 citations)

2020

- GLU Variants Improve Transformer, arXiv (7 citations)
- On the Sub-layer Functionalities of Transformer Decoder, EMNLP (3 citations)

2021

- GShard, ICLR (86 citations)
- Switch Transformer, arXiv (76 citations)
- Exploring Sparse Expert Models and Beyond, arXiv (1 citations)
- Hash Layers For Large Sparse Models, arXiv (1 citations)

Activation Function in FFN

Adapting FFN for Larger Capacity

Dropping FFN Layers

05 Position-wise FFN

Overview

Activation Function in FFN

- FFN에서 좋은 결과를 낼 수 있는 다양한 activation function

Adapting FFN for Larger Capacity

- FFN에서 모델의 capacity를 늘리는 방법

Dropping FFN Layers

- FFN 없이 학습 하는 경우

05 Position-wise FFN

Overview

FFN 말고 다른걸 넣을 순 있지만 FFN 없이 MHA만 있으면 안됨

Rank collapse 문제 발생

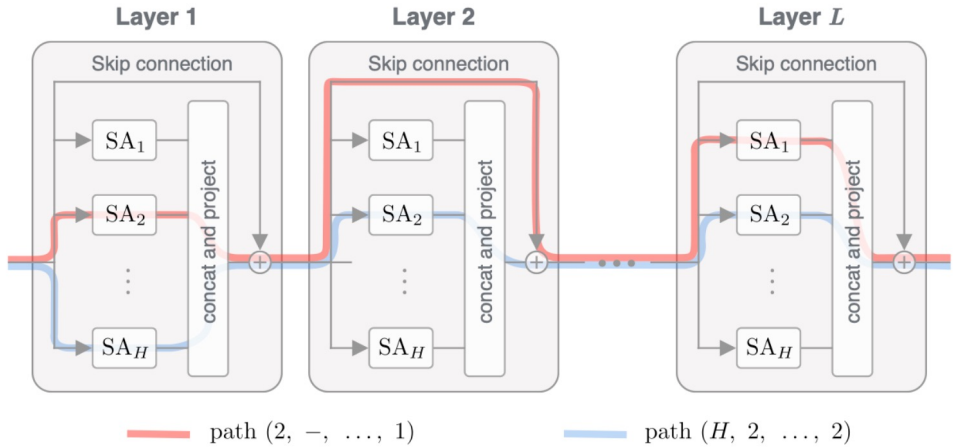


Figure 1: Two paths in a deep Self-Attention Network (SAN) with H heads and L layers. At each layer, a path can go through one of the heads or bypass the layer. Adding an MLP block after each attention layer forms the transformer architecture.

05 Position-wise FFN

Overview

Activation Function in FFN

- FFN에서 좋은 결과를 낼 수 있는 다양한 activation function

ReLU

Swish

- 강화학습 기반 Activation Search

GELU

GLU

05 Position-wise FFN

Activation Function in FFN

ReLU | **Swish** | GELU | GLU

- 강화학습 기반 Activation Search

Swish

Searching for Activation Functions, 2018, ICLR (1,177 citations)

Function	RN	WRN	DN
ReLU $[\max(x, 0)]$	93.8	95.3	94.8
$x \cdot \sigma(\beta x)$	94.5	95.5	94.9
$\max(x, \sigma(x))$	94.3	95.3	94.8
$\cos(x) - x$	94.1	94.8	94.6
$\min(x, \sin(x))$	94.0	95.1	94.4
$(\tan^{-1}(x))^2 - x$	93.9	94.7	94.9
$\max(x, \tanh(x))$	93.9	94.2	94.5
$\text{sinc}(x) + x$	91.5	92.1	92.0
$x \cdot (\sinh^{-1}(x))^2$	85.1	92.1	91.1

Table 1: CIFAR-10 accuracy.

Function	RN	WRN	DN
ReLU $[\max(x, 0)]$	74.2	77.8	83.7
$x \cdot \sigma(\beta x)$	75.1	78.0	83.9
$\max(x, \sigma(x))$	74.8	78.6	84.2
$\cos(x) - x$	75.2	76.6	81.8
$\min(x, \sin(x))$	73.4	77.1	74.3
$(\tan^{-1}(x))^2 - x$	75.2	76.7	83.1
$\max(x, \tanh(x))$	74.8	76.0	78.6
$\text{sinc}(x) + x$	66.1	68.3	67.9
$x \cdot (\sinh^{-1}(x))^2$	52.8	70.6	68.1

Table 2: CIFAR-100 accuracy.

05 Position-wise FFN

Activation Function in FFN

ReLU	Swish	GELU	GLU
	• 강화학습 기반 Activation Search		

Swish

Searching for Activation Functions, 2018, ICLR (1,177 citations)

$$f(x) = x \cdot \text{sigmoid}(\beta x)$$

β는 상수로 쓰거나 학습 파라미터로 사용 가능

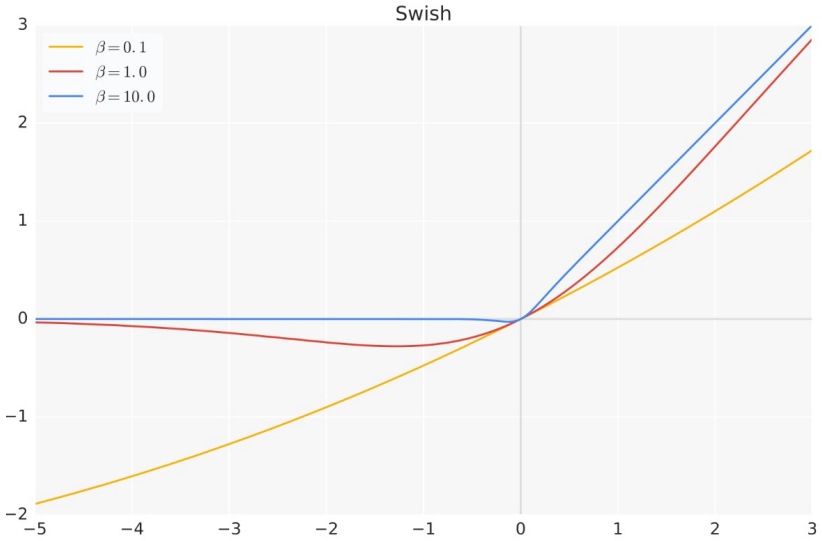


Figure 4: The Swish activation function.

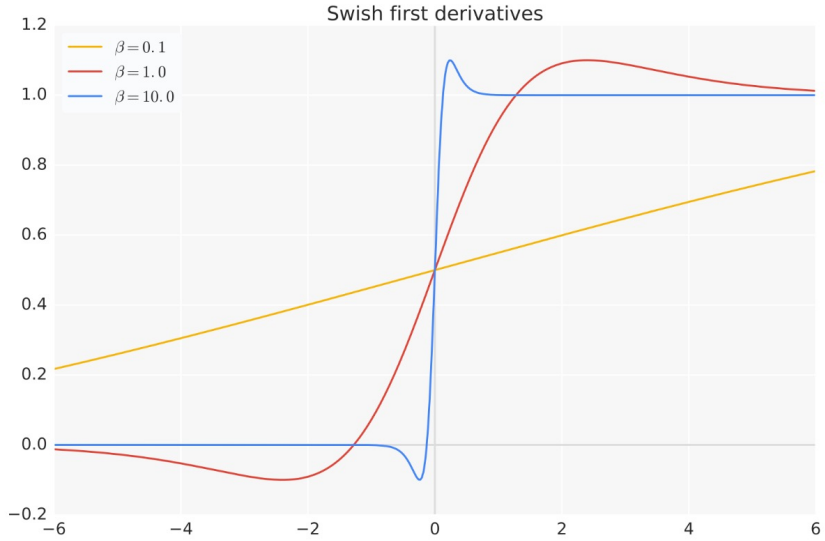


Figure 5: First derivatives of Swish.

05 Position-wise FFN

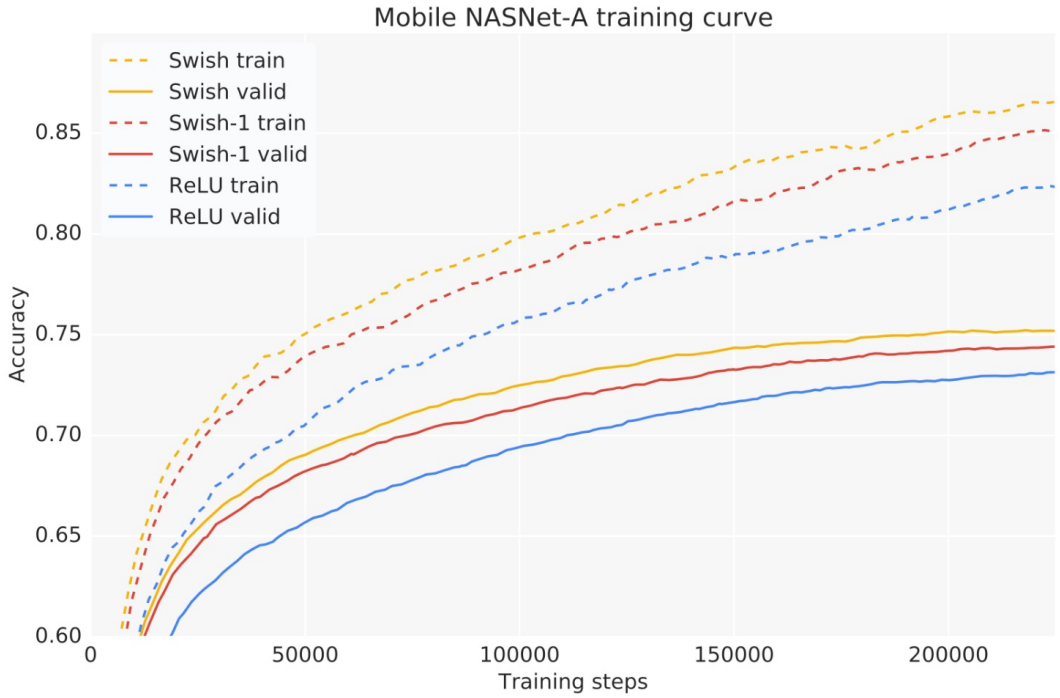
Activation Function in FFN

ReLU	Swish	GELU	GLU
	• 강화학습 기반 Activation Search		

Swish

Searching for Activation Functions, 2018, ICLR (1,177 citations)

ImageNet



Model	newstest2013	newstest2014	newstest2015	newstest2016
LReLU	26.2	27.9	29.8	33.4
PReLU	26.3	27.7	29.7	33.1
Softplus	23.4	23.6	25.8	29.2
ELU	24.6	25.1	27.7	32.5
SELU	23.7	23.5	25.9	30.5
GELU	25.9	27.3	29.5	33.1
ReLU	26.1	27.8	29.8	33.3
Swish-1	26.2	28.0	30.1	34.0
Swish	26.5	27.6	30.0	33.1

Table 11: BLEU score of a 12 layer Transformer on WMT English→German.

05 Position-wise FFN

Activation Function in FFN

ReLU	Swish	GELU	GLU

- 강화학습 기반 Activation Search

GLU (Gated Linear Units)

GLU Variants Improve Transformer, arXiv (7 citations)

$$\mathrm{GLU}(x, W, V, b, c) = \sigma(xW + b) \otimes (xV + c)$$

$$\mathrm{Bilinear}(x, W, V, b, c) = (xW + b) \otimes (xV + c)$$

Table 2: GLUE Language-Understanding Benchmark [Wang et al., 2018] (dev).

	Score Average	CoLA MCC	SST-2 Acc	MRPC F1	MRPC Acc	STSB PCC	STSB SCC	QQP F1	QQP Acc	MNLIm Acc	MNLImm Acc	QNLI Acc	RTE Acc
$\mathrm{FFN}_{\mathrm{ReLU}}$	83.80	51.32	94.04	**93.08**	**90.20**	89.64	89.42	89.01	91.75	85.83	86.42	92.81	80.14
$\mathrm{FFN}_{\mathrm{GELU}}$	83.86	53.48	94.04	92.81	**90.20**	89.69	89.49	88.63	91.62	85.89	86.13	92.39	80.51
$\mathrm{FFN}_{\mathrm{Swish}}$	83.60	49.79	93.69	92.31	89.46	89.20	88.98	88.84	91.67	85.22	85.02	92.33	81.23
$\mathrm{FFN}_{\mathrm{GLU}}$	84.20	49.16	94.27	92.39	89.46	89.46	89.35	88.79	91.62	86.36	86.18	92.92	**84.12**
$\mathrm{FFN}_{\mathrm{GEGLU}}$	84.12	53.65	93.92	92.68	89.71	90.26	90.13	89.11	91.85	86.15	86.17	92.81	79.42
$\mathrm{FFN}_{\mathrm{Bilinear}}$	83.79	51.02	**94.38**	92.28	89.46	90.06	89.84	88.95	91.69	**86.90**	**87.08**	92.92	81.95
$\mathrm{FFN}_{\mathrm{SwiGLU}}$	84.36	51.59	93.92	92.23	88.97	**90.32**	**90.13**	**89.14**	**91.87**	86.45	86.47	**92.93**	83.39
$\mathrm{FFN}_{\mathrm{ReGLU}}$	**84.67**	**56.16**	**94.38**	92.06	89.22	89.97	89.85	88.86	91.72	86.20	86.40	92.68	81.59
[Raffel et al., 2019]	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28
ibid. stddev.	0.235	1.111	0.569	0.729	1.019	0.374	0.418	0.108	0.070	0.291	0.231	0.361	1.393

Table 4: SQuAD [Rajpurkar et al., 2016] v1.1 (dev).

	EM	F1
$\mathrm{FFN}_{\mathrm{ReLU}}$	83.18	90.87
$\mathrm{FFN}_{\mathrm{GELU}}$	83.09	90.79
$\mathrm{FFN}_{\mathrm{Swish}}$	83.25	90.76
$\mathrm{FFN}_{\mathrm{GLU}}$	82.88	90.69
$\mathrm{FFN}_{\mathrm{GEGLU}}$	83.55	91.12
$\mathrm{FFN}_{\mathrm{Bilinear}}$	**83.82**	91.06
$\mathrm{FFN}_{\mathrm{SwiGLU}}$	83.42	91.03
$\mathrm{FFN}_{\mathrm{ReGLU}}$	83.53	**91.18**
[Raffel et al., 2019]	80.88	88.81
ibid. Standard Deviation	0.343	0.226

05 Position-wise FFN

Overview

Adapting FFN for Larger Capacity

- FFN에서 모델의 capacity를 늘리는 방법

MoE

MoE in Transformer

MoE가 MoE요?

MoE (Mixture of Experts)

MoE, 2017, ICLR (679 citations)

- Gating Network와 Expert는 입력 차원과 출력 차원이 동일함

Since its introduction more than two decades ago (Jacobs et al., 1991; Jordan & Jacobs, 1994)

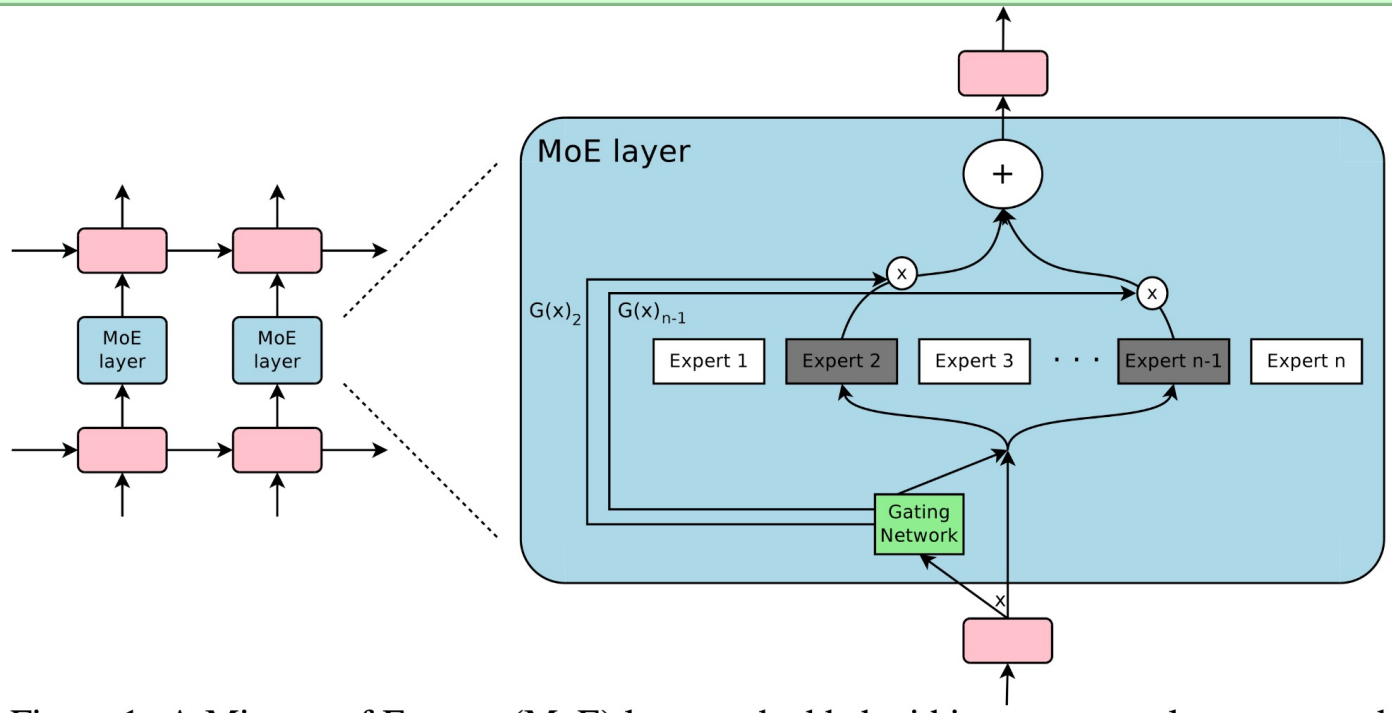


Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

Position-wise FFN

Adapting FFN for Larger Capacity

MoE (Mixture of Experts)

MoE, 2017, ICLR (679 citations)

- Gating Network와 Expert는 입력 차원과 출력 차원이 동일함

$$y = \sum_{i=1}^{n} G(x)_i E_i(x)$$

Softmax Gating

$$G_\sigma(x) = Softmax(x \cdot W_g)$$

Noisy Top-K Gating

$$G(x) = Softmax(KeepTopK(H(x), k))$$

$$H(x)_i = (x \cdot W_g)_i + StandardNormal() \cdot Softplus((x \cdot W_{noise})_i)$$

$$KeepTopK(v, k)_i = \begin{cases} v_i & \text{if } v_i \text{ is in the top } k \text{ elements of } v. \\ -\infty & \text{otherwise.} \end{cases}$$

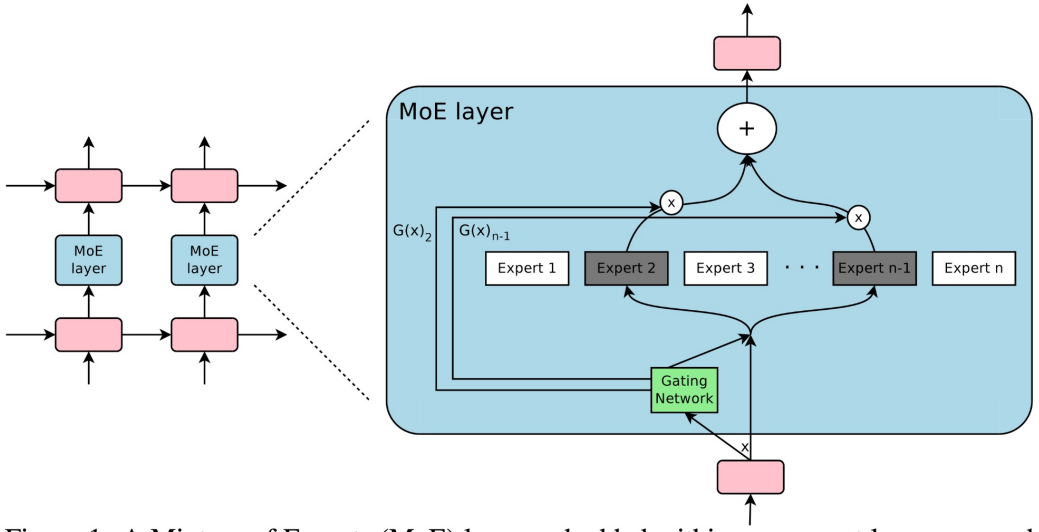


Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

MoE in Transformer

Switch Transformer, 2021, arXiv (76 citations)

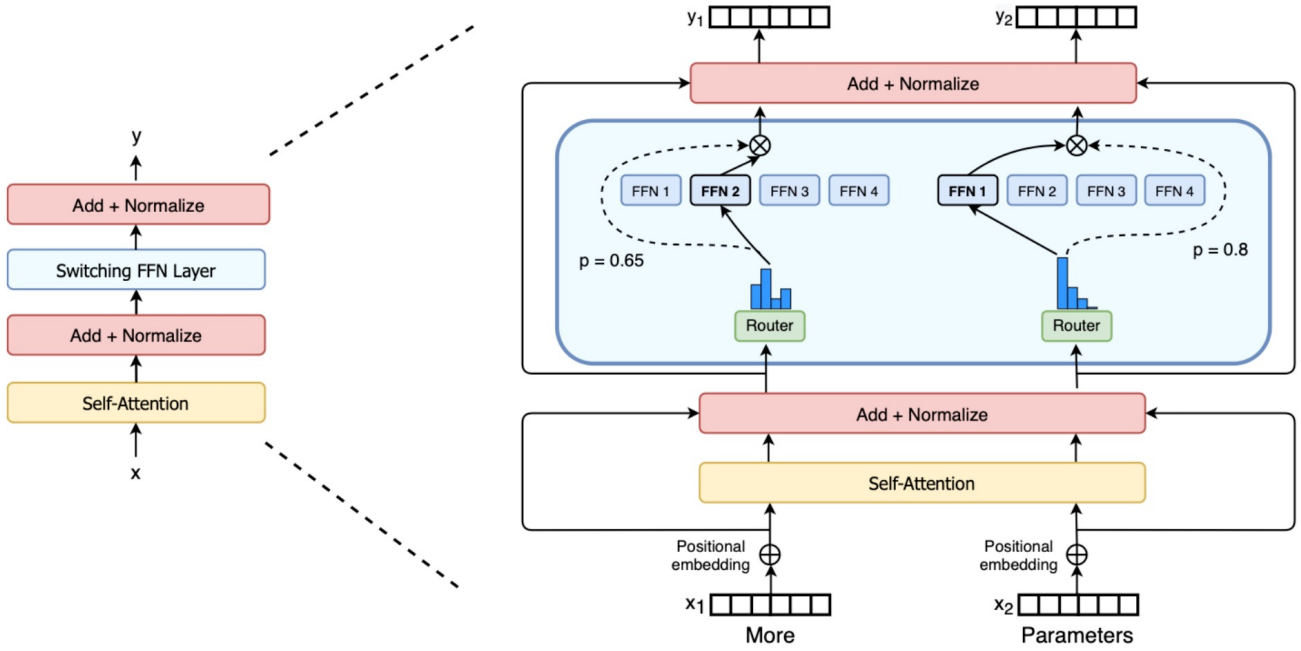


Figure 2: **Illustration of a Switch Transformer encoder block**. We replace the dense feed forward network (FFN) layer present in the Transformer with a sparse Switch FFN layer (light blue). The layer operates independently on the tokens in the sequence. We diagram two tokens (x_1 = "More" and x_2 = "Parameters" below) being routed (solid lines) across four FFN experts, where the router independently routes each token. The switch FFN layer returns the output of the selected FFN multiplied by the router gate value (dotted-line).

Position-wise FFN

Adapting FFN for Larger Capacity

MoE in Transformer

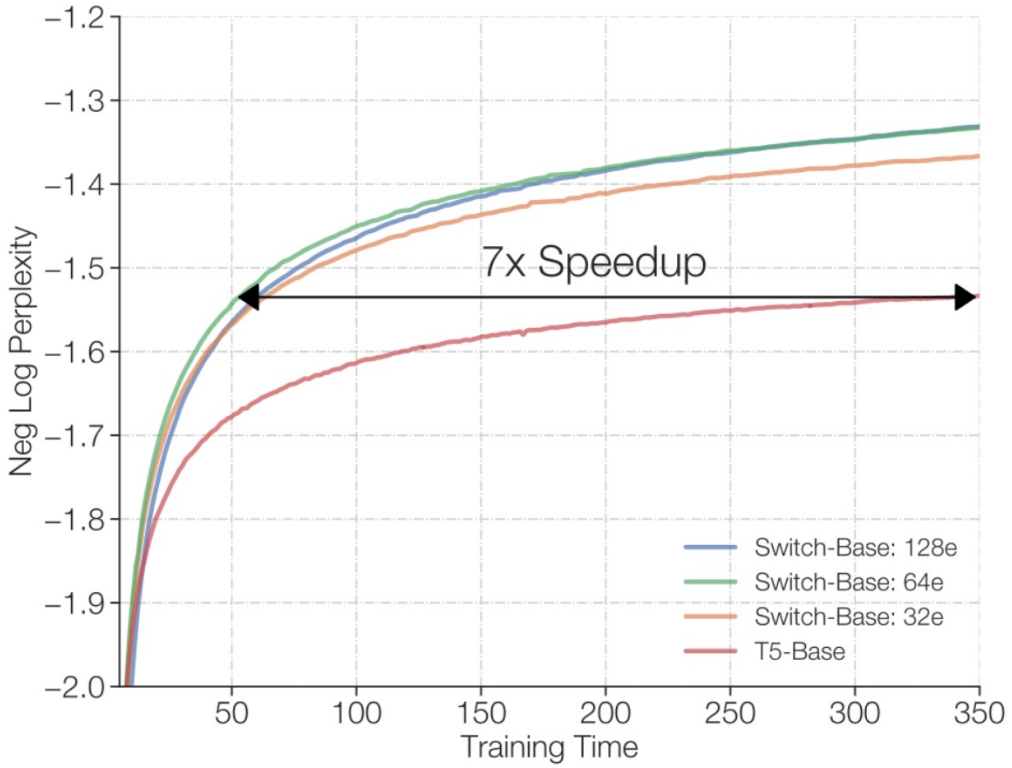
Switch Transformer, 2021, arXiv (76 citations)

Model	GLUE	SQuAD	SuperGLUE	Winogrande (XL)
T5-Base	84.3	85.5	75.1	66.6
Switch-Base	**86.7**	**87.2**	**79.5**	**73.3**
T5-Large	87.8	88.1	82.7	79.1
Switch-Large	**88.5**	**88.6**	**84.7**	**83.0**

Model	XSum	ANLI (R3)	ARC Easy	ARC Chal.
T5-Base	18.7	51.8	56.7	**35.5**
Switch-Base	**20.3**	**54.0**	**61.3**	32.8
T5-Large	20.9	56.6	**68.8**	**35.5**
Switch-Large	**22.3**	**58.6**	66.0	**35.5**

Model	CB Web QA	CB Natural QA	CB Trivia QA
T5-Base	26.6	25.8	24.5
Switch-Base	**27.4**	**26.8**	**30.7**
T5-Large	27.7	27.6	29.5
Switch-Large	**31.3**	**29.5**	**36.9**

Table 6: **Fine-tuning results.** Fine-tuning results of T5 baselines and Switch models across a diverse set of natural language tests (validation sets; higher numbers are better). We compare FLOP-matched Switch models to the T5-Base and T5-Large baselines. For most tasks considered, we find significant improvements of the Switch-variants. We observe gains across both model sizes and across both reasoning and knowledge-heavy language tasks.



05 Position-wise FFN

Overview

Dropping FFN Layers

- FFN 없이 학습 하는 경우

Transformer decoder 사실 FFN 필요 없다

05 Position-wise FFN

Dropping FFN Layers

Transformer Decoder 사실 FFN 필요 없다

On the Sub-layer Functionalities of Transformer Decoder, 2020, EMNLP (3 citations)

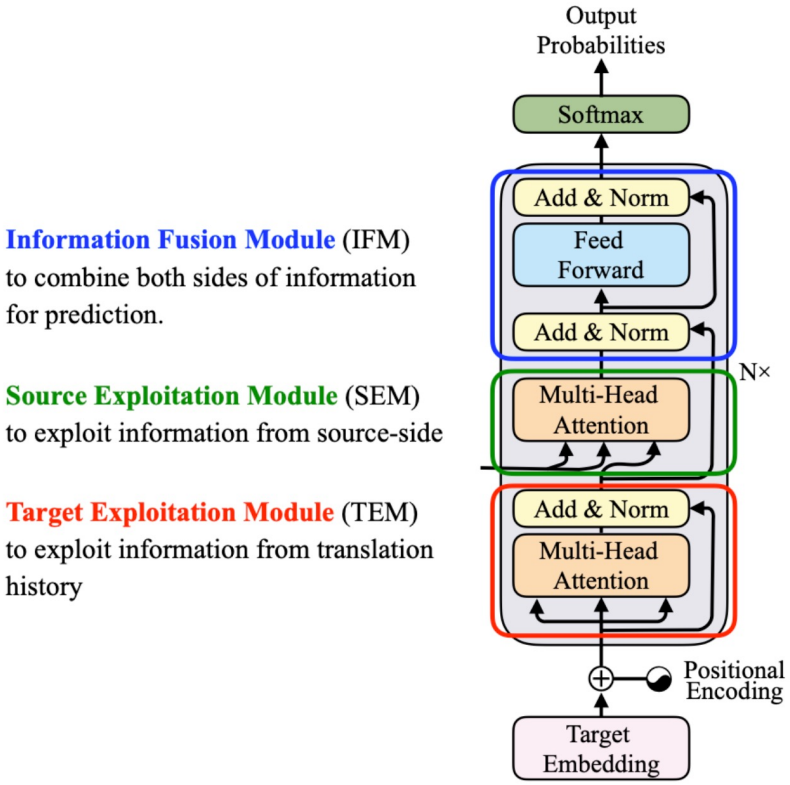


Figure 1: A sub-layer splitting of Transformer decoder with respect to their functionalities.

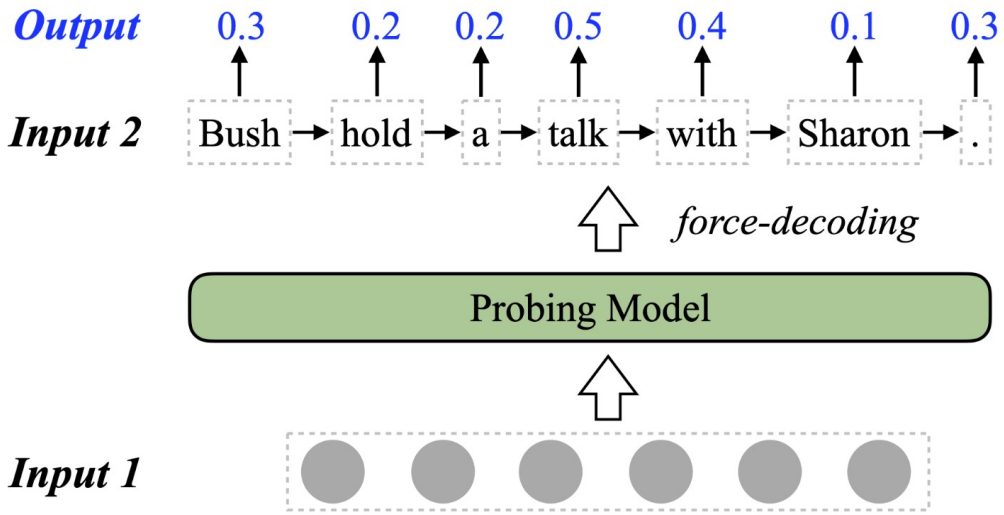


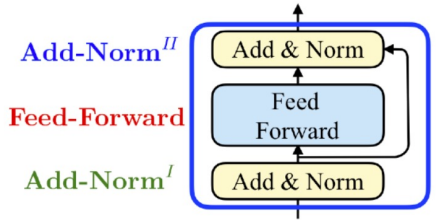
Figure 2: Illustration of the information probing model, which reads the representation of a decoder module ("Input 1") and the word sequence to recover ("Input 2"), and outputs the generation probability ("Output").

05 Position-wise FFN

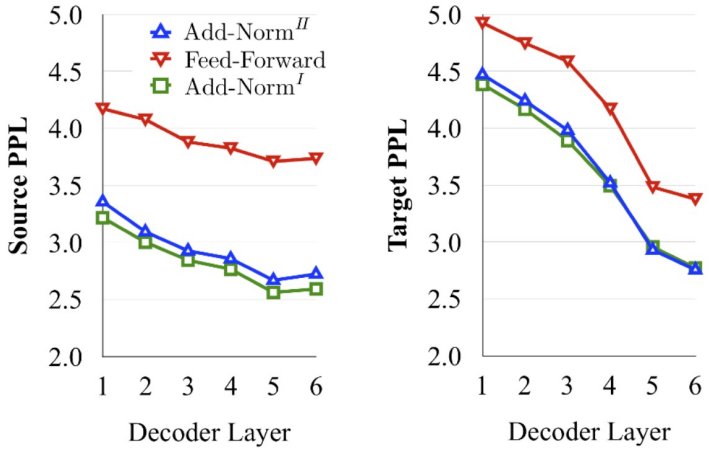
Dropping FFN Layers

Transformer Decoder 사실 FFN 필요 없다

On the Sub-layer Functionalities of Transformer Decoder, 2020, EMNLP (3 citations)



(a) Three Operations in IFM



(b) Source PPL (c) Target PPL

Figure 8: Illustration of (a) three operations within IFM, and (b,c) the source and target information evolution within IFM on En-De task.

05 Position-wise FFN

Dropping FFN Layers

Transformer Decoder 사실 FFN 필요 없다

On the Sub-layer Functionalities of Transformer Decoder, 2020, EMNLP (3 citations)

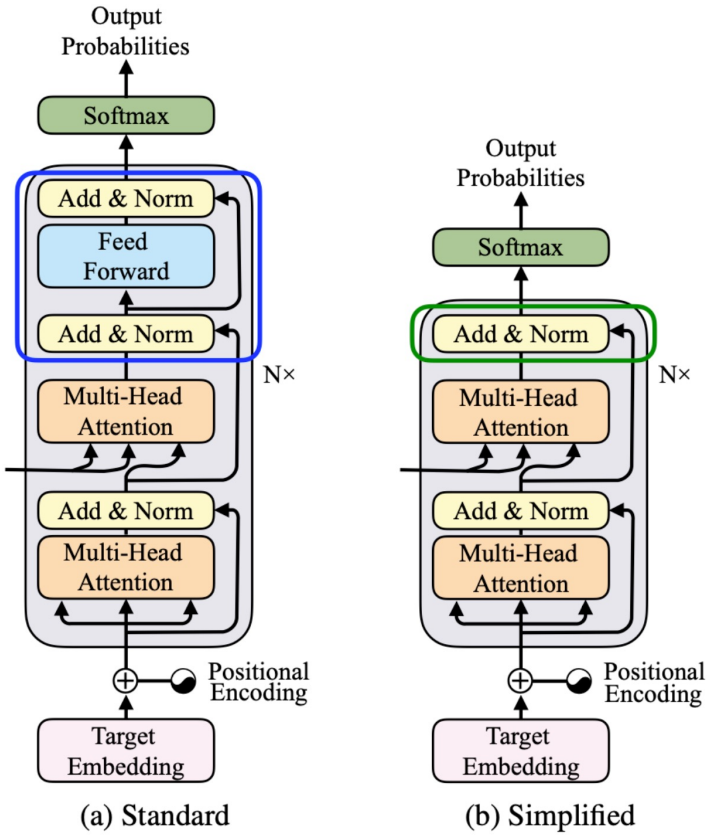


Figure 9: Illustration of (a) the standard decoder, and (b) the simplified decoder with simplified IFM.

	Decoder	BLEU	#Train.	#Infer.
En-De	Standard	27.45	63.93K	65.35
	Simplified	27.29	71.08K	72.93
	△	*-0.16*	*+11.18%*	*+11.60%*
En-Zh	Standard	32.24	32.49K	38.55
	Simplified	33.15	36.59K	54.06
	△	*+0.91*	*+12.62%*	*+40.23%*
En-Fr	Standard	40.39	68.28K	58.97
	Simplified	40.07	76.03K	67.23
	△	*-0.32*	*+11.35%*	*+14.01%*

Table 4: Performance of the simplified Base decoder. "#Train" denotes the training speed (words per second) and "#Infer." denotes the inference speed (sentences per second). Results are averages of three runs.

감사합니다