RoFormer : Enhanced Transformer with Rotary Position Embedding

윤세환

목차

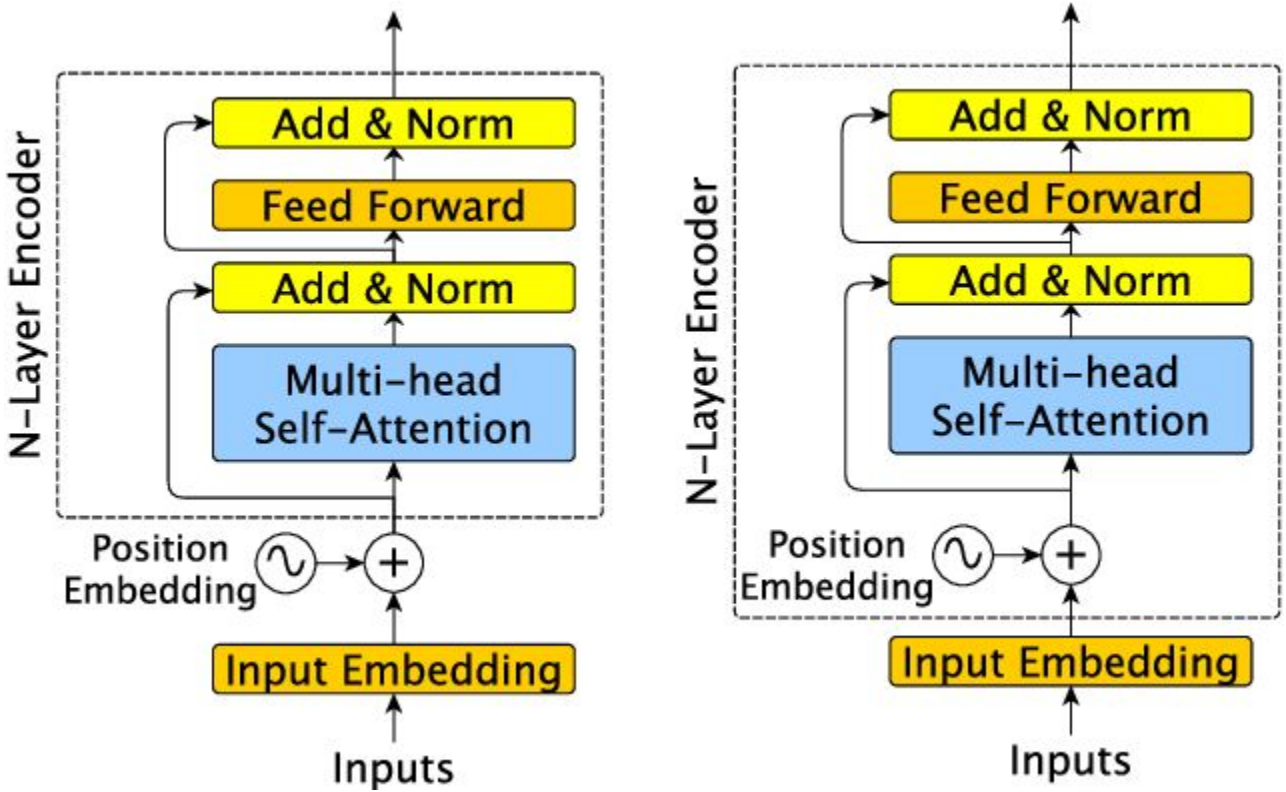
기존 Position Embedding 소개

기존 Position Embedding 방식 - absolute PE

- 각 단어의 위치값을 사용하여 위치 정보를 임베딩하고, 해당 값을 단어 임베딩값에 더하여 사용하는 방식

$$\begin{cases} \boldsymbol{p}_{i,2t} & = \sin(k/10000^{2t/d}) \\ \boldsymbol{p}_{i,2t+1} & = \cos(k/10000^{2t/d}) \end{cases}$$

$$\begin{aligned} \langle f_q(x_m, m), f_k(x_n, n) \rangle &= g(x_m, x_n, m, n) \\ &= (W_q(x_m + p_m))^T W_k(x_n + p_n) \end{aligned}$$


N-Layer Encoder
Add & Norm
Feed Forward
Add & Norm
Multi-head
Self-Attention
Position
Embedding
Input Embedding
Inputs
N-Layer Encoder
Add & Norm
Feed Forward
Add & Norm
Multi-head
Self-Attention
Position
Embedding
Input Embedding
Inputs

기존 Position Embedding 방식 - relative PE

- query와 key단어간 상대적 거리를 사용할 때, 두 단어간 거리가 특정 값 이상일 경우 거리가 더 멀어져도 더 이상 유의미한 정보를 제공하지 않을 것이라는 가설에 따라 거리차이에 대해 clipping을 수행할 수 있다.
- 두 거리의 간격을 r이라 할 경우 다음과 같이 표현된다.

$$r = clip(m - n, r_{min}, r_{max})$$

r_min과 r_max가 위에서 언급한 “두 단어간 거리차이에 대한 특정 값"이 된다.

기존 Position Embedding 방식 - relative PE

- absolute position embedding에서 사용된 수식을 전개해보면…

$$(W_q(x_m + p_m))^T W_k(x_n + p_n) =$$
$$x_m^T W_q^T W_k x_n + x_m^T W_q^T W_k p_n + p_m^T W_q^T W_k x_n + p_m^T W_q^T W_k p_n$$

위와 같은 수식이 되는데, x는 단어(content), p는 위치정보이므로 이를 토대로 생각해보면
순서대로 q단어-k단어, q단어-k위치, q위치-k단어, q위치-k위치를 의미하게 된다.

위 수식의 위치 정보를 각 단어의 위치가 아닌, 상대적인 위치 차이를 사용하는 방식이 relative position embedding

relative position embedding - XL, T5

- XL
  - query에 대해 위치정보를 사용하지 않는 대신, key의 위치정보를 query-key의 상대거리정보로 대체한다.
  - 대신 query 위치정보 자리에 학습 가능한 파라미터(u, v)를 사용한다.

$$x_m^T W_q^T W_k x_n + x_m^T W_q^T W_k p_{m-n} + u^T W_q^T W_k x_n + v^T W_q^T W_k p_{m-n}$$

- T5
  - q단어-k단어 외 나머지 부분을 모델이 알아서 학습하도록 만드는 방법

$$x_m^T W_q^T W_k x_n + b_{n,m}$$

relative position embedding - TUPE

- 단어와 절대적인 위치간의 상관관계가 매우 작은 것을 확인하였고,
  이에 기존 단어-위치간 관계를 나타내는 2개의 항을 통합한 항을 추가
- 단어 임베딩에 사용되는 가중치(W)와 위치 임베딩값이 사용되는 가중치(U)를
  분리

$$x_m^T W_q^T W_k x_n + p_m^T U_q^T U_k p_n + b_{n,m}$$

relative position embedding - DeBERTa

- 기존 단어-위치간 관계를 나타내는 2개의 항에서 사용되던 위치 값을 모두 relative 위치로 교체
- 이미 상대적인 위치를 고려했으므로, 마지막 항인 위치-위치간 관계는 더 이상 많은 정보를 제공하지 못한다고 판단하여 제거

$$x_m^T W_q^T W_k x_n + x_m^T W_q^T W_k p_{m-n} + p_{m-n}^T W_q^T W_k x_n$$

Rotary Position Embedding

Rotary Position Embedding

본 논문에서 소개한 Rotary Position Embedding은 아래 목적을 달성하기 위해 사용됨.

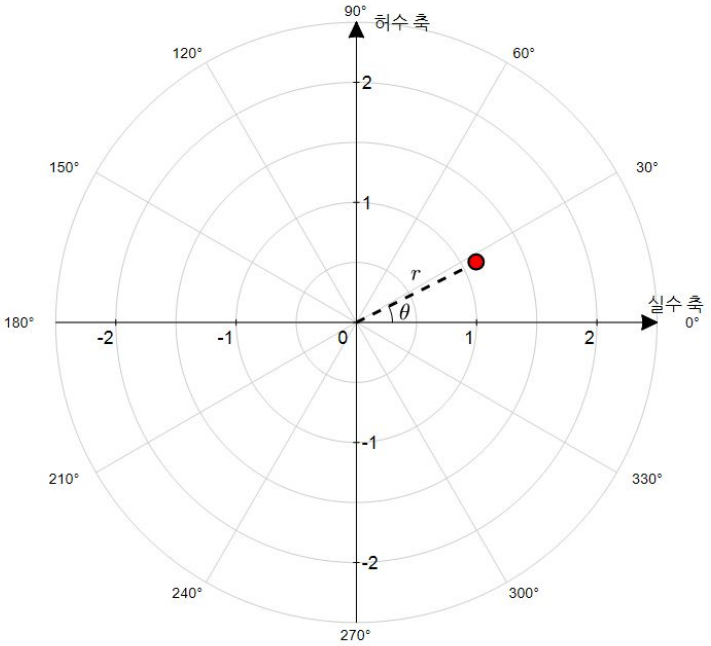
- query, key단어 임베딩값과 상대적인 거리를 사용하는 g함수를 정의하는 것

$$\langle f_q(x_m, m), f_k(x_n, n) \rangle = g(x_m, x_n, m-n)$$

- 이 때, 기존의 position embedding방식인, 값을 더하는 방식이 아닌 특정 값을 곱하는 방식을 적용하며, 이 곱해지는 matrix가 회전 변환을 수행하는 matrix가 된다.
- 즉, 두 단어간 상대적인 거리차이를 얼만큼 회전했는가로 나타내는 방식

필요 사전 지식 - complex number와 회전 변환

y축을 허수측, x축을 실수측이라고 가정하여 극 좌표계와 직교 좌표계간의 변환을 고려



빨간 점을 z라고 한다면, z의 직교 좌표계 상의 좌표는

$$z = x + jy$$

이며, x와 y는 다음과 같이 표시할 수 있다.

$$\begin{cases} x &= r\cos(\theta) \\ y &= r\sin(\theta) \end{cases}$$

즉, r=1 이라면 z는

$$z = \cos(\theta) + j\sin(\theta)$$

필요 사전 지식 - complex number와 회전 변환

$$z = \cos(\theta) + j\sin(\theta)$$

theta에 대해 미분

$$\frac{dz}{d\theta} = -\sin(\theta) + j\cos(\theta) = j^2\sin(\theta) + j\cos(\theta) = j(\cos(\theta) + j\sin(\theta)) = jz$$

1/z로 양변을 나눔

$$\frac{1}{z}\frac{dz}{d\theta} = j$$

theta에 대해 적분

$$\int \frac{1}{z}dz = \int jd\theta \implies \ln(z) = j\theta + C \implies j\theta$$

(theta에 0을 대입하면 C=0임을 쉽게 알 수 있습니다.)

$$z = e^{j\theta} = \cos(\theta) + j\sin(\theta)$$

Rotary Position Embedding

회전 변환을 통해 위치 정보를 반영한다고 가정했을시, f_q, f_k, g를 아래와 같이 정의할 수 있다.

$$f_q(x_m, m) = (W_q x_m) e^{im\theta}$$

$$f_k(x_n, n) = (W_k x_n) e^{in\theta}$$

$$g(x_m, x_n, m-n) = Re\left[(W_q x_m)(W_k x_n)^* e^{i(m-n)\theta}\right]$$

이전 페이지에서 나온 결과를 보면, e의 지수항에 있는 부분이 얼만큼 회전하는가를 의미하며 이는 곧 위치 정보를 의미하게 된다.
따라서 위 복소수의 실수 부분은 단어에 대한 정보, 허수에 대한 부분은 위치에 대한 정보를 나타낸다고 볼 수 있다.

Rotary Position Embedding

$$g(x_m, x_n, m-n) = Re\left[(W_q x_m)(W_k x_n)^* e^{i(m-n)\theta}\right]$$

위 수식에서 Re는 복소수의 실수 부분을 의미하며, *는 conjugate complex number를 의미한다.

$$Re[a+bi] = a \qquad \overline{a+bi} = a-bi$$

- Re의 경우, 실제 attention score를 구하는 과정에서는 허수부를 사용할 수 없기에 사용됨
- conjugate를 수행한 이유는, 회전변환시 두 위치간의 차이를 반영해야 하므로, 위치 정보를 표시하는 허수부의 기호를 변경하기 위해서 사용됨

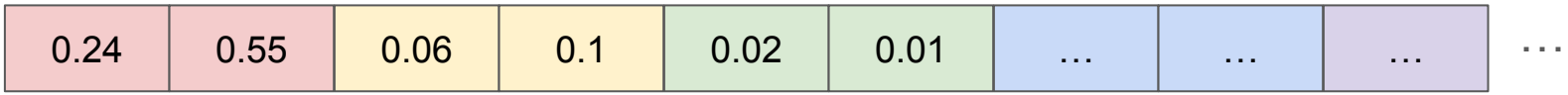
$$\left(W_q x_m e^{im\theta}\right)\left(\overline{W_k x_n e^{in\theta}}\right)$$

Rotary Position Embedding 과정

단어 embedding 벡터의 element에 대해, 2개씩 묶어서 아래 회전 변환을 수행한다.
(수식에서는 query에 대해 수행하지만, key에 대해서도 수식 자체는 동일)

$$f_q(x_m, m) = \begin{pmatrix} \cos(m\theta) & -\sin(m\theta) \\ \sin(m\theta) & \cos(m\theta) \end{pmatrix} \begin{pmatrix} W_q^{11} & W_q^{12} \\ W_q^{21} & W_q^{22} \end{pmatrix} \begin{pmatrix} x_m^1 \\ x_m^2 \end{pmatrix}$$

단어 임베딩값



Rotary Position Embedding 과정

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

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

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

Rotary Position Embedding 과정

$$\boldsymbol{R}^{d}_{\Theta,m}\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{d-1} \\ x_d \end{pmatrix} \otimes \begin{pmatrix} \cos m\theta_1 \\ \cos m\theta_1 \\ \cos m\theta_2 \\ \cos m\theta_2 \\ \vdots \\ \cos m\theta_{d/2} \\ \cos m\theta_{d/2} \end{pmatrix} + \begin{pmatrix} -x_2 \\ x_1 \\ -x_4 \\ x_3 \\ \vdots \\ -x_d \\ x_{d-1} \end{pmatrix} \otimes \begin{pmatrix} \sin m\theta_1 \\ \sin m\theta_1 \\ \sin m\theta_2 \\ \sin m\theta_2 \\ \vdots \\ \sin m\theta_{d/2} \\ \sin m\theta_{d/2} \end{pmatrix}$$

Rotary Position Embedding 과정

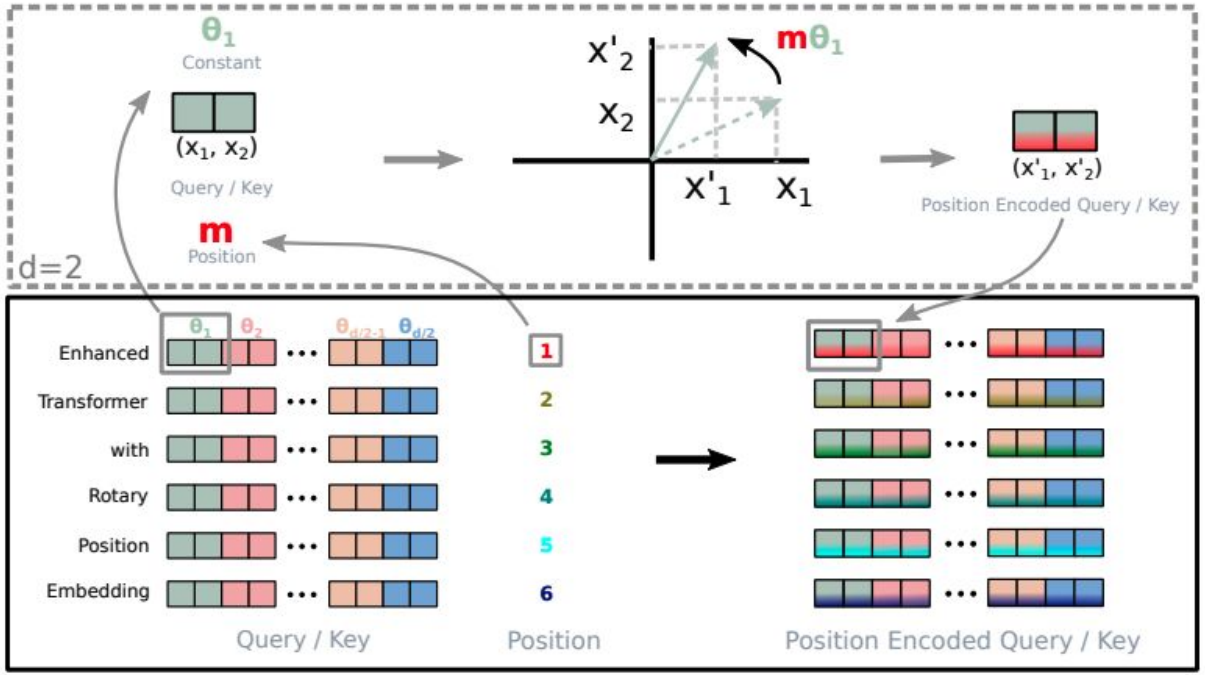


Figure 1: Implementation of Rotary Position Embedding(RoPE).

Rotary Position Embedding - Long term decay

- Long term decay
  - transformer와 동일하게 $\theta_i = 10000^{-2i/d}$ 사용
  - Relative Position이 멀어지면 Inner product값이 감소하는 효과를 얻을 수 있으며, 이는 RelativePosition에서 사용한 clipping과 유사한 효과를 낼 수 있다.

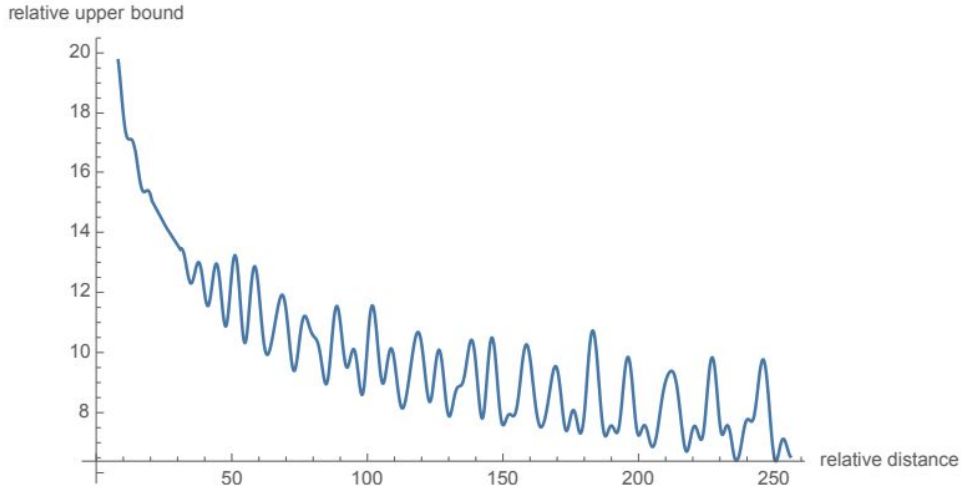


Figure 2: Long-term decay of RoPE.

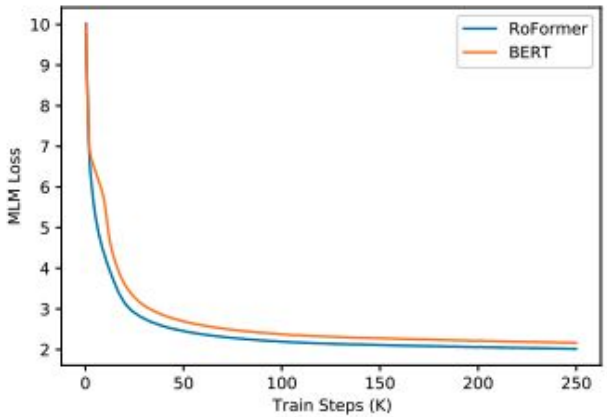
Experiments

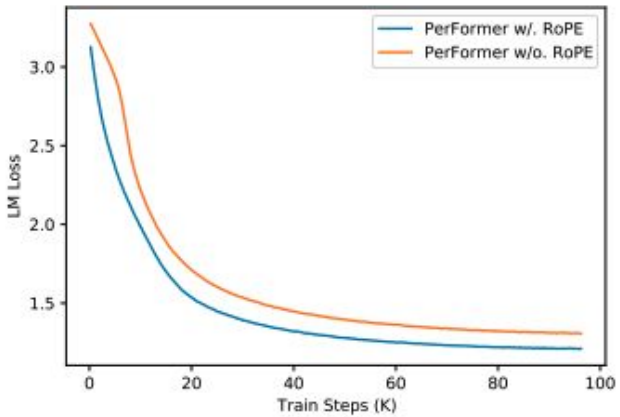
- 기계번역
  - 영어를 독일어로 번역하는 task에 적용하였으며, 일반적인 transformer와 Roformer를 사용한 transformer를 BLEU 스코어로 비교
  - 기본 transformer 모델에 비해 약간 더 높은 BLEU 스코어를 보여줌

Model	BLEU
Transformer-baseVaswani et al. [2017]	27.3
RoFormer	**27.5**

Experiments

- pre-training language modeling
  - BERT 모델에서 position embedding으로 RoFormer 사용하여 pre-training 수행
  - BERT 모델의 손실함수 중 MLM (Masked Language Modeling) 에 대한 손실값을 비교지표로 사용





Experiments

- fine-tuning on GLUE tasks
  (Global Language Understanding Evaluation, 모델들의 자연어 이해 능력을 평가)
  - MRPC, QQP, STS-B는 f1 score, 나머지는 정확도를 사용

Table 2: Comparing RoFormer and BERT by fine tuning on downstream GLEU tasks.

Model	MRPC	SST-2	QNLI	STS-B	QQP	MNLI(m/mm)
BERTDevlin et al. [2019]	88.9	93.5	90.5	85.8	71.2	84.6/83.4
RoFormer	**89.5**	90.7	88.0	**87.0**	**86.4**	80.2/79.8

Experiments

- 중국어 데이터 평가
  - 긴 텍스트에 대한 성능을 보여주기 위해, 평가시에는 512개 이상의 단어로 구성된 데이터를 사용
  - 3개의 입력을 받고 (A, B, C) A와 B 그리고 A와 C중 어떤 것이 더 가까이 위치했는지 맞추는

Model	BERTDevlin et al. [2019]	WoBERTSu [2020]	NEZHAWei et al. [2019]	RoFormer
Tokenization level	char	word	char	word
Position embedding	abs.	abs.	rel.	RoPE

Stage	Max seq length	Batch size	Training steps	Loss	Accuracy
1	512	256	200k	1.73	65.0%
2	1536	256	12.5k	1.61	66.8%
3	256	256	120k	1.75	64.6%
4	128	512	80k	1.83	63.4%
5	1536	256	10k	1.58	67.4%
6	512	512	30k	1.66	66.2%

Model	Validation	Test
BERT-512	64.13%	67.77%
WoBERT-512	64.07%	68.10%
RoFormer-512	64.13%	68.29%
RoFormer-1024	**66.07%**	**69.79%**

관련 정보

- 본 논문은 EleutherAI에 적용되기 시작하면서 유명해졌으며, 실제 해당 논문에 RoFormer을 적용한 결과를 통해 RoFormer의 효과를 확인할 수 있음.

- 이후 ALiBi라는 방식이 제시되었는데, 구현이 더 쉽고 성능 및 속도 측면에서도 RoFormer 방식보다 더 높은 성능을 보임

참고 링크

- 복소수 기초
  https://angeloyeo.github.io/2022/01/05/complex_number_basic.html
- 회전변환
  https://laplandjin.tistory.com/entry/%ED%9A%8C%EC%A0%84-%EB%B3%80%ED%99%98
- 논문
  https://arxiv.org/abs/2104.09864