Efficient Transformers

A naive review

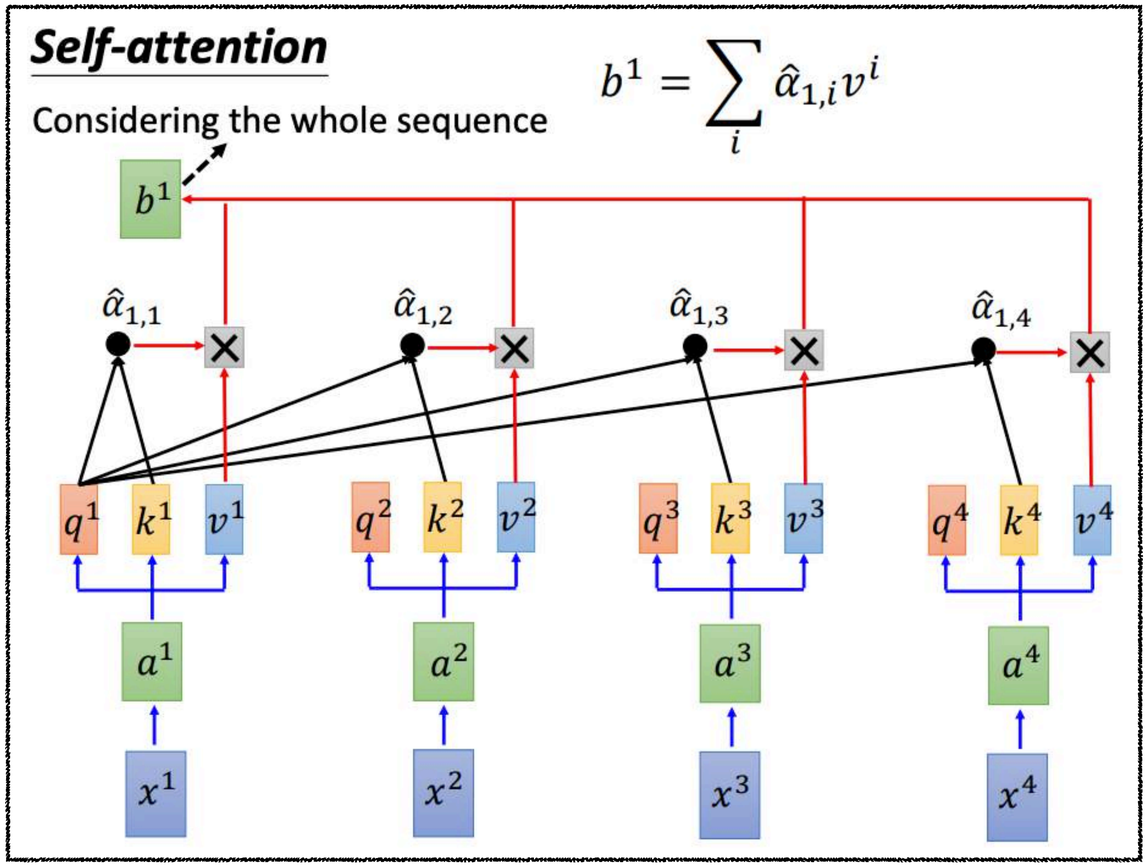
Xiachong Feng 2020.10.31. Thanks Zekun Wang for discussing papers~👍

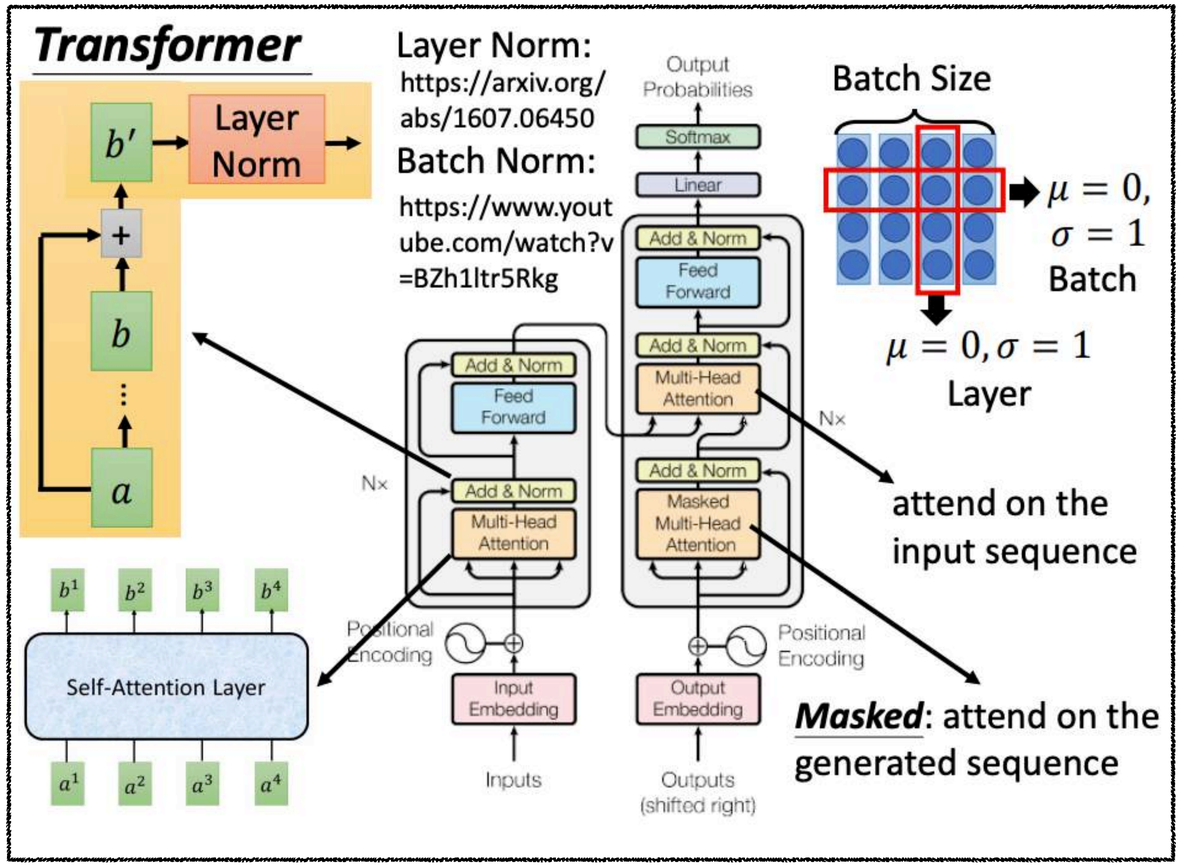
Content

- Transformer Basic
- Papers
  - Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context
  - Star-Transformer
  - BP-Transformer: Modelling Long-Range Context via Binary Partitioning
  - Reformer: The Efficient Transformer
  - Longformer: The Long-Document Transformer
  - Big Bird: Transformers for Longer Sequences
- Conclusion

Goal: Know the core idea of each model~

Transformer





http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2019/Lecture/Transformer%20(v5).pdf

Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, Ruslan Salakhutdinov

Carnegie Mellon University, Google Brain

ACL19

How to Use Vanilla Transformer Language Models for Long Document

- Split the entire corpus into shorter segments
- Problem:
  - Information never flows across segments
  - Computation inefficient

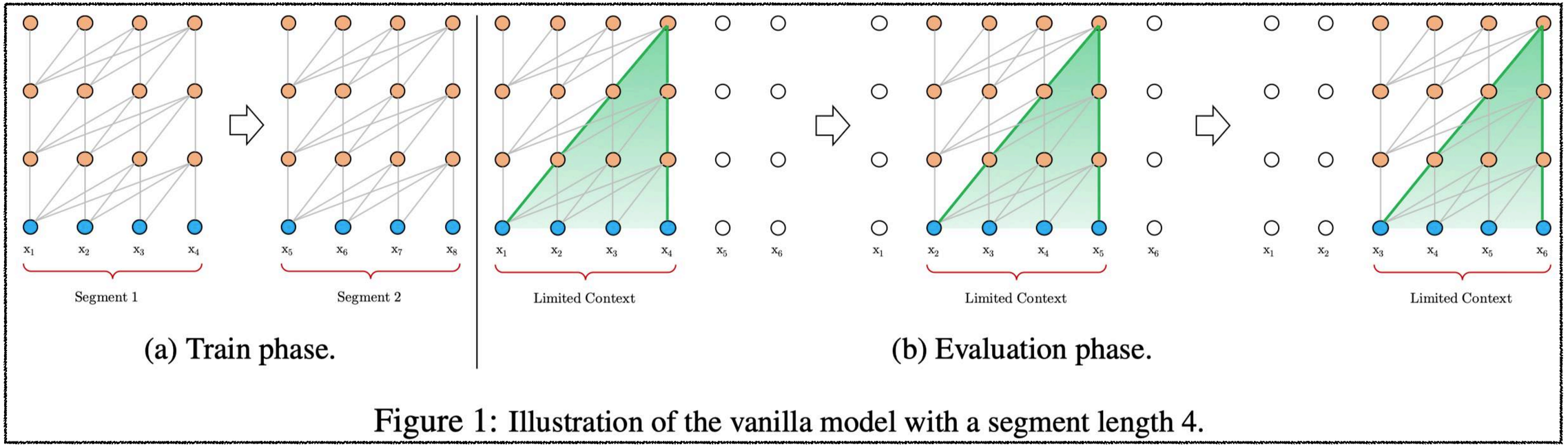
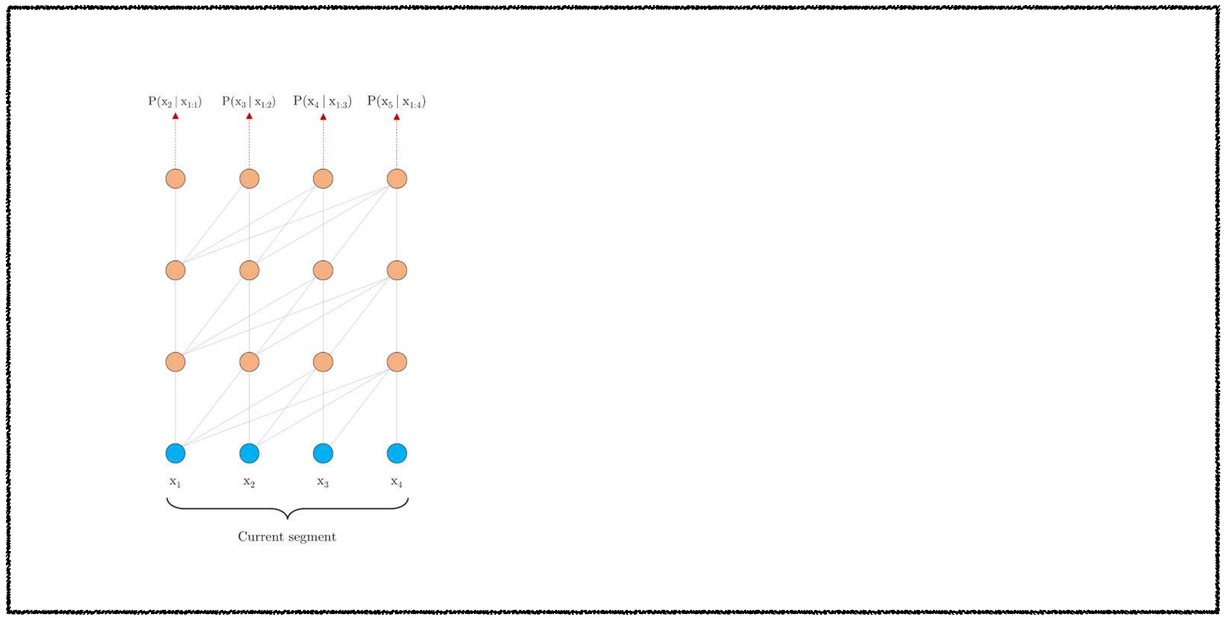


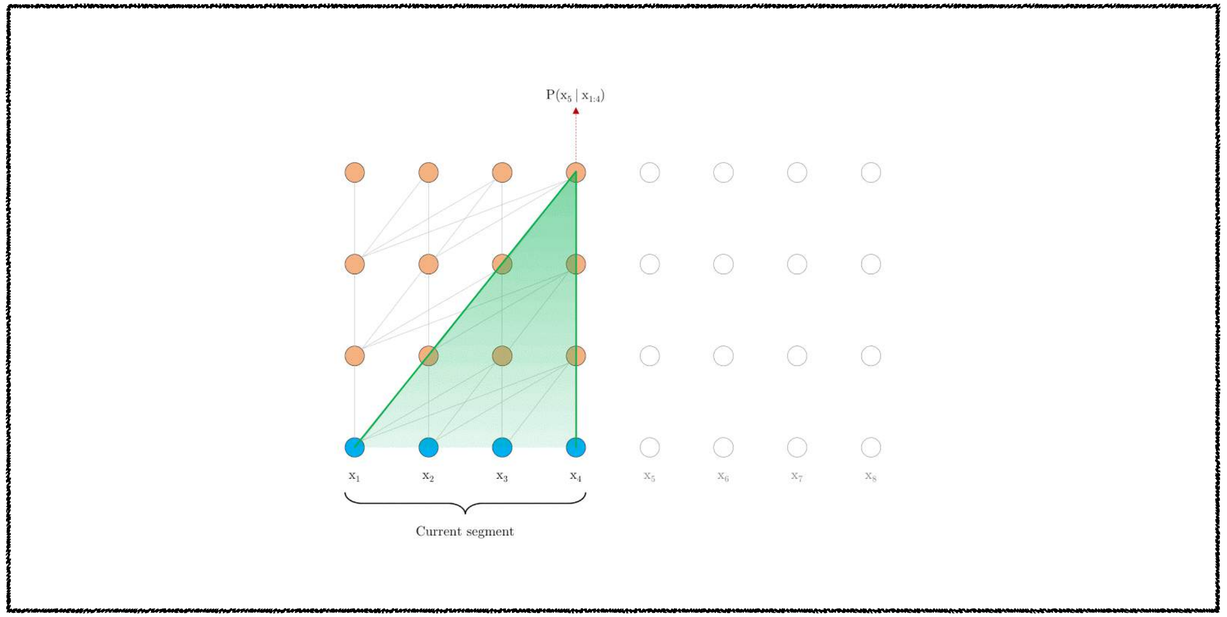
Figure 1: Illustration of the vanilla model with a segment length 4.

How to Use Vanilla Transformer Language Models for Long Document

- Split the entire corpus into shorter segments
- Problem: Information never flows across segments Computation inefficient



Vanilla Transformer with a fixed-length context at training time

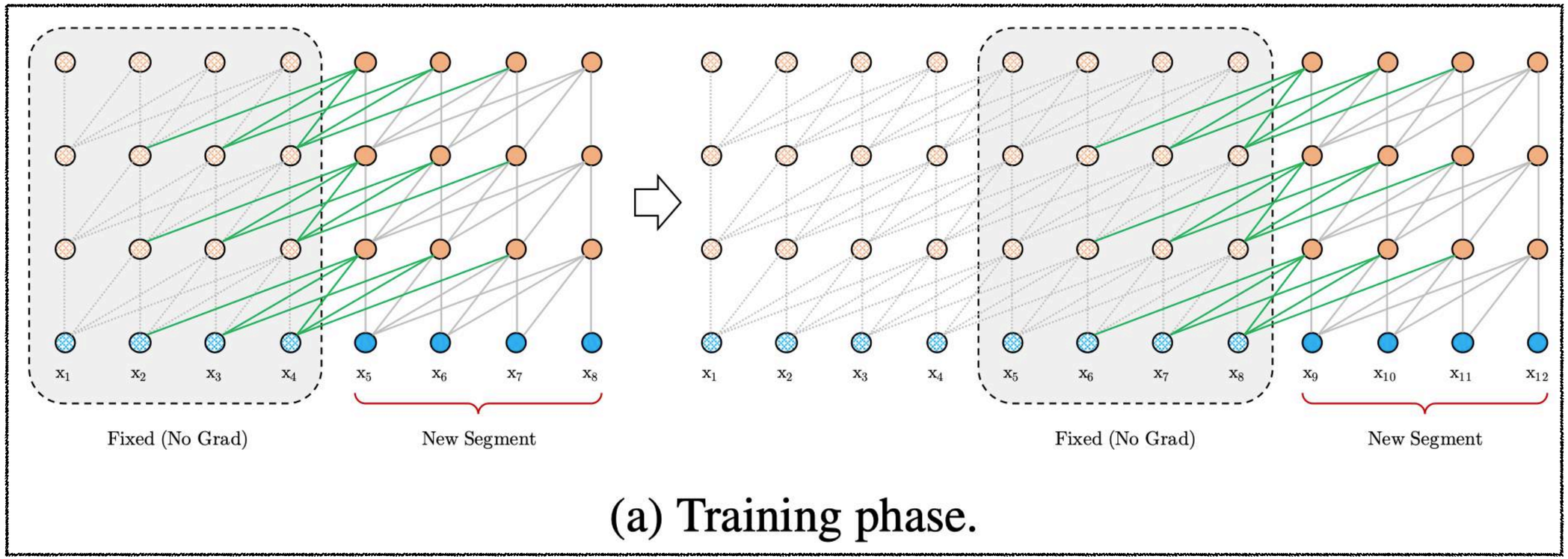


Vanilla Transformer with a fixed-length context at evaluation time

https://ai.googleblog.com/2019/01/transformer-xl-unleashing-potential-of.html

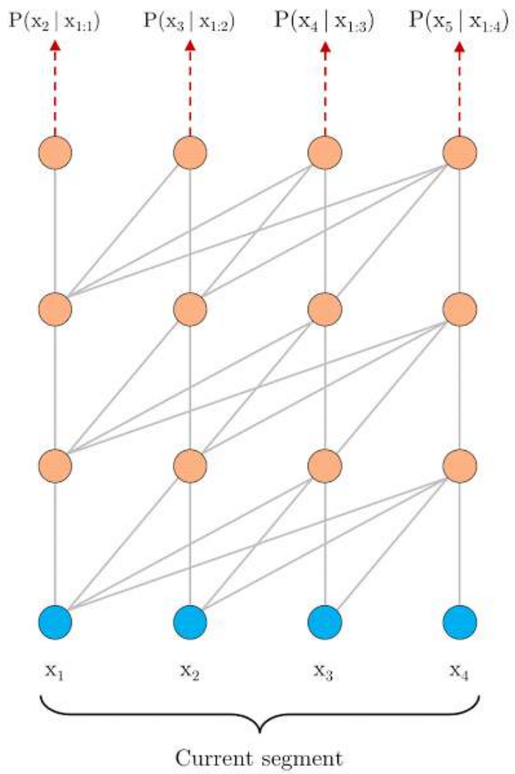
Segment-Level Recurrence with State Reuse

- During training, the hidden state sequence computed for the previous segment is fixed and cached to be reused as an extended context when the model processes the next new segment



Transformer-XL model with a segment length 4

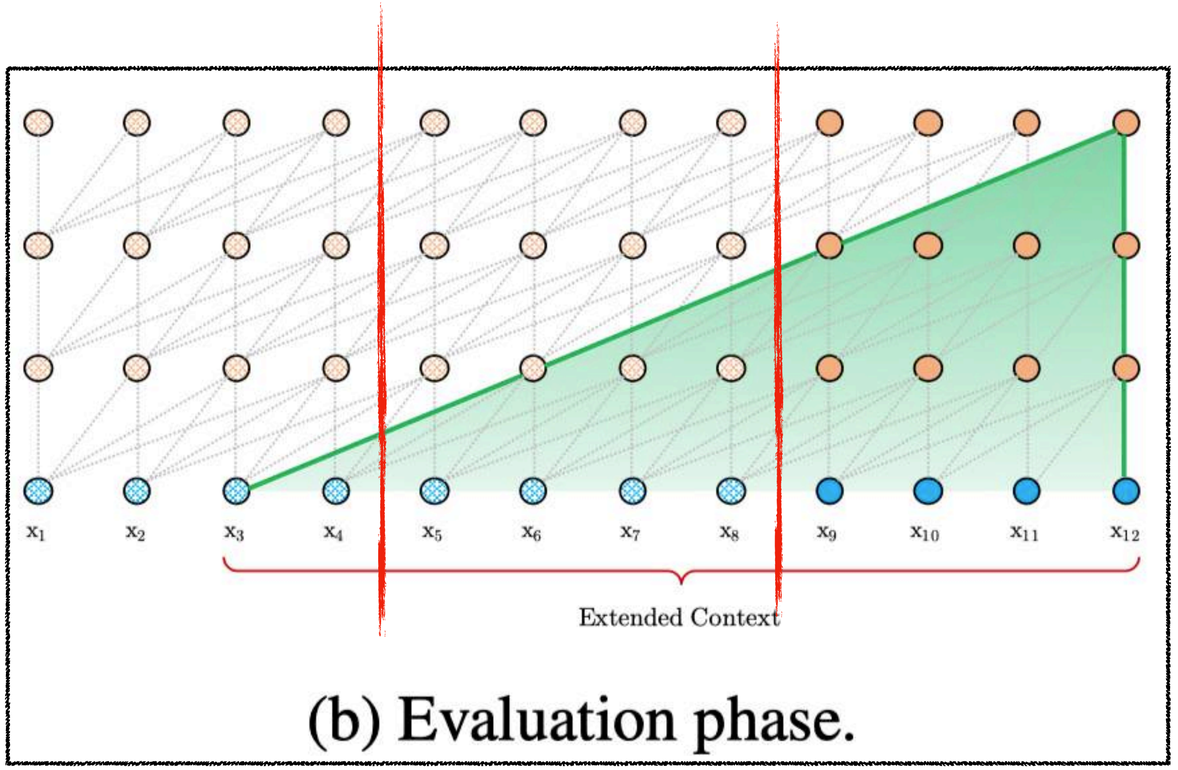
Segment-Level Recurrence with State Reuse



https://ai.googleblog.com/2019/01/transformer-xl-unleashing-potential-of.html

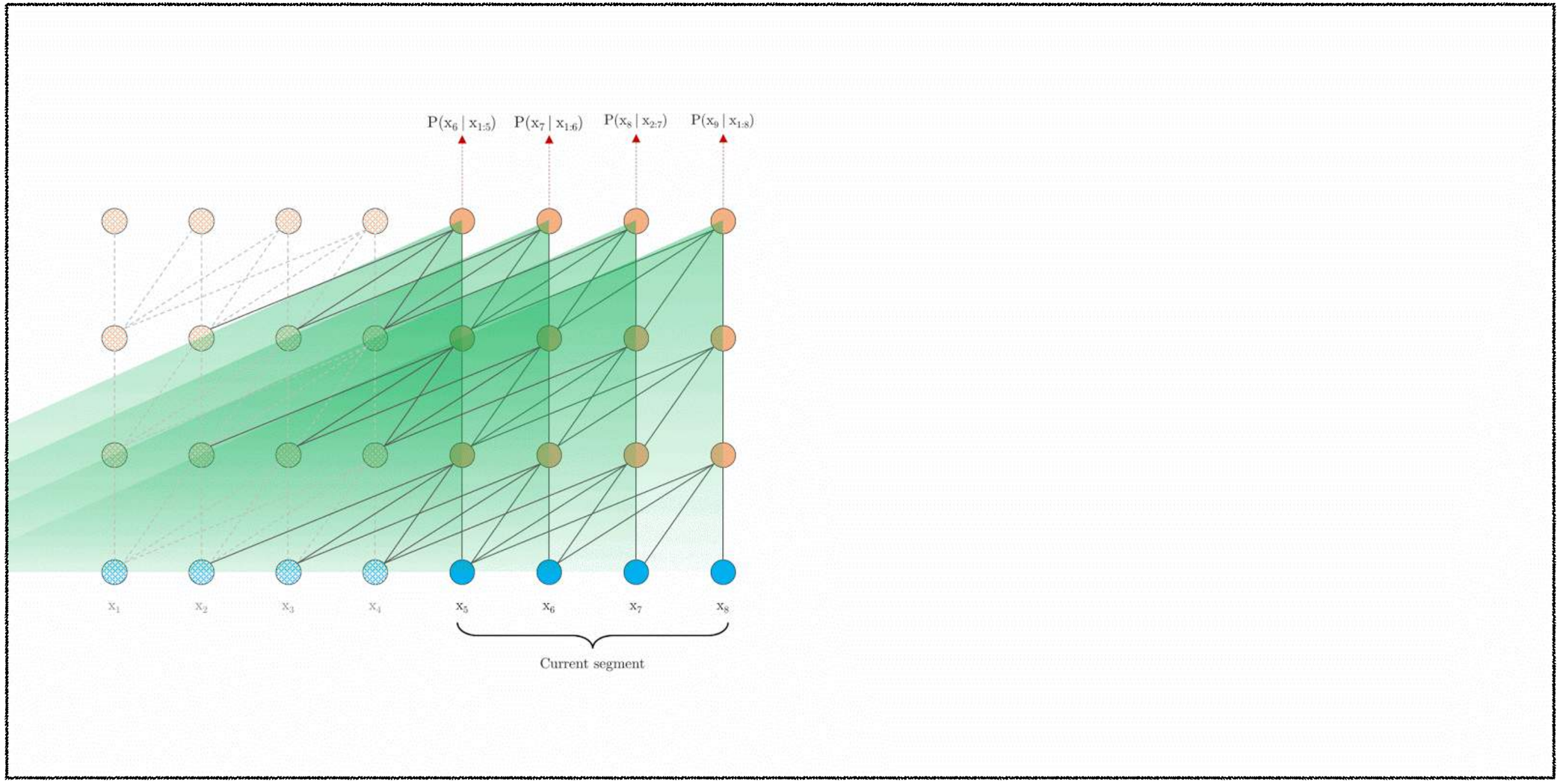
Segment-Level Recurrence with State Reuse

- the largest possible dependency length grows linearly w.r.t. the number of layers as well as the segment length : $O(N \times L)$



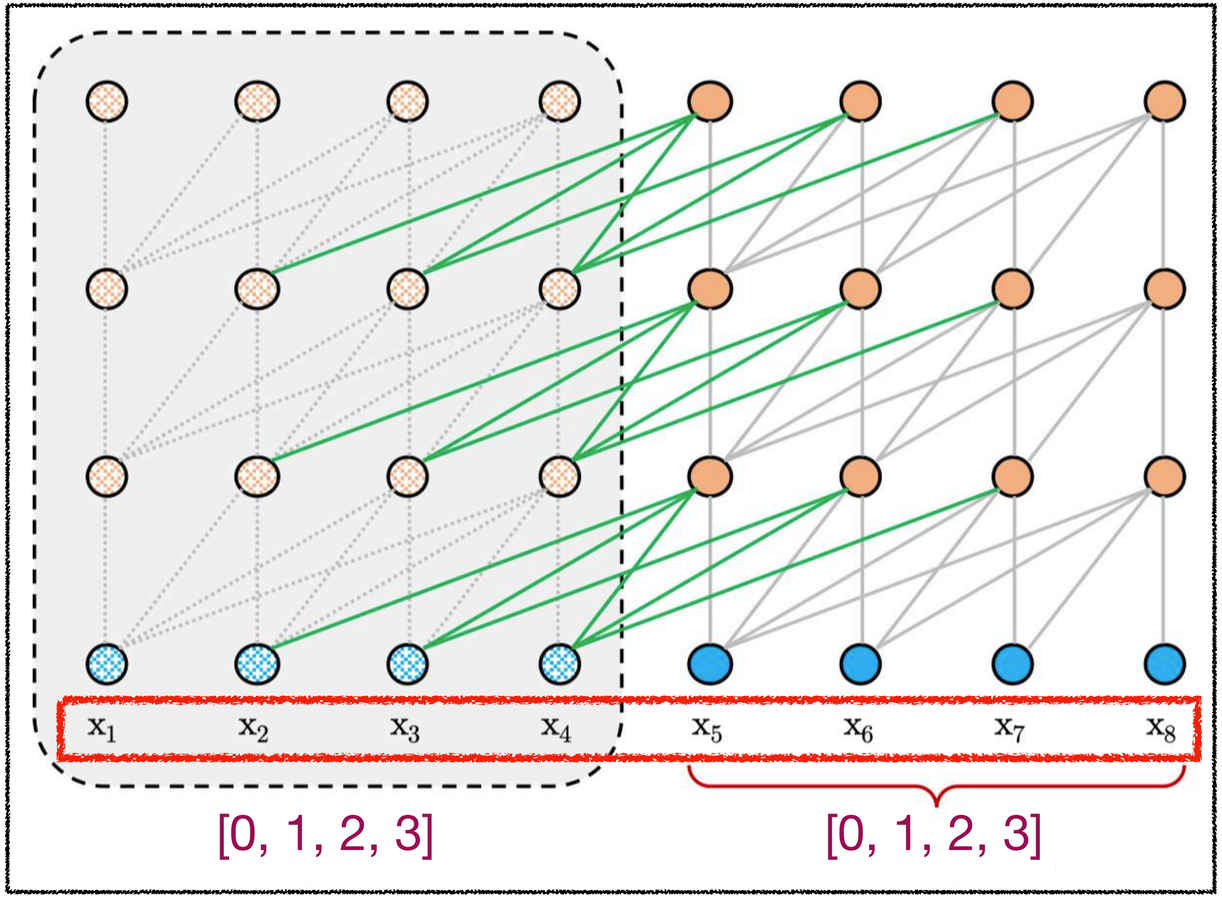
(b) Evaluation phase.

Segment-Level Recurrence with State Reuse



https://ai.googleblog.com/2019/01/transformer-xl-unleashing-potential-of.html

Positional Encodings?



Relative Positional Encodings

query q_i → key vector k_j

$$\mathbf{A}^{\text{abs}}_{i,j} = \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(a)} + \underbrace{\mathbf{E}_{x_i}^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(b)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{E}_{x_j}}_{(c)} + \underbrace{\mathbf{U}_i^\top \mathbf{W}_q^\top \mathbf{W}_k \mathbf{U}_j}_{(d)} .$$

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

1. R is a sinusoid encoding matrix (Vaswani et al., 2017) without learnable parameters.
2. Introduce a trainable parameter u and v to replace the query $U_i^T W_q^T$
3. Separate the two weight matrices $W_{k,E}$ and $W_{k,R}$

Experiments

- word-level and character-level language modeling

Model	#Param	PPL
Grave et al. (2016b) - LSTM	-	48.7
Bai et al. (2018) - TCN	-	45.2
Dauphin et al. (2016) - GCNN-8	-	44.9
Grave et al. (2016b) - LSTM + Neural cache	-	40.8
Dauphin et al. (2016) - GCNN-14	-	37.2
Merity et al. (2018) - QRNN	151M	33.0
Rae et al. (2018) - Hebbian + Cache	-	29.9
Ours - Transformer-XL Standard	151M	**24.0**
Baevski and Auli (2018) - Adaptive Input$^\diamond$	247M	20.5
Ours - Transformer-XL Large	257M	**18.3**

Table 1: Comparison with state-of-the-art results on WikiText-103. $^\diamond$ indicates contemporary work.

Model	#Param	bpc
Ha et al. (2016) - LN HyperNetworks	27M	1.34
Chung et al. (2016) - LN HM-LSTM	35M	1.32
Zilly et al. (2016) - RHN	46M	1.27
Mujika et al. (2017) - FS-LSTM-4	47M	1.25
Krause et al. (2016) - Large mLSTM	46M	1.24
Knol (2017) - cmix v13	-	1.23
Al-Rfou et al. (2018) - 12L Transformer	44M	1.11
Ours - 12L Transformer-XL	41M	**1.06**
Al-Rfou et al. (2018) - 64L Transformer	235M	1.06
Ours - 18L Transformer-XL	88M	1.03
Ours - 24L Transformer-XL	277M	**0.99**

Table 2: Comparison with state-of-the-art results on enwik8.

Model	#Param	bpc
Cooijmans et al. (2016) - BN-LSTM	-	1.36
Chung et al. (2016) - LN HM-LSTM	35M	1.29
Zilly et al. (2016) - RHN	45M	1.27
Krause et al. (2016) - Large mLSTM	45M	1.27
Al-Rfou et al. (2018) - 12L Transformer	44M	1.18
Al-Rfou et al. (2018) - 64L Transformer	235M	1.13
Ours - 24L Transformer-XL	277M	**1.08**

Table 3: Comparison with state-of-the-art results on text8.

Model	#Param	PPL
Shazeer et al. (2014) - Sparse Non-Negative	33B	52.9
Chelba et al. (2013) - RNN-1024 + 9 Gram	20B	51.3
Kuchaiev and Ginsburg (2017) - G-LSTM-2	-	36.0
Dauphin et al. (2016) - GCNN-14 bottleneck	-	31.9
Jozefowicz et al. (2016) - LSTM	1.8B	30.6
Jozefowicz et al. (2016) - LSTM + CNN Input	1.04B	30.0
Shazeer et al. (2017) - Low-Budget MoE	~5B	34.1
Shazeer et al. (2017) - High-Budget MoE	~5B	28.0
Shazeer et al. (2018) - Mesh Tensorflow	4.9B	24.0
Baevski and Auli (2018) - Adaptive Input$^\diamond$	0.46B	24.1
Baevski and Auli (2018) - Adaptive Input$^\diamond$	1.0B	23.7
Ours - Transformer-XL Base	0.46B	23.5
Ours - Transformer-XL Large	0.8B	**21.8**

Table 4: Comparison with state-of-the-art results on One Billion Word. $^\diamond$ indicates contemporary work.

Star-Transformer

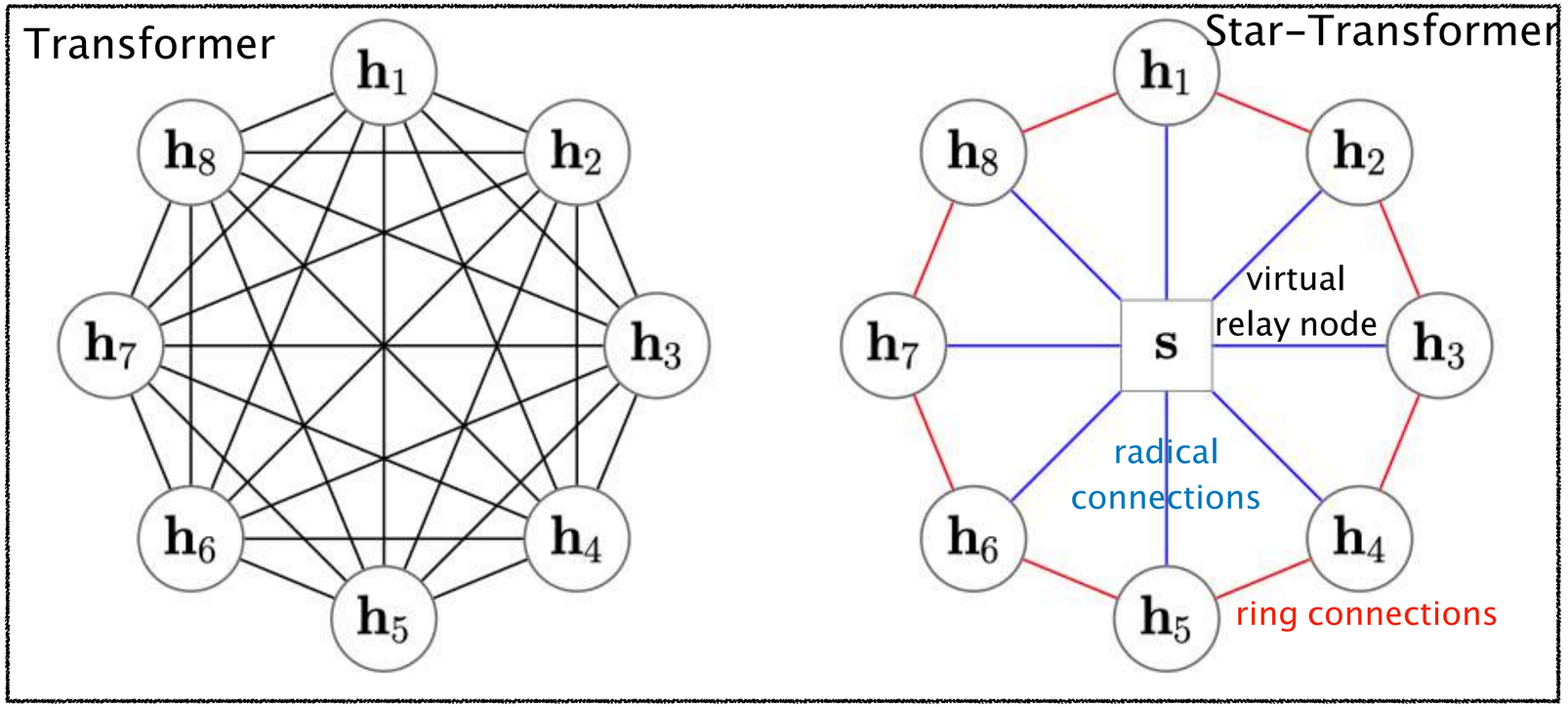
Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, Zheng Zhang

Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
School of Computer Science, Fudan University
New York University

NAACL19

Motivation

- The **local compositionality** is already a robust inductive bias for modeling the text sequence.



The Update of Star-Transformer

Algorithm 1 The Update of Star-Transformer

// *Initialization*

$\mathbf{h}_1^0, \cdots, \mathbf{h}_n^0 \leftarrow \mathbf{e}_1, \cdots, \mathbf{e}_n$

$\mathbf{s}^0 \leftarrow average(\mathbf{e}_1, \cdots, \mathbf{e}_n)$

for t **from** 1 **to** T **do**

  // *update the satellite nodes*

  for i **from** 1 **to** n **do**

    $\mathbf{C}_i^t = [\mathbf{h}_{i-1}^{t-1}; \mathbf{h}_i^{t-1}; \mathbf{h}_{i+1}^{t-1}; \mathbf{e}^i; \mathbf{s}^{t-1}]$

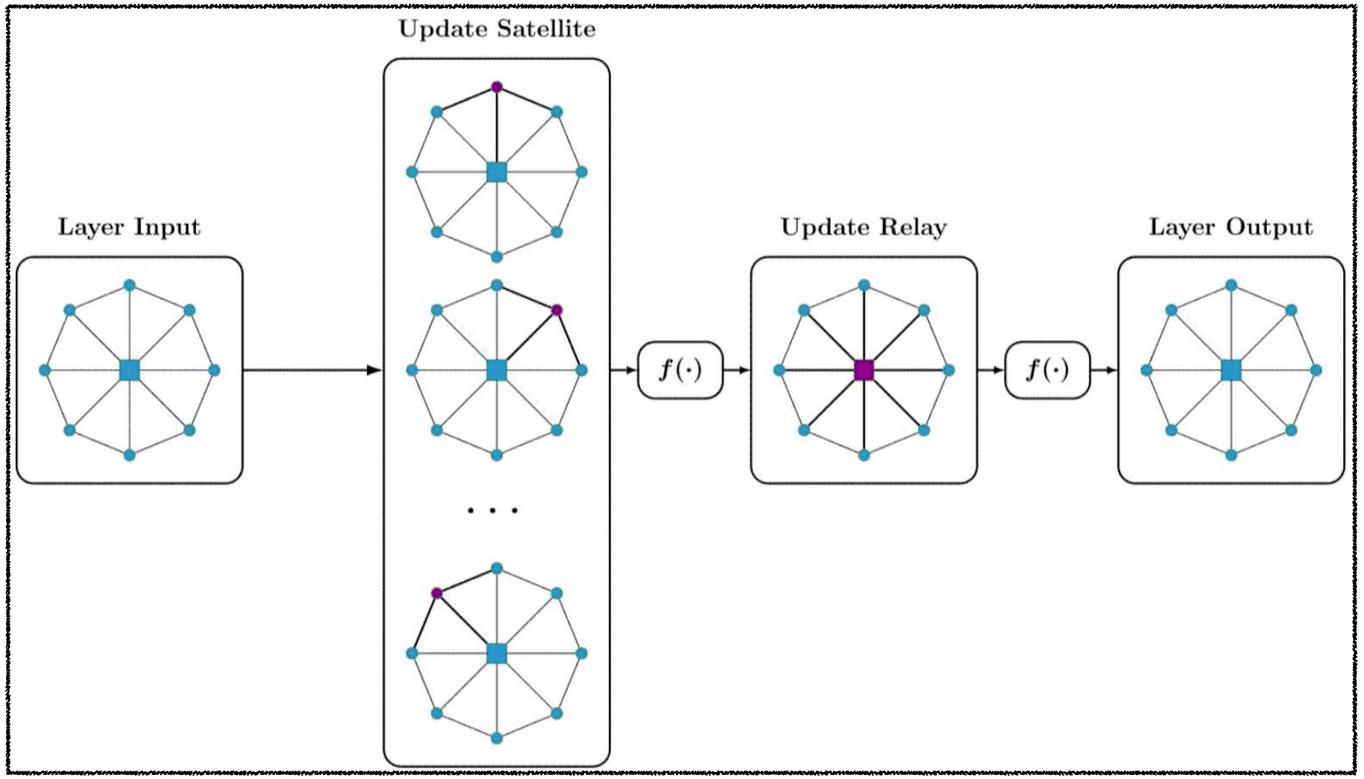
    $\mathbf{h}_i^t = \mathrm{MultiAtt}(\mathbf{h}_i^{t-1}, \mathbf{C}_i^t)$

    $\mathbf{h}_i^t = \mathrm{LayerNorm}(\mathrm{ReLU}(\mathbf{h}_i^t))$

  // *update the relay node*

  $\mathbf{s}^t = \mathrm{MultiAtt}(\mathbf{s}^{t-1}, [\mathbf{s}^{t-1}; \mathbf{H}^t])$

  $\mathbf{s}^t = \mathrm{LayerNorm}(\mathrm{ReLU}(\mathbf{s}^t))$



https://xpqiu.github.io/slides/20200613-CAAI-自然语言处理中的自注意力模型.pdf

Experiments

	Task	Dataset	Metric	Transformer	Star-Transformer
1	Text Classification	MTL 16	Acc	82.78	**86.98**
2	NLI	SNLI	Acc	82.2	**86.0**
3	NER	CoNLL2003	F1	86.48	**90.93**
4	NER	CoNLL2012	F1	83.57	**86.30**
5	POS	PTB	Acc	96.31	**97.14**

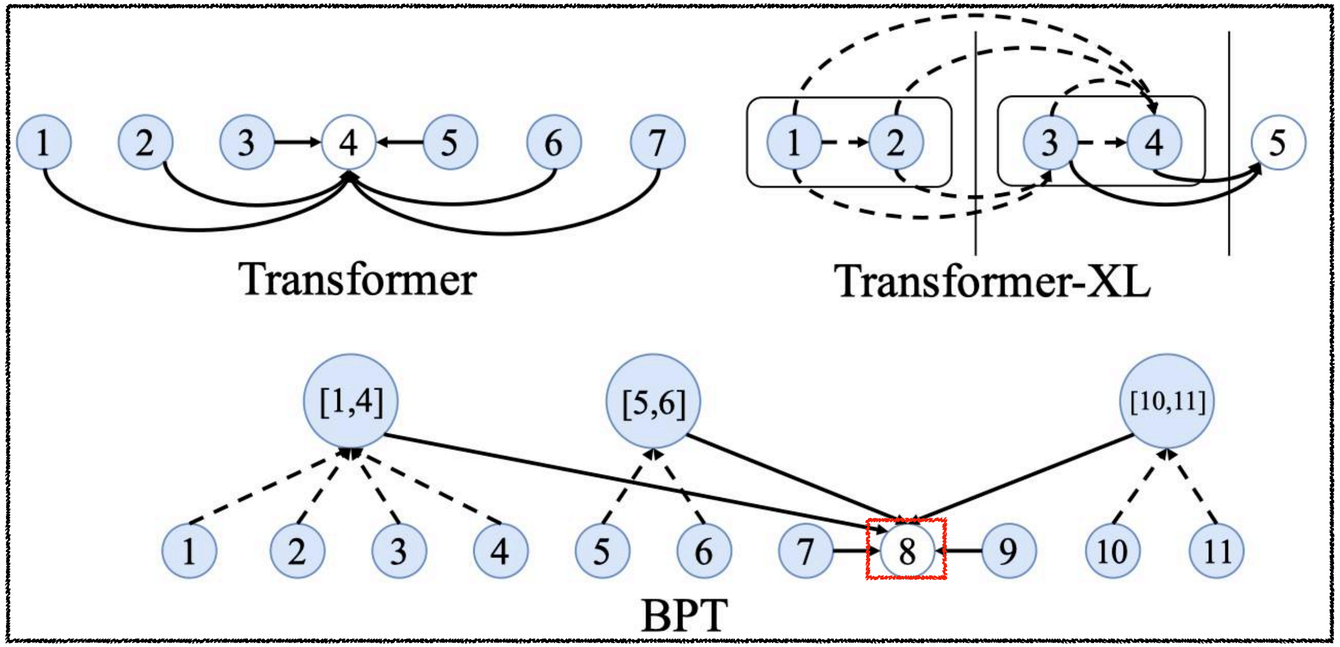
BP-Transformer: Modelling Long-Range Context via Binary Partitioning

Zihao Ye, Qipeng Guo, Quan Gan, Xipeng Qiu, Zheng Zhang

AWS Shanghai AI Lab
Fudan University
New York University Shanghai

Motivation (inductive bias)

- Attending the context information from fine-grain to coarse-grain as the relative distance increases.
- The farther the context information is, the coarser its representation is.
- A token node can attend the smaller-scale span for the closer context and the larger-scale span for the longer-distance context.



Binary partitioning

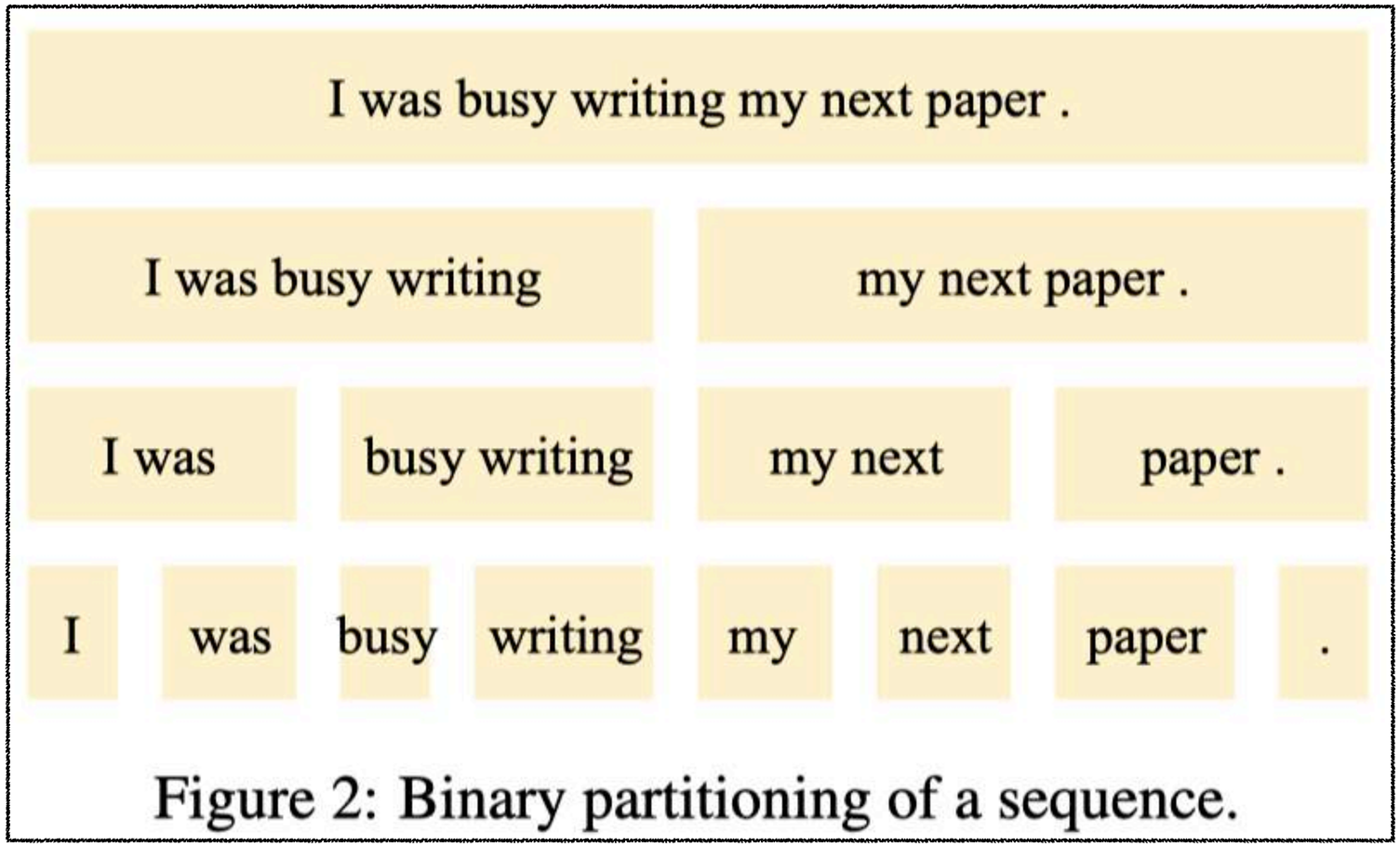


Figure 2: Binary partitioning of a sequence.

Span nodes

Token nodes

binary tree

Edge Construction

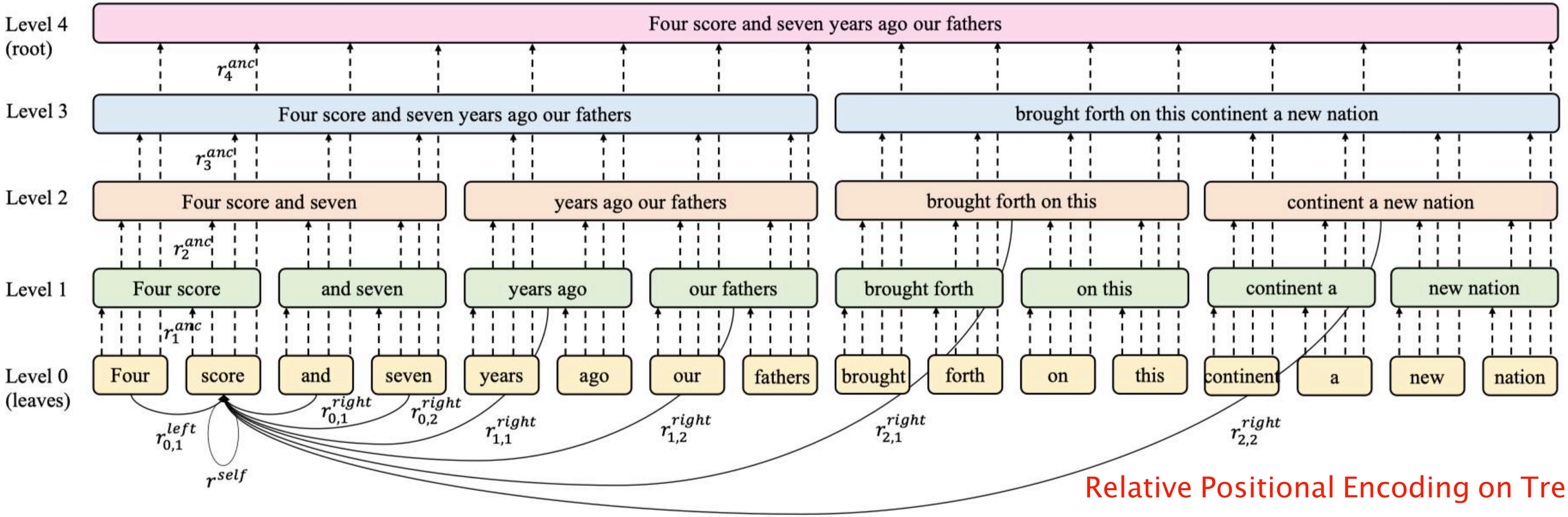


Figure 3: The figure illustrates how to build the graph: nodes at different levels are colored differently, dashed lines are edges connects token nodes to span nodes; solid lines are edges connect to token nodes. The r_*^* are relative positions assigned to edges.

The update of graph

Algorithm 2 The update of graph

Require: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ the underlying graph, N the number of layers, $\mathbf{H}^0$ initial hidden states

1: **for** $i := 1$ to N **do:** Graph Self Attention

2: $\quad \mathbf{Z}^i \leftarrow \text{norm}\left(\mathbf{H}^{i-1} + \text{GSA}^{(i)}\left(\mathcal{G}, \mathbf{H}^{i-1}\right)\right)$

3: $\quad \mathbf{H}^i \leftarrow \text{norm}\left(\mathbf{Z}^i + \text{FFN}^{(i)}\left(\mathbf{Z}^i\right)\right)$

4: **end for**

5: **return** $\mathbf{H}^N$

Experiments

Model	SST-5	IMDB
BPT	52.71(0.32)	**92.12(0.11)**
Star Transformer	52.9	90.50
Transformer	50.4	89.24
Bi-LSTM (Li et al., 2015)	49.8	-
Tree-LSTM (Socher et al., 2013)	51.0	-
QRNN (Bradbury et al., 2017)	-	91.4
BCN+Char+CoVe (McCann et al., 2017)	53.7	91.8

Table 1: Test accuracy on SST-5 and IMDB. In BPT, $k = 2$ and $k = 4$ for SST and IMDB respectively. The last model used word embeddings pretrained with translation and additional character-level embeddings.

Classification

Model	Enwiki8	Text8	Params
HM-LSTM (Chung et al., 2017)	-	1.29	35M
Recurrent Highway (Zilly et al., 2017)	-	1.27	45M
mLSTM (Krause et al., 2016)	1.24	1.27	45M
Transformer (Al-Rfou et al., 2018)	1.11	1.18	44M
Transformer-XL (Dai et al., 2019)	1.06	-	41M
Adaptive Span (Sukhbaatar et al., 2019)	1.02	1.11	39M
BPT ($k = 64, l = 8192$)	**1.02**	**1.11**	**38M**

Table 3: Test BPC on Enwiki8/Text8. Note that Transformer-XL can be only used for language modeling. l denotes the context length.

Language Modeling

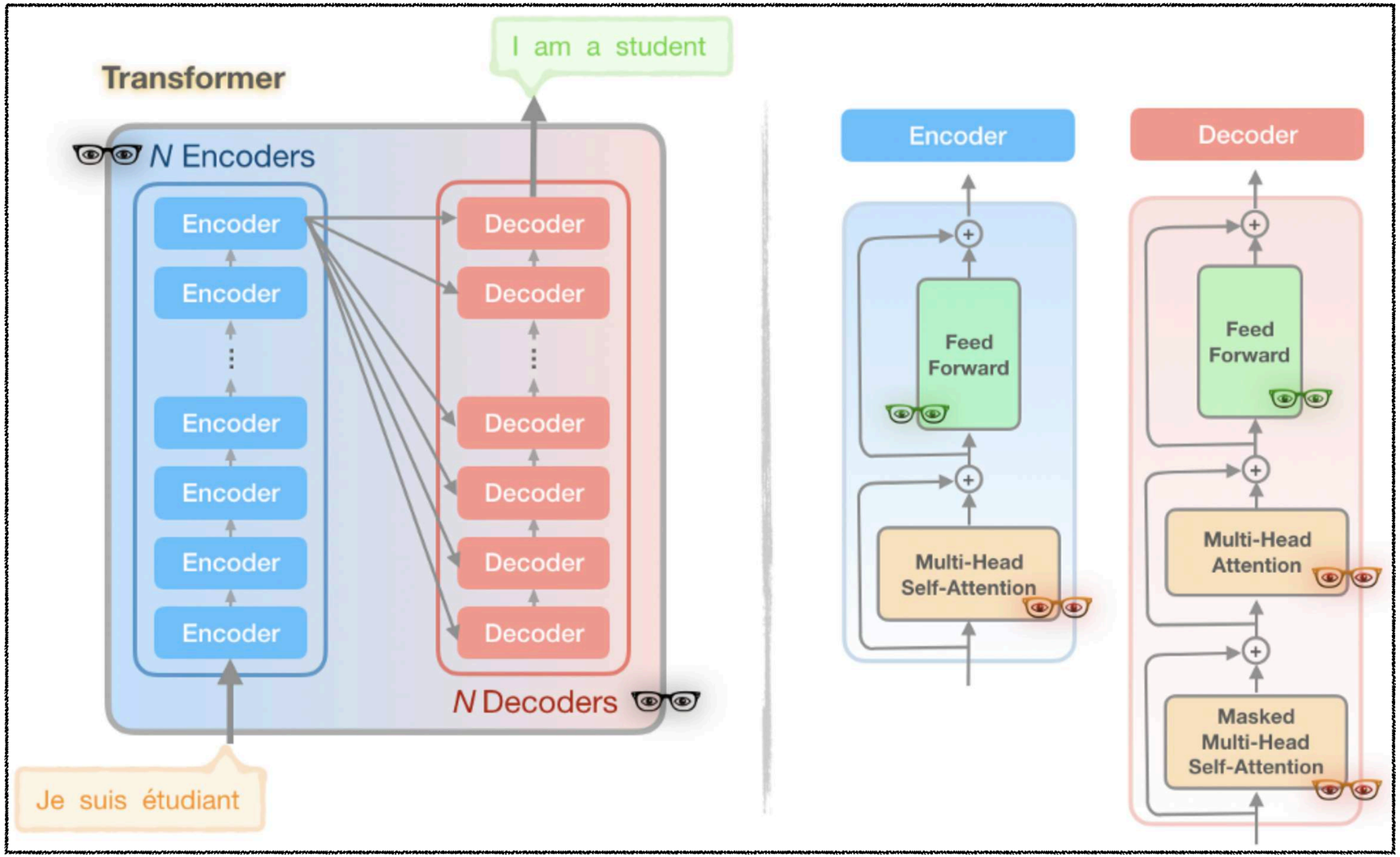
Reformer: The Efficient Transformer

Nikita Kitaev, Łukasz Kaiser, Anselm Levskaya

U.C. Berkeley & Google Research

ICLR2020

Problems of Transformers



- Attention computation
- Large number of layers
- Depth of feed-forward layers

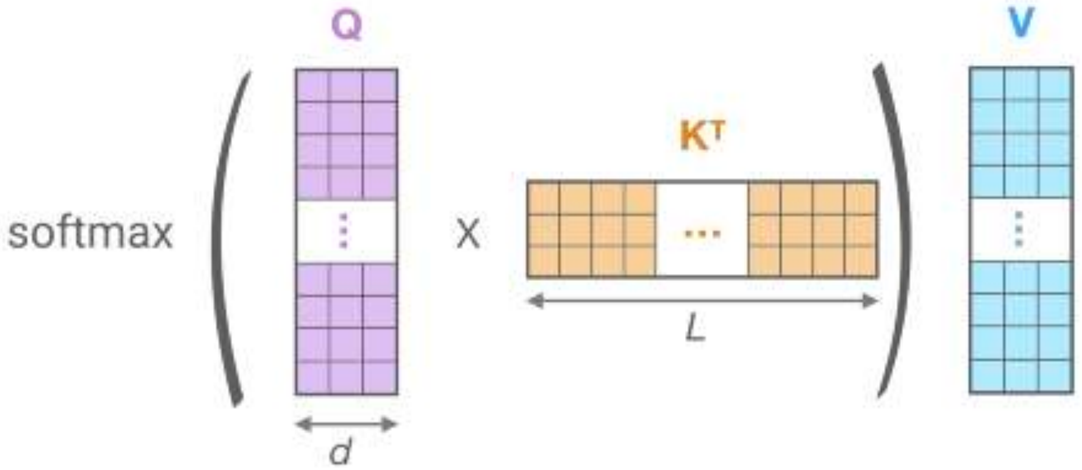
https://towardsdatascience.com/illustrating-the-reformer-393575ac6ba0

Problem1: Attention computation

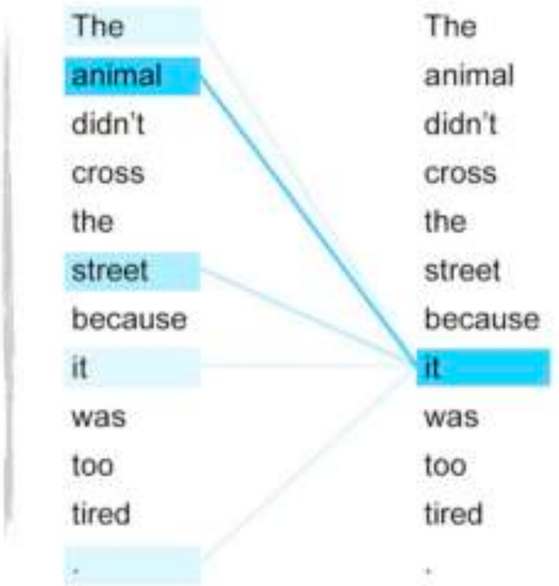
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Q: L **Q**ueries of size d, to attend for
K: L **K**eys of size d, to attend to
V: L **V**alues of size d

L: length of sequence
d: depth of attention

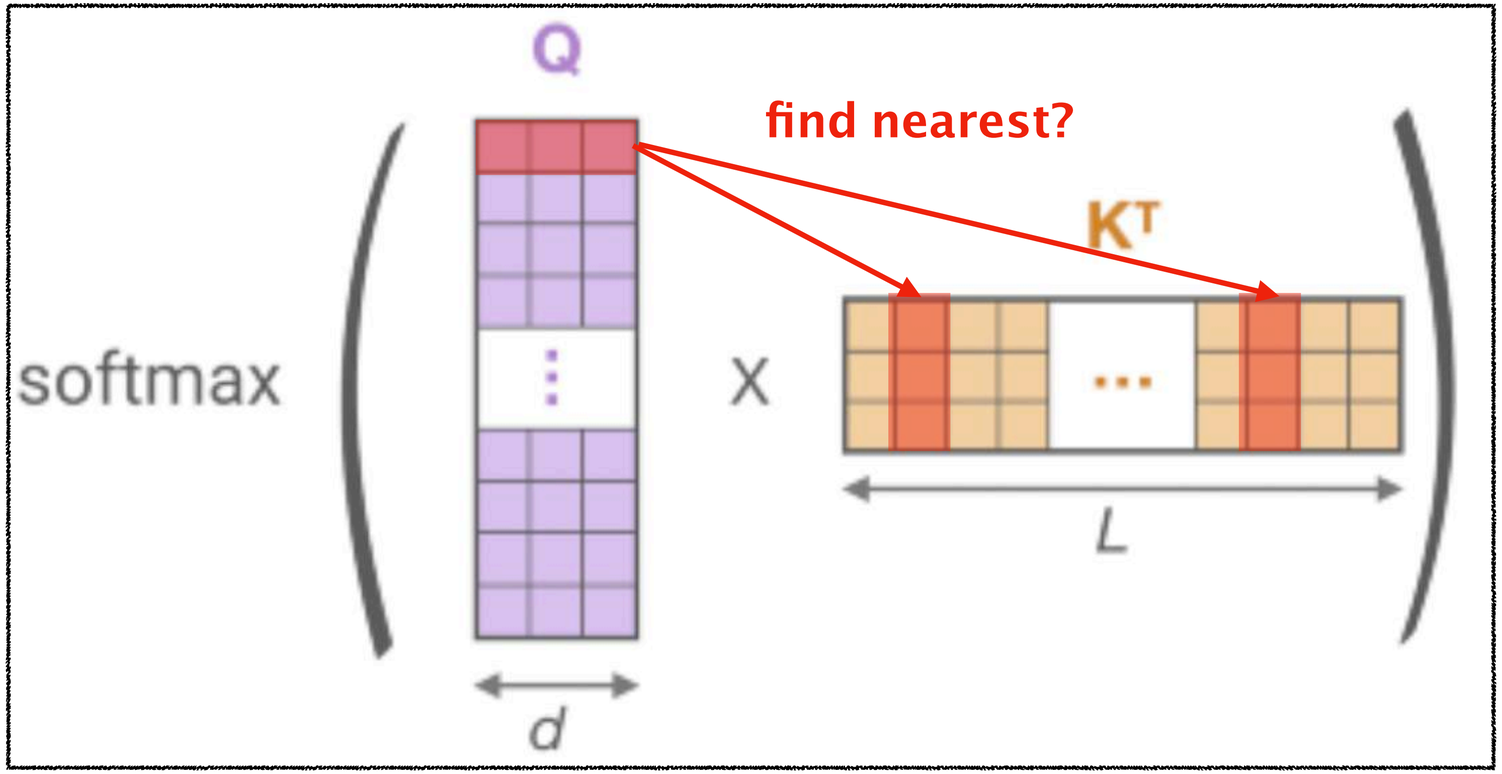


$[batch\_size, length, length]$



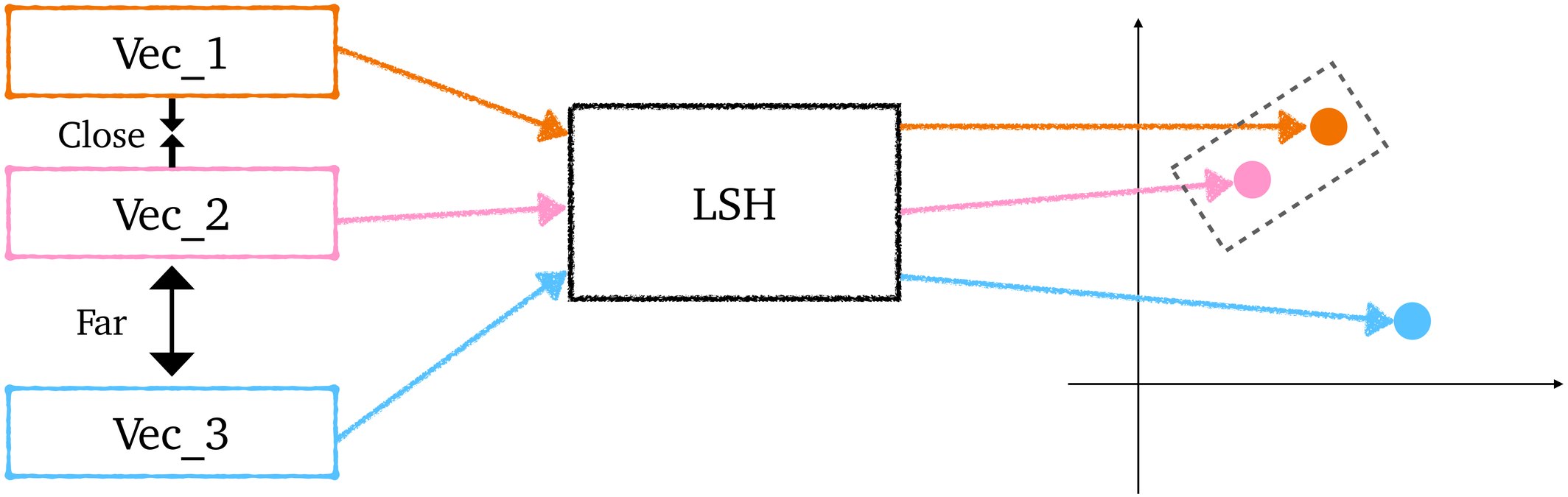
https://towardsdatascience.com/illustrating-the-reformer-393575ac6ba0

Solution1: find the nearest ones to attend?



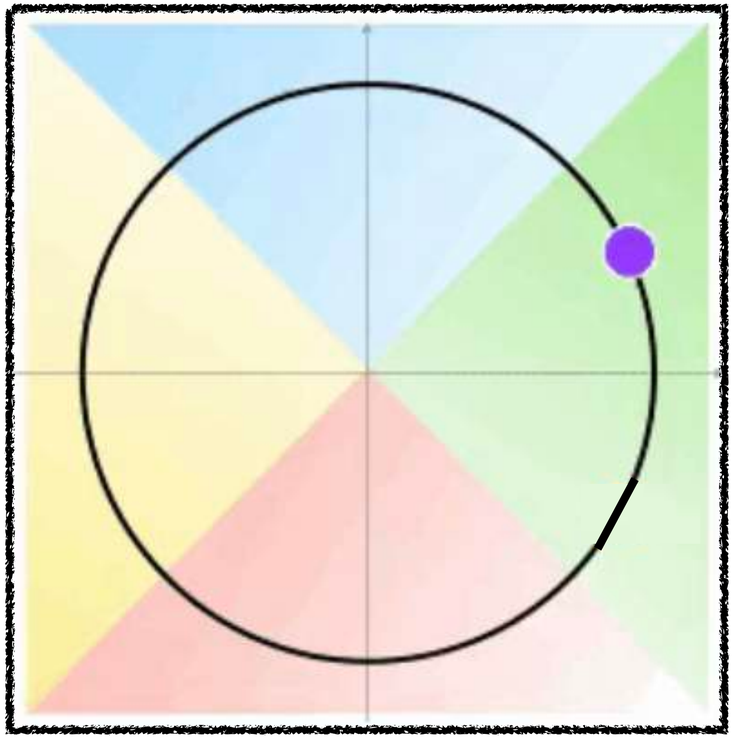
Solution1: Locality sensitive hashing (LSH) Attention

The main idea behind LSH is to **select *hash* functions** such that for two points '*p*' and '*q*', if '*q*' is close to '*p*' then with good enough probability we have '*hash(q) == hash(p)*'.



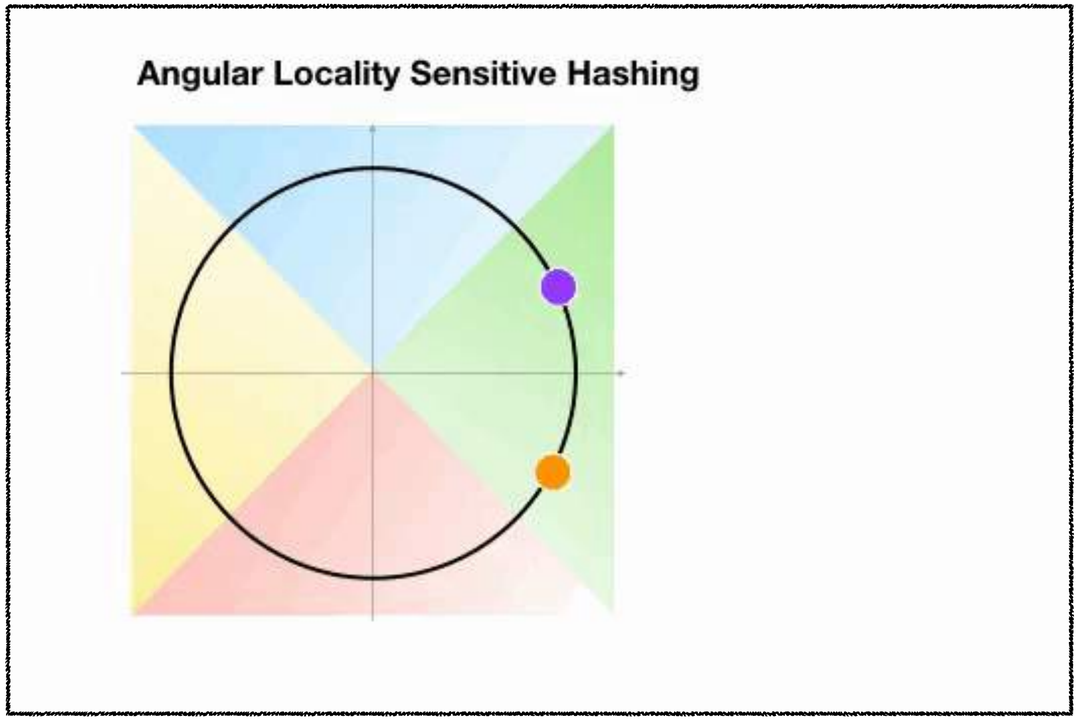
Solution1: Locality sensitive hashing (LSH) Attention

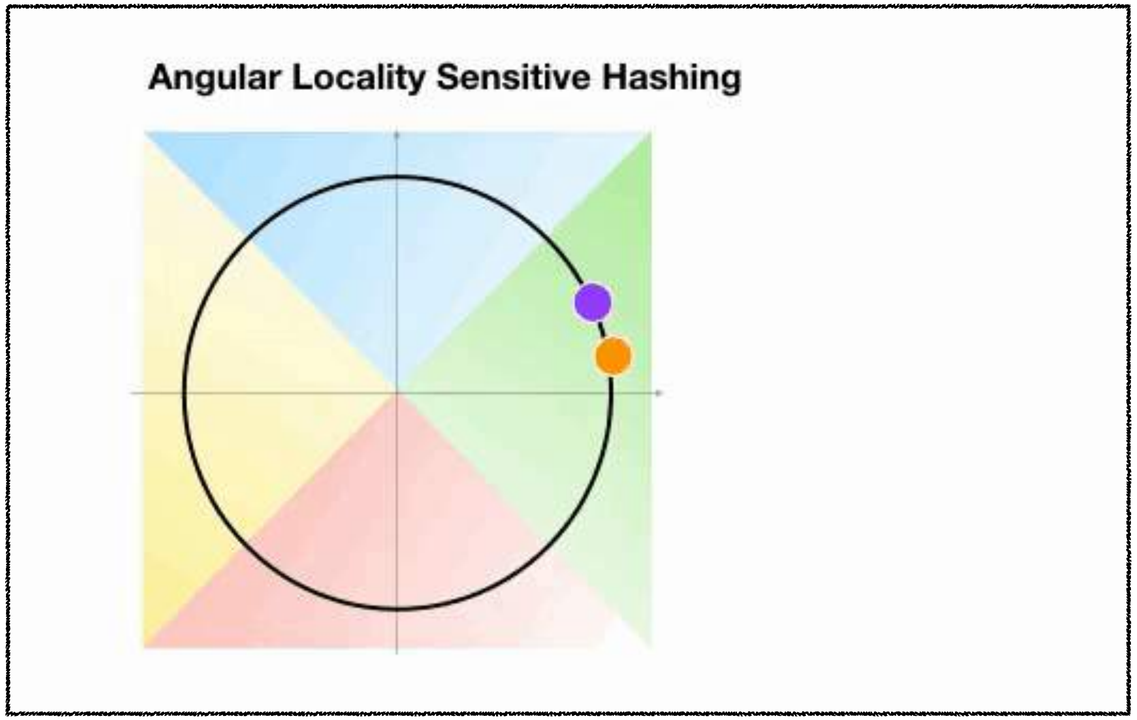
- projects the points on a unit sphere which has been divided into predefined regions each with a distinct code.



Angular LSH

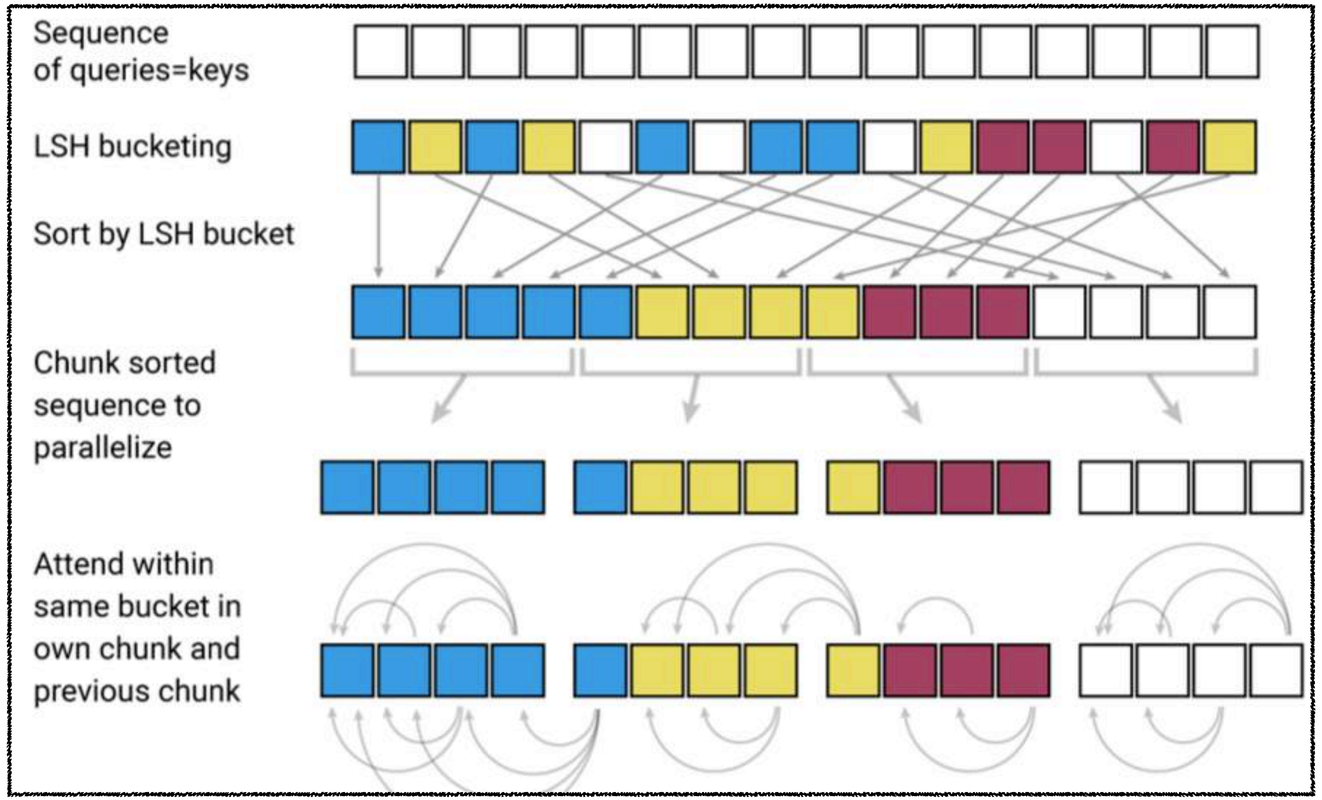
Solution1: Locality sensitive hashing (LSH) Attention

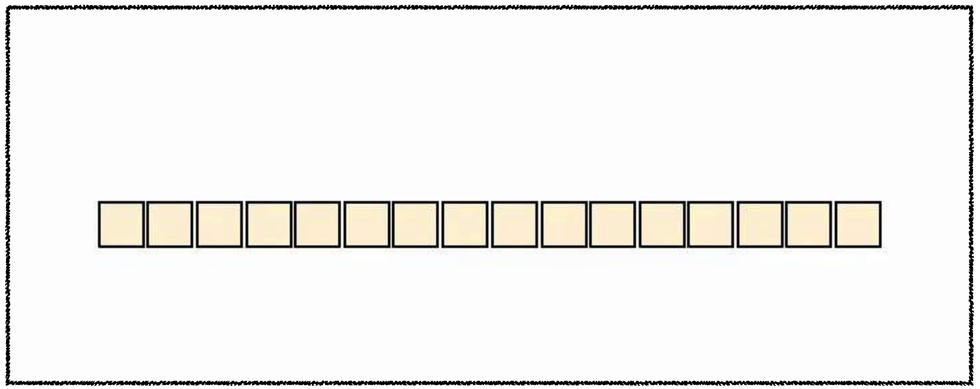




https://towardsdatascience.com/illustrating-the-reformer-393575ac6ba0

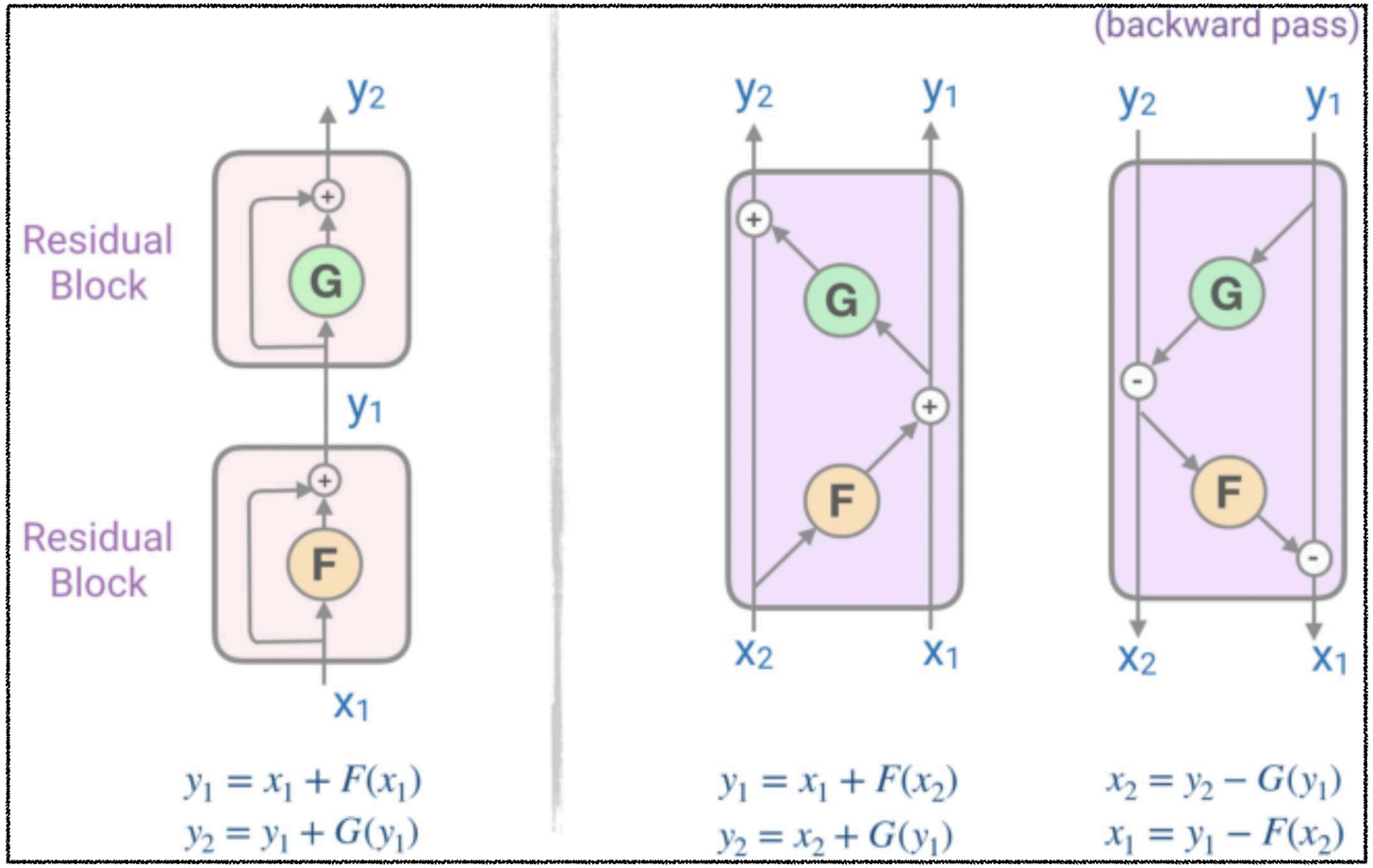
Solution1: Locality sensitive hashing (LSH) Attention





https://towardsdatascience.com/illustrating-the-reformer-393575ac6ba0

Solution2: Reversible residual Network



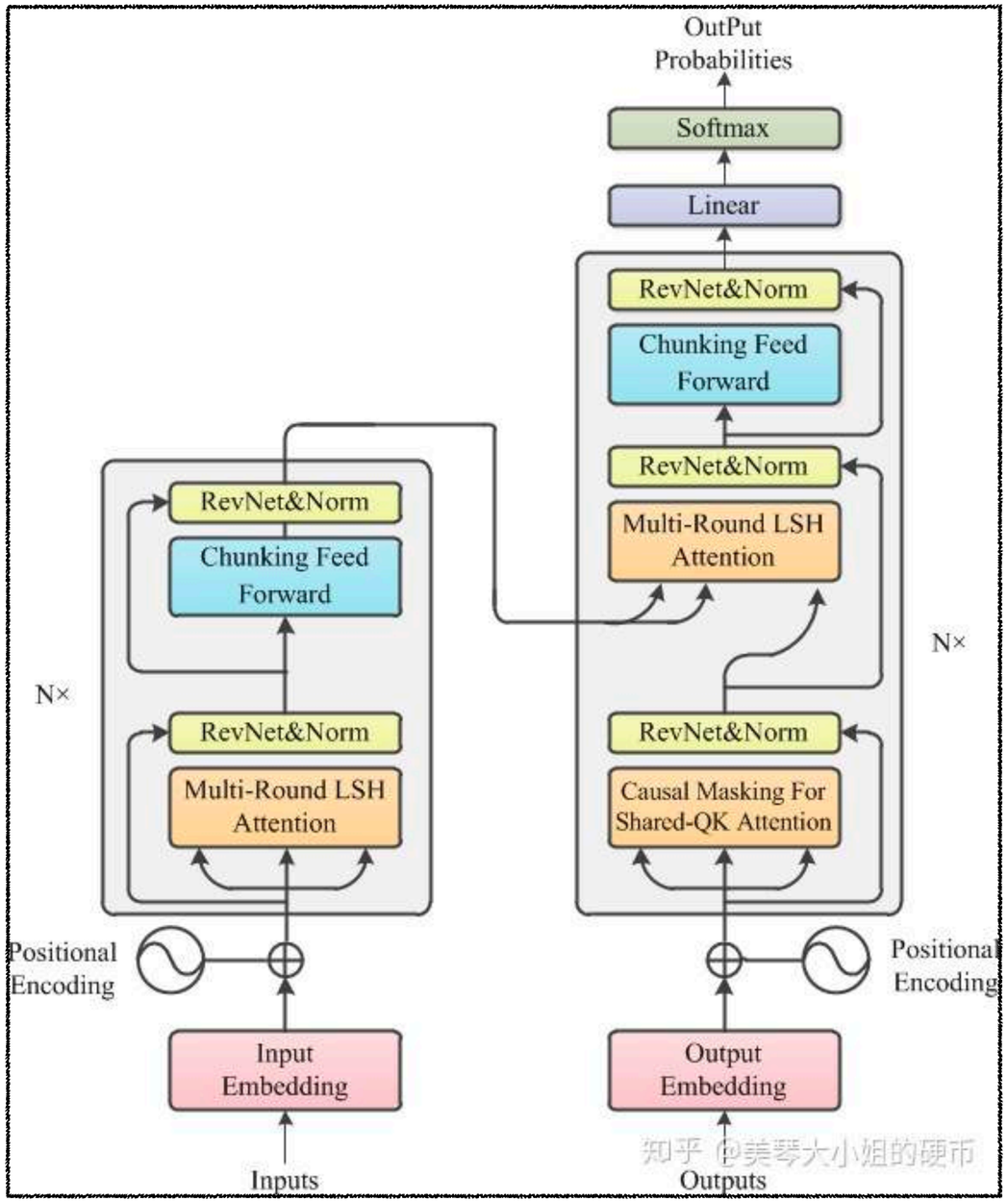
$$Y_1 = X_1 + Attention(X_2),$$
$$Y_2 = X_2 + FeedForward(Y_1)$$

https://towardsdatascience.com/illustrating-the-reformer-393575ac6ba0

Solution3: Chunking

$$
\begin{aligned}
y_2 &= x_2 + FFN(y_1) \\
&= [y_2^{(1)}; y_2^{(2)}; \ldots; y_2^{(c)}] \\
&= [x_2^{(1)} + FFN(y_1^{(1)}); x_2^{(2)} + FFN(y_1^{(2)}); \ldots; x_2^{(c)} + FFN(y_1^{(c)})]
\end{aligned}
$$

Reformer



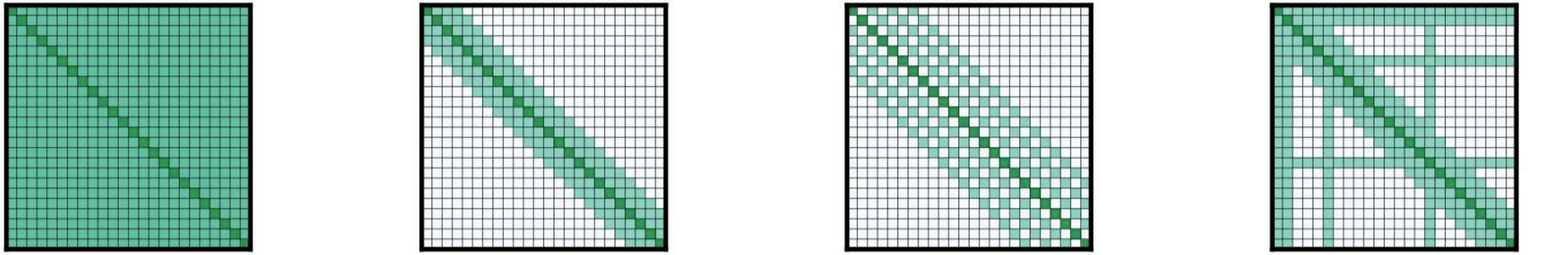
https://zhuanlan.zhihu.com/p/267095360

Longformer: The Long-Document Transformer

Iz Beltagy, Matthew E. Peters, Arman Cohan

Allen Institute for Artificial Intelligence, Seattle, WA, USA

Attention Pattern



(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

Figure 2: Comparing the full self-attention pattern and the configuration of attention patterns in our Longformer.

Pre-training

- RoBERTa uses learned absolute position embeddings with the maximum position being 512.
- To support longer documents, we add extra position embeddings to support up to position 4,096.
- To leverage RoBERTa's pretrained weights, instead of randomly initializing the new position embeddings, we initialize them by copying the 512 position embeddings from RoBERTa multiple times

Source	Tokens	Avg doc len
Books (Zhu et al., 2015)	0.5B	95.9K
English Wikipedia	2.1B	506
Realnews (Zellers et al., 2019)	1.8B	1.7K
Stories (Trinh and Le, 2018)	2.1B	7.8K

Table 5: Pretraining data

Experiment

Model	#Param	Dev	Test
Dataset `text8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.18
Adaptive (Sukhbaatar et al., 2019)	38M	1.05	1.11
BP-Transformer (Ye et al., 2019)	39M	-	1.11
Our Longformer	41M	1.04	**1.10**
Dataset `enwik8`			
T12 (Al-Rfou et al., 2018)	44M	-	1.11
Transformer-XL (Dai et al., 2019)	41M	-	1.06
Reformer (Kitaev et al., 2020)	-	-	1.05
Adaptive (Sukhbaatar et al., 2019)	39M	1.04	1.02
BP-Transformer (Ye et al., 2019)	38M	-	1.02
Our Longformer	41M	1.02	**1.00**

Table 2: *Small* model BPC on `text8` & `enwik8`

	QA			Coref.	Classification	
Model	WikiHop	TriviaQA	HotpotQA	OntoNotes	IMDB	Hyperpartisan
RoBERTa-base	72.4	74.3	63.5	78.4	95.3	87.4
Longformer-base	**75.0**	**75.2**	**64.4**	**78.6**	**95.7**	**94.8**

Table 8: Summary of finetuning results on QA, coreference resolution, and document classification. Results are on the development sets comparing our Longformer-base with RoBERTa-base. TriviaQA, Hyperpartisan metrics are F1, WikiHop and IMDB use accuracy, HotpotQa is joint F1, OntoNotes is average F1.

Big Bird: Transformers for Longer Sequences

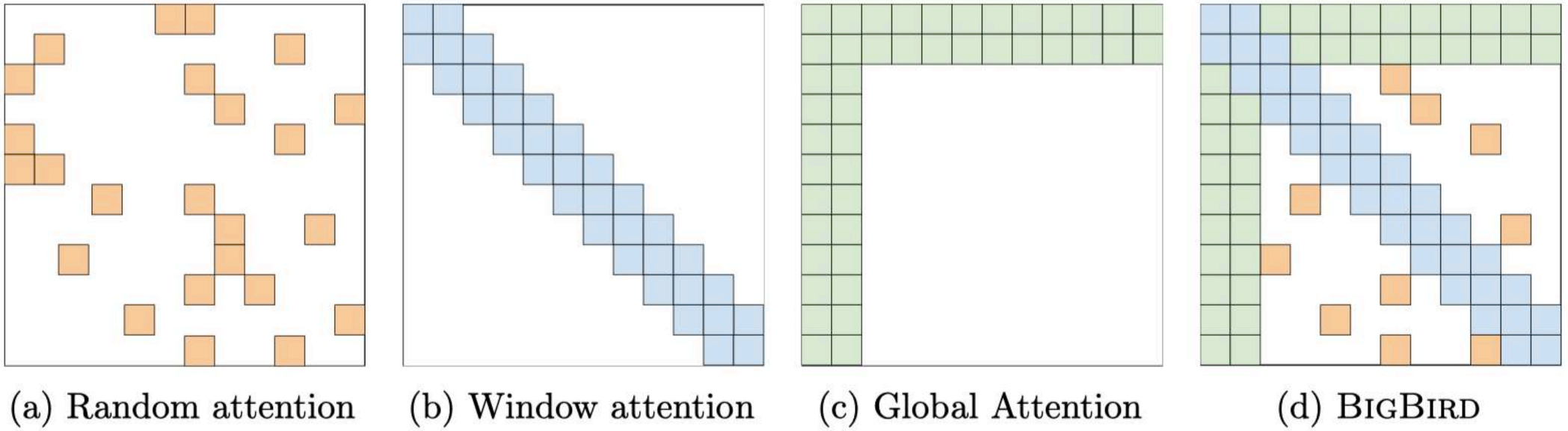
Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, Amr Ahmed

Google Research

Big Bird?

http://speech.ee.ntu.edu.tw/~tlkagk/courses/DLHLP20/BERT%20train%20(v8).pdf

Attention mechanism



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

Figure 1: Building blocks of the attention mechanism used in BIGBIRD. White color indicates absence of attention. (a) random attention with $r = 2$, (b) sliding window attention with $w = 3$ (c) global attention with $g = 2$. (d) the combined BIGBIRD model.

Pre-training

- warm-starting from the public RoBERTa checkpoint

Dataset	# tokens	Avg. doc len.
Books [110]	1.0B	37K
CC-News [35]	7.4B	561
Stories [90]	7.7B	8.2K
Wikipedia	3.1B	592

Table 2: Dataset used for pre training.

Summarization

	Instances			Input Length		Output Length	
Dataset	Training	Dev	Test	Median	90%-ile	Median	90%-ile
Arxiv [21]	203037	6436	6440	6151	14405	171	352
PubMed [21]	119924	6633	6658	2715	6101	212	318
BigPatent [79]	1207222	67068	67072	3082	7693	123	197

Table 7: Statistics of datasets used for summarization.

Summarization

- we used sparse BigBird attention only for encoder, while keeping the full attention for decoder.

Parameter		Base: BIGBIRD-RoBERTa	Large: BIGBIRD-Pegasus
Block length, b		64	64
Global token location		ITC	ITC
# of global token, g		$2 \times b$	$2 \times b$
Window length, w		$3 \times b$	$3 \times b$
# of random token, r		$3 \times b$	$3 \times b$
Max. encoder sequence length	BBC-XSUM:	1024	1024
	CNN/DM:	2048	2048
	Others:	3072	3072
Max. decoder sequence length	BBC-XSUM:	64	64
	CNN/DM:	128	128
	Others:	256	256
Beam size		5	5
Length penalty	BBC-XSUM:	0.7	0.7
	Others:	0.8	0.8
# of heads		12	16
# of hidden layers		12	16
Hidden layer size		768	1024
Batch size		128	128
Loss		teacher forced cross-entropy	teacher forced cross-entropy
Activation layer		gelu	gelu
Dropout prob		0.1	0.1
Attention dropout prob		0.1	0.1
Optimizer		Adam	Adafactor
Learning rate		1×10^{-5}	1×10^{-4}
Compute resources		4×4 TPUv3	4×8 TPUv3

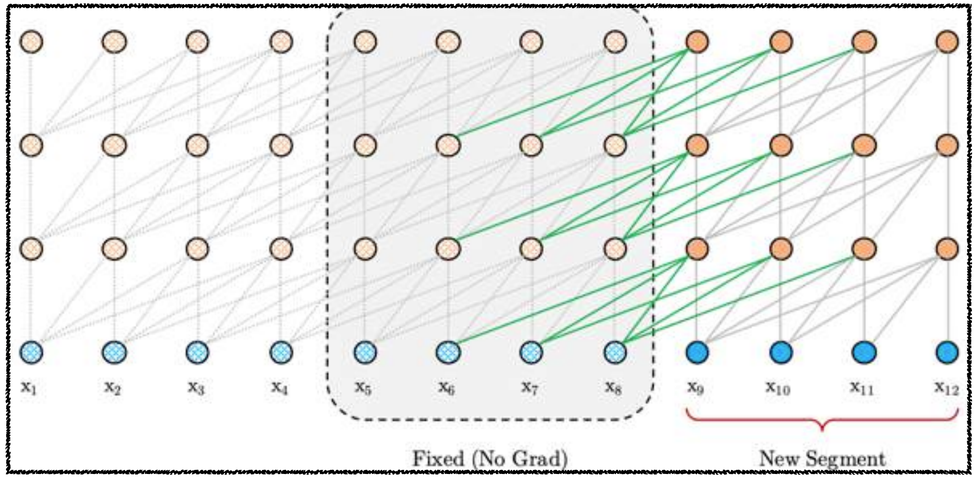
Table 18: Encoder hyperparameters for Summarization. We use full attention in decoder

Summarization

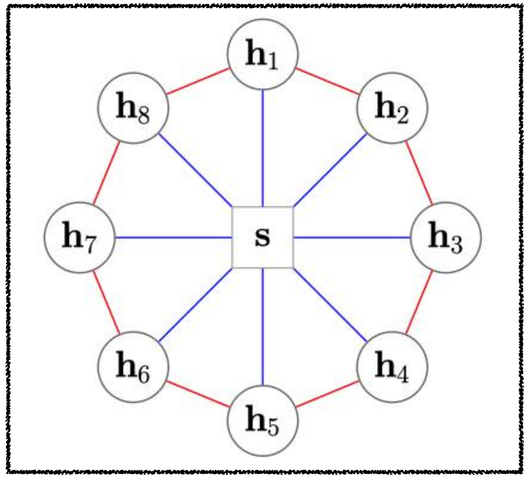
	Model	Arxiv			PubMed			BigPatent		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Prior Art	SumBasic [69]	29.47	6.95	26.30	37.15	11.36	33.43	27.44	7.08	23.66
	LexRank [26]	33.85	10.73	28.99	39.19	13.89	34.59	35.57	10.47	29.03
	LSA [98]	29.91	7.42	25.67	33.89	9.93	29.70	-	-	-
	Attn-Seq2Seq [86]	29.30	6.00	25.56	31.55	8.52	27.38	28.74	7.87	24.66
	Pntr-Gen-Seq2Seq [78]	32.06	9.04	25.16	35.86	10.22	29.69	33.14	11.63	28.55
	Long-Doc-Seq2Seq [21]	35.80	11.05	31.80	38.93	15.37	35.21	-	-	-
	Sent-CLF [82]	34.01	8.71	30.41	45.01	19.91	41.16	36.20	10.99	31.83
	Sent-PTR [82]	42.32	15.63	38.06	43.30	17.92	39.47	34.21	10.78	30.07
	Extr-Abst-TLM [82]	41.62	14.69	38.03	42.13	16.27	39.21	38.65	12.31	34.09
	Dancer [32]	42.70	16.54	38.44	44.09	17.69	40.27	-	-	-
Base	Transformer	28.52	6.70	25.58	31.71	8.32	29.42	39.66	20.94	31.20
	+ RoBERTa [77]	31.98	8.13	29.53	35.77	13.85	33.32	41.11	22.10	32.58
	+ Pegasus [107]	34.81	10.16	30.14	39.98	15.15	35.89	43.55	20.43	31.80
	BIGBIRD-RoBERTa	41.22	16.43	36.96	43.70	19.32	39.99	55.69	37.27	45.56
Large	Pegasus (Reported) [107]	44.21	16.95	38.83	45.97	20.15	41.34	52.29	33.08	41.75
	Pegasus (Re-eval)	43.85	16.83	39.17	44.53	19.30	40.70	52.25	33.04	41.80
	BIGBIRD-Pegasus	**46.63**	**19.02**	**41.77**	**46.32**	**20.65**	**42.33**	**60.64**	**42.46**	**50.01**

Table 8: Summarization ROUGE score for long documents.

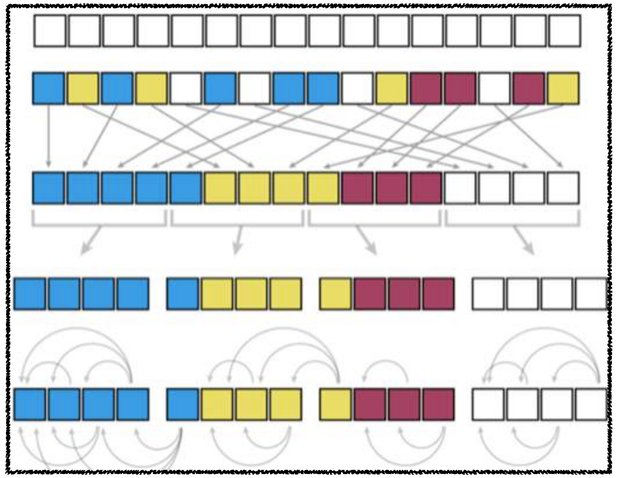
Conclusions



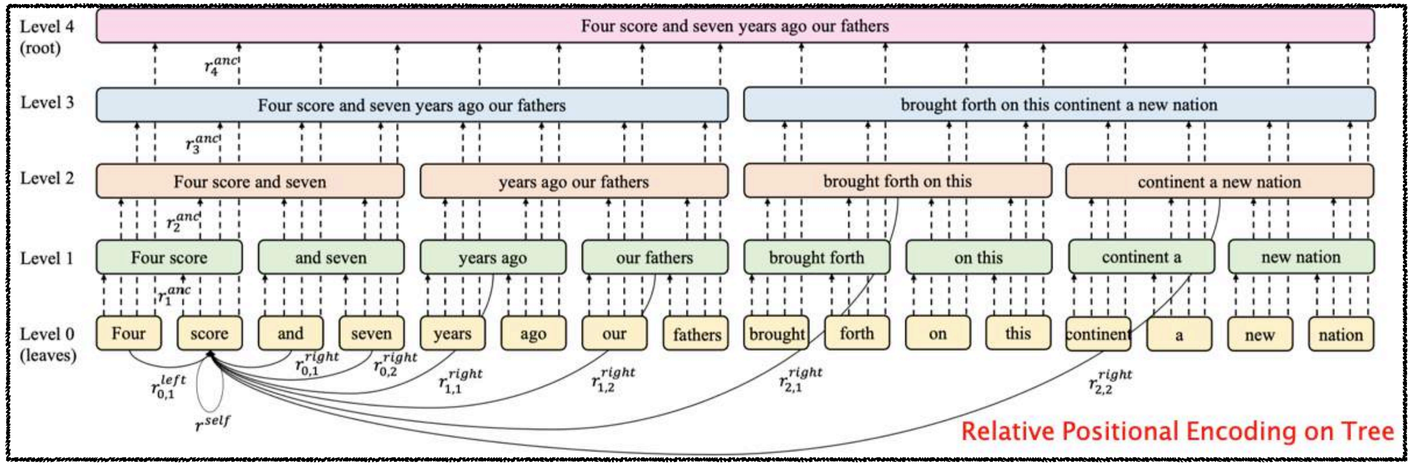
Transformer-XL



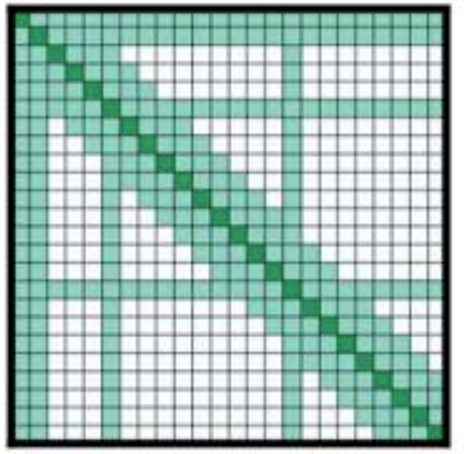
Star-Transformer



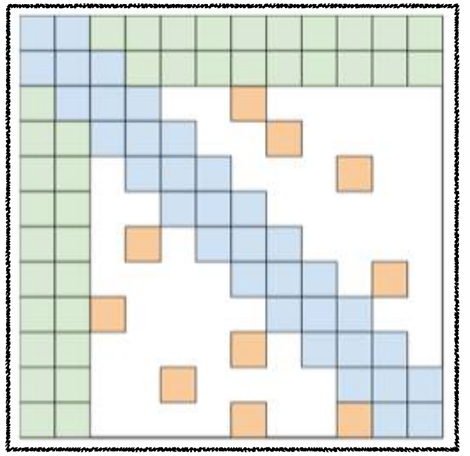
Reformer



BP-Transformer

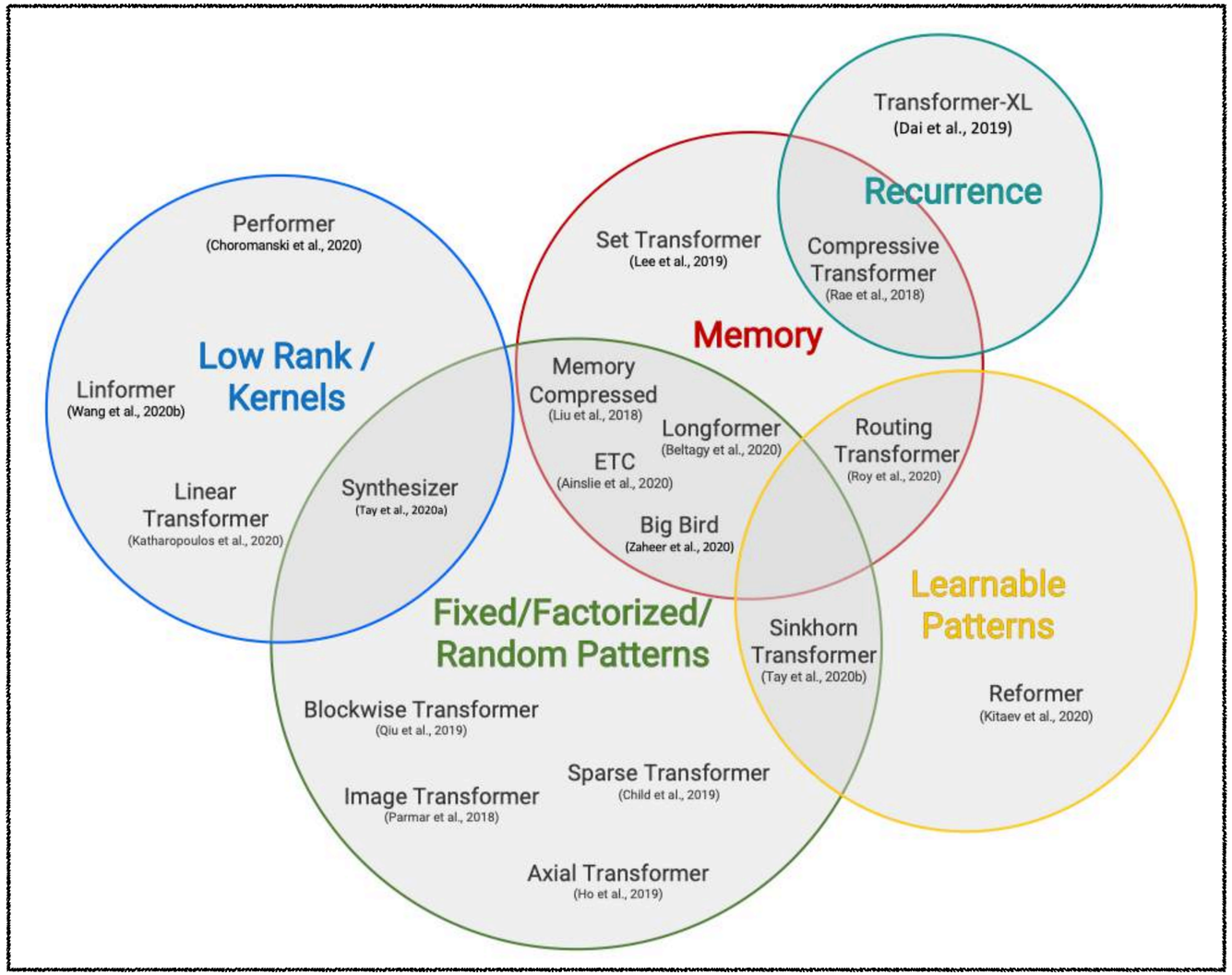


Longformer



Big Bird

For more



Efficient Transformers: A Survey

For more

- **[Survey]** Efficient Transformers: A Survey https://arxiv.org/abs/2009.06732
- **[BLOG]** A Survey of Long-Term Context in Transformers https://www.pragmatic.ml/a-survey-of-methods-for-incorporating-long-term-context/
- **[Repo]** Separius/awesome-fast-attention https://github.com/Separius/awesome-fast-attention

Thanks!