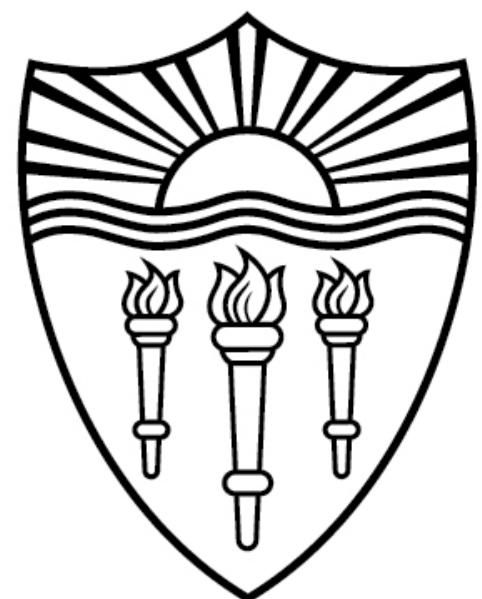


CSCI 544: Applied Natural Language Processing

Advanced Transformers

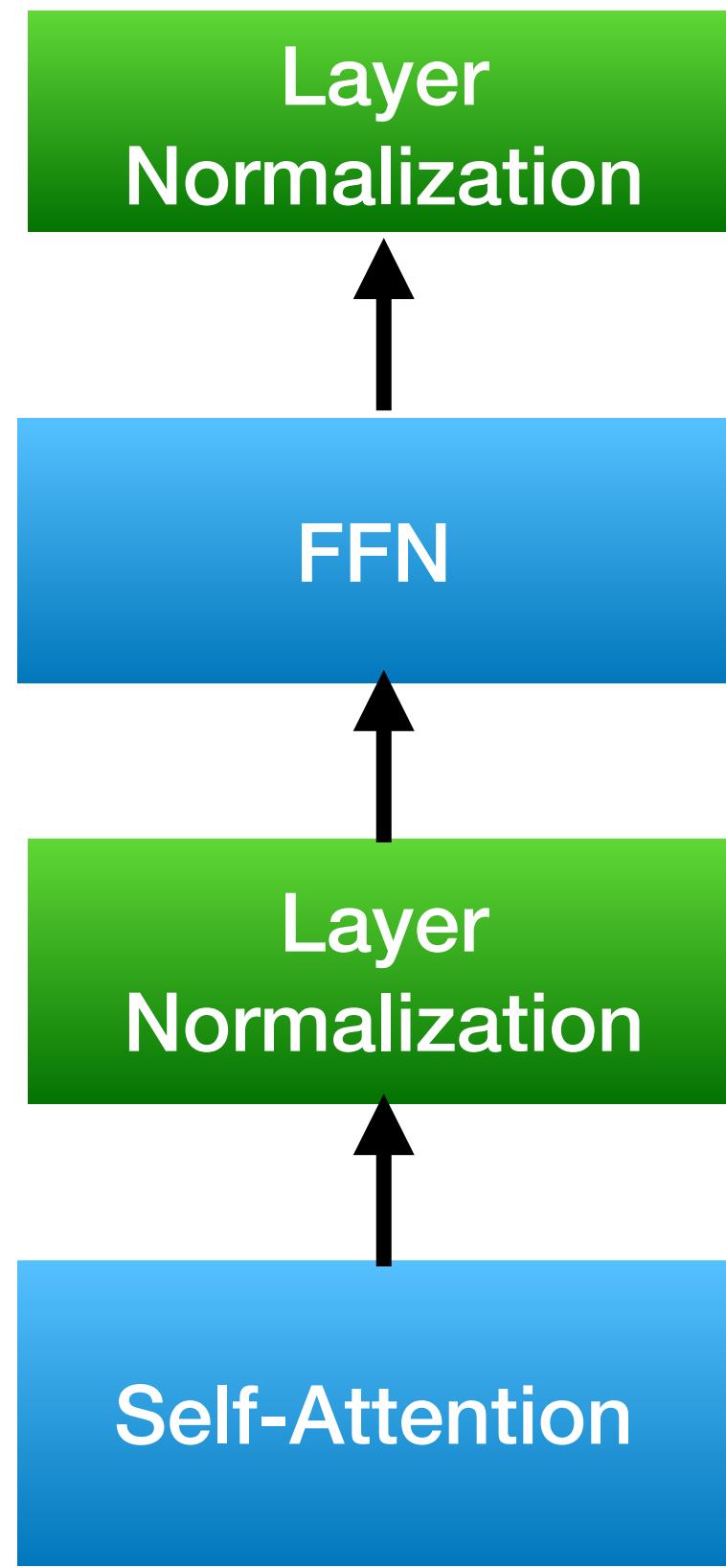
Xuezhe Ma (Max)



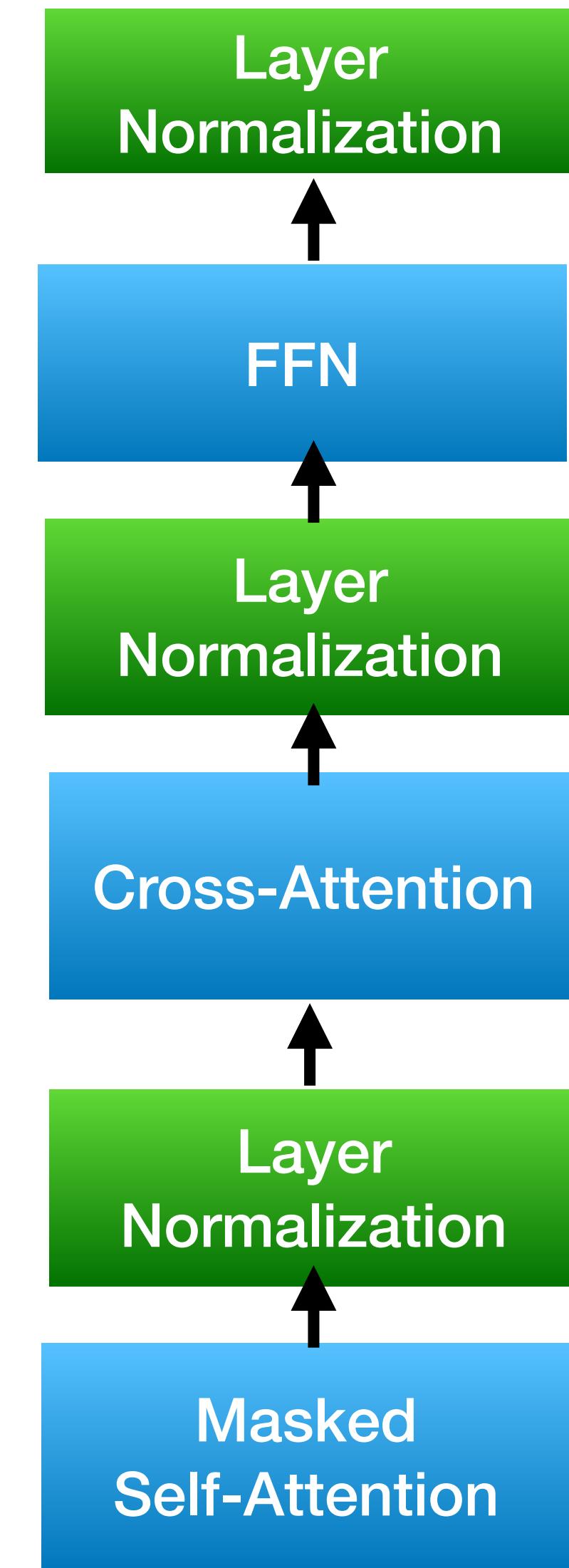
USC University of
Southern California

Recap: Transformers

Transformer Encoder Block



Transformer Decoder Block



Attention

- **Easier to capture dependencies:** we draw attention between every pair of words!
- **Easier to parallelize:**

$$\begin{aligned}\text{MultiHead}(X) &= \text{concat}(h_1, \dots, h_k)W_O \\ h_i &= \text{attn}(Q_i, K_i, V_i) \\ Q_i &= (XW_Q)^i, K_i = (XW_K)^i, V_i = (XW_V)^i\end{aligned}$$

$$\text{attn}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

Open Challenges in LLM Research

Aug 16, 2023 • Chip Huyen

Never before in my life had I seen so many smart people working on the same goal: making LLMs better. After talking to many people working in both industry and academia, I noticed the 10 major research directions that emerged. The first two directions, hallucinations and context learning, are probably the most talked about today. I'm the most excited about numbers 3 (multimodality), 5 (new architecture), and 6 (GPU alternatives).

Open challenges in LLM research

1. Reduce and measure hallucinations
 2. Optimize context length and context construction
 3. Incorporate other data modalities
 4. Make LLMs faster and cheaper
 5. Design a new model architecture
 6. Develop GPU alternatives
 7. Make agents usable
 8. Improve learning from human preference
 9. Improve the efficiency of the chat interface
 10. Build LLMs for non-English languages
-

From Huyen's Blog

Open Challenges in LLM Research

Aug 16, 2023 • Chip Huyen

Never before in my life had I seen so many smart people working on the same goal: making LLMs better. After talking to many people working in both industry and academia, I noticed the 10 major research directions that emerged. The first two directions, hallucinations and context learning, are probably the most talked about today. I'm the most excited about numbers 3 (multimodality), 5 (new architecture), and 6 (GPU alternatives).

Open challenges in LLM research

1. Reduce and measure hallucinations
2. Optimize context length and context construction
3. Incorporate other data modalities
4. Make LLMs faster and cheaper
5. Design a new model architecture
6. Develop GPU alternatives
7. Make agents usable
8. Improve learning from human preference
9. Improve the efficiency of the chat interface
10. Build LLMs for non-English languages

Long Context Modeling

From Huyen's Blog

Why Long Context?

- Longer Context = Higher Intelligence
- Long Context = Better Applications
 - Book-level summarization
 - Doc-level translation
 - Customized assistant
 - High-resolution & long video generation
 - ...



Pic from game of thrones

Long Context is hard for Transformer

- Complexity

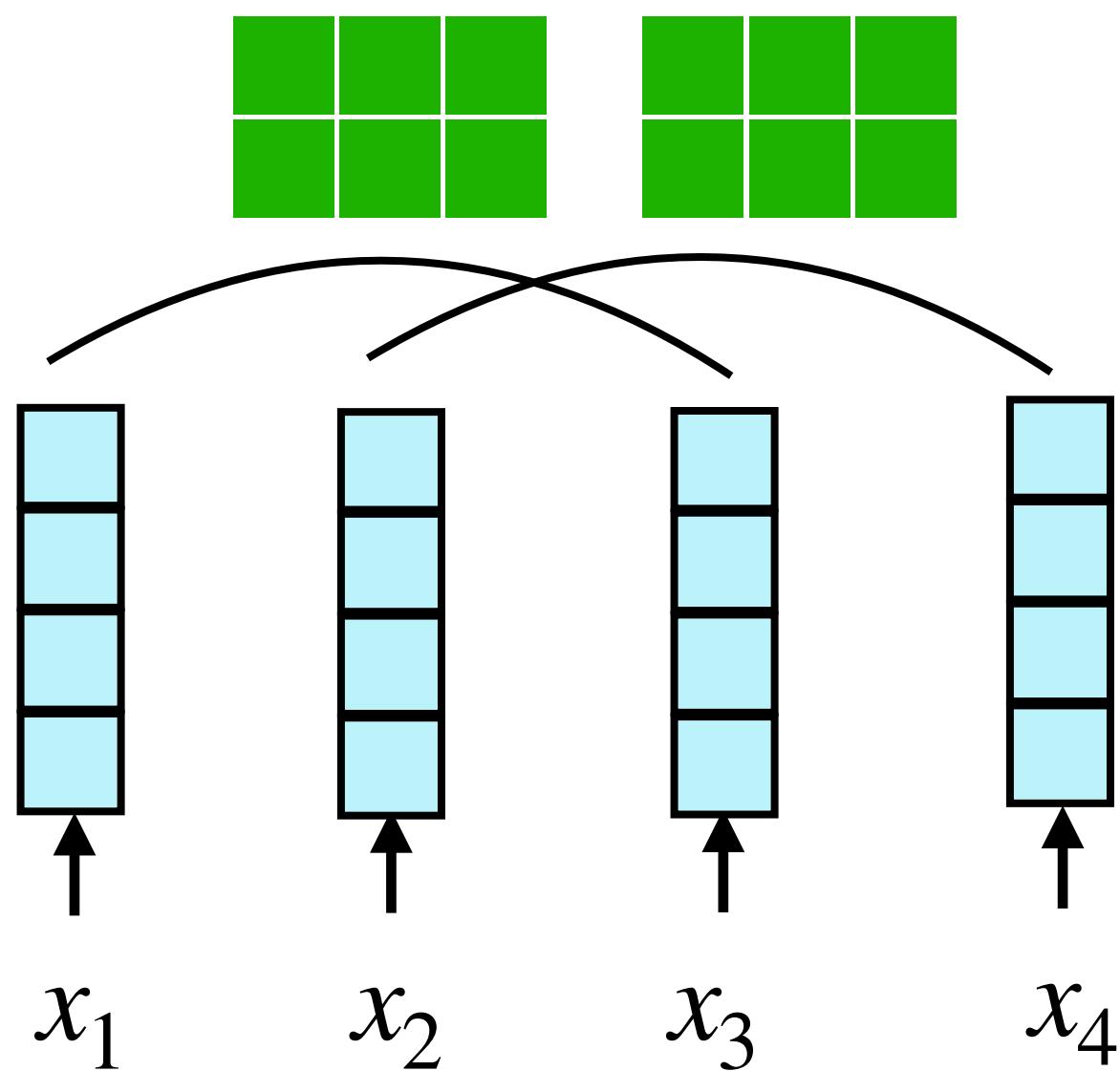
- Expensive: quadratic complexity
 - Both time and space
 - $O(hn^2)$: h heads and sequence length of n

- Inductive Bias on Length Generalization

- Hard to be extended to longer contexts than pre-training
 - Almost no prior knowledge of dependency patterns (weak inductive bias)
 - Position information only from absolute/relative positional embeddings

Inductive Bias: CNN & RNN

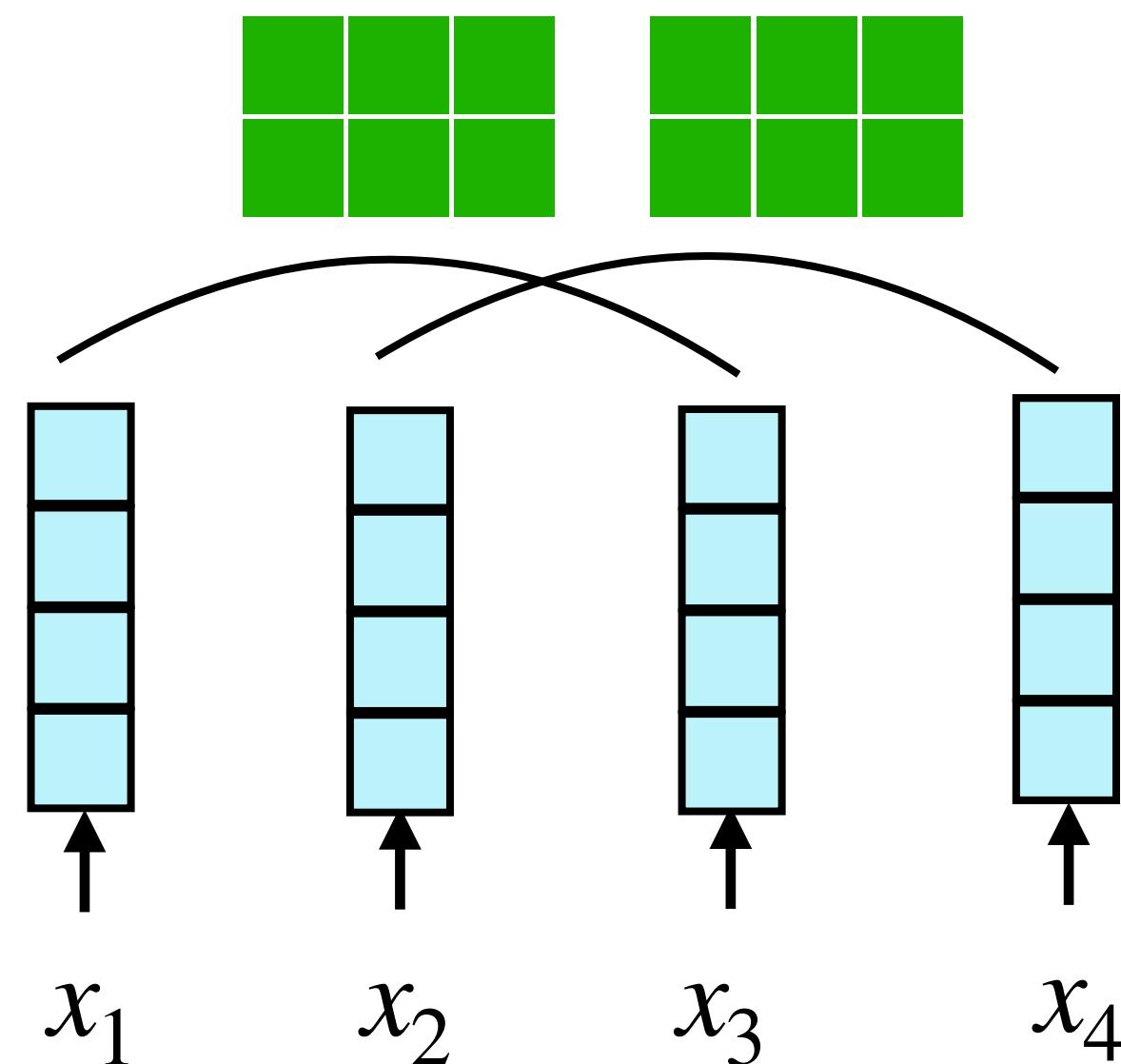
CNN



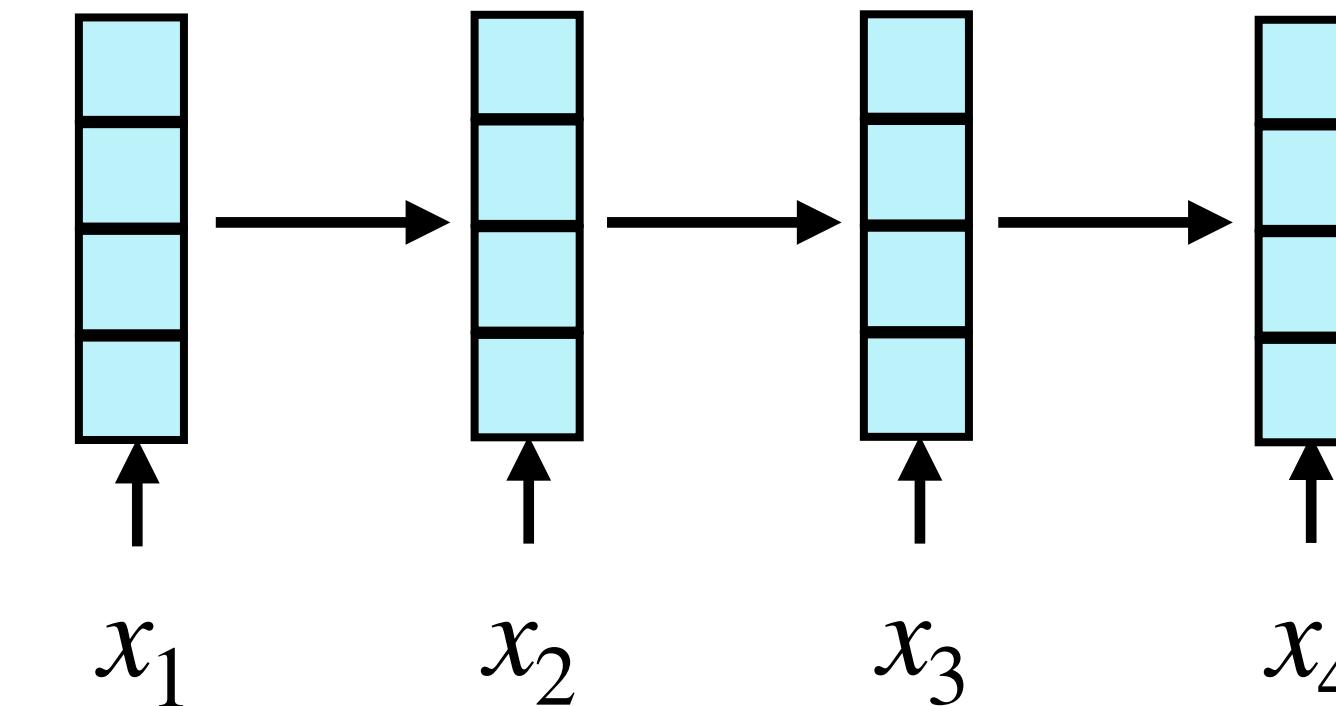
- Local dependencies
 - Window size is usually small (e.g. 3 or 5)
- Time-invariant kernel

Inductive Bias: CNN & RNN

CNN



RNN



- Local dependencies
 - Window size is usually small (e.g. 3 or 5)
- Time-invariant kernel

- Sequential dependencies
- Time-invariant recurrence

Weak Inductive Bias on Length Generalization

- Transformer Hard to be extended to longer contexts than pre-training

Attention score $A \in \mathbb{R}^{n \times n}$

	k_1	k_2	\dots	k_n
q_1			\dots	
q_2			\dots	
\vdots	\vdots	\vdots	\dots	\vdots
q_n			\dots	

- Pair-wise interaction
- Order-invariant if no positional embeddings

Efficient Attention Mechanisms

Attention Mechanism

Inputs: $Q \in \mathbb{R}^{n \times d}, C \in \mathbb{R}^{m \times d}$

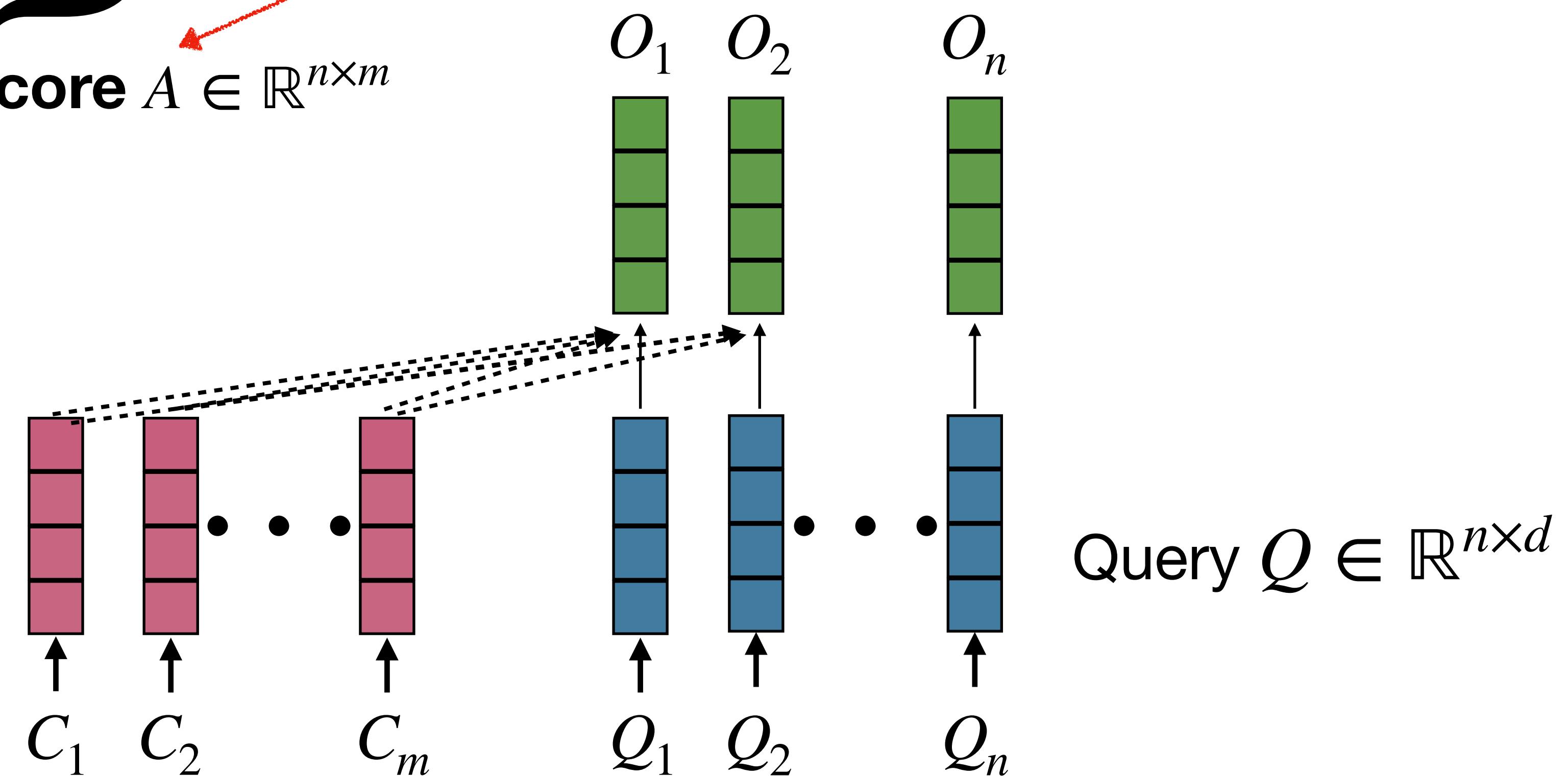
Outputs: $O = Attn(Q, C) = \sigma \left(\frac{QK^T}{\sqrt{d}} \right) V$

Attention score $A \in \mathbb{R}^{n \times m}$

time and memory consuming

$K = CW_K, V = CW_V, \sigma$ is the softmax

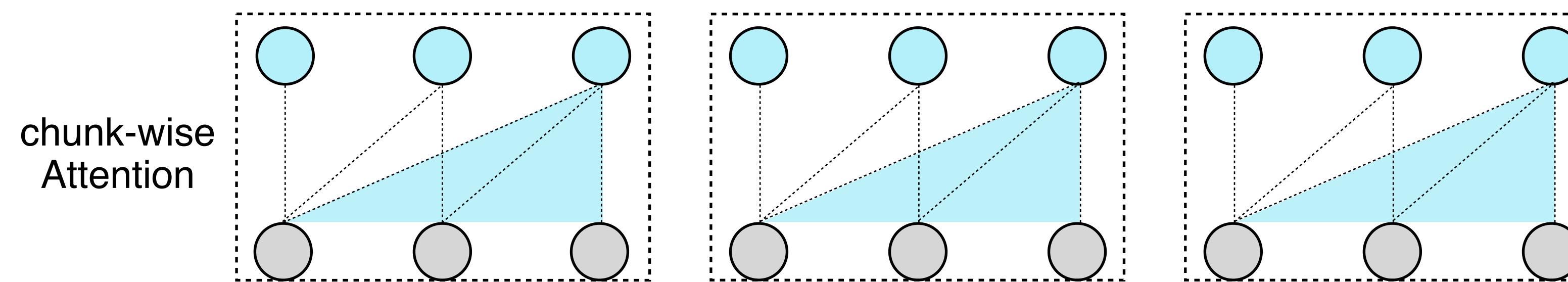
Context $C \in \mathbb{R}^{m \times d}$



Query $Q \in \mathbb{R}^{n \times d}$

Why not Chunk-wise Attention?

- Applying attention individually to each chunk with fixed length



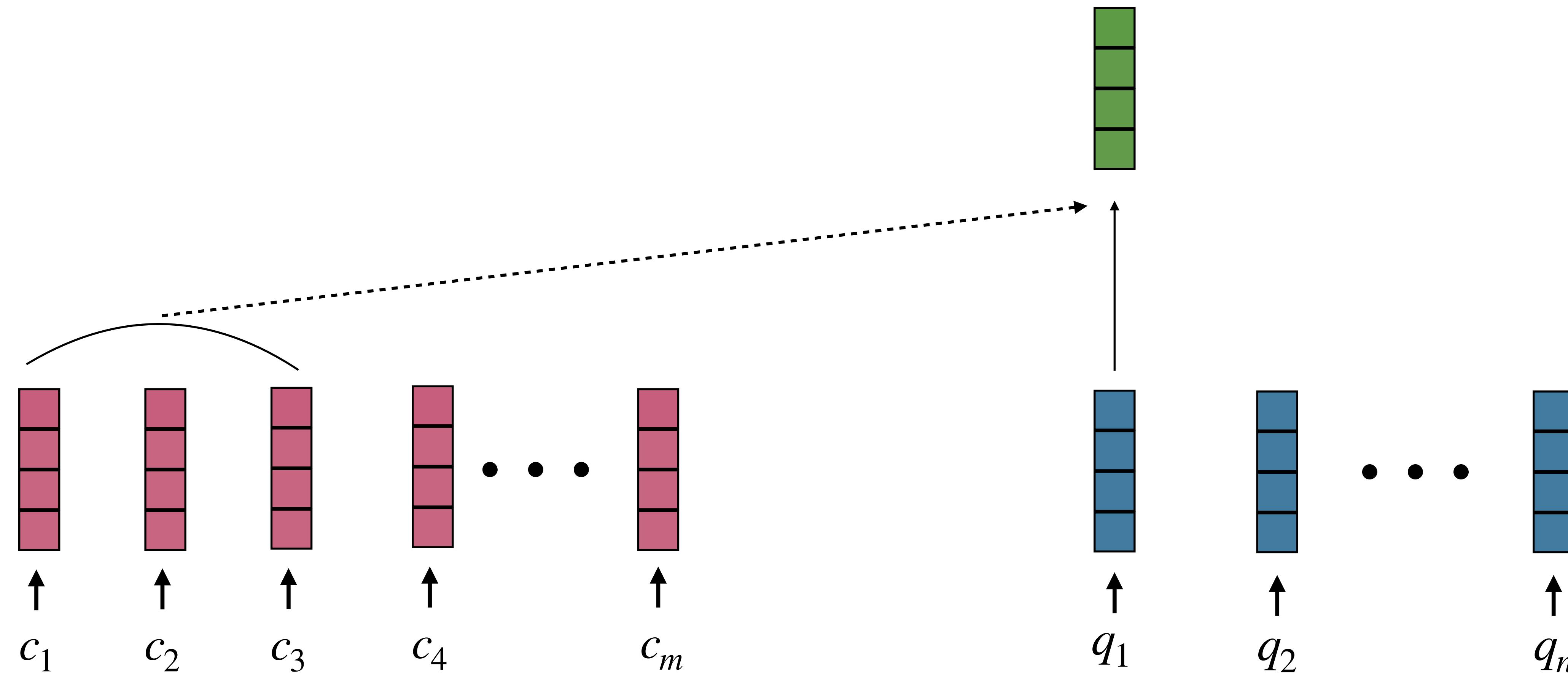
Previous Work: Efficient Attention

- **Sparse Attention**

- Local attention
 - Image Transformer (Parmar et al., 2018)
- Stride attention
 - Sparse Transformer (Child et al., 2019)
 - Longformer (Beltagy et al., 2020)
- Attention with learnable patterns
 - Reformer (Kitaev et al., 2020)
 - Sinkhorn attention (Tay et al., 2020)

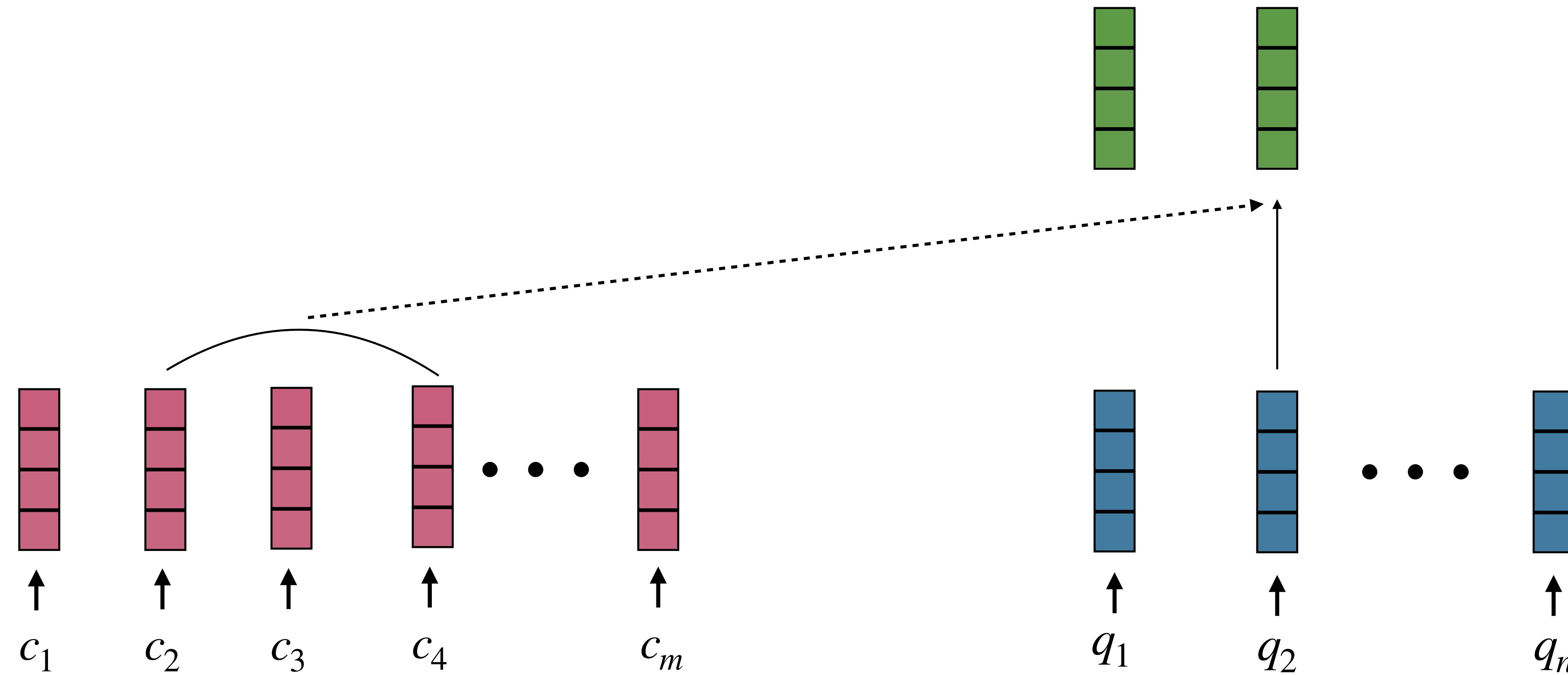
LongFormer

- Sliding Window Attention

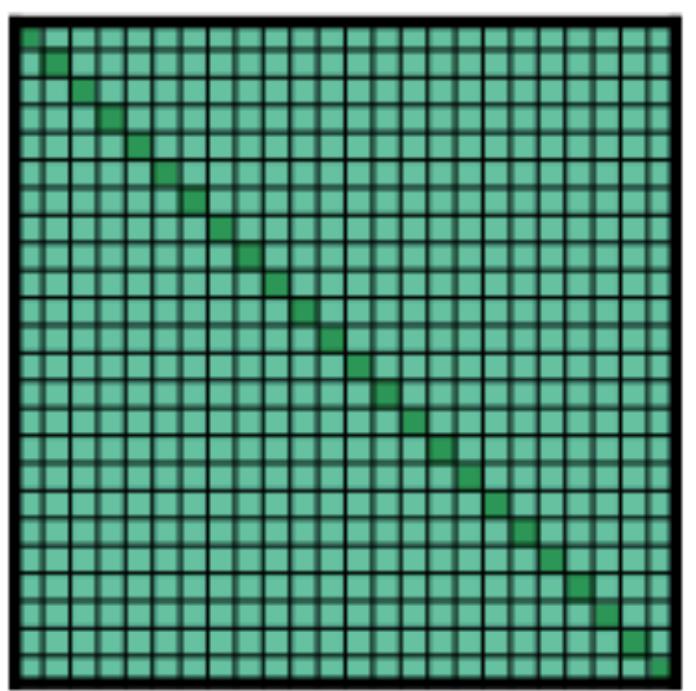


LongFormer

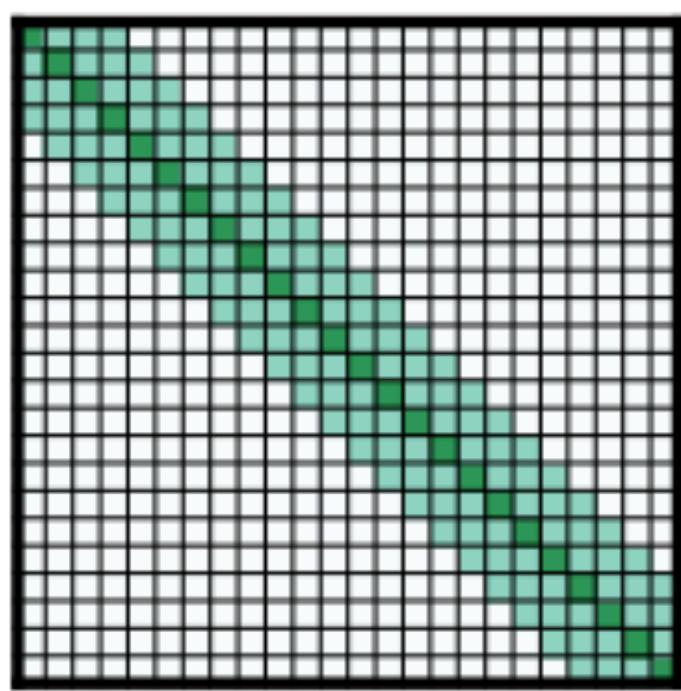
- Sliding Window Attention



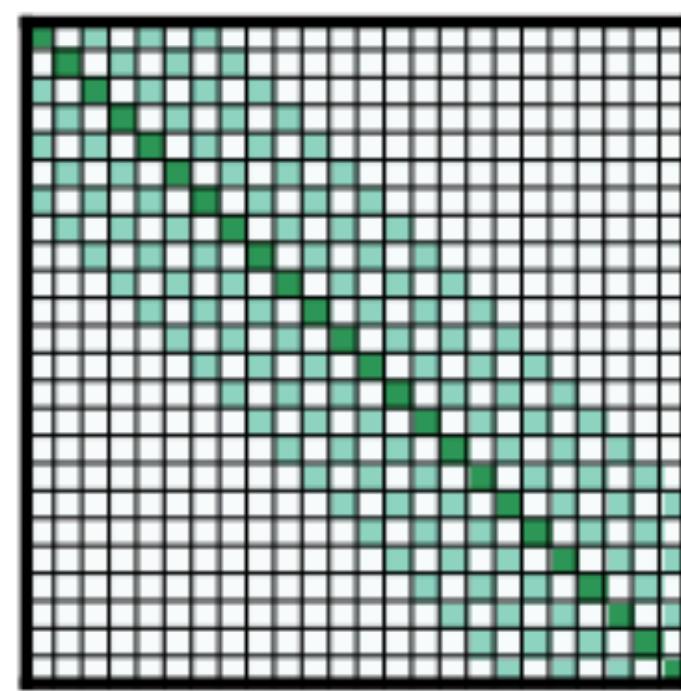
Sliding Window Attention



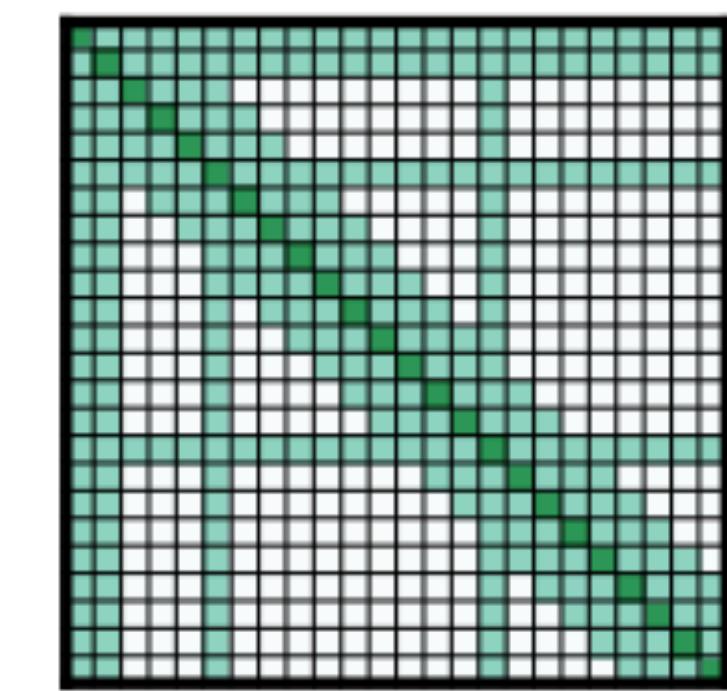
(a) Full n^2 attention



(b) Sliding window attention



(c) Dilated sliding window



(d) Global+sliding window

Implementation is NOT easy

Previous Work: Efficient Attention

- **Sparse Attention**

- Local attention
 - Image Transformer (Parmar et al., 2018)
- Stride attention
 - Sparse Transformer (Child et al., 2019)
 - Longformer (Beltagy et al., 2020)
- Attention with learnable patterns
 - Reformer (Kitaev et al., 2020)
 - Sinkhorn attention (Tay et al., 2020)

- **Linear Attention: Decoupling Attention Matrix**

- Lambda Network (Bello, et al., 2021)
- Performer (Chor, et al., 2021)
- Random Feature Attention (RFA) (Peng et al., 2021)
- Recent works: H3, RetNet, RWKV, ...

Linear Attention: Decoupling Attention Matrix

$$\sigma \left(\frac{QK^T}{\sqrt{d}} \right) V = \underbrace{A}_{n \times m} \times \underbrace{V}_{m \times d}$$

$$\approx \underbrace{(Q')}_{n \times k} \times \underbrace{(K')^T}_{k \times m} \times \underbrace{V}_{m \times d}$$

$$\approx \underbrace{Q'}_{n \times k} \times \left(\underbrace{(K')^T}_{k \times m} \times \underbrace{V}_{m \times d} \right)$$

$$k \ll n, m$$

Decoupling Attention Matrix

- Lambda Network (Bello, 2021)

$$Q' = Q, \quad (K')^T = \sigma\left(\frac{K^T}{\sqrt{d}}\right)$$

Decoupling Attention Matrix

- Kernel Methods (Chor, et al., 2021, Peng et al., 2021)

$$Q' = \phi(Q), \quad (K')^T = \phi(K^T)$$

$$\phi(x) = \frac{1}{\sqrt{k}} [\sin(w_1^T x), \dots, \sin(w_k^T x), \cos(w_1^T x), \dots, \cos(w_k^T x)]$$

$$w_i \sim \mathcal{N}(0, \sigma^2 I)$$

$$\mathbb{E}_W [\phi(x)^T \phi(y)] = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

Previous Work: Efficient Attention

- **Sparse Attention**

- Local attention
 - Image Transformer (Parmar et al., 2018)
- Stride attention
 - Sparse Transformer (Child et al., 2019)
 - Longformer (Beltagy et al., 2020)
- Attention with learnable patterns
 - Reformer (Kitaev et al., 2020)
 - Sinkhorn attention (Tay et al., 2020)

- **Decoupling Attention Matrix**

- Lambda Network (Bello, et al., 2021)
- Performer (Chor, et al., 2021)
- Random Feature Attention (RFA) (Peng et al., 2021)

- **Low-rank Approximation**

- Linformer (Wang et al., 2020)
- Luna (Ma et al., 2021)

Low-Rank Approximation

- Motivation: projecting context C into a shorter length

Linear Attention (Linformer, Wang et al., 2020)

$$\sigma \left(\frac{Q(E\mathbf{C})^T}{\sqrt{d}} \right) (\mathbf{F}\mathbf{C})$$

$E, F \in \mathbb{R}^{l \times m}$
learnable parameters

$n \times l$ $l \times d$

$l \ll n, m$

Problems:

- Cannot model sequences with **various lengths**
- E and F do **NOT** contain **contextual information**

Linear Nested Attention: Pack & Unpack

- Motivation: first packing context C into a shorter length, then unpacking with query Q

Linear Attention

$$\sigma \left(\frac{Q(EC)^T}{\sqrt{d}} \right)$$

A nested attention

$n \times l$ $l \times d$

Pack Attention

$$C' = \sigma \left(\frac{PC^T}{\sqrt{d}} \right) C$$

$l \times m$ $m \times d$

$P \in \mathbb{R}^{l \times d}$
learnable parameters

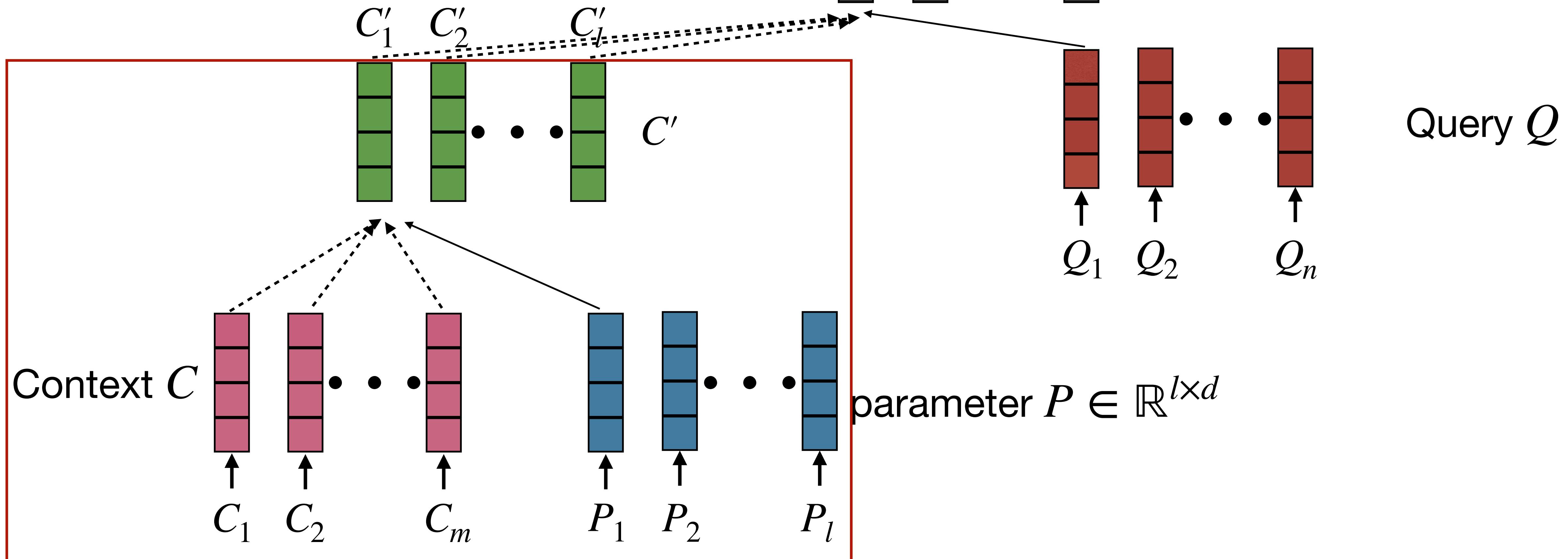
Unpack Attention

$$O = \sigma \left(\frac{Q(C')^T}{\sqrt{d}} \right) C'$$

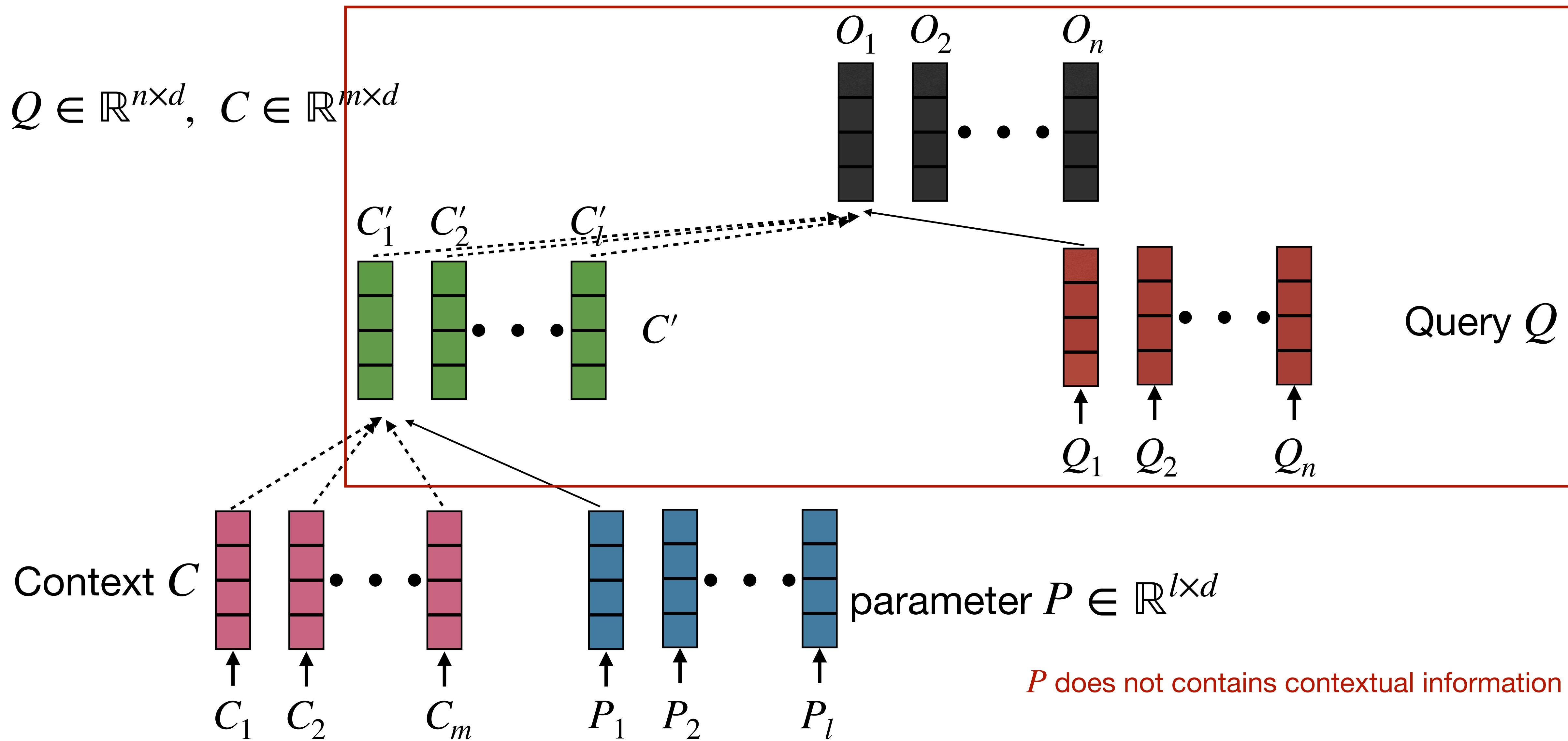
$n \times l$ $l \times d$

Linear Nested Attention: Pack & Unpack

$$Q \in \mathbb{R}^{n \times d}, C \in \mathbb{R}^{m \times d}$$



Linear Nested Attention: Pack & Unpack



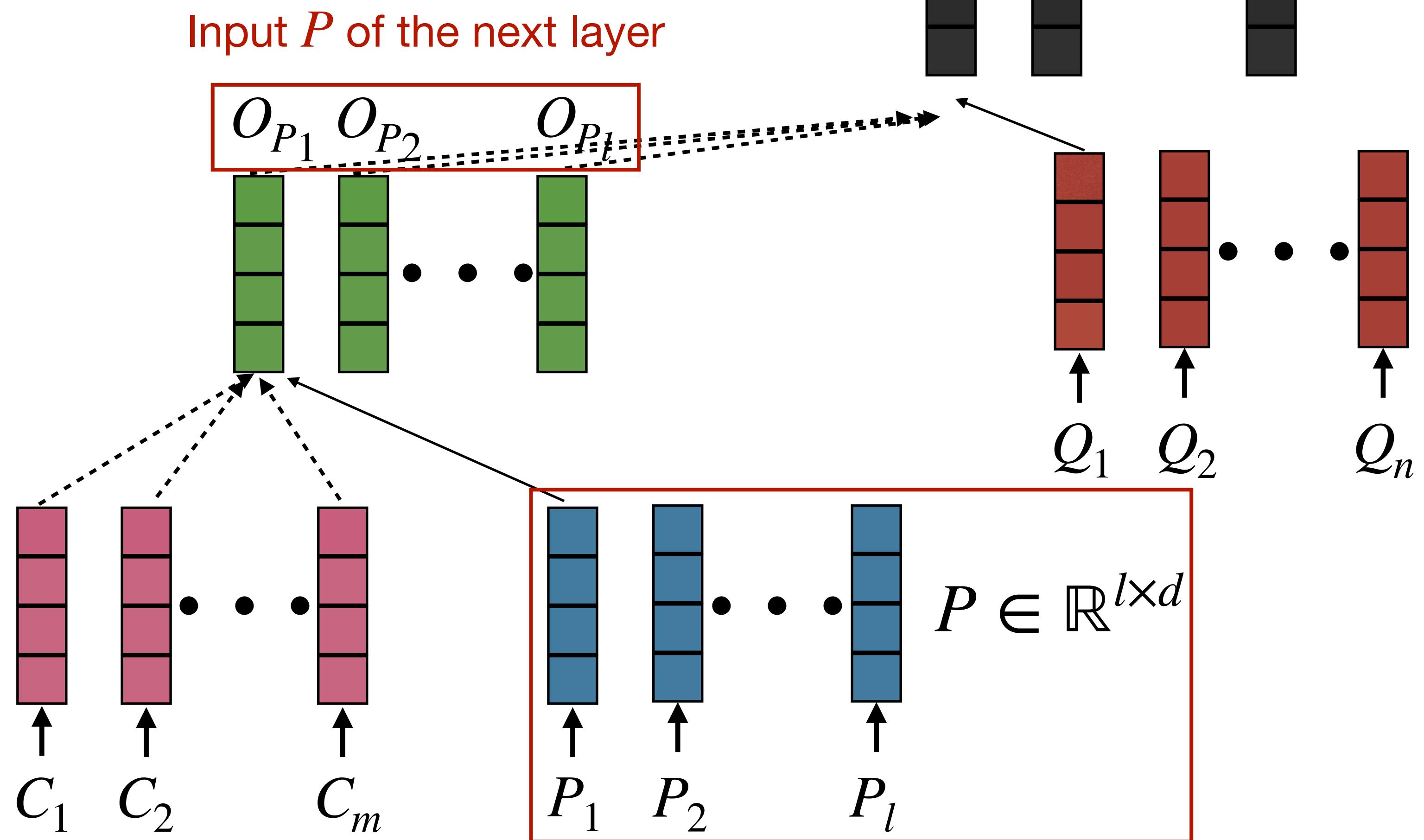
Luna: Linear Unified Nested Attention (Ma, et al., 2021)

Inputs: $Q \in \mathbb{R}^{n \times d}$, $C \in \mathbb{R}^{m \times d}$, $P \in \mathbb{R}^{l \times d}$

Outputs:

$$O_P = C' = \sigma \left(\frac{PC^T}{\sqrt{d}} \right) C$$

$$O_Q = \sigma \left(\frac{Q(C')^T}{\sqrt{d}} \right) C'$$



LRA: Long Range Arena

- A benchmark of 6 tasks for long-range sequence modeling

- Intentionally designed to be challenging

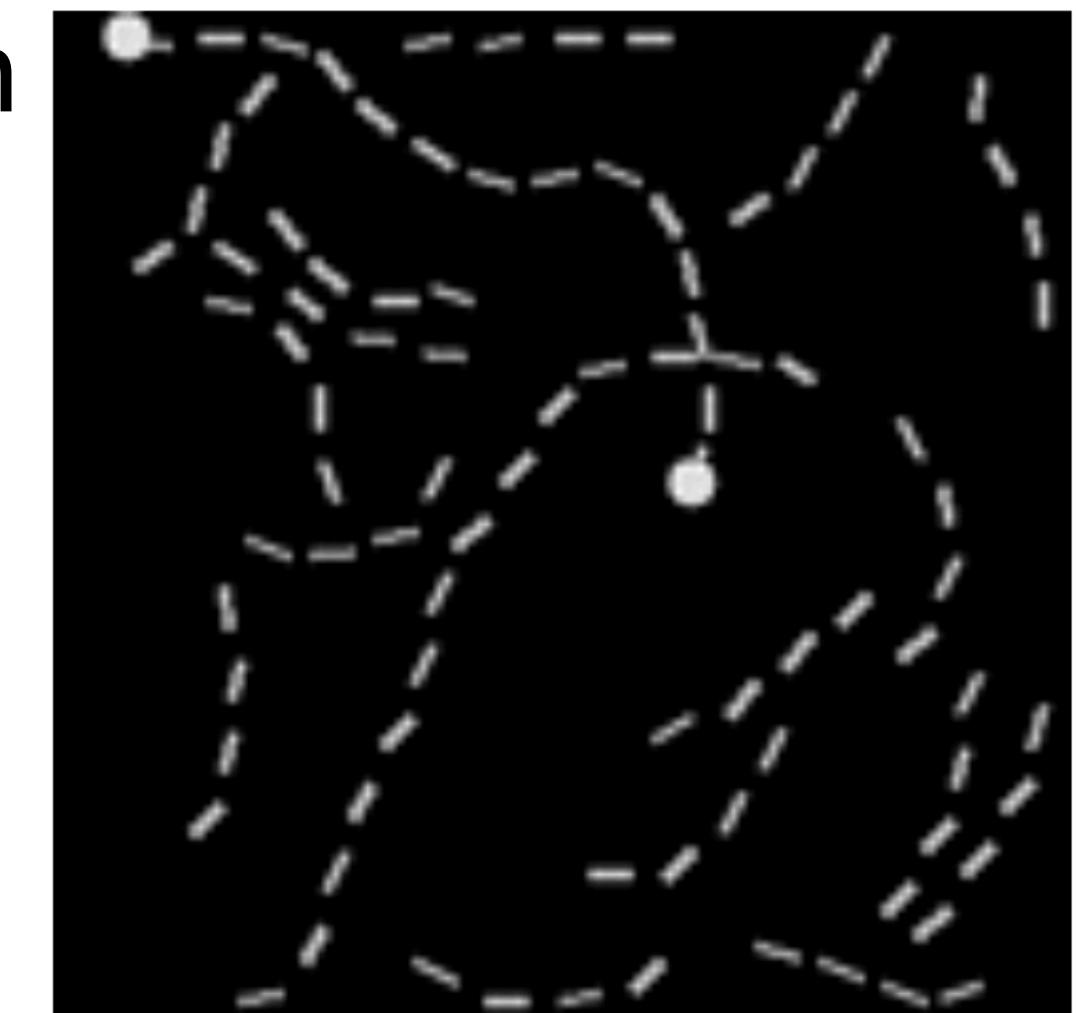
- Tasks

- Long ListOpts:

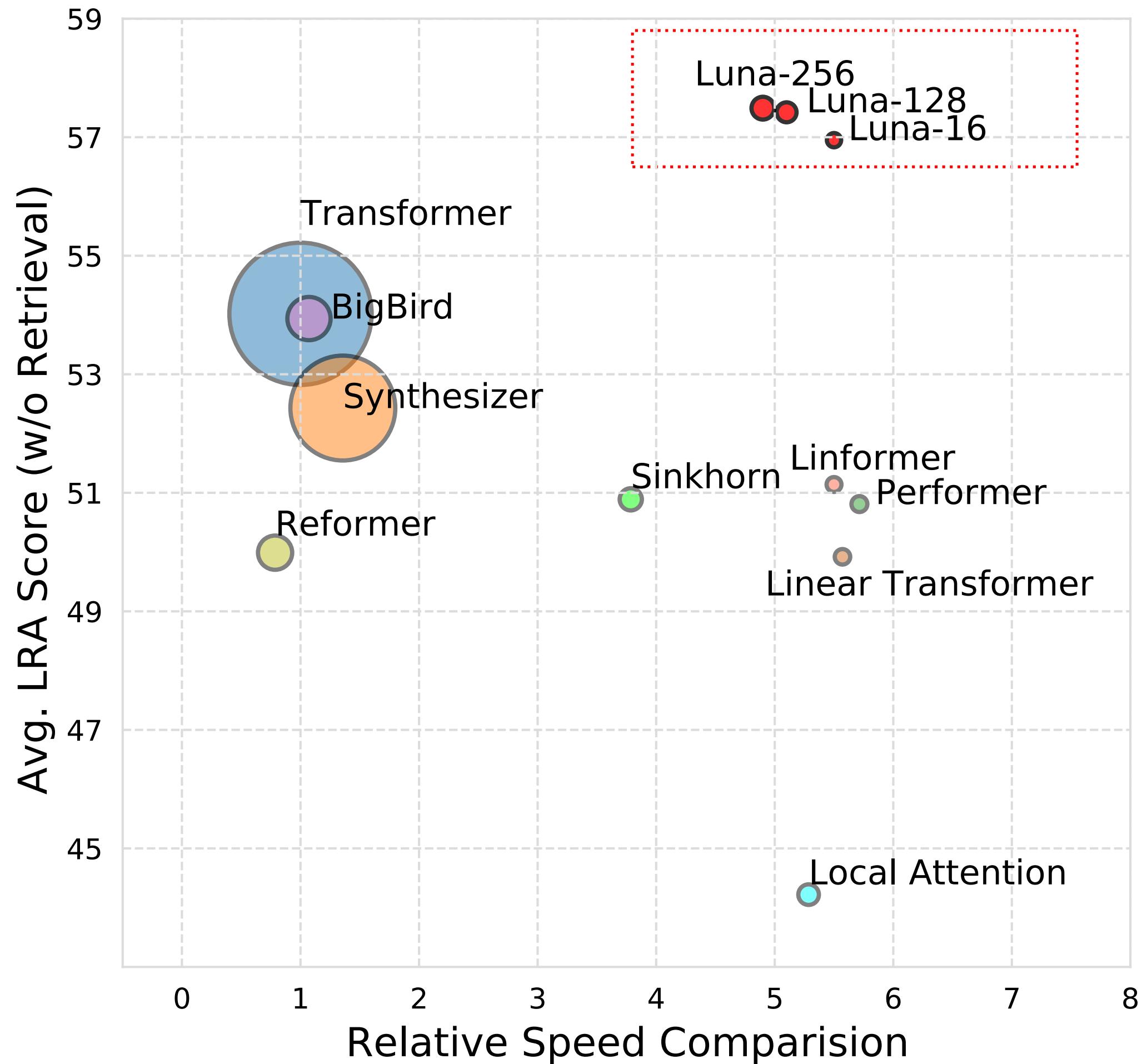
INPUT: [MAX 4 3 [MIN 2 3] 1 0 [MEDIAN 1 5 8 9, 2]]

OUTPUT: 5

- Byte-level text classification on IMDB (> 4k tokens)
 - Byte-level document retrieval on ACL Anthology Network (ANN) (> 2k tokens)
 - Pixel-level image classification on CIFAR-10 ($32 \times 32 = 1024$ token)
 - PathFinder ($32 \times 32 = 1024$ tokens)
 - Path-X (Extreme Length: $128 \times 128 = 16,384$ tokens)

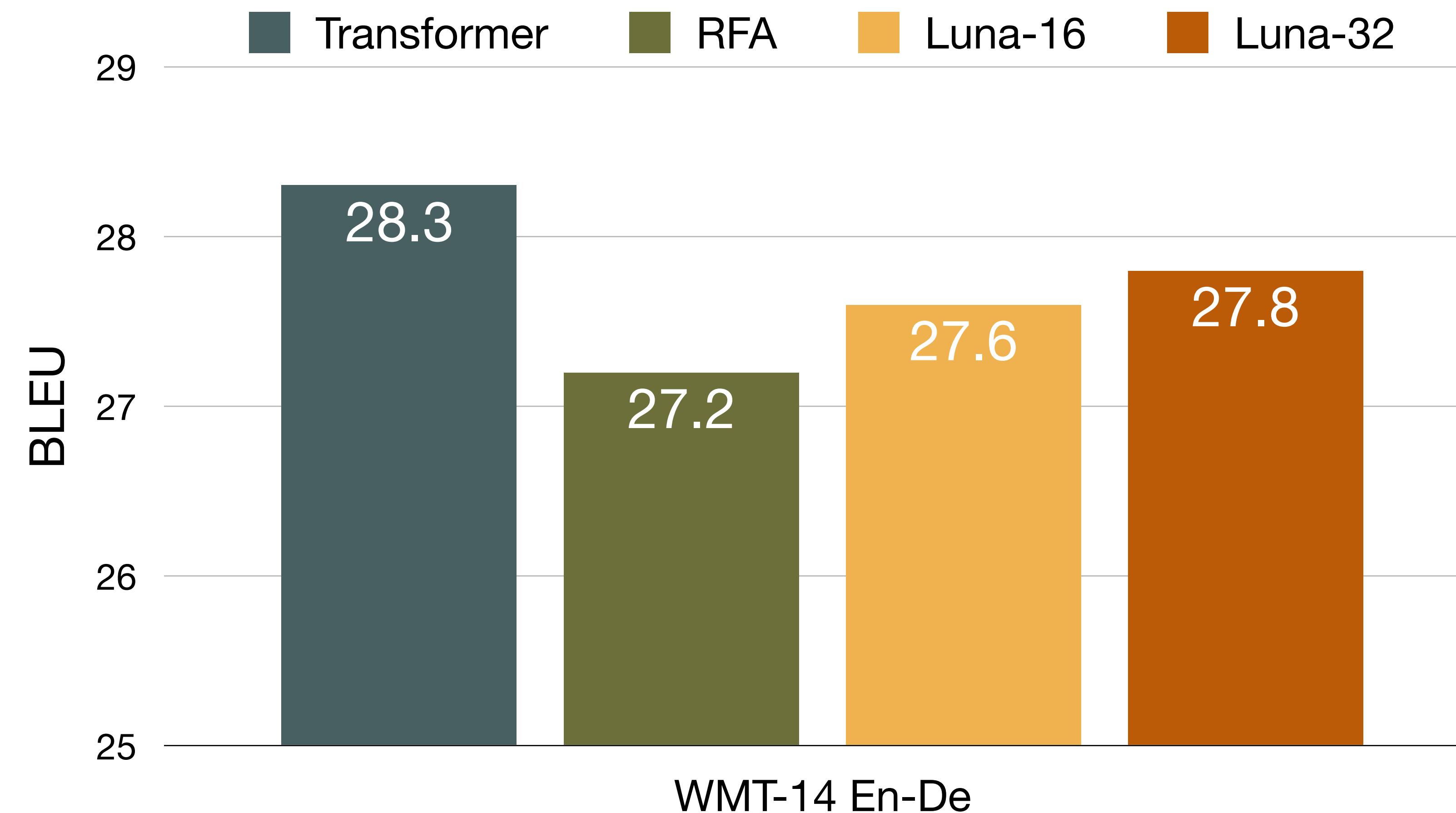


Performance on LRA

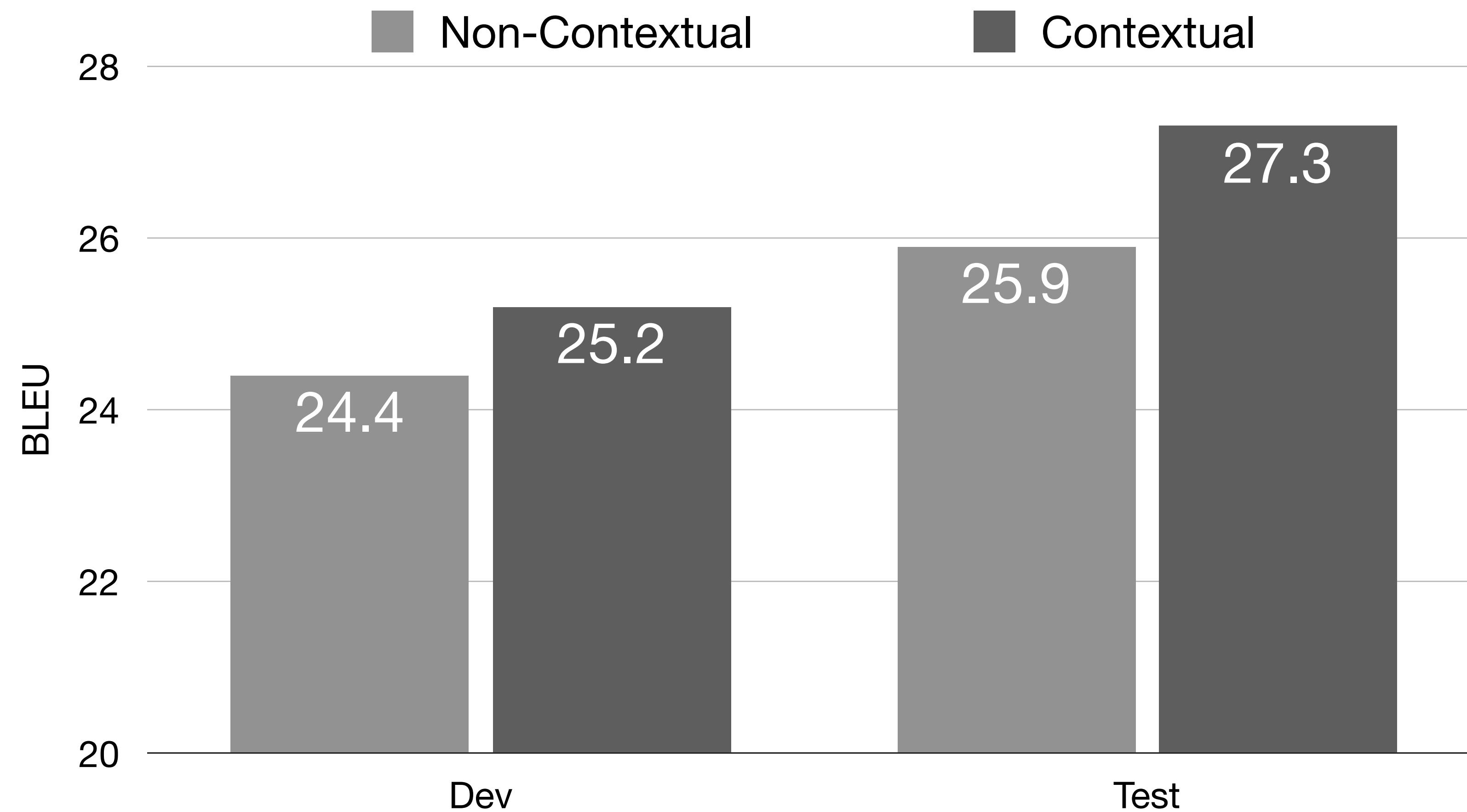


Trade-off between accuracy (**y-axis**), speed (**x-axis**)
and memory (**circle-radius**)

Results: Machine Translation



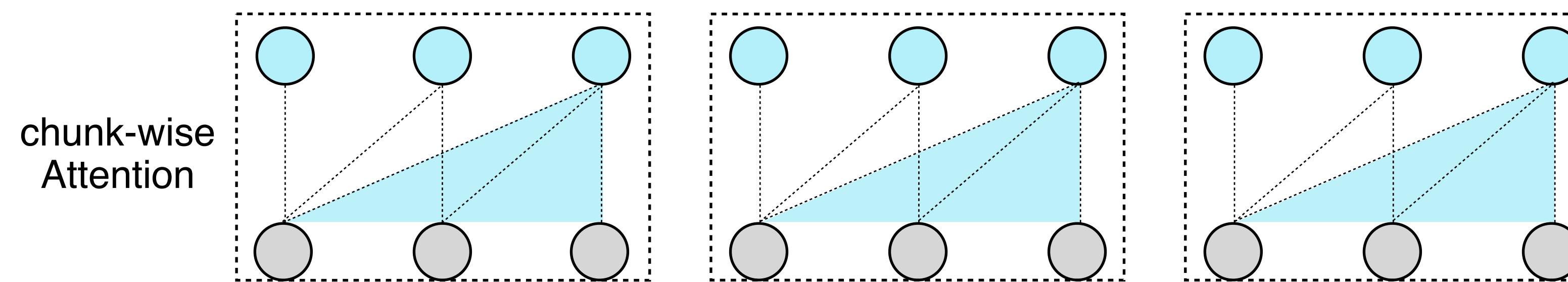
Effect of Contextual P



All models were optimized with Apollo (Ma 2020)

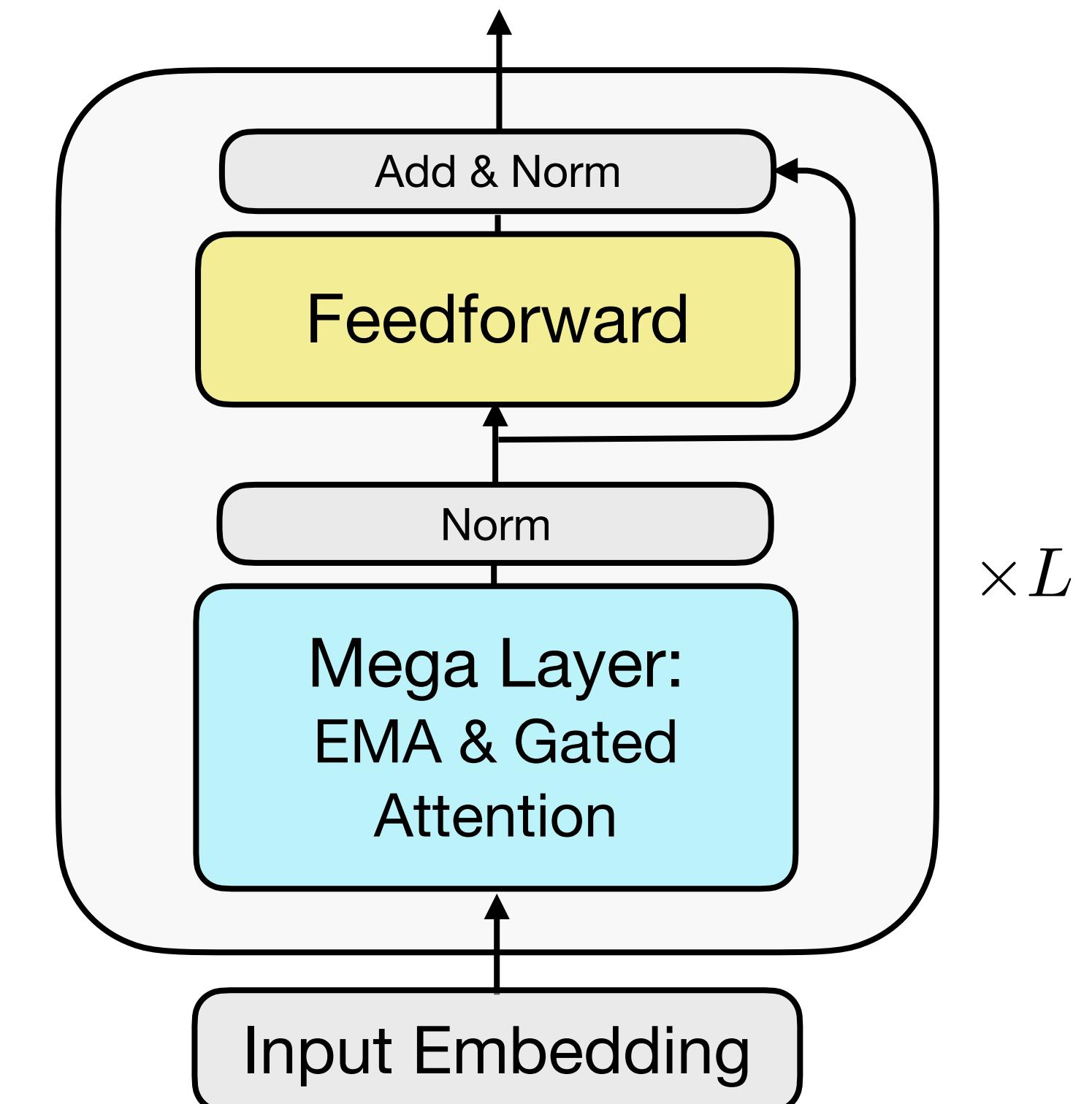
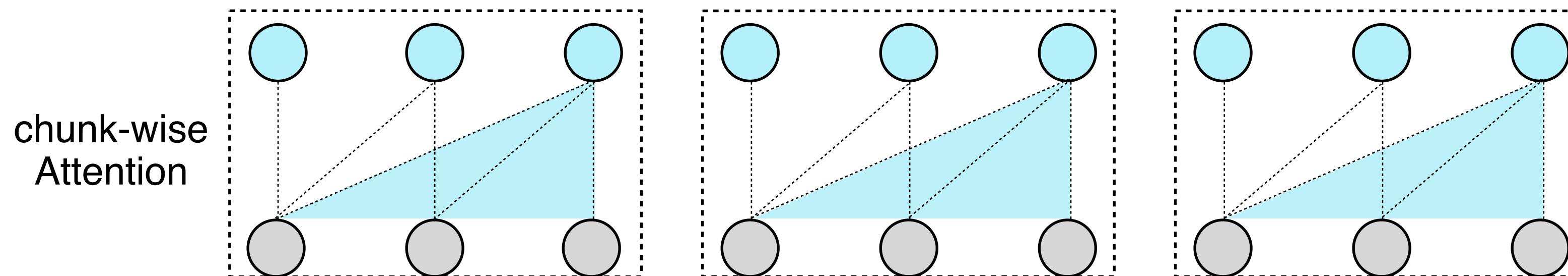
Why not Chunk-wise Attention?

- Applying attention individually to each chunk with fixed length



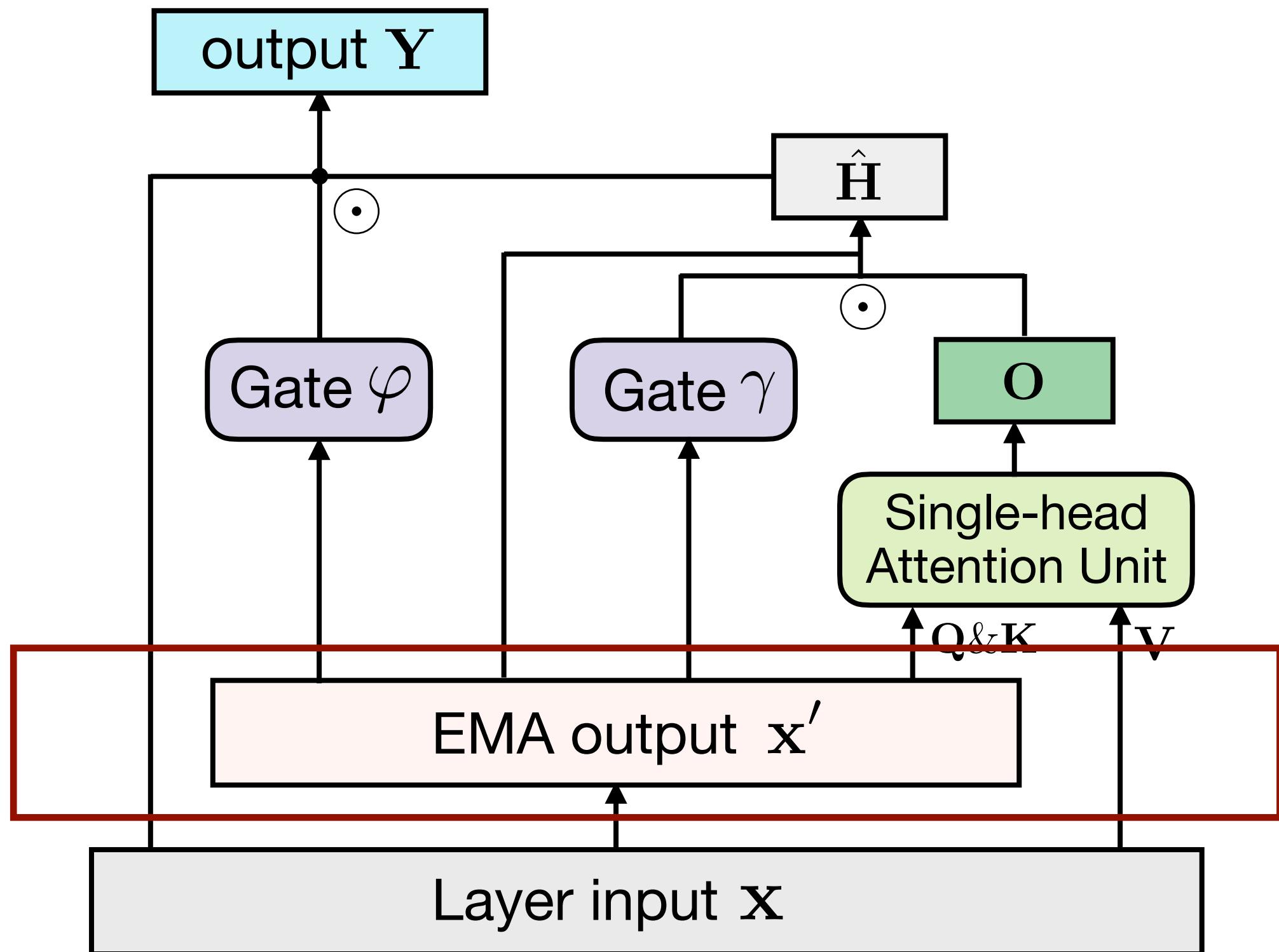
Mega: Executive Summary

- Effective and efficient drop-in replacement of attention for long sequence modeling
 - Outstanding results on various types of data
 - text, images and audios
 - Exponential Moving Average (EMA)
 - Mega-chunk: linear complexity of time and space



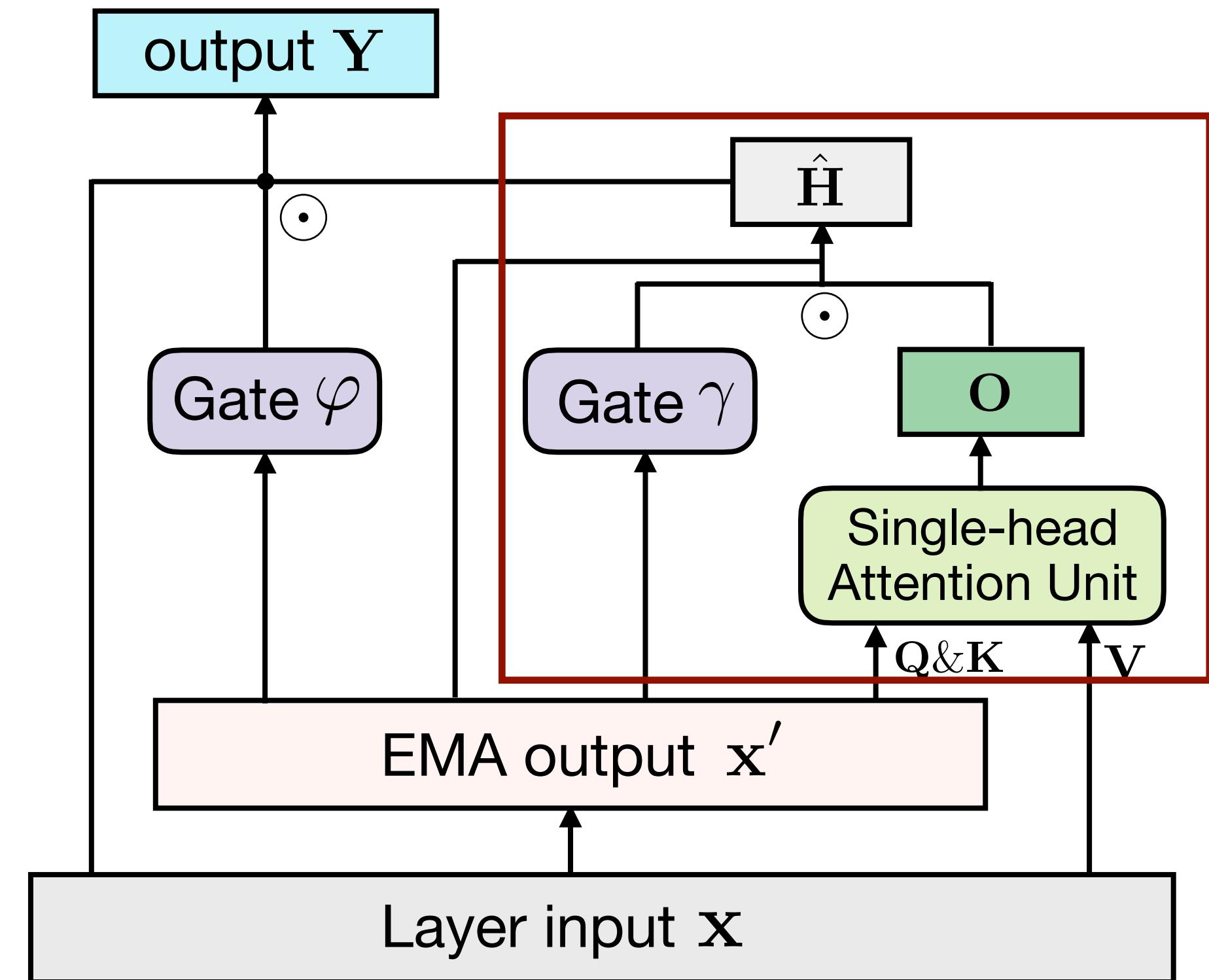
Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time



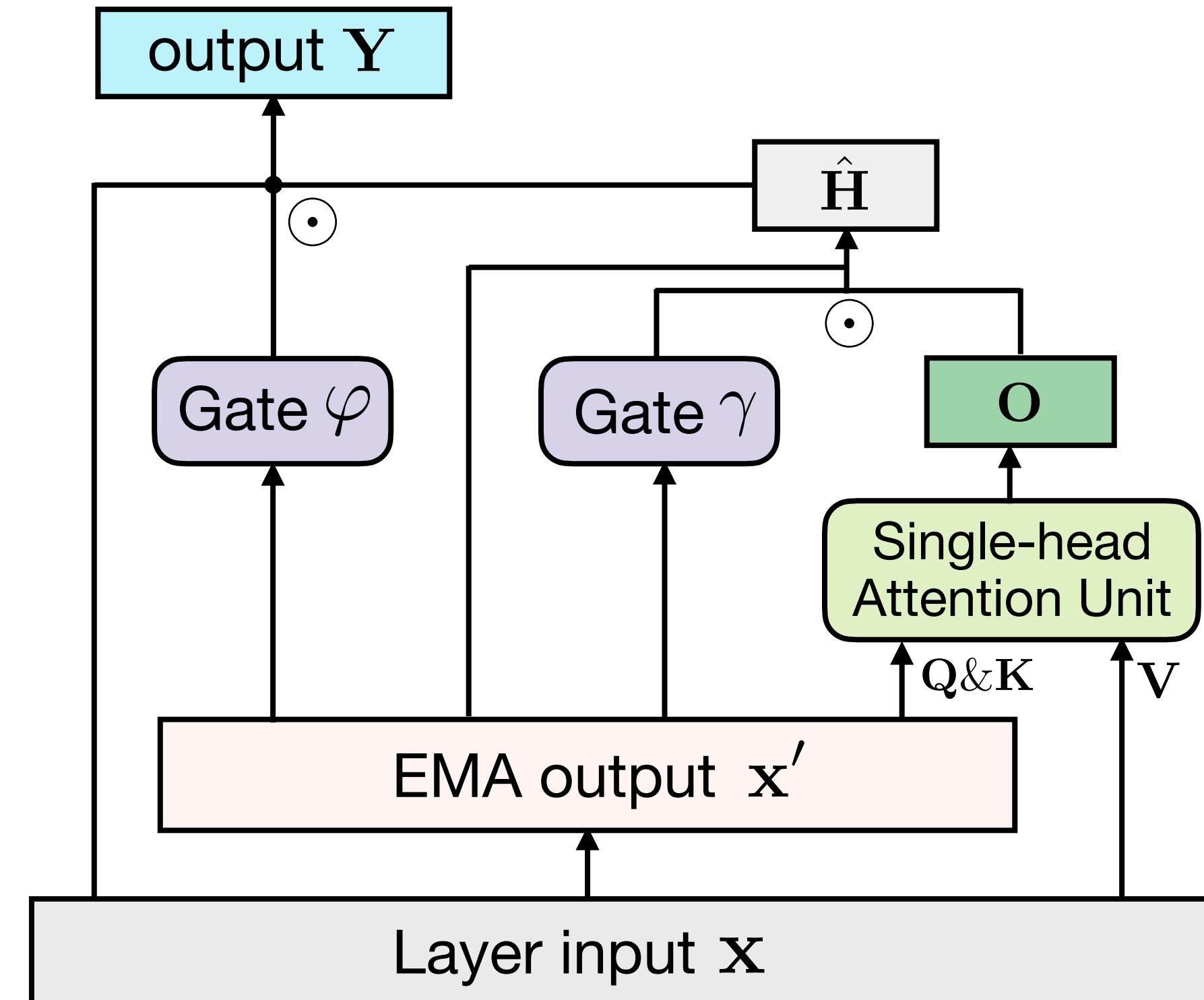
Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time
- **Single-head Gated Attention**
 - Adding a reset gate to the attention output
 - Theoretically proving that **single-head** gated attention is as expressive as **multi-head** one



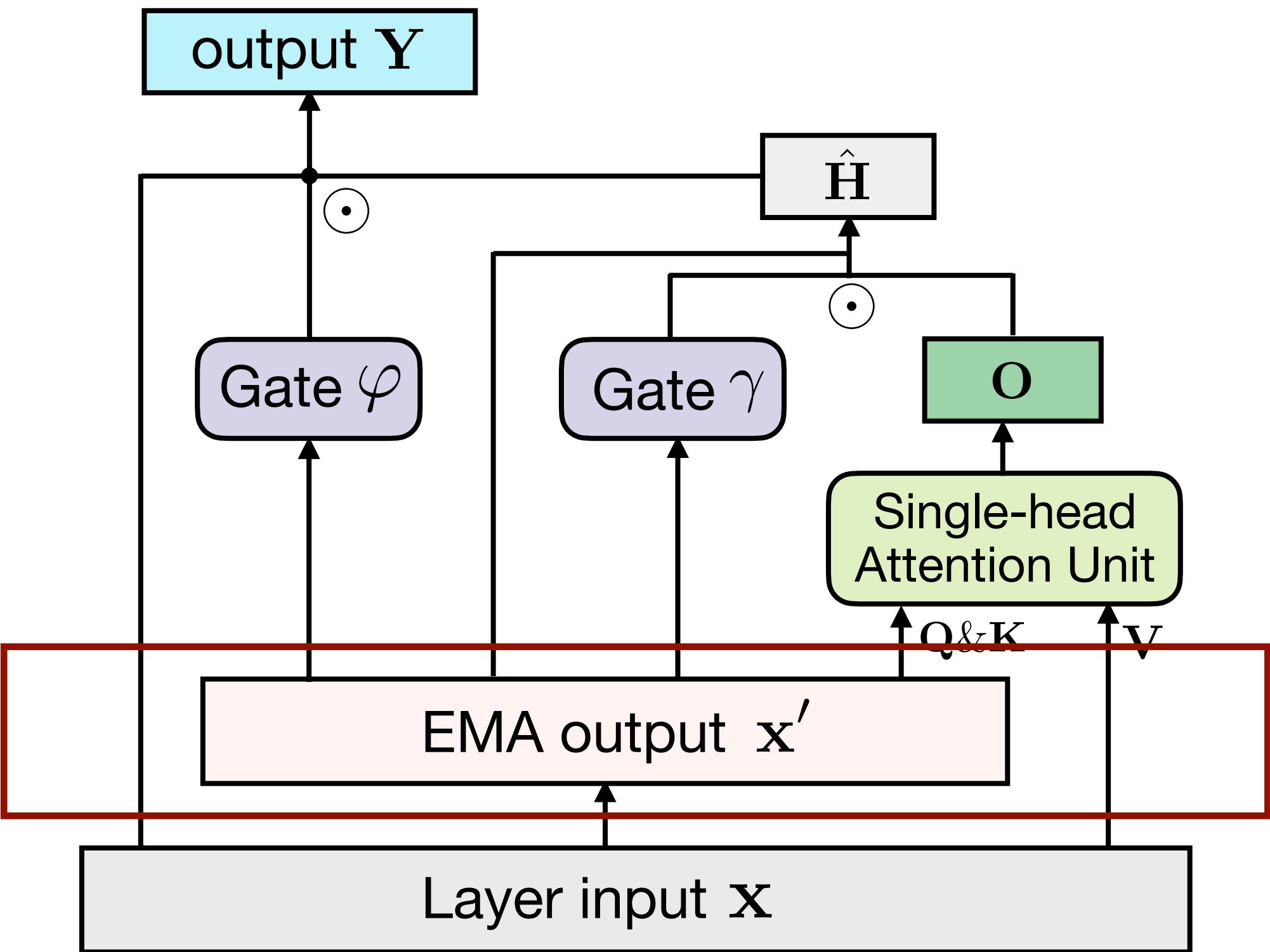
Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time
- **Single-head Gated Attention**
 - Adding a reset gate to the attention output
 - Theoretically proving that **single-head** gated attention is as expressive as **multi-head** one
- **Mega-Chunk**
 - Applying attention to local chunks of fixed length
 - Reducing quadratic complexity to linear

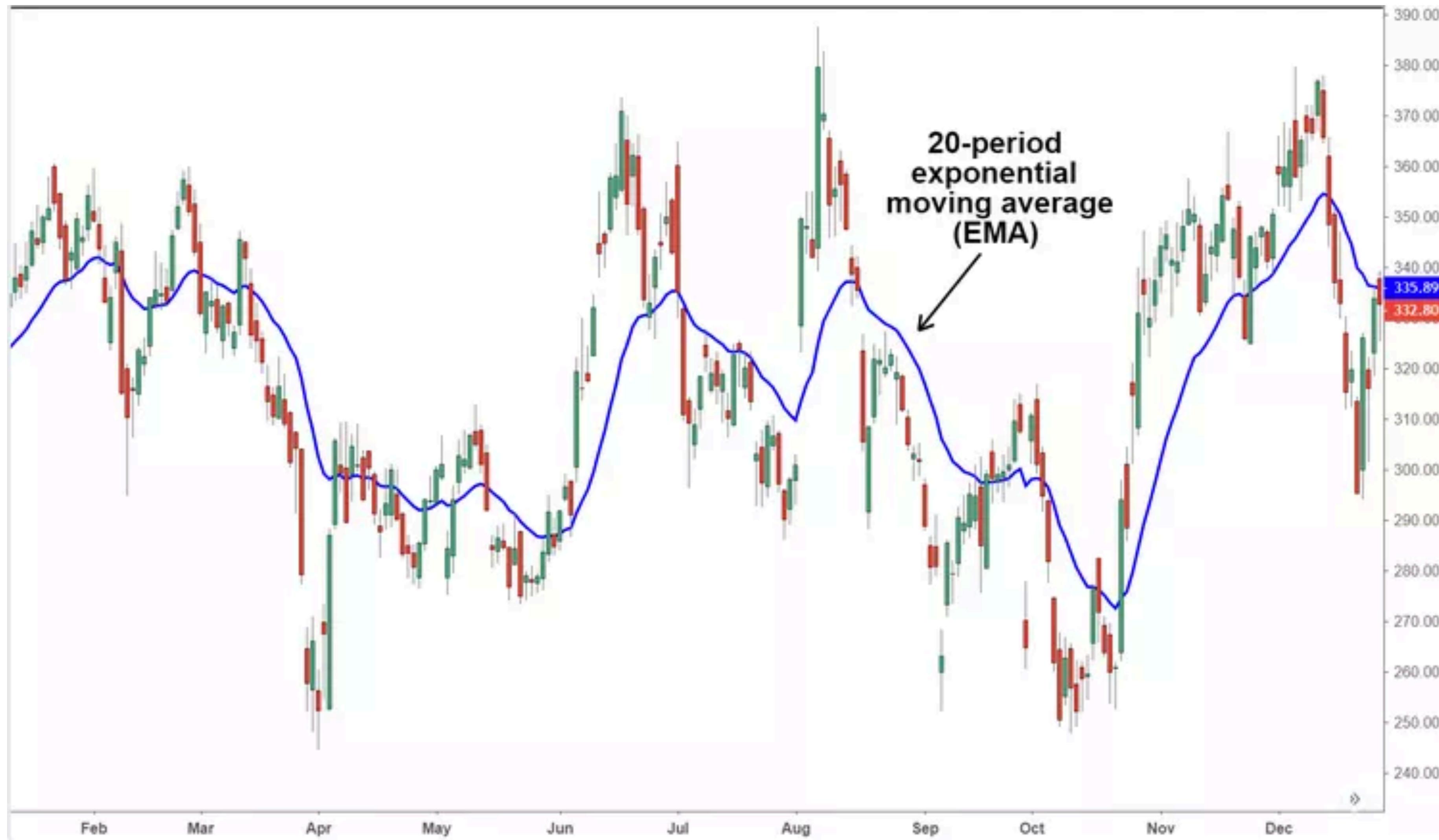


Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time
- **Single-head Gated Attention**
 - Adding a reset gate to the attention output
 - Theoretically proving that **single-head** gated attention is as expressive as multi-head one
- **Mega-Chunk**
 - Applying attention to local chunks of fixed length
 - Reducing quadratic complexity to linear



Exponential Moving Average (EMA)



Exponential Moving Average (EMA)

Notations: Assuming 1-dim input sequence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, $\mathbf{x}_t \in \mathbb{R}$

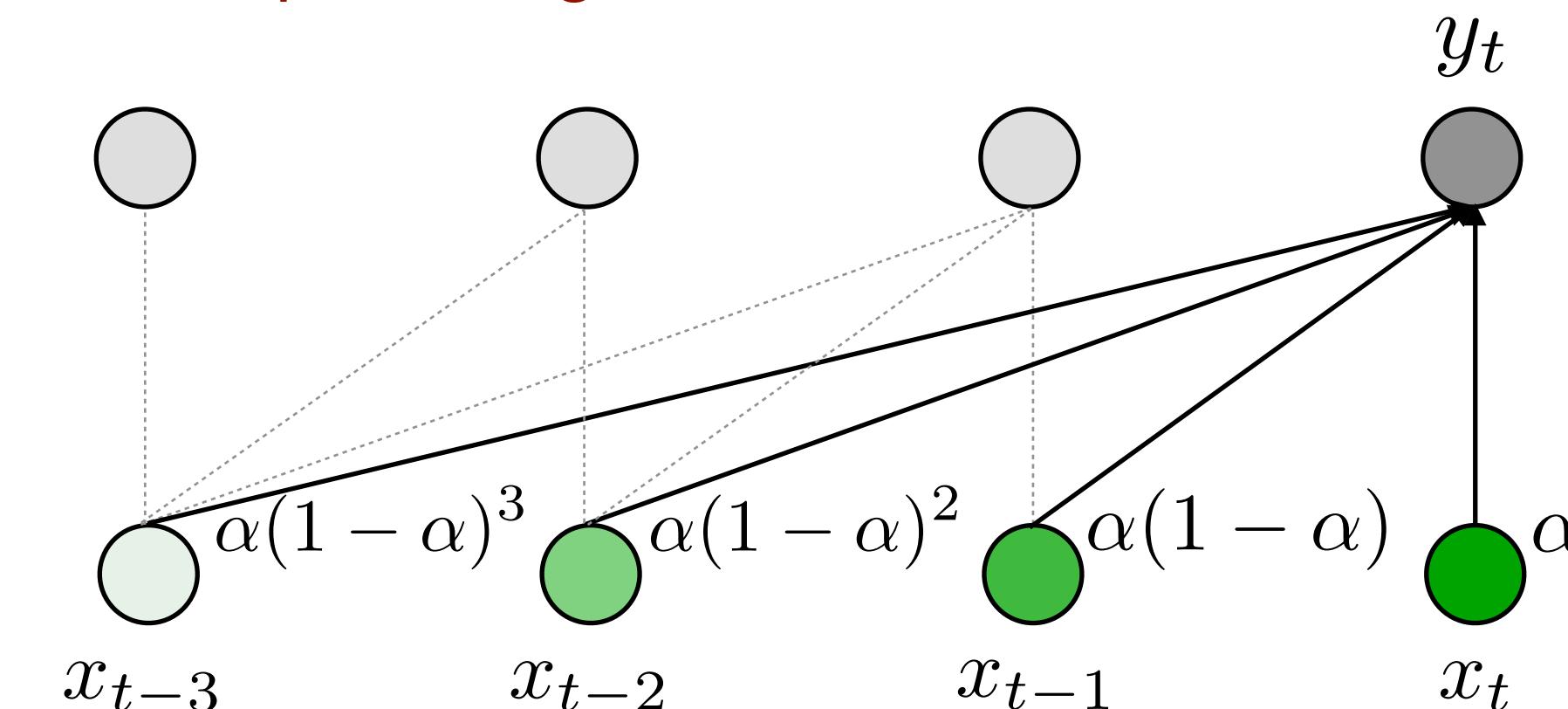
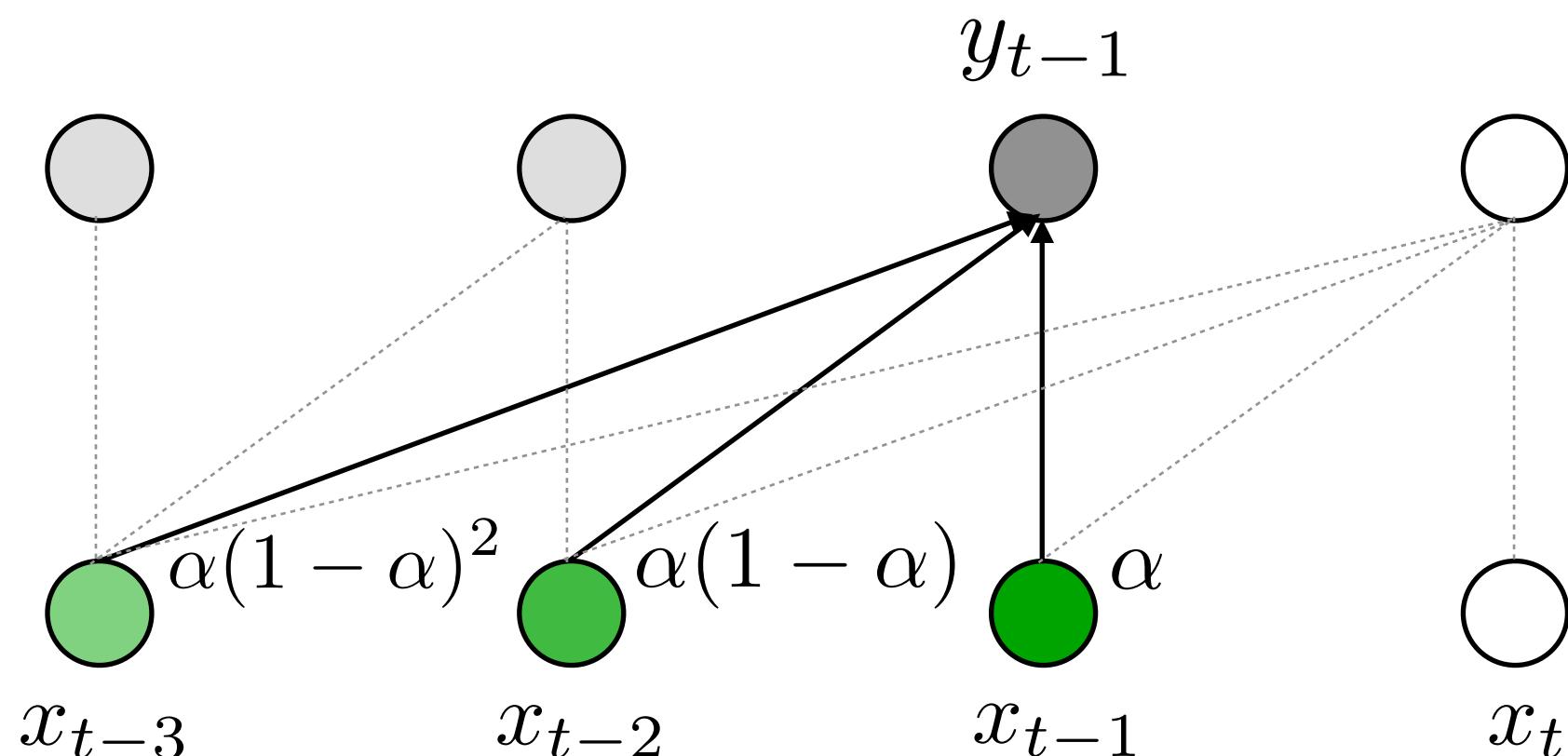
EMA

$$\mathbf{y}_t = \boxed{\alpha} \odot \mathbf{x}_t + \boxed{(1 - \alpha)} \odot \mathbf{y}_{t-1}, \quad \alpha \in (0, 1)$$

Damped EMA

$$\mathbf{y}_t = \alpha \odot \mathbf{x}_t + (1 - \alpha \odot \delta) \odot \mathbf{y}_{t-1}, \quad \delta \in (0, 1)$$

Relaxing the coupled weights



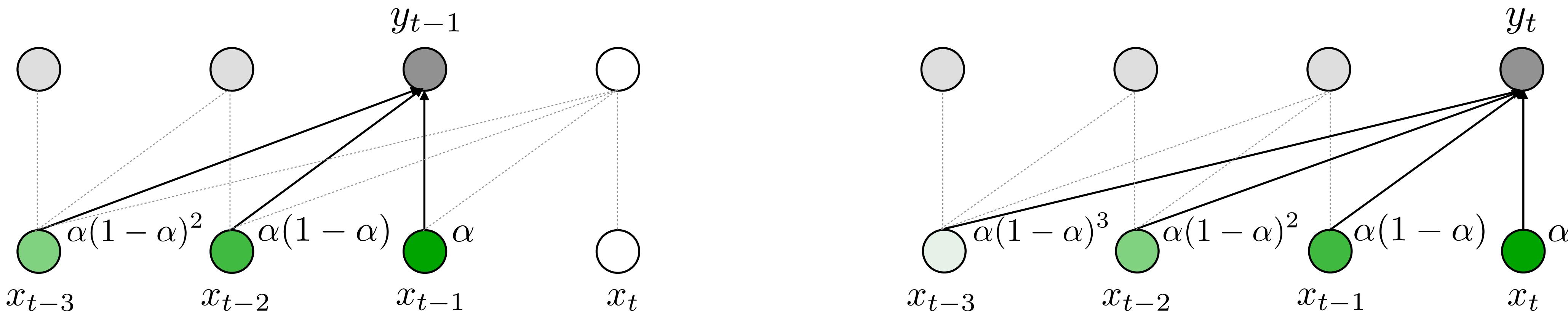
Learnable
Parameters

Efficient Algorithm for EMA

- Efficiently compute EMA outputs of all tokens in parallel

$$\mathbf{y}_t = \boxed{\alpha} \odot \mathbf{x}_t + \boxed{(1 - \alpha \odot \delta)} \odot \mathbf{y}_{t-1}$$

EMA weights are input independent



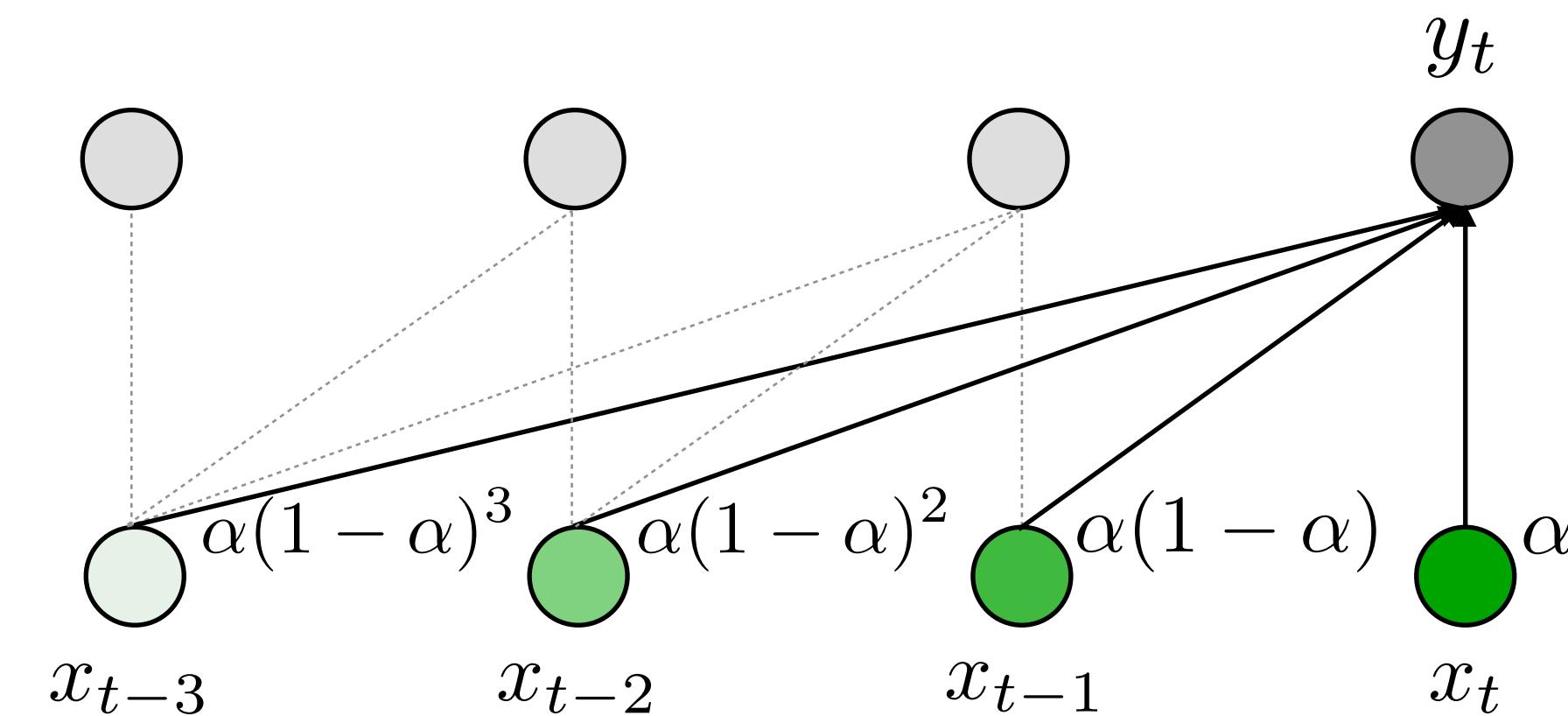
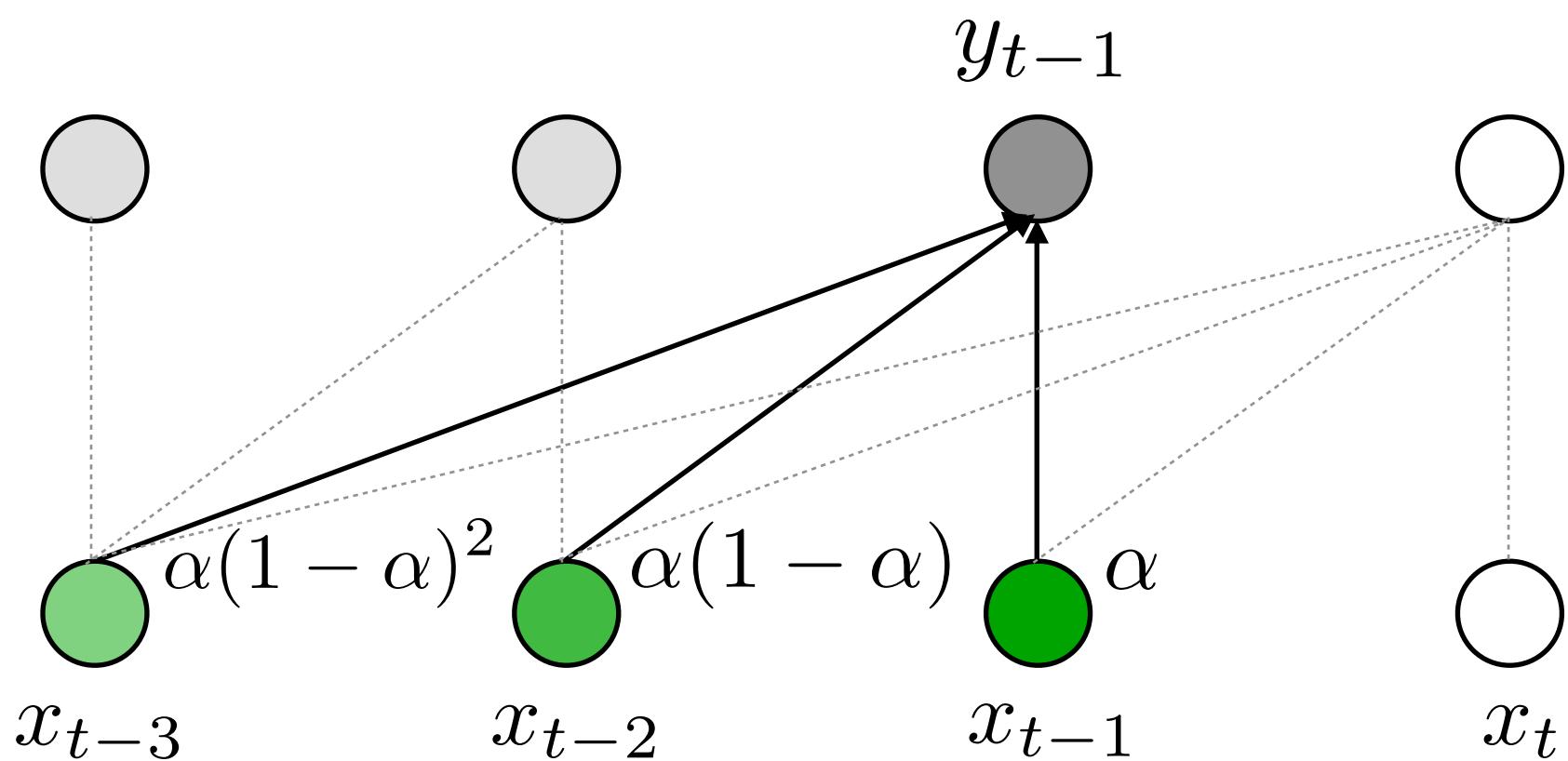
We can compute the weights for each input tokens in advance and compute EMA with FFTs.

Extending EMA to Multi-dimensional Input

- How to apply EMA to vector input $\mathbf{X}_t \in \mathbb{R}^d$
 - Stacking d individual EMA layers
 - Dimension-independent: no interactions across different dimensions
 - Implementation via vectorization

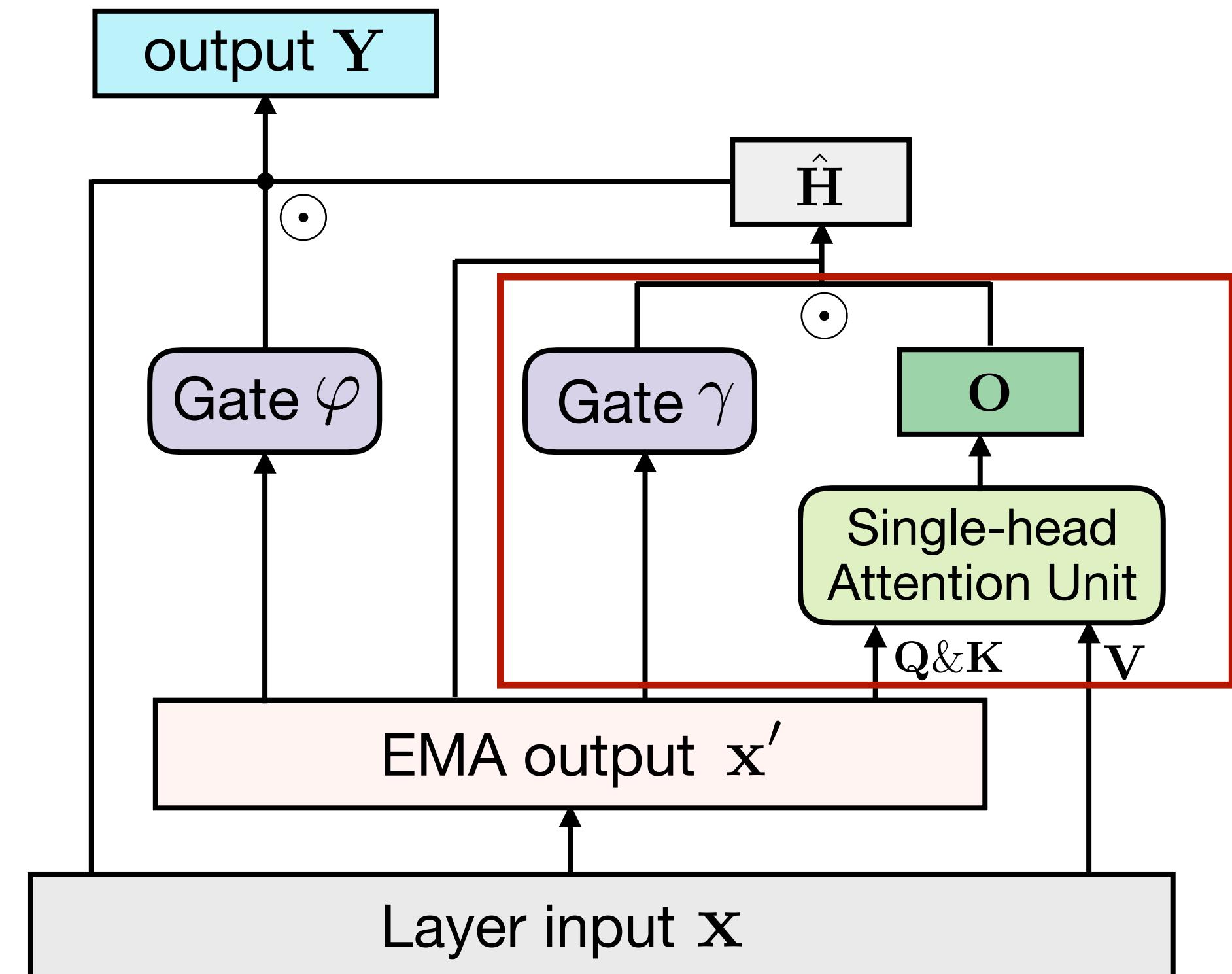
EMA: Summary

- **Strong Inductive Bias**
 - Favors recent, local contexts
 - In principle, can peek **unbounded** context history
- **Efficient to compute and easy to learn**
 - Parallel computation
 - Less parameters & capacity
 - Extremely simplified state space model



Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time
 - Connection with State Space Models (e.g S4)
- **Single-head Gated Attention**
 - Adding a reset gate to the attention output
 - Theoretically proving that **single-head** gated attention is as expressive as multi-head one
- **Mega-Chunk**
 - Applying attention to local chunks of fixed length
 - Reducing quadratic complexity to linear

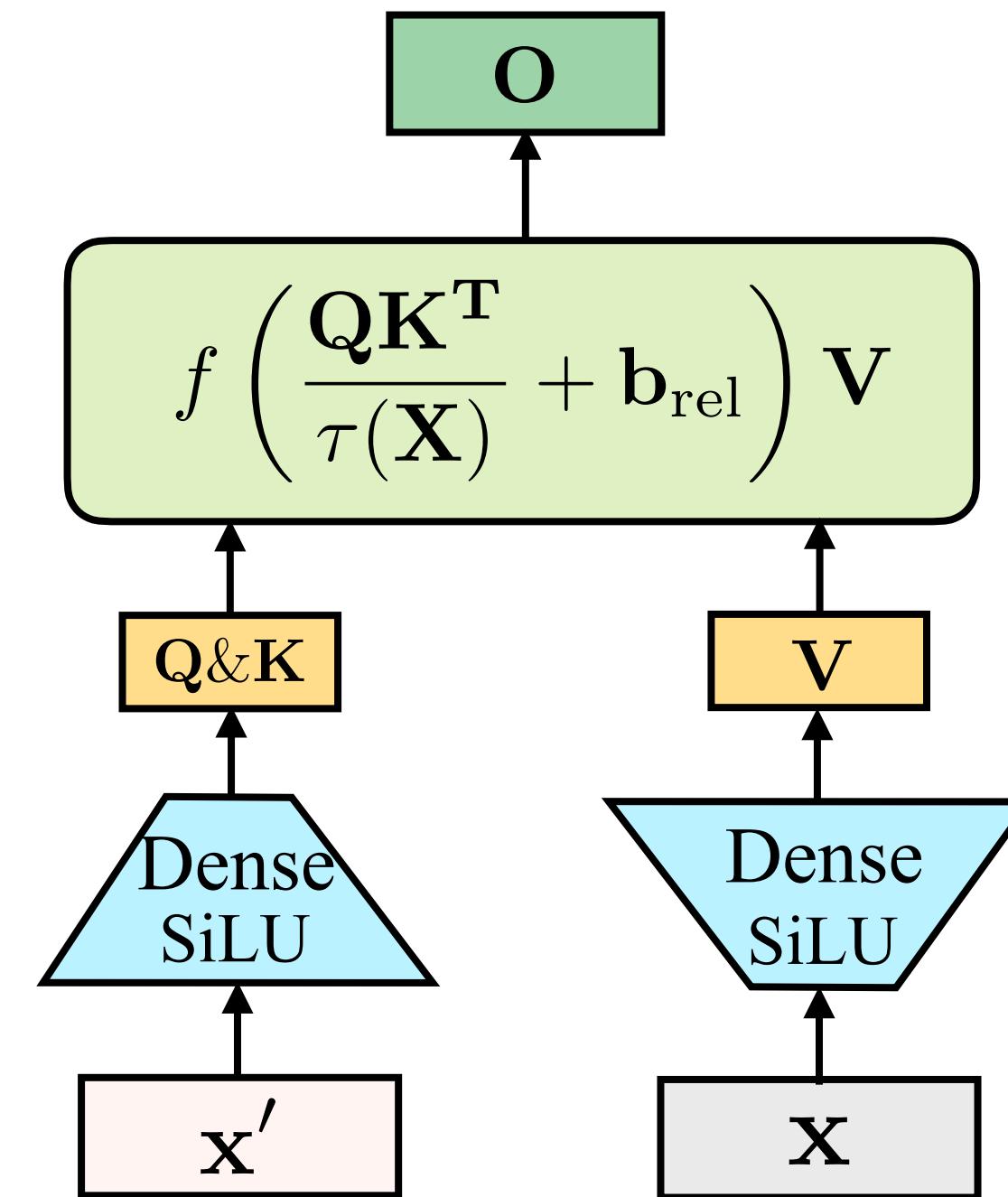


Single-head Gated Attention

- Adding a reset gate to the attention output

Attention:

$$O = \text{softmax} \left(\frac{QK^T}{\tau(X)} + b_{\text{rel}} \right) V$$



Single-head Gated Attention

- Adding a reset gate to the attention output

Attention:

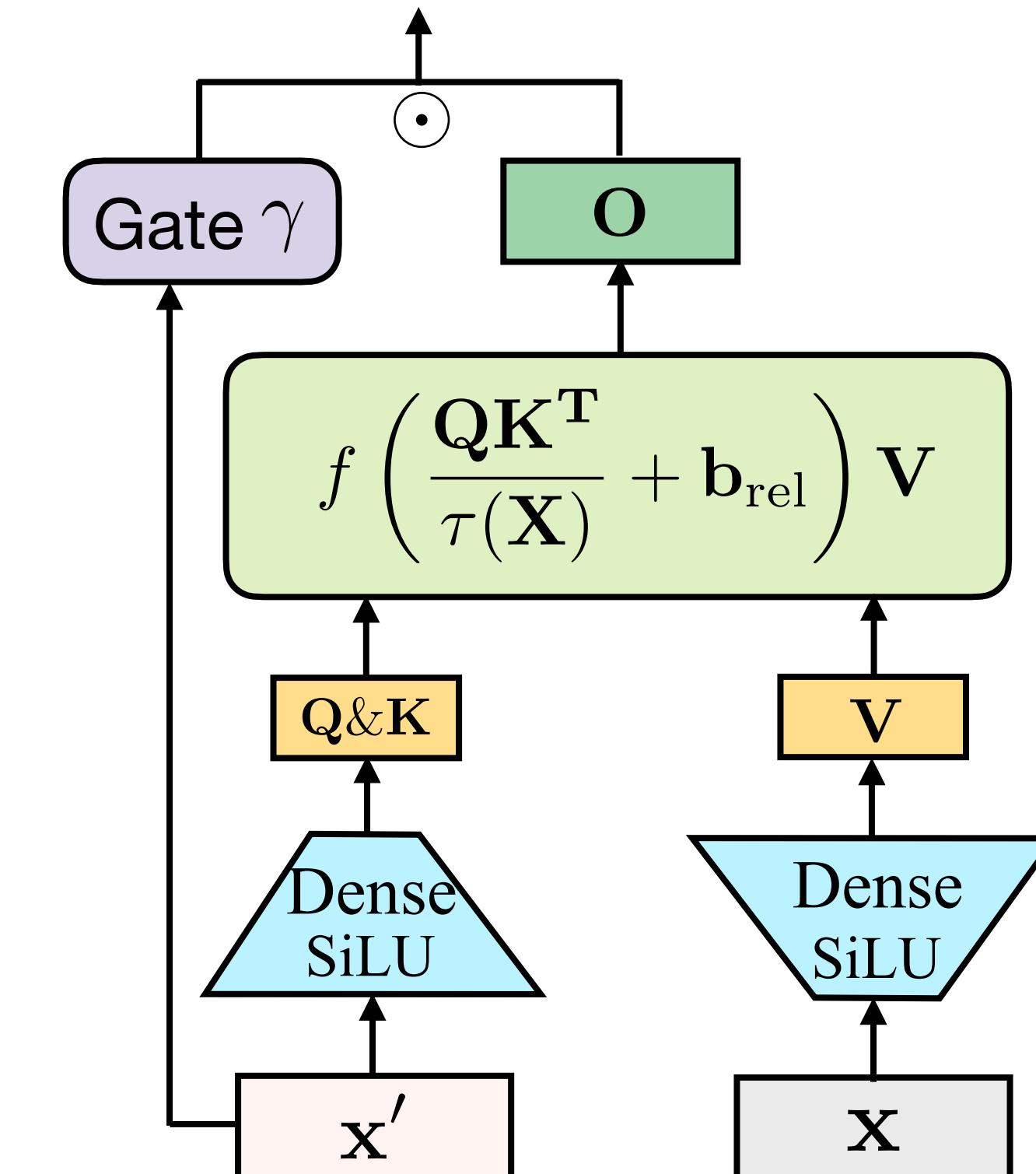
$$O = \text{softmax} \left(\frac{QK^T}{\tau(X)} + b_{\text{rel}} \right) V$$

Gated Attention:

$$\gamma = \mathcal{G}(X)$$

$$O_{\text{SHGA}} = O \odot \gamma$$

$\mathcal{G}()$ is a transform, e.g. an MLP



Single-head Gated Attention

- Adding a reset gate to the attention output

Attention:

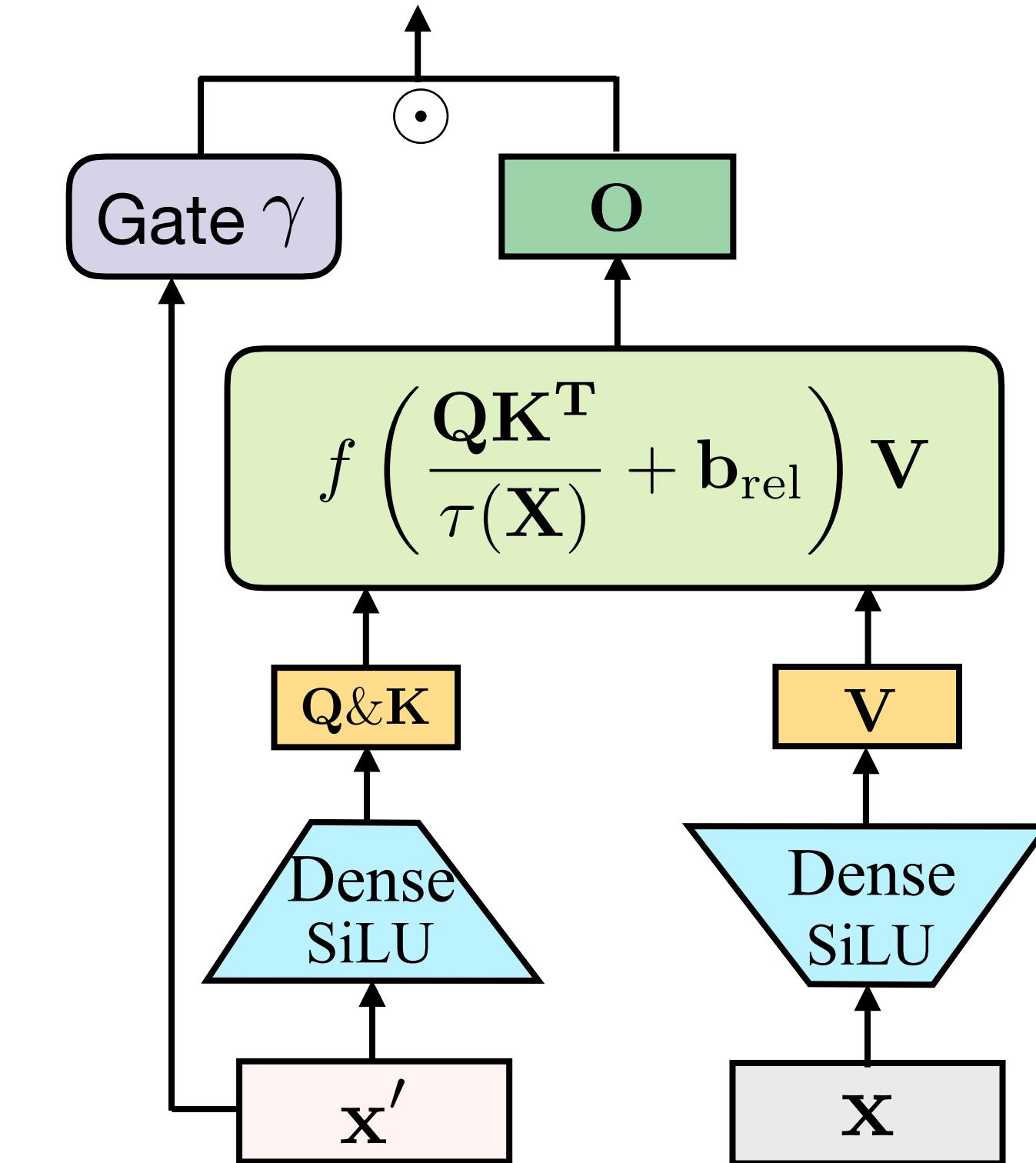
$$O = \text{softmax} \left(\frac{QK^T}{\tau(X)} + b_{\text{rel}} \right) V$$

Gated Attention:

$$\gamma = \mathcal{G}(X)$$

$$O_{\text{SHGA}} = O \odot \gamma$$

$\mathcal{G}()$ is a transform, e.g. an MLP



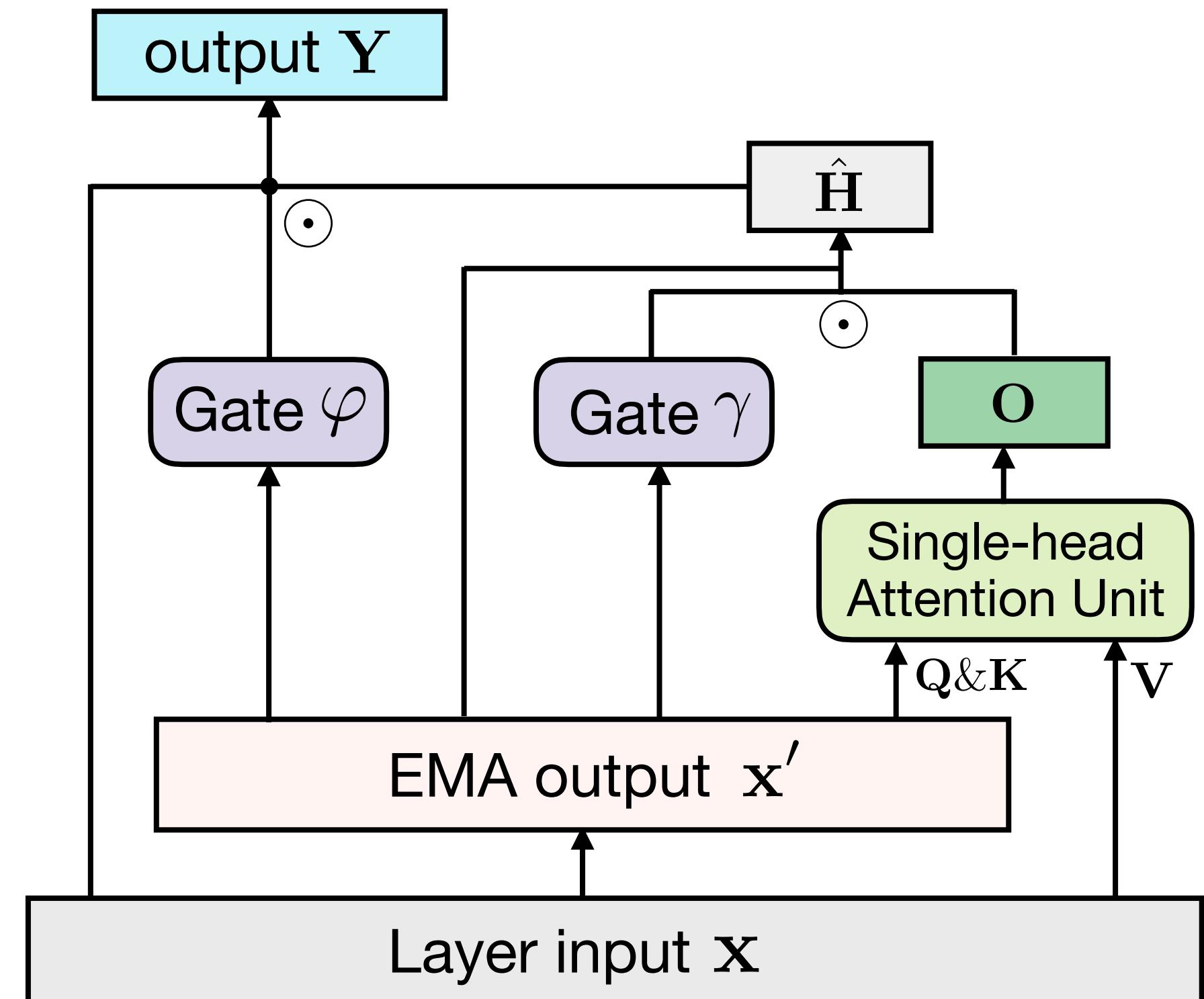
- Single-head gated attention is as expressive as multi-head one

Theorem: Suppose the transformation \mathcal{G} is a universal function approximator.

Then, for each X there exists $\gamma = \mathcal{G}(X)$ such that $O_{\text{SHGA}} = O_{\text{MHA}}$

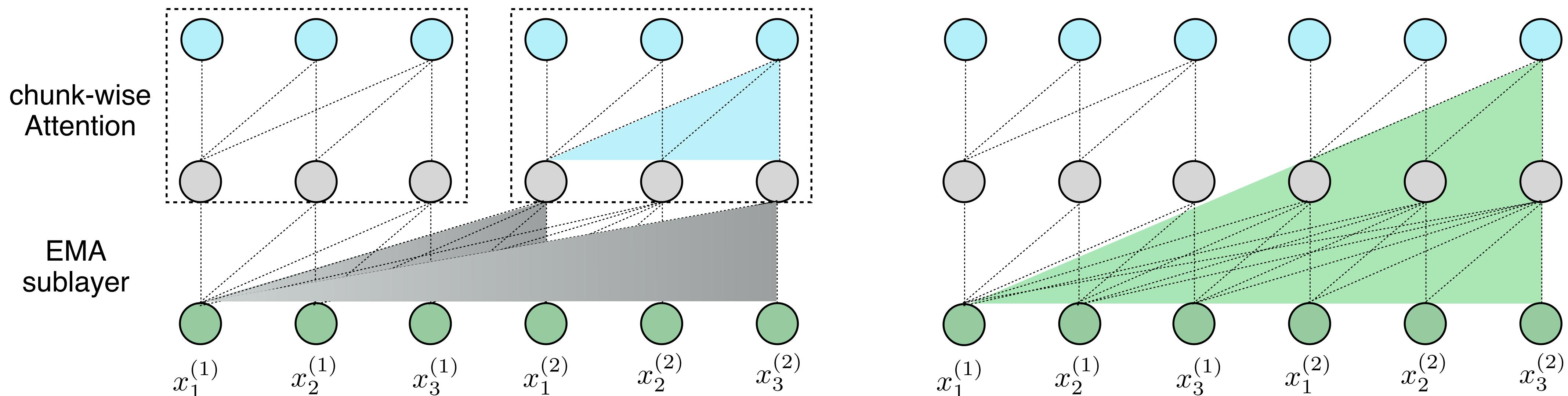
Mega Architecture: Outline

- **Exponential Moving Average (EMA)**
 - Local dependencies that decaying exponentially over time
 - Connection with [State Space Models](#) (e.g S4)
- **Single-head Gated Attention**
 - Adding a reset gate to the attention output
 - Theoretically proving that [single-head](#) gated attention is as expressive as multi-head one
- **Mega-Chunk**
 - Applying attention to local chunks of fixed length
 - Reducing quadratic complexity to linear



Mega-Chunk: Efficient Mega

- Split input sequences into multiple chunks with fixed length
- Applying attention individually to each chunk
 - Linear complexity & easy implementation
 - Losing contextual information between chunks
 - Fortunately, EMA preserves the loss of information from previous chunks



Experiments

- **Long Range Arena (LRA)**

- 3 tasks on byte-level text classification
- 3 tasks on pixel-level image classification

- **Language Modeling**

- Enwiki8 (character-level)
- WikiText-103 (Word-level)

- **Machine Translation**

- WMT'14 English - German

- **Image Classification**

- ImageNet-1K

- **Raw Speech Classification**

- Speech Commands

Experimental Results

	LRA	WMT'14	WikiText-103	ImageNet	Raw-SC
XFM	59.24	27.68	18.66	81.80	31.24
S4	85.86	—	20.95	—	97.50
Mega	88.21	29.01	18.07	82.35	97.30

Analysis on LRA: Accuracy & Efficiency

Models	ListOps	Text	Retrieval	Image	Pathfinder	Path-X	Avg.	Speed	Mem.
XFM	36.37	64.27	57.46	42.44	71.40	✗	54.39	–	–
XFM‡	37.11	65.21	79.14	42.94	71.83	✗	59.24	1×	1×
Reformer	37.27	56.10	53.40	38.07	68.50	✗	50.67	0.8×	0.24×
Linformer	35.70	53.94	52.27	38.56	76.34	✗	51.36	5.5×	0.10×
BigBird	36.05	64.02	59.29	40.83	74.87	✗	55.01	1.1×	0.30×
Performer	18.01	65.40	53.82	42.77	77.05	✗	51.41	5.7×	0.11×
Luna-256	37.98	65.78	79.56	47.86	78.55	✗	61.95	4.9×	0.16×
S4-v1	58.35	76.02	87.09	87.26	86.05	88.10	80.48	–	–
S4-v2	59.60	86.82	90.90	88.65	94.20	96.35	86.09	–	–
S4-v2‡	59.10	86.53	90.94	88.48	94.01	96.07	85.86	4.8×	0.14×
MEGA	63.14	90.43	91.25	90.44	96.01	97.98	88.21	2.9×	0.31×
MEGA-chunk	58.76	90.19	90.97	85.80	94.41	93.81	85.66	5.5×	0.13×

Limitations of Mega

- Mode Capacity vs. Full Attention
 - Limited capacity of EMA sub-layer
 - Mega-chunk fails behind Mega w. full attention

Analysis on LRA: Accuracy & Efficiency

Models	ListOps	Text	Retrieval	Image	Pathfinder	Path-X	Avg.	Speed	Mem.
XFM	36.37	64.27	57.46	42.44	71.40	✗	54.39	–	–
XFM‡	37.11	65.21	79.14	42.94	71.83	✗	59.24	1×	1×
Reformer	37.27	56.10	53.40	38.07	68.50	✗	50.67	0.8×	0.24×
Linformer	35.70	53.94	52.27	38.56	76.34	✗	51.36	5.5×	0.10×
BigBird	36.05	64.02	59.29	40.83	74.87	✗	55.01	1.1×	0.30×
Performer	18.01	65.40	53.82	42.77	77.05	✗	51.41	5.7×	0.11×
Luna-256	37.98	65.78	79.56	47.86	78.55	✗	61.95	4.9×	0.16×
S4-v1	58.35	76.02	87.09	87.26	86.05	88.10	80.48	–	–
S4-v2	59.60	86.82	90.90	88.65	94.20	96.35	86.09	–	–
S4-v2‡	59.10	86.53	90.94	88.48	94.01	96.07	85.86	4.8×	0.14×
MEGA	63.14	90.43	91.25	90.44	96.01	97.98	88.21	2.9×	0.31×
MEGA-chunk	58.76	90.19	90.97	85.80	94.41	93.81	85.66	5.5×	0.13×

Limitations of Mega

- Mode Capacity vs. Full Attention

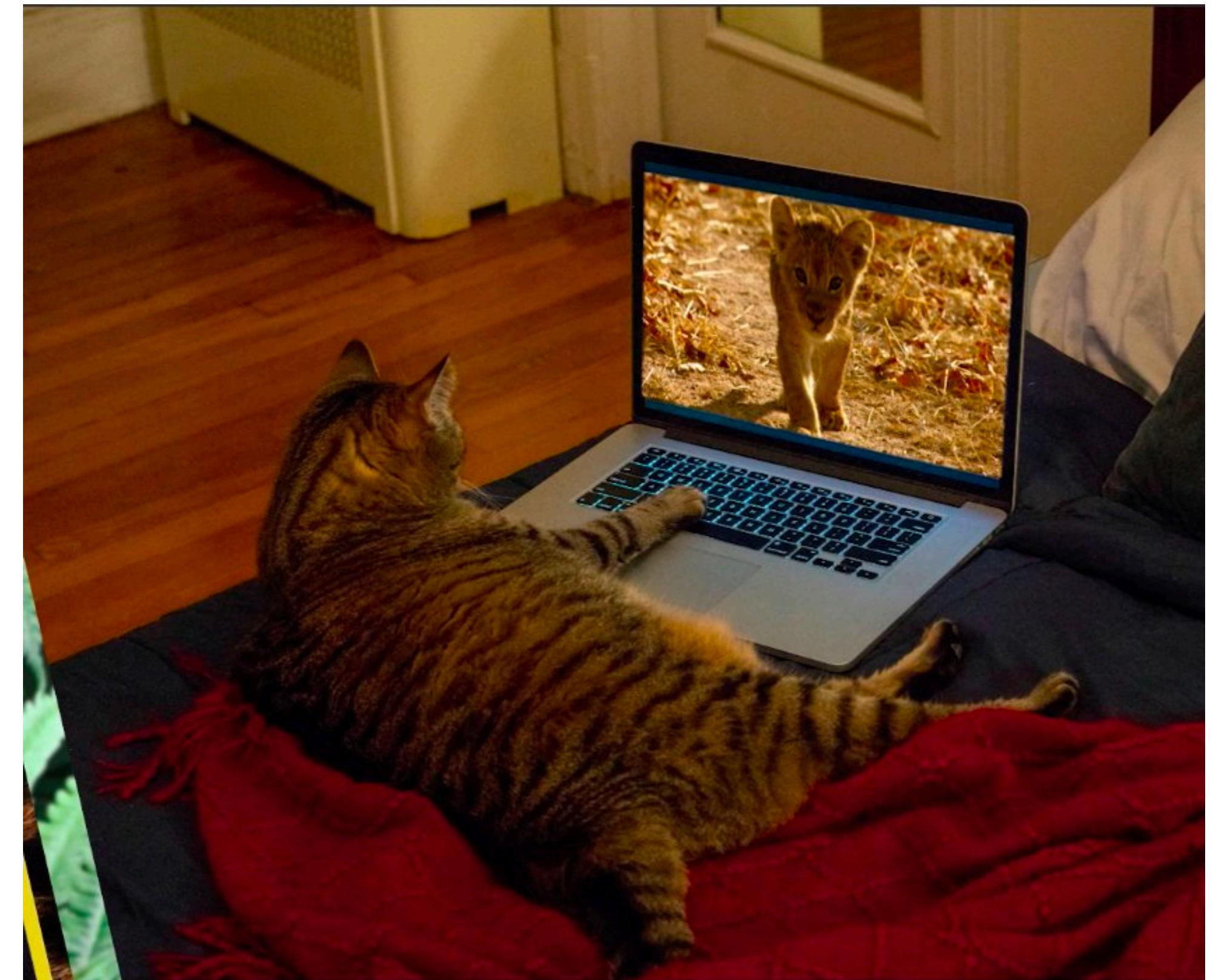
- Limited capacity of EMA sub-layer
 - Mega-chunk fails behind Mega w. full attention

- Architectural Divergence on Difference Data Modalities

- Not only for Mega, but for almost all architectures
- Different normalization layers on different positions
 - Batch Norm vs. Layer Norm vs. RMS Norm vs Scale Norm vs. ...
 - Pre-Norm vs. Post-Norm vs. QK-Norm vs. ...
- Different attention functions
 - Softmax vs. ReLU2 vs. Laplace vs ...
- ...

Unified Architecture for Multi-Modality

- Why a Unified Architecture Important for Multi-modality?
 - A unified foundation model learning from various types of signals



Limitations of Mega

- Mode Capacity vs. Full Attention
 - Limited capacity of EMA sub-layer
 - Mega-chunk fails behind Mega w. full attention
- Architectural Divergence on Difference Data Modalities
 - Not only for Mega, but for almost all architectures
 - Different normalization layers and positions
 - Batch Norm vs. Layer Norm vs. RMS Norm vs Scale Norm vs. ...
 - Pre-Norm vs. Post-Norm vs. QK-Norm vs. ...
 - Different attention functions
 - Softmax vs. ReLU2 vs. Laplace vs ...
 - ...
- No Evidences or Results for Mega's Scalability
 - Large-scale pretraining with Mega?

Megalodon: An Improved Version of Mega

- **Complex Exponential Moving Average**
 - Extending EMA to the Complex Field
- **Timestep Normalization**
 - Auto-regressive group normalization across sequential dimension
- **Numerical Stability**
 - Normalized Attention
 - Pre-Norm w. Top-hop Residual

Megalodon: Results on Small-Scale Benchmarks

	LRA	SC	ImageNet	Enwiki8	WT103
S4	85.86	97.50	—	—	20.95
XFM	59.24	xx	81.8	1.08	18.66
Mega-chunk	85.66	96.92	—	1.02	18.07
Mega	88.21	—	82.35	—	—
Megalodon-chunk	87.62	98.14	—	1.00	17.23
Megalodon	88.63	—	83.12	—	—

Importantly! All the experiments were conducted using exactly the same architecture!

Megalodon on LLM Pretraining

- Pretraining Data

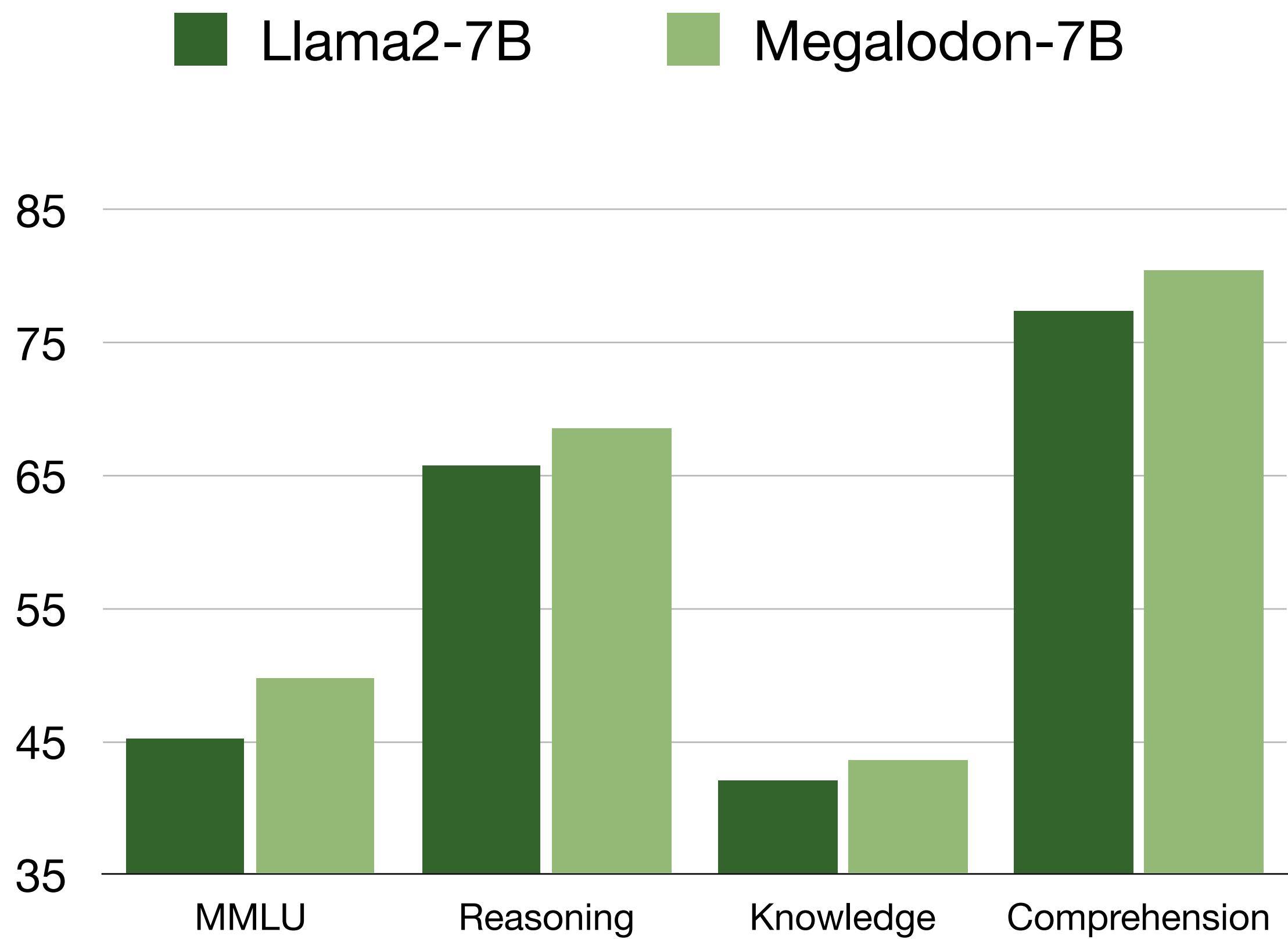
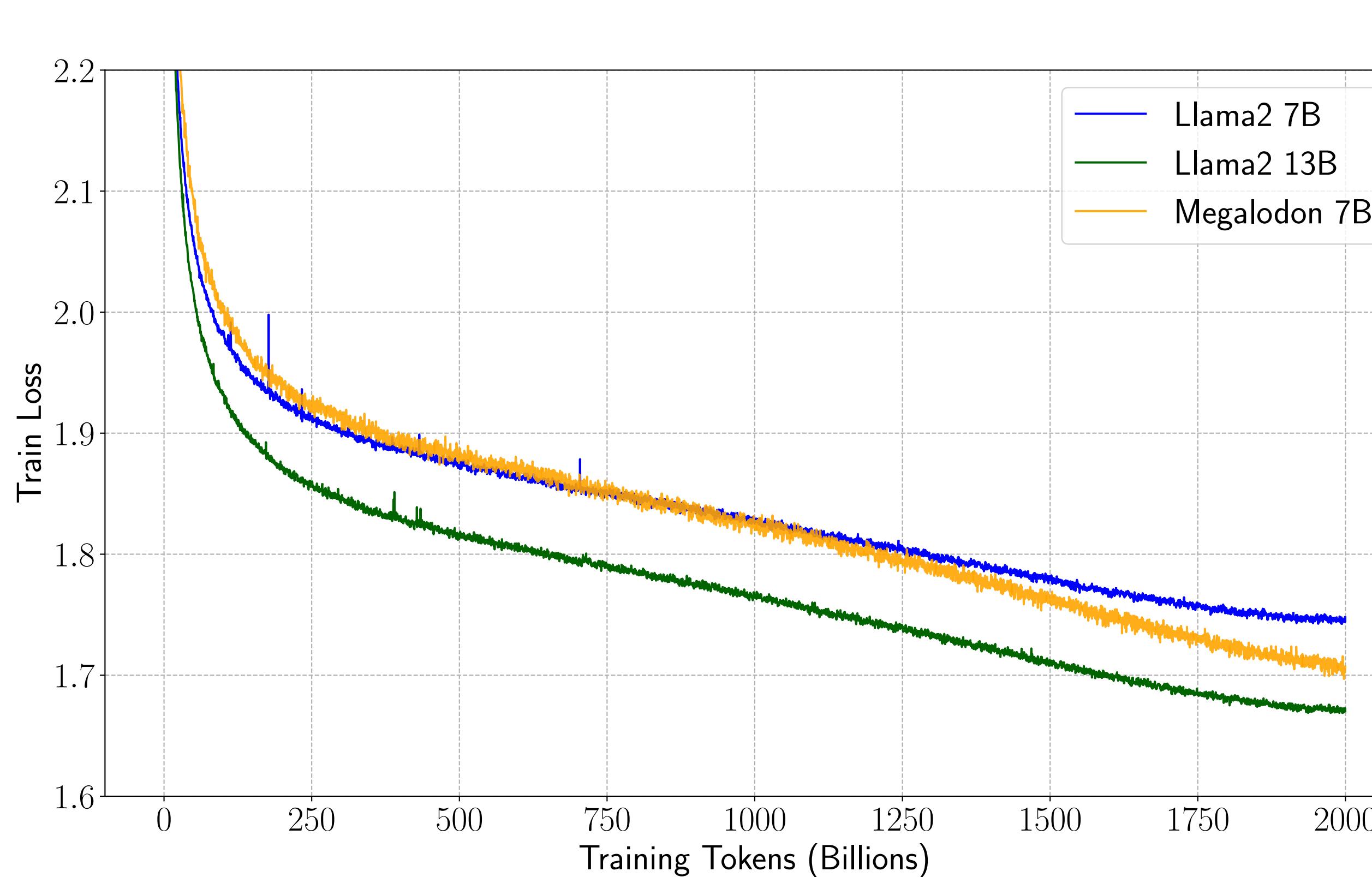
- 2 trillion tokens
- Exactly the same with Llama2 for head-to-head comparison

- Architecture Hyperparameters

- Closely following Llama2
 - 7B parameters
 - 32 blocks, model dimension 4096
 - Rotary positional embedding (RoPE)
- Differences
 - 4 attention heads (32 in Llama2)
 - 32K context length w. 4K attention chunk size (4K full attention in Llama2)

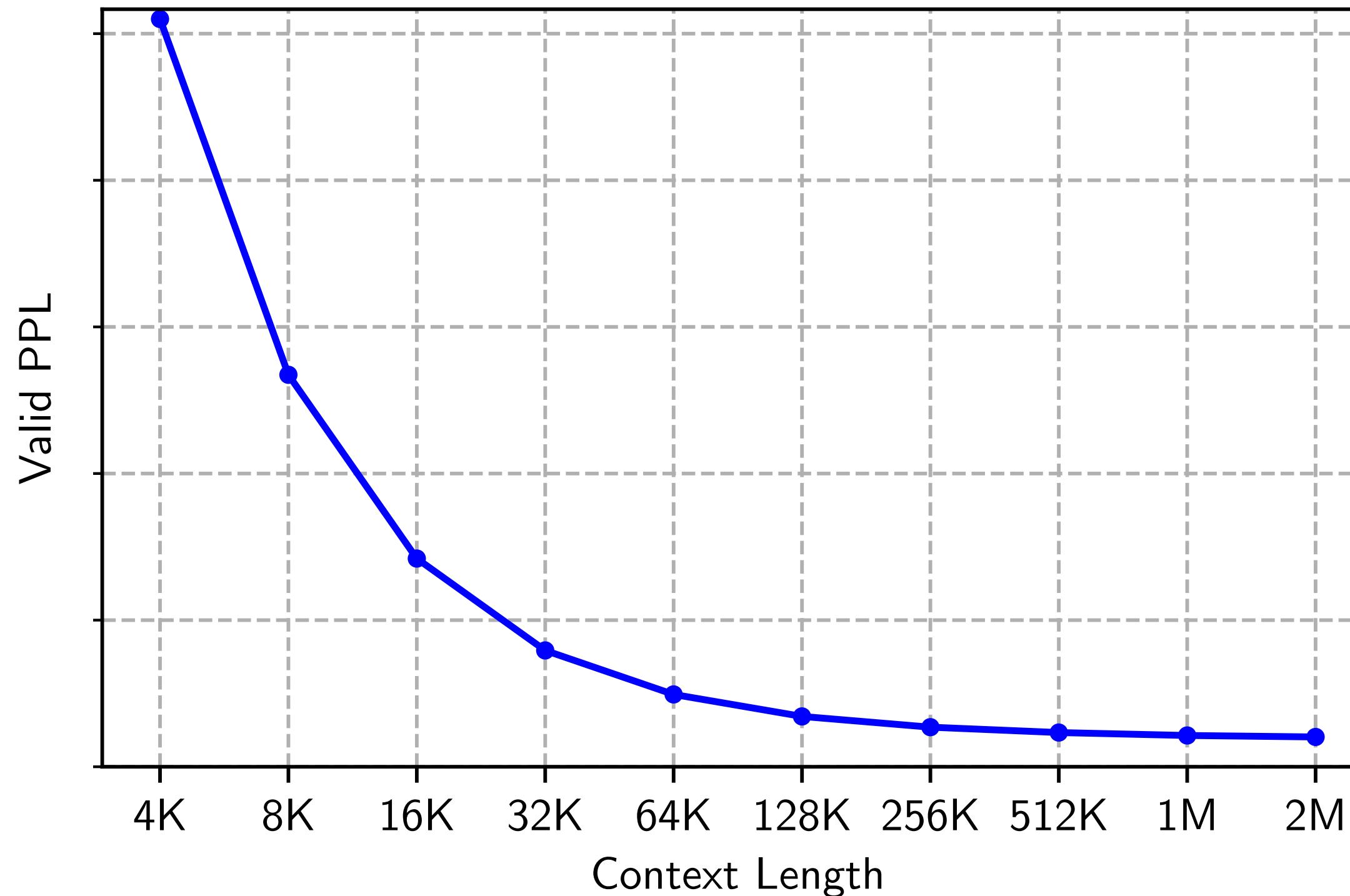
Efficiency of Megalodon-7B

- Data Efficiency



Long-Context Modeling

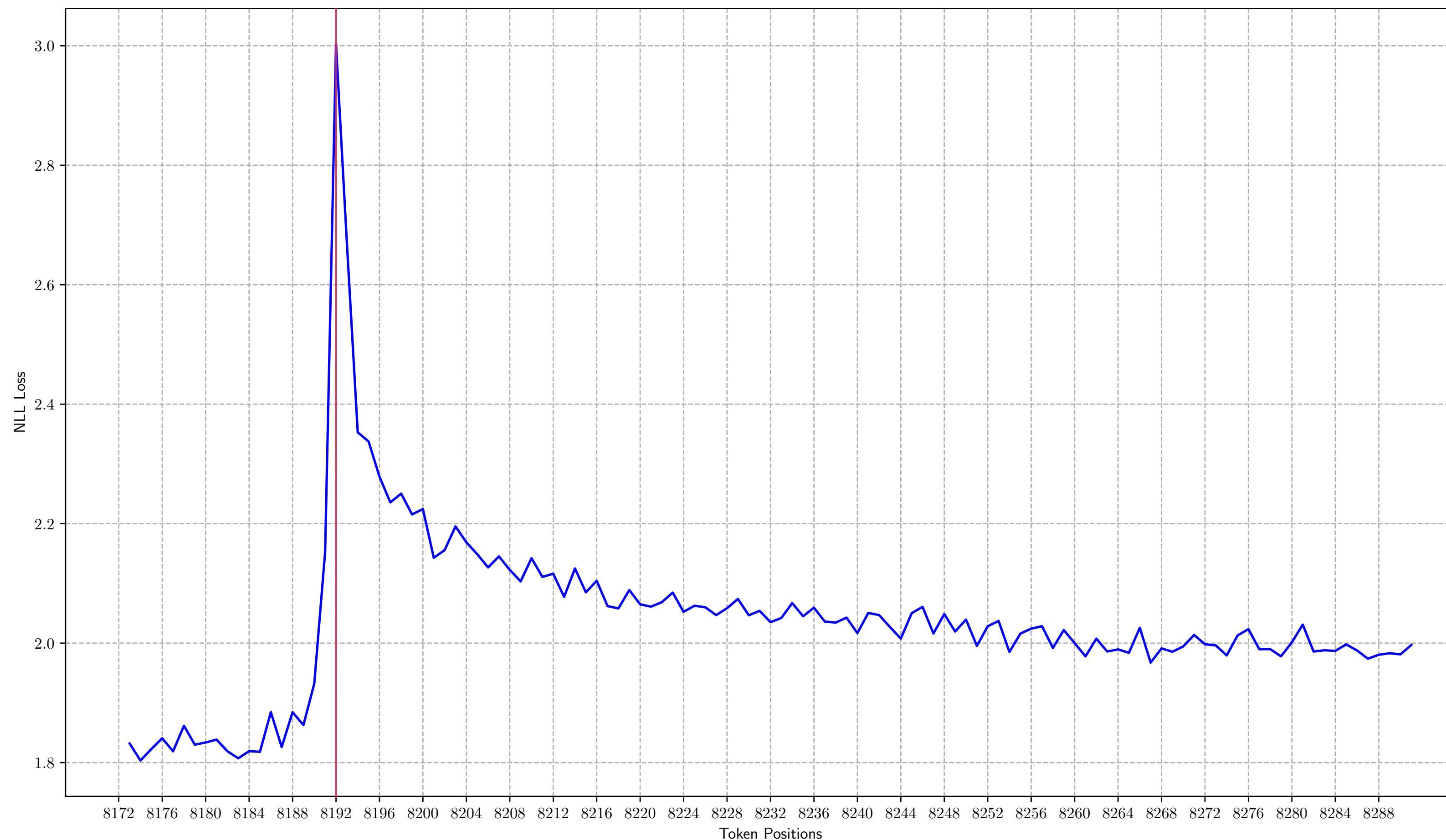
- Long-Context Evaluation of Megalodon-7B
 - Perplexity w. various context lengths
 - Long-context QA tasks in Scrolls



Model	NaQA	Qasper	QMSum
Xgen	17.4	20.5	6.8
MPT	18.8	24.7	8.8
Yarn	20.9	26.2	11.4
LLAMA2	18.8	19.8	10.1
LLAMA2-L*	23.5	28.3	14.5
MEGALODON	23.9	28.0	13.1

Existing Problems of Megalodon

- Validation losses on positions **before** and **after** chunk boundaries increase significantly



Thanks!
Q&A