Your Transformer May Not be as Powerful as You Expect

Shengjie Luo

School of Intelligence Science and Technology, Peking University





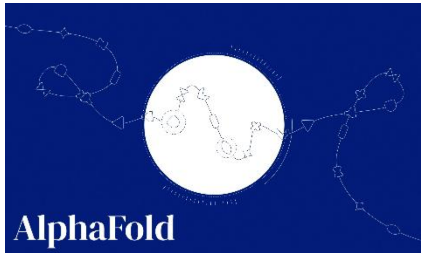

Shengjie Luo*	Shanda Li*	Shuxin Zheng	Tie-Yan Liu	Liwei Wang†	Di He†
Peking University	CMU	Microsoft Research	Microsoft Research	Peking University	Peking University

Transformer – SOTA deep learning model

NLP Tasks: Machine Translation, Language Pre-training

CV Tasks: Classification, Detection, Segmentation

Graph-Learning Tasks: Node prediction, Graph prediction



Transformer Model Recap (original)

Attention

Make all tokens (semantics) interact with each other

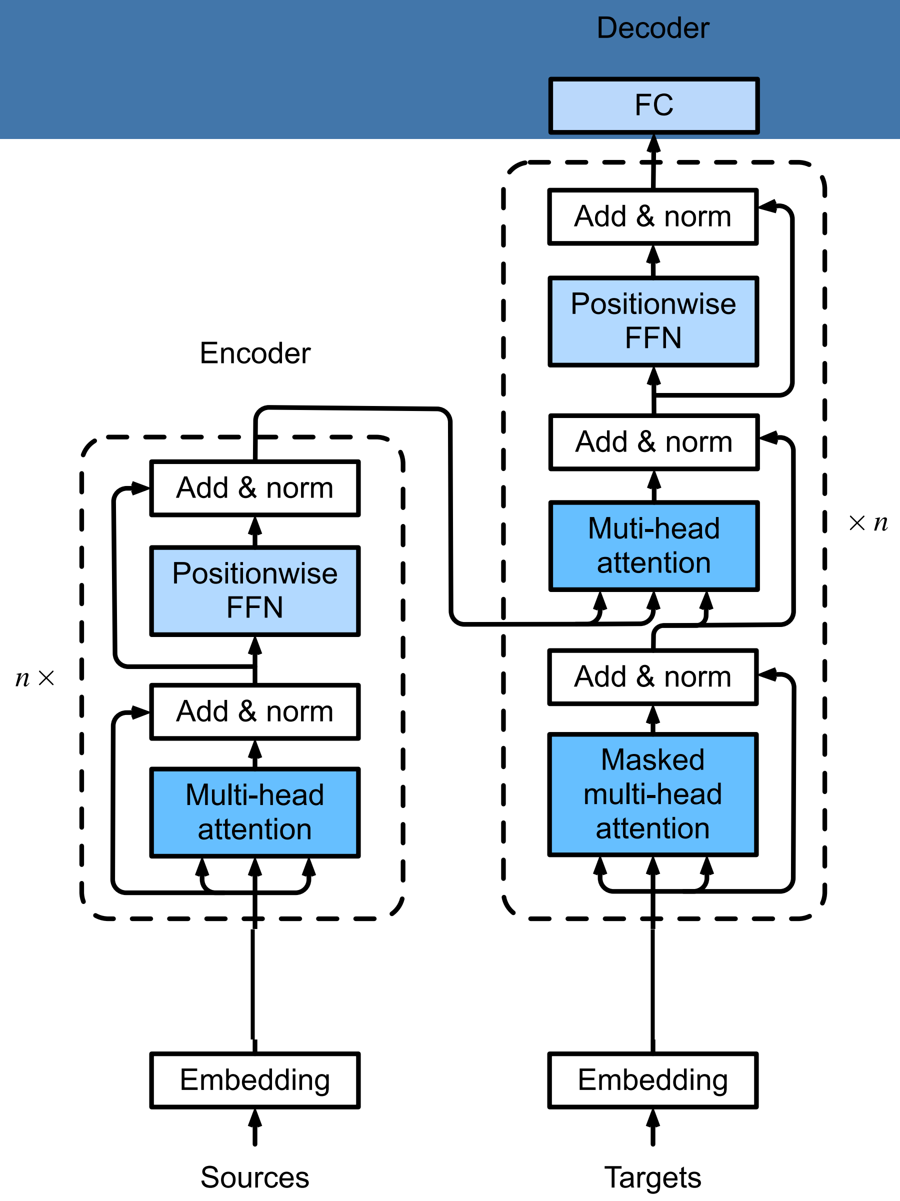
$$\boldsymbol{A}^h(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}_Q^h(\boldsymbol{X}\boldsymbol{W}_K^h)^\top\right);$$

$$\mathrm{Attn}(\boldsymbol{X}) = \boldsymbol{X} + \sum_{h=1}^{H} \boldsymbol{A}^h(\boldsymbol{X})\boldsymbol{X}\boldsymbol{W}_V^h\boldsymbol{W}_O^h;$$

Position-wise FFN

Learn abstractive contextual representation

$$\mathrm{FFN}(\boldsymbol{X}) = \boldsymbol{X} + \mathrm{ReLU}(\boldsymbol{X}\boldsymbol{W}_1)\boldsymbol{W}_2$$



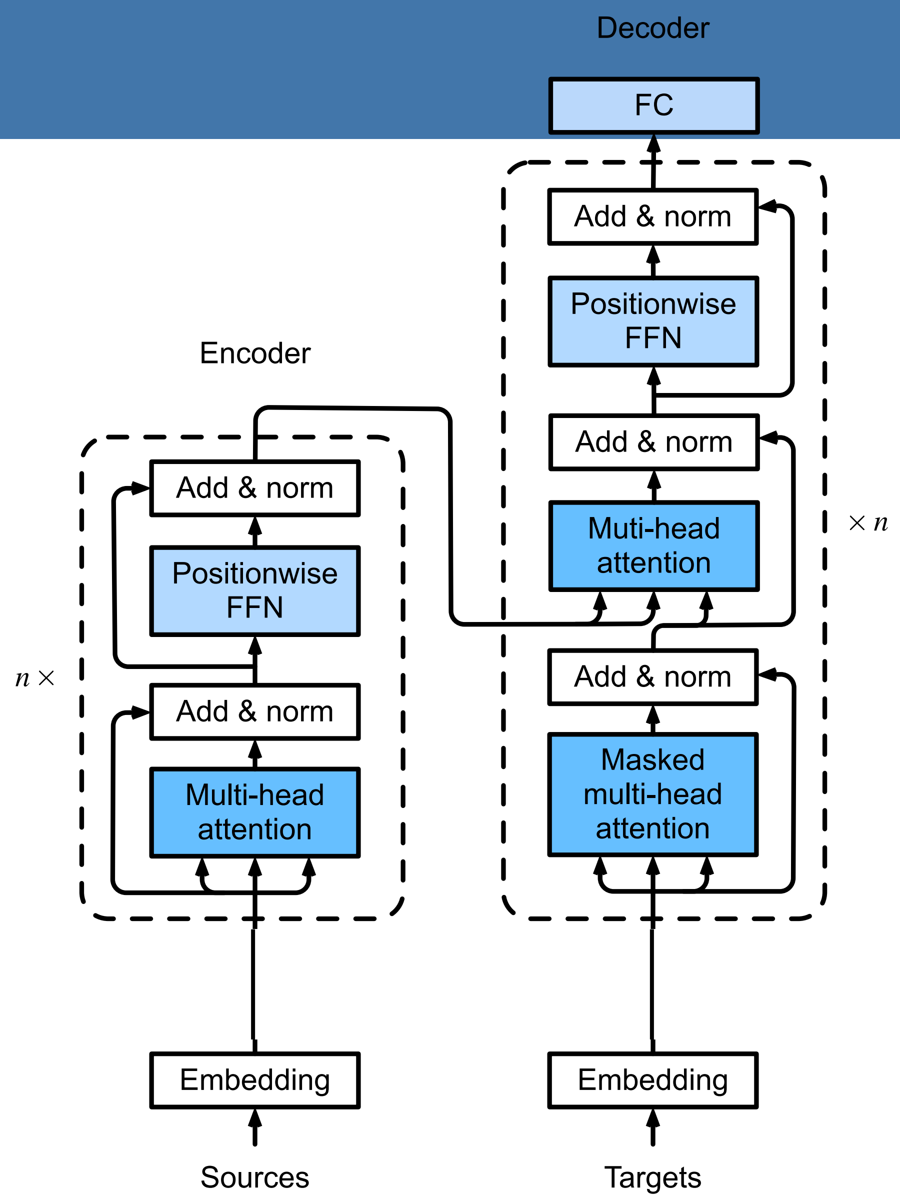
Transformer Model Recap (original)

Attention & FFN are position-insensitive.

$$\boldsymbol{A}^h(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}_Q^h(\boldsymbol{X}\boldsymbol{W}_K^h)^\top\right);$$

$$\mathrm{Attn}(\boldsymbol{X}) = \boldsymbol{X} + \sum_{h=1}^{H} \boldsymbol{A}^h(\boldsymbol{X})\boldsymbol{X}\boldsymbol{W}_V^h\boldsymbol{W}_O^h;$$

$$\mathrm{FFN}(\boldsymbol{X}) = \boldsymbol{X} + \mathrm{ReLU}(\boldsymbol{X}\boldsymbol{W}_1)\boldsymbol{W}_2$$



Transformer Model Recap (original)

Attention & FFN are position-insensitive.

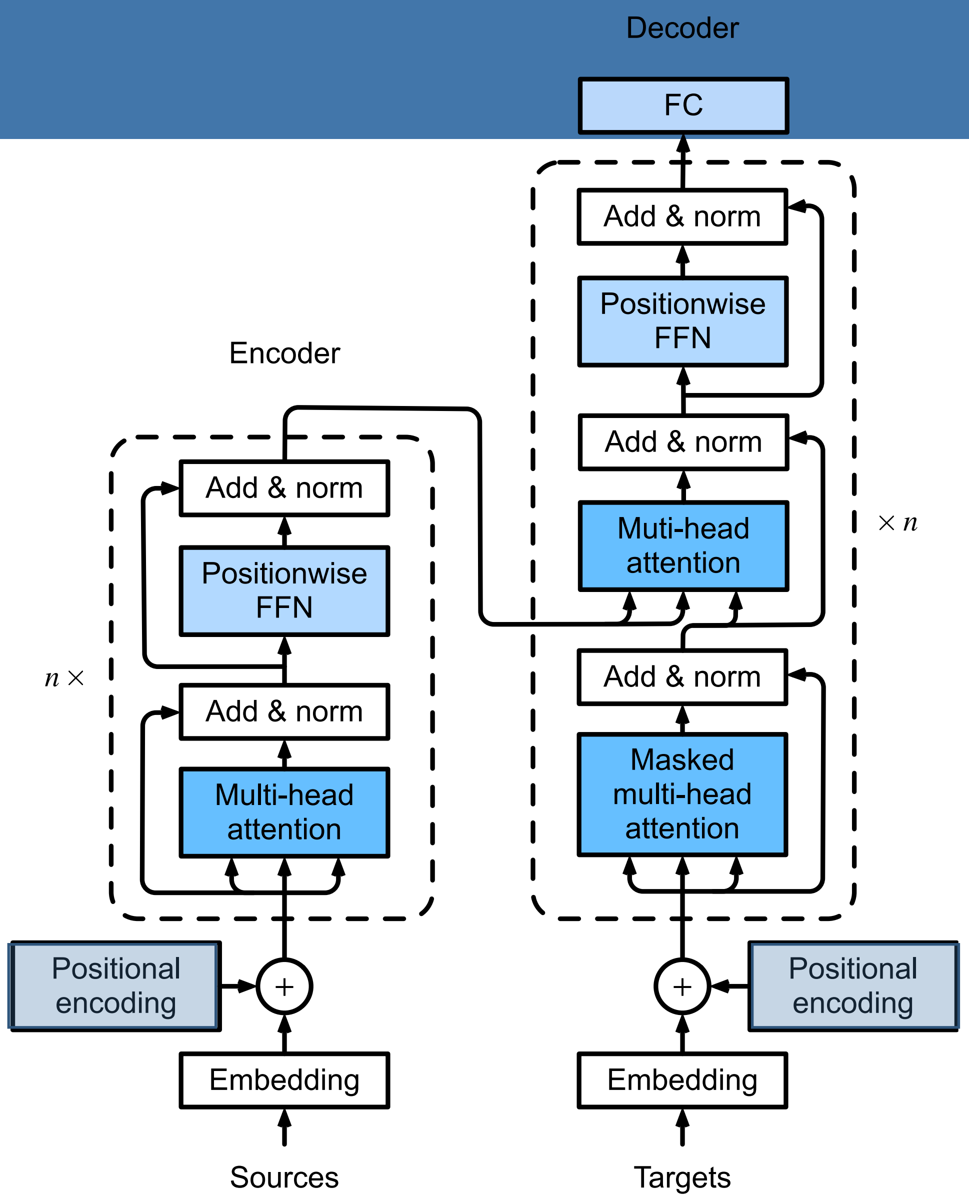
$$\boldsymbol{A}^h(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}_Q^h(\boldsymbol{X}\boldsymbol{W}_K^h)^\top\right);$$

$$\mathrm{Attn}(\boldsymbol{X}) = \boldsymbol{X} + \sum_{h=1}^{H} \boldsymbol{A}^h(\boldsymbol{X})\boldsymbol{X}\boldsymbol{W}_V^h\boldsymbol{W}_O^h;$$

$$\mathrm{FFN}(\boldsymbol{X}) = \boldsymbol{X} + \mathrm{ReLU}(\boldsymbol{X}\boldsymbol{W}_1)\boldsymbol{W}_2$$

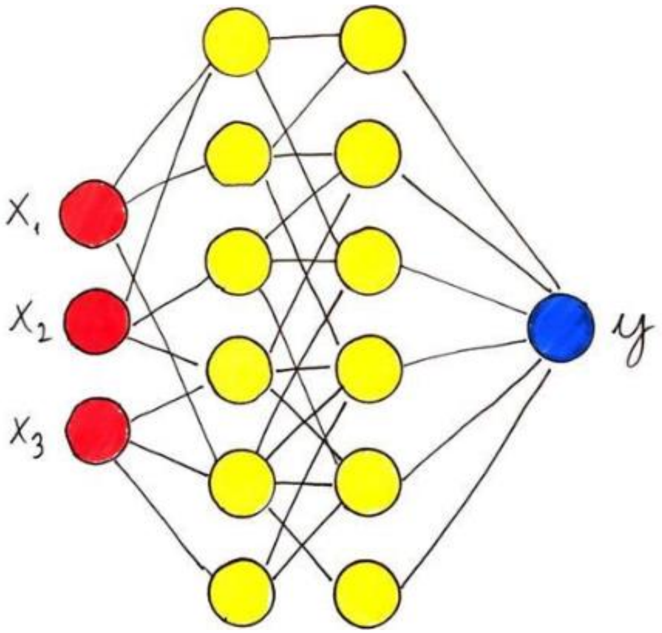
(Absolute) Positional Encoding (APE)

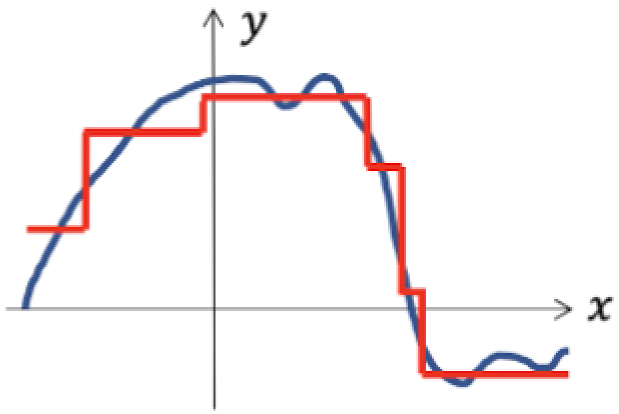
Assign an embedding vector to each position index



How Powerful are APE-based Transformers?

- **Definition [Universal approximator]**
  - A neural network can approximate any continuous functions in R^n.
  - Shallow and wide networks (Funahashi, 1989; Cybenko, 1989; Barron, 1994;)
  - Deep and thin networks (Lu, 2017; Hanin, 2017; Lin, 2018)





How Powerful are APE-based Transformers?

Theorem (informal) (Yun et al., ICLR 2020)

Given any fixed input length n, Transformers with APE can approximate any continuous sequence-to-sequence function with arbitrary precision under mild assumptions.

But in practice, absolute positional encoding is not so popular now……

Problems

- Extrapolation
  - APE-based Transformer usually generalizes poorly to **longer sequences**, as those positional embeddings for large indexes are hardly trained.

Problems

- Extrapolation
  - APE-based Transformer usually generalizes poorly to **longer sequences**, as those positional embeddings for large indexes are hardly trained.
- Relative information
  - Empirically, people find that absolute positional encoding cannot capture **relative positional signal** well

Problems

- Extrapolation
  - APE-based Transformer usually generalizes poorly to **longer sequences**, as those positional embeddings for large indexes are hardly trained.
- Relative information
  - Empirically, people find that absolute positional encoding cannot capture **relative positional signal** well
- Apply to other data modality
  - Image and graph data require several **transformation-invariant properties**, such as rotation and translation, which are difficult to be satisfied by APE.

From Absolute-PE to Relative-PE

- Relative Positional Encoding (RPE) encodes **relative distance** $i - j$ for each position pair (i, j) in the **attention module**

$$\boldsymbol{A}^h_{\mathrm{RPE}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}^h_Q(\boldsymbol{X}\boldsymbol{W}^h_K)^\top + \boldsymbol{B}\right)$$

The (i, j)-th entry of ***B*** models the interaction between the i-th and j-th position.

From Absolute-PE to Relative-PE

- Relative Positional Encoding (RPE) encodes **relative distance** $i - j$ for each position pair (i, j) in the **attention module**

$$\boldsymbol{A}^h_{\mathrm{RPE}}(\boldsymbol{X}) = \operatorname{softmax}\left(\boldsymbol{X}\boldsymbol{W}^h_Q(\boldsymbol{X}\boldsymbol{W}^h_K)^\top + \boldsymbol{B}\right)$$

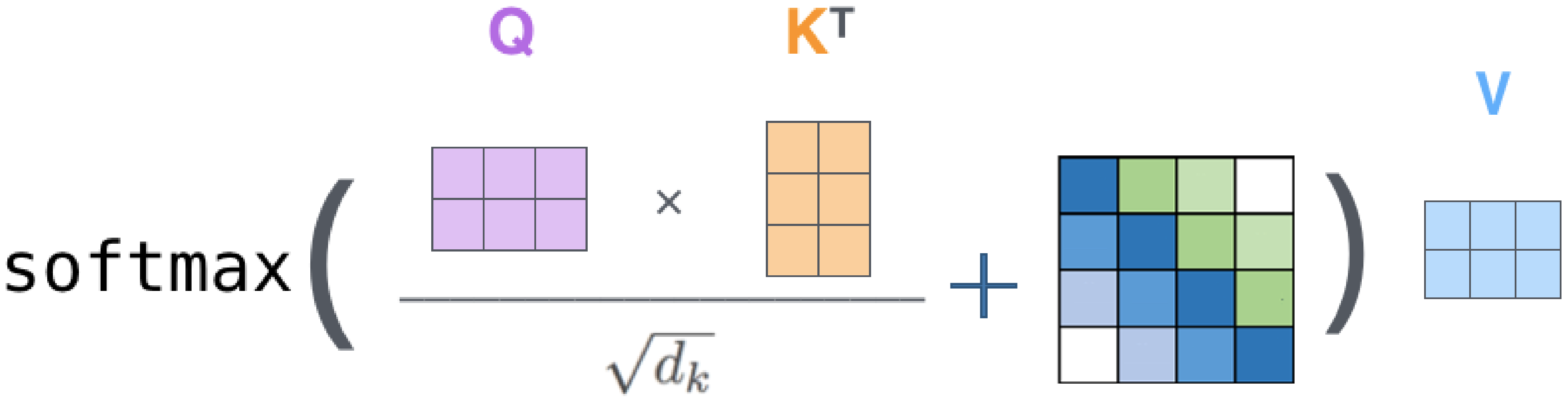
The (i, j)-th entry of ***B*** models the interaction between the i-th and j-th position.

- RPE can be easily applied to various forms of data (e.g., graphs, images, etc) and generalize better to longer sequences.

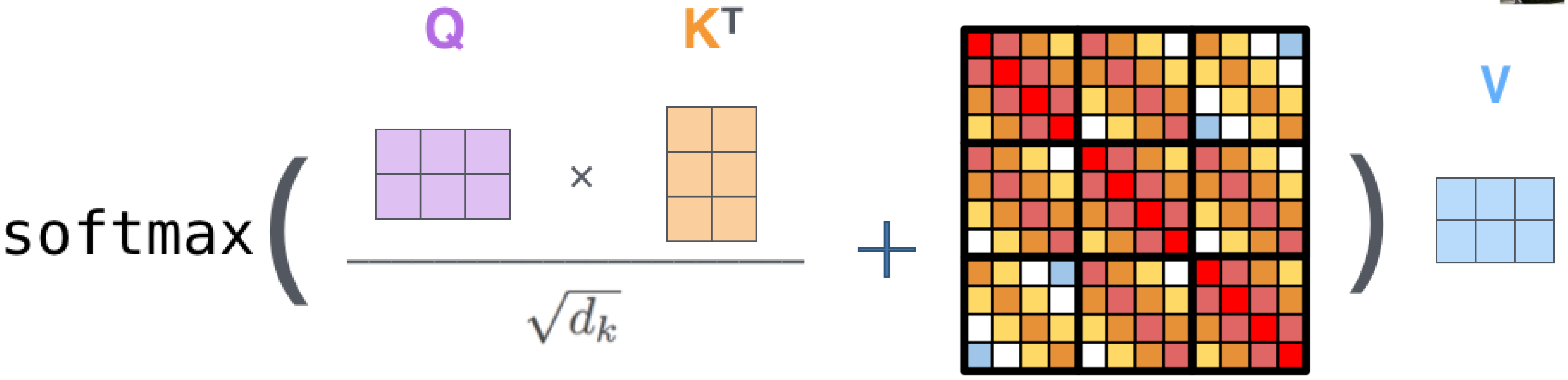
Examples of RPE: T5

$$\boldsymbol{A}^h_{\mathrm{RPE}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}^h_Q(\boldsymbol{X}\boldsymbol{W}^h_K)^\top + \boldsymbol{B}\right)$$

B is parametrized as a fully learnable **Toeplitz matrix**, i.e., $b_{ij} = m_i$



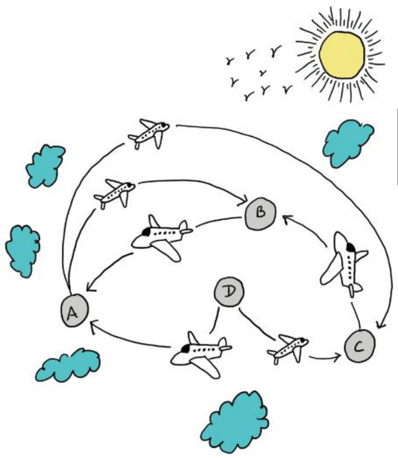
Examples of RPE: Swin-Transformer (SOTA in CV)



Examples of RPE: Graphormer (SOTA in Graph Learning)

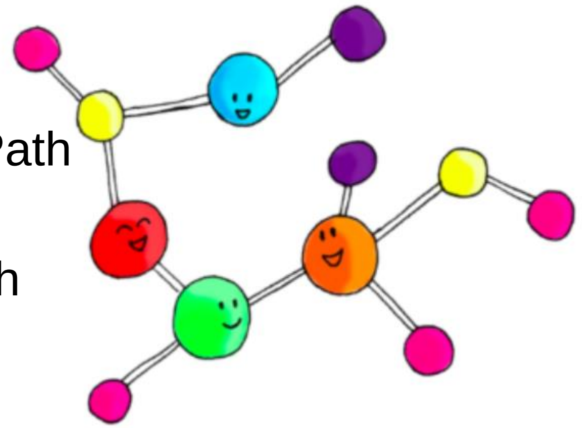
$$\mathrm{softmax}\left(\frac{Q \times K^T}{\sqrt{d_k}} + b_{\phi(v_i,v_j)}\right) V$$

$\phi(v_i, v_j)$: Any Metric that Measures the Distance Between v_i & v_j.



☑ Unweighted Shortest Path

Weighted Shortest Path



List

- Shaw's RPE [58]: $b_{ij} = \boldsymbol{X}_i \boldsymbol{W}_Q^h \boldsymbol{r}_{i-j}^\top$, where $\boldsymbol{r}_{i-j}$ are learnable vectors.
- T5 [54]: $b_{ij} = m_{i-j}$, where m_{i-j} are learnable scalars, i.e., $\boldsymbol{B}$ is parameterized as a Toeplitz matrix [22, 45].
- DeBERTa [25]: $b_{ij} = \boldsymbol{X}_i \boldsymbol{W}_Q^h \boldsymbol{r}_{i-j}^\top + \boldsymbol{s}_{i-j}(\boldsymbol{X}_j \boldsymbol{W}_K^h)^\top$, where $\boldsymbol{r}_{i-j}$ and $\boldsymbol{s}_{i-j}$ are learnable vectors.
- Transformer-XL [10]: $b_{ij} = \boldsymbol{X}_i \boldsymbol{W}_Q^h (\boldsymbol{r}_{i-j} \tilde{\boldsymbol{W}}_K^h)^\top + \boldsymbol{u}(\boldsymbol{X}_j \boldsymbol{W}_K^h)^\top + \boldsymbol{v}(\boldsymbol{r}_{i-j} \tilde{\boldsymbol{W}}_K^h)^\top$, where $\boldsymbol{u}, \boldsymbol{v}$ and $\tilde{\boldsymbol{W}}_K^h$ are all learnable vectors/matrix, and $\boldsymbol{r}_{i-j}$ are sinusoidal positional encoding vectors fixed during training.

Question

	Transformer with APE	Transformer with RPE
Generalize to longer sequence	No	Yes
Easily extend to image/graph data	No	Yes
Universal approximation	Yes	? (This work)

Negative Results

- All currently used RPE-Transformers are **not** universal approximators for continuous sequence-to-sequence functions!

Negative Results

- There exist continuous sequence-to-sequence functions that RPE-based Transformers **cannot approximate no matter how deep and wide the model is**!

Negative Results

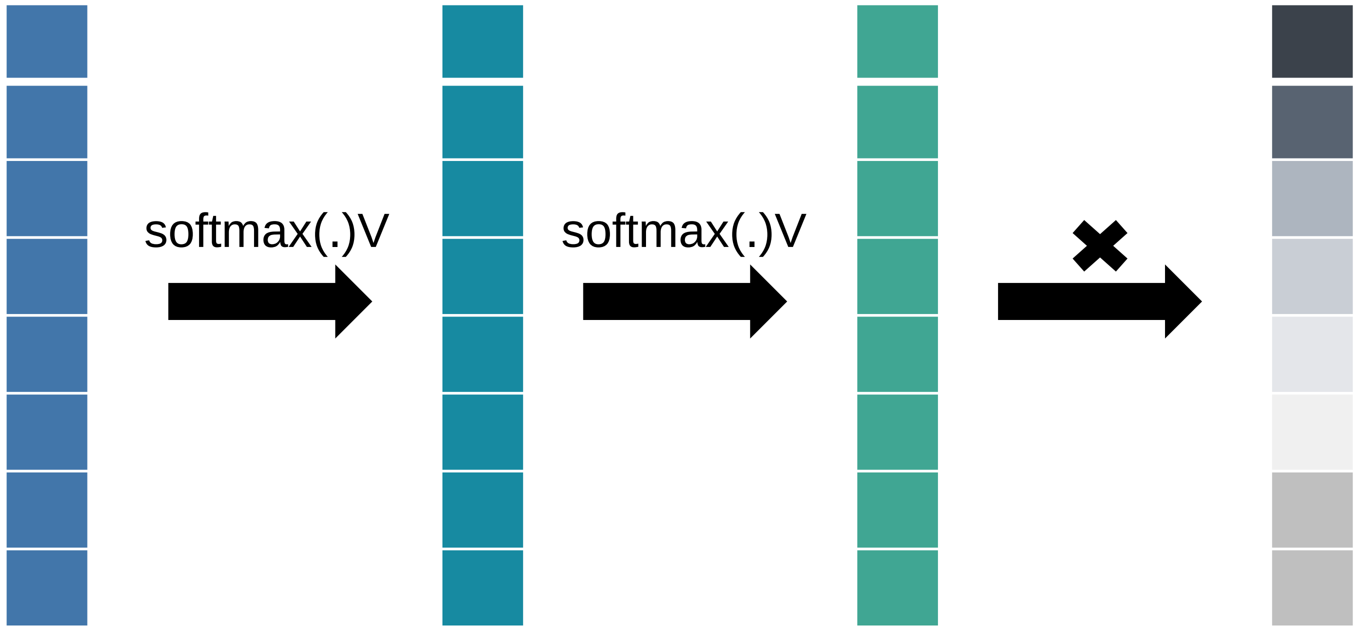
- There exist continuous sequence-to-sequence functions that RPE-based Transformers **cannot approximate no matter how deep and wide the model is**!

- Motivating example

$$\boldsymbol{A}^h_{\mathrm{RPE}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}^h_Q(\boldsymbol{X}\boldsymbol{W}^h_K)^\top + \boldsymbol{B}\right)$$

$$\mathrm{Attn}(\boldsymbol{X}) = \boldsymbol{X} + \sum_{h=1}^{H} \boldsymbol{A}^h(\boldsymbol{X})\boldsymbol{X}\boldsymbol{W}^h_V\boldsymbol{W}^h_O;$$

Identical tokens



Position-dependent outputs

See detailed proofs in our paper https://arxiv.org/abs/2205.13401

Negative Results

- There exist continuous sequence-to-sequence functions that RPE-based Transformers **cannot approximate no matter how deep and wide the model is**!

- Motivating example

$$A^h_{\mathrm{RPE}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}^h_Q(\boldsymbol{X}\boldsymbol{W}^h_K)^\top + \boldsymbol{B}\right)$$

$$\mathrm{Attn}(\boldsymbol{X}) = \boldsymbol{X} + \sum_{h=1}^{H} \boldsymbol{A}^h(\boldsymbol{X})\boldsymbol{X}\boldsymbol{W}^h_V\boldsymbol{W}^h_O;$$

Attention matrix is always a right stochastic matrix. This restricts the network from capturing rich positional information in the RPEs **(B)** and limits model's capacity.

See detailed proofs in our paper https://arxiv.org/abs/2205.13401

Question

	Transformer with APE	Transformer with RPE
Generalize to longer sequence	No	Yes
Easily extend to image/graph data	No	Yes
Universal approximation	Yes	**No**

Overcome the problem

- Sufficient conditions for universal attention module (See detailed proofs in our paper)
  - Attentive condition: the module should <u>cover the originally defined attention</u>.
  - Position-aware condition: The module needs to <u>break the right-stochastic-matrix limitation</u>

Overcome the problem

- Sufficient conditions for universal attention module (See detailed proofs in our paper)
  - Attentive condition: the module should <u>cover the originally defined attention</u>.
  - Position-aware condition: The module needs to <u>break the right-stochastic-matrix limitation</u>

Theorem 3. *Given $n, d \in \mathbb{N}^*$, $p \in [1, +\infty)$, $\varepsilon > 0$, a compact set $\mathcal{D} \subseteq \mathbb{R}^{n\times d}$, and a continuous sequence-to-sequence function $f : \mathcal{D} \to \mathbb{R}^{n\times d}$. Assume that $\boldsymbol{A}_U^h$ satisfies the following conditions:*

- ***Attentive condition.*** *For any $\boldsymbol{u} \in \mathbb{R}^{d\times 1}$ and $c \in \mathbb{R}$, there exists a parametrization of $\boldsymbol{A}_U^h$, such that $\boldsymbol{A}_U^h(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{u}(\boldsymbol{X}\boldsymbol{u} - c\boldsymbol{1})^\top\right)$.*
- ***Position-aware condition.*** *There exists a parametrization of $\boldsymbol{A}_U^h$ and a vector $\boldsymbol{v} \in \mathbb{R}^n$ whose entries are all distinct, such that $\boldsymbol{A}_U^h(\boldsymbol{X})\boldsymbol{1} = \boldsymbol{v}$ for any $\boldsymbol{X} \in \mathbb{R}^{n\times d}$.*

Then there exists a Transformer network $g \in \Omega_{\mathrm{U}}^{2,1,4}$ such that $\left(\int_{\mathcal{D}} \|f(\boldsymbol{X}) - g(\boldsymbol{X})\|_p^p \mathrm{d}\boldsymbol{X}\right)^{\frac{1}{p}} < \varepsilon$, where $\|\cdot\|_p$ denotes the entry-wise ℓ^p norm for matrices.

Practical instantiation

- A Universal RPE-based Transformer (Universal RPE-based attention module)

$$A_{\mathrm{U}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top + \boldsymbol{B}\right) \odot \boldsymbol{C},$$

Practical instantiation

- A Universal RPE-based Transformer (Universal RPE-based attention module)

$$A_{\mathrm{U}}(\boldsymbol{X}) = \mathrm{softmax}\left(\boldsymbol{X}\boldsymbol{W}_Q(\boldsymbol{X}\boldsymbol{W}_K)^\top + \boldsymbol{B}\right) \odot \boldsymbol{C},$$

- Capacity: URPE-based Transformers are universal approximators
- Parameter efficiency: introduce **0.01%** additional parameters
- Compatible: can be used in both NLP generation and understanding tasks, can be applied to tasks beyond NLP.

Overview

	Transformer with APE	Transformer with RPE	Transformer with URPE
Generalize to longer sequence	No	Yes	Yes
Easily extend to image/graph data	No	Yes	Yes
Universal approximation	Yes	No	Yes

Experiments: Synthetic Tasks

- Verify whether the theoretical claims are correct
  - **Position Identification** (PI): to predict the position index of each token.

$$f_{PI}(w_1, w_2, \dots, w_n) = (1, 2, \dots, n)$$

  - **Even Token Prediction** (ETP): to output the input tokens at positions with even number index.

$$f_{ETP}(w_1, w_2, \dots, w_n) = (w_2, w_4, \dots, w_n, EOS, \dots, EOS)$$

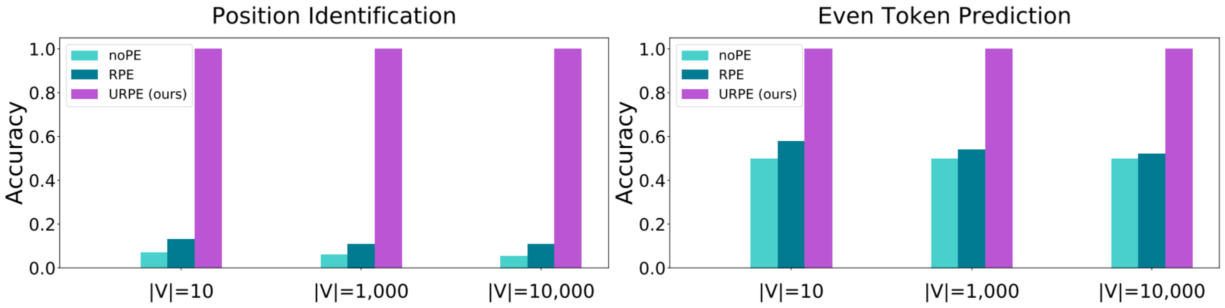


Figure 1: Results on synthetic sequence-to-sequence tasks: (1) Position Identification (Left Panel); (2) Even Token Prediction (Right Panel). $|V|$ is the vocabulary size. The URPE-based Transformer model consistently solves both tasks across different settings while other methods fail.

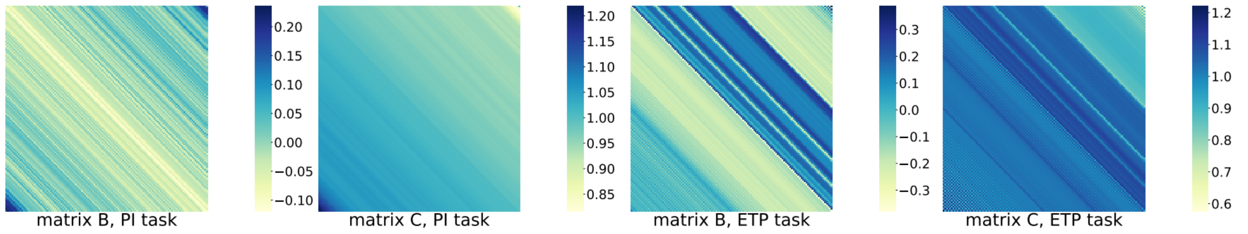


Figure 2: Visualizations of the learned Universal RPE (matrix B and C in Eq.(8)). It can be easily seen that the matrix $\boldsymbol{B}$ and $\boldsymbol{C}$ capture different aspects of positional information.

Experiments: Long-sequence Language Modelling

Table 1: Language model perplexity scores on WikiText-103 validation and test set. We use * to indicate the best performance. All the results of the baseline methods are reported in [10]

Model	#Params	Valid Perplexity	Test Perplexity
LSTM [21]	-	/	48.7
TCN [2]	-	/	45.2
GCNN-8 [11]	-	/	44.9
LSTM+Neural cache [21]	-	/	40.8
GCNN-14 [11]	-	/	37.2
QRNN [46]	151M	/	33.0
Hebbian+Cache [53]	-	/	29.9
Transformer-XL Base [10]	151M	23.1	24.0
Transformer-XL Base + URPE-based Attention (ours)	151M	22.4*	23.2*

Experiments: Graph Learning (Molecular Property Prediction)

Table 2: Mean Absolute Error (MAE) on ZINC test set. We use * to indicate the best performance.

Model	#Params	Test MAE on ZINC-Subset	Test MAE on ZINC-Full
GIN [66]	509,549	0.526±0.051	0.088±0.002
GraphSAGE [23]	505,341	0.398±0.002	0.126±0.003
GAT [62]	531,345	0.384±0.007	0.111±0.002
GCN [35]	505,079	0.367±0.011	0.113±0.002
MoNet [48]	504,013	0.292±0.006	0.090±0.002
GatedGCN-PE [5]	505,011	0.214±0.006	-
MPNN(sum) [20]	480,805	0.145±0.007	-
HIMP [17]	614,516	0.151±0.006	0.036±0.002
PNA [8]	387,155	0.142±0.010	-
GT [15]	588,929	0.226±0.014	-
SAN [37]	508,577	0.139±0.006	-
Graphormer [67]	489,321	0.122±0.006	0.052±0.005
Graphormer+URPE-based Attention (ours)	491,737	0.086±0.007*	0.028±0.002*

Table 3: Results on PCQM4M from OGB-LSC. We use * to indicate the best performance. The results of the baselines are reported in [67, 29].

Model	#Params	Valid MAE
GCN [35]	2.0M	0.1691
GIN [66]	3.8M	0.1537
GCN-VN [35, 20]	4.9M	0.1485
GIN-VN [66, 20]	6.7M	0.1395
GINE-VN [6, 20]	13.2M	0.1430
DeeperGCN-VN [38, 20]	25.5M	0.1398
GT [15]	0.6M	0.1400
GT-Wide [15]	83.2M	0.1408
Graphormer [67]	12.5M	0.1264
Graphormer + URPE-based Attention (ours)	12.5M	0.1238*

Thanks!

For further questions, feel free to email
luosj@stu.pku.edu.cn

Personal website: https://lsj2408.github.io