Transformers are RNNs:
Fast Autoregressive Transformers with Linear Attention

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, François Fleuret

Rak Arina, AMI171
HSE Research Seminar
2020

Recap | Transformers

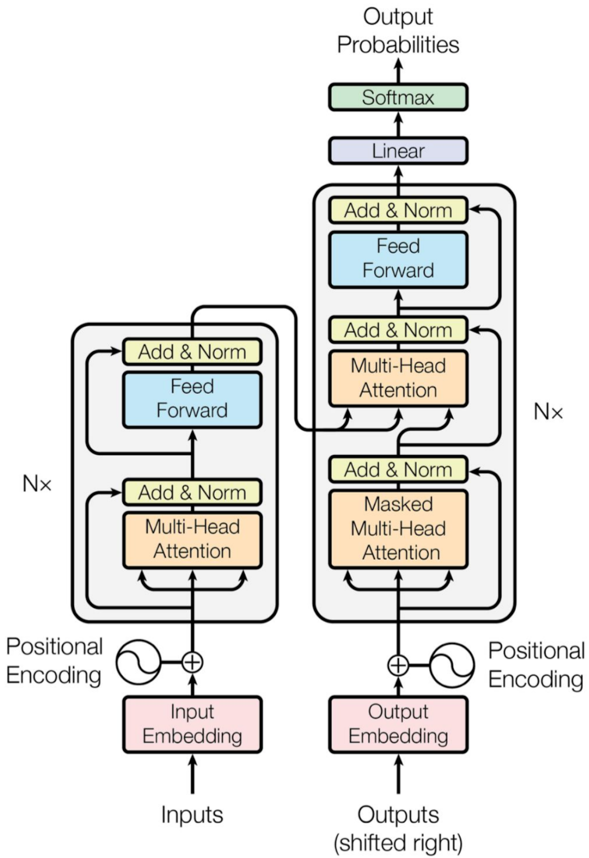
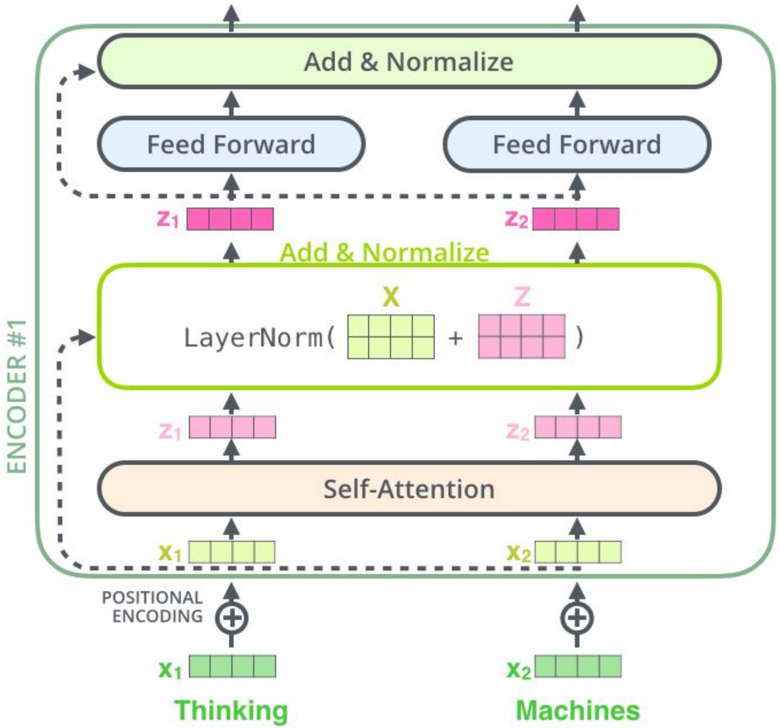


Figure 1: The Transformer - model architecture.



Recap | Transformers Self-attention

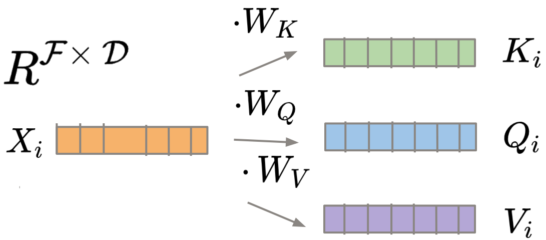
X - input
K - key
Q - query
V - value
Y - output

$$X \in \mathbb{R}^{\mathcal{N} \times \mathcal{F}}, \; W_K, W_Q \in R^{\mathcal{F} \times \mathcal{M}}, \; W_V \in R^{\mathcal{F} \times \mathcal{D}}$$

$$K = XW_K \in R^{\mathcal{N} \times \mathcal{M}}$$

$$Q = XW_Q \in R^{\mathcal{N} \times \mathcal{M}}$$

$$V = XW_V \in R^{\mathcal{N} \times \mathcal{D}}$$



$$y_i = \sum_j w_j V_j = softmax\left(\frac{Q_i K^T}{\sqrt{\mathcal{D}}}\right)V, \quad w_j = \frac{\exp(Q_i^T \cdot K_j / \sqrt{\mathcal{D}})}{\exp(Q_i^T \cdot K_j / \sqrt{\mathcal{D}})}$$

$$Y = softmax\left(\frac{QK^T}{\sqrt{\mathcal{D}}}\right) V$$

QK^T – takes $O(\mathcal{N}^2)$ both memory and space

Self-attention

$$y_i = \sum_j w_j V_j = \frac{\sum_j sim(Q_i , K_j) V_j}{\sum_j sim(Q_i, K_j)}$$

For softmax attention:

$$sim(Q_i, K_j) = \exp\left(\frac{Q_i^T K_j}{\sqrt{\mathcal{D}}}\right)$$

Self-attention

For softmax attention:

$$y_i \;=\; \sum_j w_j\, V_j = \;\sum_j \frac{sim(Q_i\;,\; K_j)\, V_j}{sim(Q_i,\; K_j)}$$

$$sim(Q_i,\; K_j) \;=\; \exp\left(\frac{Q_i^T K_j}{\sqrt{\mathcal{D}}}\right)$$

Kernelization

Lets choose a kernel $k(\cdot,\;\cdot)$ with a feature space ϕ as $a\ sim$:

$$y_i \;=\; \sum_j \frac{k(Q_i\;,\; K_j)\, V_j}{k(Q_i,\; K_j)} \;=\; \sum_j \frac{\phi(Q_i\;)^T\,\phi(K_j)V_j}{\phi(Q_i\;)^T\,\phi(K_j)}$$

$$\phi(x) \;=\; elu(x) \;+\; 1$$

Self-attention

For softmax attention:

$$sim(Q_i,\, K_j) \;=\; \exp\left(\frac{Q_i^T K_j}{\sqrt{\mathcal{D}}}\right)$$

$$y_i \;=\; \sum_j w_j\, V_j = \; \sum_j \frac{sim(Q_i\,,\; K_j)\, V_j}{sim(Q_i,\; K_j)}$$

Kernelization

Lets choose a kernel $k(\cdot,\ \cdot)$ with a feature space ϕ as $a\ sim$:

$$y_i \;=\; \sum_j \frac{k(Q_i\,,\; K_j)\, V_j}{k(Q_i,\; K_j)} \;=\; \sum_j \frac{\phi(Q_i\,)^T\, \phi(K_j) V_j}{\phi(Q_i\,)^T\, \phi(K_j)}$$

$$\phi(x) \;=\; elu(x) \;+\; 1$$

Associativity

$$y_i \;=\; \frac{\phi(Q_i\,)^T \sum_j \phi(K_j) V_j^T}{\phi(Q_i\,)^T \sum_j \phi(K_j)}$$

Self-attention

For softmax attention:

$$sim(Q_i,\, K_j) \;=\; \exp\left(\frac{Q_i^T K_j}{\sqrt{\mathcal{D}}}\right)$$

$$y_i \;=\; \sum_j w_j\, V_j = \sum_j \frac{sim(Q_i\,,\, K_j)\, V_j}{sim(Q_i,\, K_j)}$$

Kernelization

Lets choose a kernel $k(\cdot,\ \cdot)$ with a feature space ϕ as $a\ sim$:

$$y_i \;=\; \sum_j \frac{k(Q_i\,,\, K_j)\, V_j}{k(Q_i,\, K_j)} \;=\; \sum_j \frac{\phi(Q_i\,)^T\, \phi(K_j) V_j}{\phi(Q_i\,)^T\, \phi(K_j)}$$

$$\phi(x) \;=\; elu(x) \,+\, 1$$

Associativity

$$y_i \;=\; \frac{\phi(Q_i\,)^T \sum_j \phi(K_j) V_j^T}{\phi(Q_i\,)^T \sum_j \phi(K_j)}$$

$$\sum_j \phi(K_j) V_j^T \;-\; \text{precaculating in } O(\mathcal{N})$$

$$\underbrace{Y \;\propto\; \phi(Q) \underbrace{\left(\phi(K)^T V\right)}_{O(\mathcal{N})}}_{O(\mathcal{N})}$$

$$K,\, Q \in \mathbb{R}^{\mathcal{N} \times \mathcal{M}},\, V \in \mathbb{R}^{\mathcal{N} \times \mathcal{D}},\, \phi : \mathbb{R}^{\mathcal{F}} \to \mathbb{R}^{\mathcal{C}}$$
$$\phi(x) = elu(x) + 1 \implies C = M$$

Softmax attention

$$Y = softmax\left(\frac{QK^T}{\sqrt{\mathcal{D}}}\right) V$$

$$QK^T \sim O(\mathcal{N}^2\mathcal{M})$$

$$softmax\left(\frac{QK^T}{\sqrt{\mathcal{D}}}\right) \sim O(\mathcal{N}^2)$$

$$softmax\left(\frac{QK^T}{\sqrt{\mathcal{D}}}\right) V \sim O(\mathcal{N}^2\mathcal{D})$$

$$Y \sim O(\mathcal{N}^2 \max(\mathcal{D}, \mathcal{M}))$$

Kernelized attention

$$Y = \phi(Q) \left(\phi(K)^T V\right)$$

$$\phi(K) \sim O(\mathcal{N}\mathcal{M}) \cdot O(\phi(K_{ij})) = O(\mathcal{N}\mathcal{M})$$

$$\phi(K)^T V \sim O(\mathcal{N}\mathcal{D}\mathcal{C})$$

$$\phi(Q) \left(\phi(K)^T V\right) \sim O(\mathcal{N}\mathcal{D}\mathcal{C})$$

$$Y \sim O(\mathcal{N}\mathcal{D}\mathcal{C})$$

Causal masking

Non-autoregressive

$$y_i = \frac{\sum_{j=1}^{\mathcal{N}} sim(Q_i , K_j)\, V_j}{\sum_{j=1}^{\mathcal{N}} sim(Q_i, K_j)}$$

Autoregressive

$$y_i = \frac{\sum_{j=1}^{i} sim(Q_i , K_j)\, V_j}{\sum_{j=1}^{i} sim(Q_i, K_j)}$$

Causal masking

Non-autoregressive

$$y_i = \sum_{j=1}^{\mathcal{N}} \frac{sim(Q_i , K_j) V_j}{sim(Q_i, K_j)}$$

$$y_i = \frac{\phi(Q_i)^T \sum_{j=1}^{\mathcal{N}} \phi(K_j) V_j^T}{\phi(Q_i)^T \sum_{j=1}^{\mathcal{N}} \phi(K_j)}$$

Autoregressive

$$y_i = \sum_{j=1}^{i} \frac{sim(Q_i , K_j) V_j}{sim(Q_i, K_j)}$$

$$y_i = \frac{\phi(Q_i)^T \sum_{j=1}^{i} \phi(K_j) V_j^T}{\phi(Q_i)^T \sum_{j=1}^{i} \phi(K_j)}$$

Causal masking

Non-autoregressive

$$y_i \;=\; \sum_{j=1}^{\mathcal{N}} \frac{sim(Q_i\,,\,K_j)\,V_j}{sim(Q_i,\,K_j)}$$

$$y_i \;=\; \frac{\phi(Q_i)^T \overbrace{\sum_{j=1}^{\mathcal{N}} \phi(K_j)V_j^T}^{S}}{\phi(Q_i)^T \underbrace{\sum_{j=1}^{\mathcal{N}} \phi(K_j)}_{Z}}$$

Autoregressive

$$y_i \;=\; \sum_{j=1}^{i} \frac{sim(Q_i\,,\,K_j)\,V_j}{sim(Q_i,\,K_j)}$$

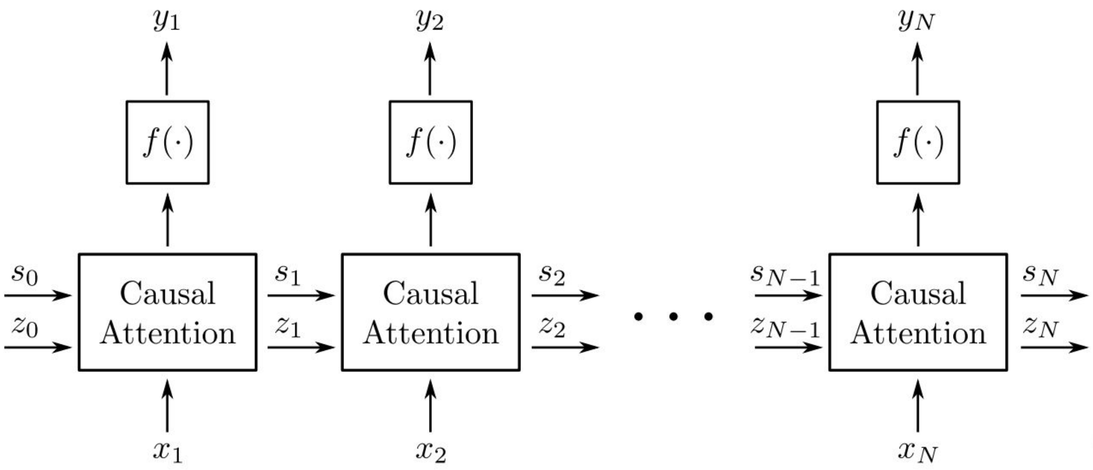
$$y_i \;=\; \frac{\phi(Q_i)^T \overbrace{\sum_{j=1}^{i} \phi(K_j)V_j^T}^{S_i}}{\phi(Q_i)^T \underbrace{\sum_{j=1}^{i} \phi(K_j)}_{Z_i}}$$

$$S_i \;=\; S_{i-1} + \phi(K_i)V_i^T$$
$$Z_i \;=\; Z_{i-1} + \phi(K_i)$$

Transformers are RNNs

$$
\begin{aligned}
s_0 &= 0 \\
z_0 &= 0 \\
&\vdots \\
s_i &= s_{i-1} + \phi(x_i \; W_K)\,(x_i W_V)^T \\
z_i &= z_{i-1} + \phi(x_i \; W_K)
\end{aligned}
$$

$$
y_i = f_l\left(\frac{\phi(x_i W_Q)^T s_i}{\phi(x_i W_Q)^T z_i} + x_i\right)
$$



The resulting RNN has two hidden states: **the attention memory** s and the **normalizer memory** z

Tansformer causal attention

- Parallelizable during training
- $O\left(\mathcal{N}^2\right)$ during inference

RNN causal attention

- Sequential during training
- $O(1)$ Space

$O(\mathcal{N})$ Time

during inference

Linear tansformer causal attention

- Parallelizable during training
- $O(1)$ Space

$O(\mathcal{N})$ Time

during inference

This results in inference **thousands of times** faster than other transformer models.

Experiments

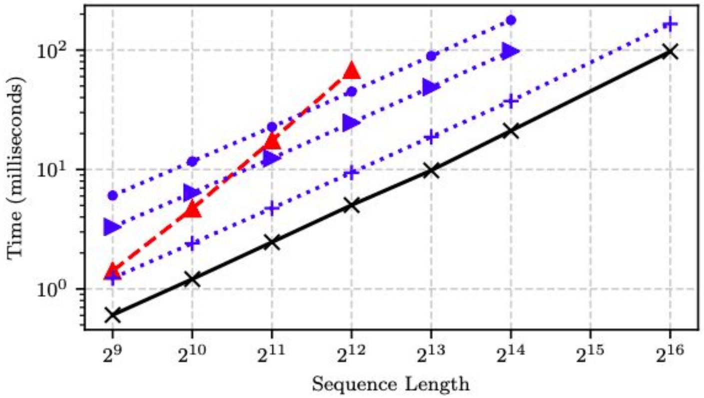
Baselines

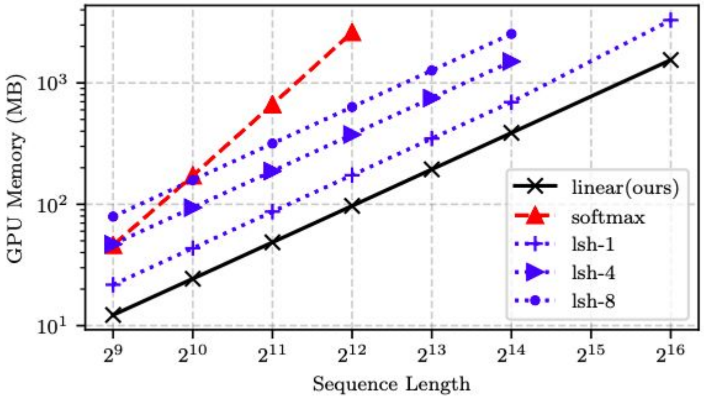
- Softmax transformer (Vaswani et al., 2017)
- LSH attention from Reformer (Kitaev et al., 2020)

Experiments

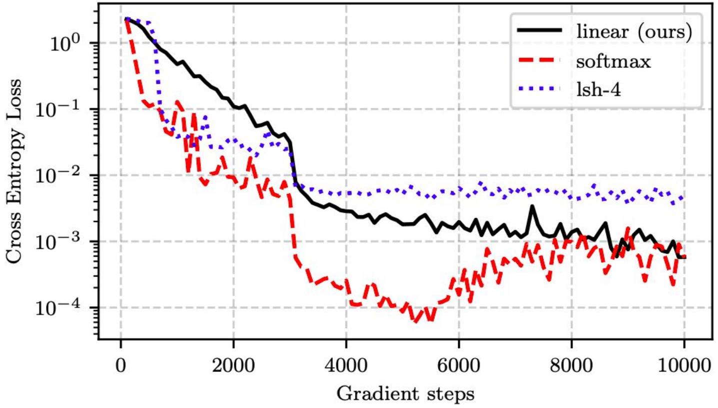
- Artificial benchmark for computational and memory requirements
- Autoregressive image generation on MNIST and CIFAR-10
- Automatic speech recognition on Wall Street Journal

Artificial copy task with causal masking



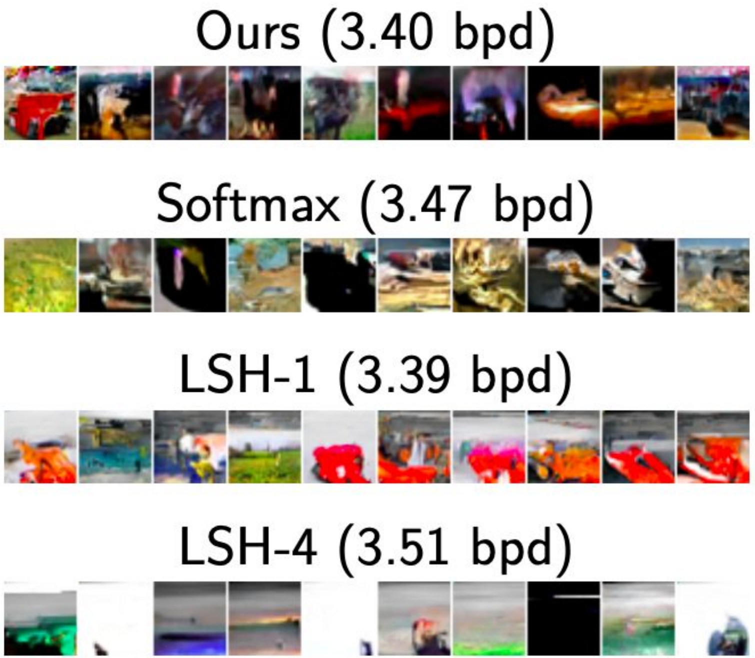


Linear and Reformer models scale linearly with the sequence length unlike softmax which scales with the square of the sequence length both in memory and time.



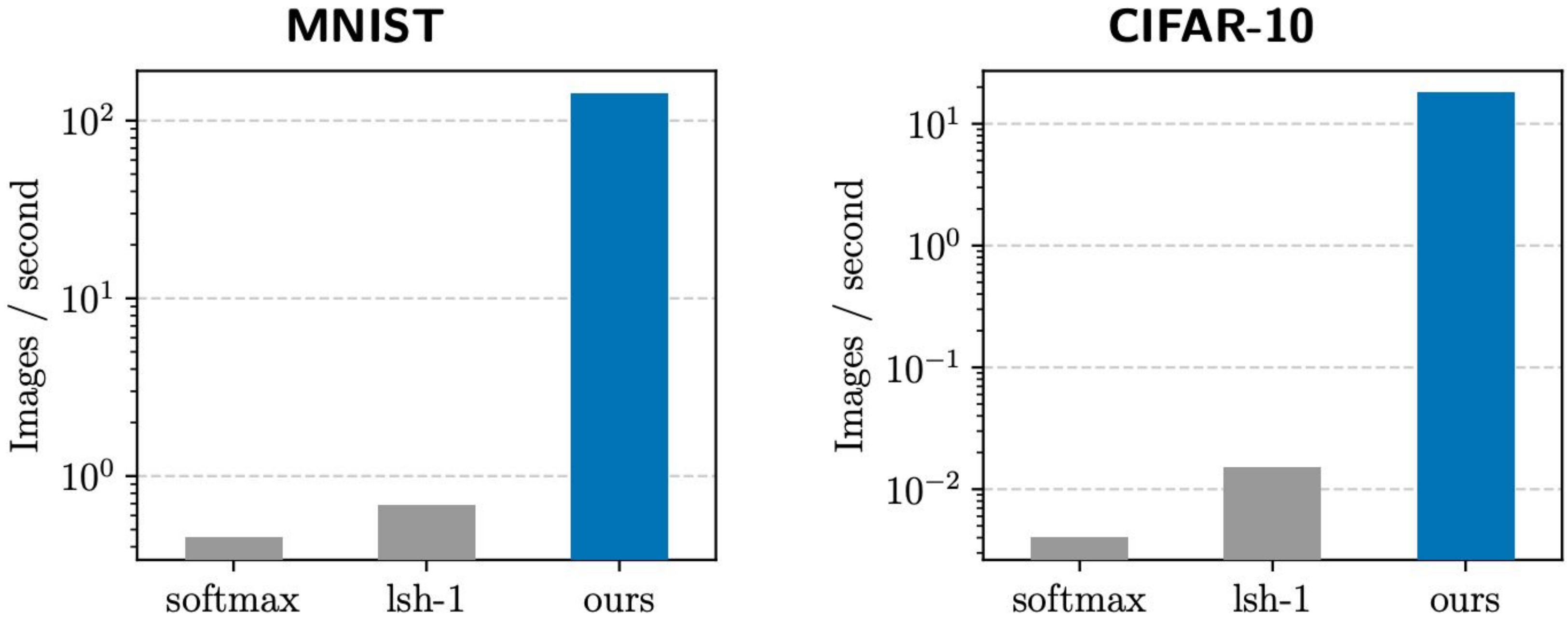
Unconditional generation after 250 epochs on MNIST

Unconditional generation after 1 GPU week on CIFAR-10

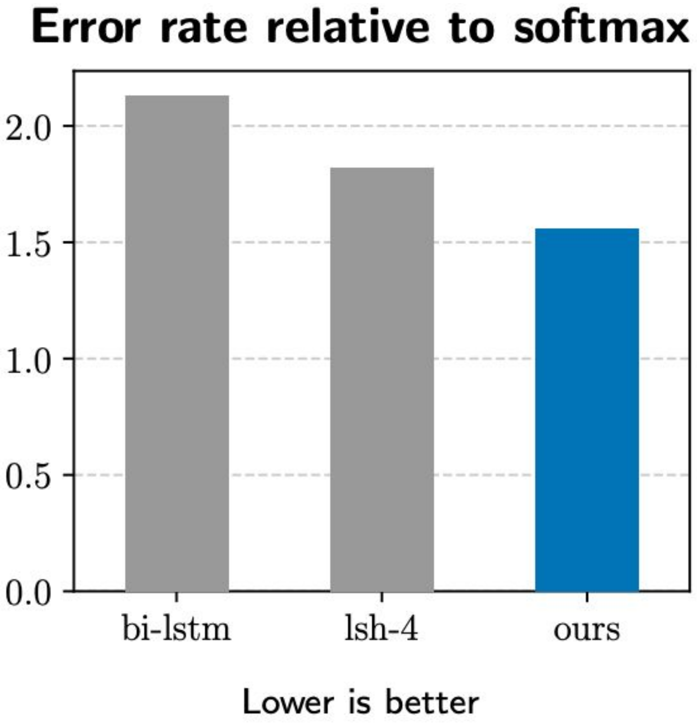


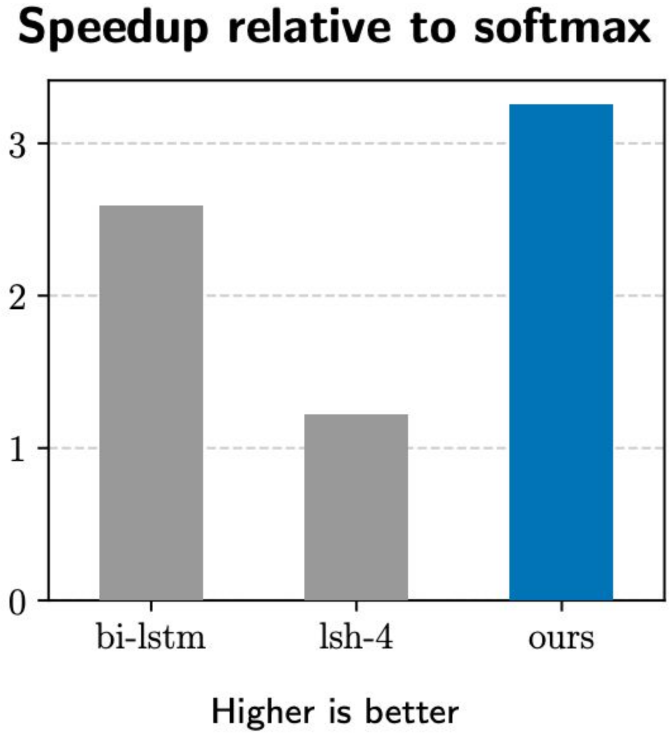
1,000 times faster image generation
Constant memory per image from the first to the last pixel.

Image generation



Automatic speech recognition





Comparison of efficient Transformers

Model	ListOps	Text	Retrieval	Image	Pathfinder	Path-X	Avg
Transformer	36.37	64.27	57.46	42.44	71.40	FAIL	54.39
Local Attention	15.82	52.98	53.39	41.46	66.63	FAIL	46.06
Sparse Trans.	17.07	63.58	**59.59**	**44.24**	71.71	FAIL	51.24
Longformer	35.63	62.85	56.89	42.22	69.71	FAIL	53.46
Linformer	35.70	53.94	52.27	38.56	76.34	FAIL	51.36
Reformer	**37.27**	56.10	53.40	38.07	68.50	FAIL	50.67
Sinkhorn Trans.	33.67	61.20	53.83	41.23	67.45	FAIL	51.39
Synthesizer	36.99	61.68	54.67	41.61	69.45	FAIL	52.88
BigBird	36.05	64.02	59.29	40.83	74.87	FAIL	**55.01**
Linear Trans.	16.13	**65.90**	53.09	42.34	75.30	FAIL	50.55
Performer	18.01	65.40	53.82	42.77	**77.05**	FAIL	51.41
Task Avg (Std)	29 (9.7)	61 (4.6)	55 (2.6)	41 (1.8)	72 (3.7)	FAIL	52 (2.4)

LONG RANGE ARENA: A BENCHMARK FOR EFFICIENT TRANSFORMERS

	Steps per second				Peak Memory Usage (GB)			
Model	1K	2K	3K	4K	1K	2K	3K	4K
Transformer	8.1	4.9	2.3	1.4	0.85	2.65	5.51	9.48
Local Attention	9.2 (1.1x)	8.4 (1.7x)	7.4 (3.2x)	7.4 (5.3x)	0.42	0.76	1.06	1.37
Linformer	9.3 (1.2x)	9.1 (1.9x)	8.5 (3.7x)	7.7 (5.5x)	**0.37**	**0.55**	0.99	**0.99**
Reformer	4.4 (0.5x)	2.2 (0.4x)	1.5 (0.7x)	1.1 (0.8x)	0.48	0.99	1.53	2.28
Sinkhorn Trans	9.1 (1.1x)	7.9 (1.6x)	6.6 (2.9x)	5.3 (3.8x)	0.47	0.83	1.13	1.48
Synthesizer	8.7 (1.1x)	5.7 (1.2x)	6.6 (2.9x)	1.9 (1.4x)	0.65	1.98	4.09	6.99
BigBird	7.4 (0.9x)	3.9 (0.8x)	2.7 (1.2x)	1.5 (1.1x)	0.77	1.49	2.18	2.88
Linear Trans.	9.1 (1.1x)	9.3 (1.9x)	8.6 (3.7x)	7.8 (5.6x)	**0.37**	0.57	**0.80**	1.03
Performer	**9.5** (1.2x)	**9.4** (1.9x)	**8.7** (3.8x)	**8.0** (5.7x)	**0.37**	0.59	0.82	1.06

Summary

- **Kernel feature maps** and **matrix associativity** yield an attention with linear complexity.
- Computing the key value matrix as a **cumulative sum** extends our efficient attention computation to the autoregressive case.
- Using the RNN formulation to perform autoregressive inference requires **constant memory** and is **many times faster**

Вопросы

- Основная идея: за счет чего происходит переход из O(n^2) в O(n) по памяти и времени в слое Self-attention?

- Как переписать задачу (слой attention), с учетом causal masking? Как добиться линейной сложности в случае autoregressive transformer?

- Как представить autoregressive transformer как RNN? Что будет являться скрытыми состояниями?