北京大学
PEKING UNIVERSITY

CR-Net: Scaling Parameter-Efficient Training with Cross-Layer Low-Rank Structure

Boao Kong*, Junzhu Liang*, Yuxi Liu*, Renjia Deng, Kun Yuan

Peking University

Presenter: Boao Kong

April 01, 2026

- **Accepted by ICLR 2026.**
- **ArXiv: https://arxiv.org/abs/2509.18993.**

ArXiv

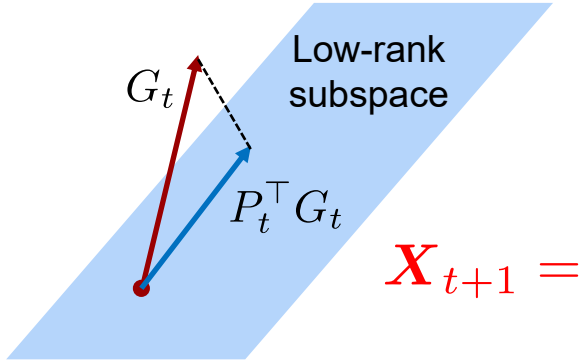
Openreview

Memory overhead in LLMs pre-training

北京大学 PEKING UNIVERSITY

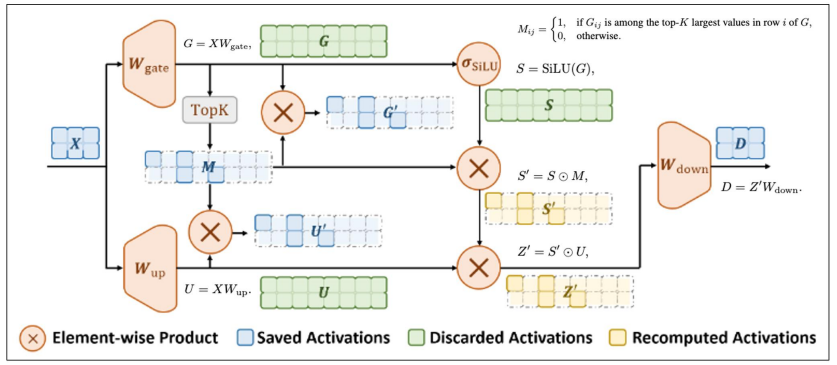
Memory = Parameter + Gradient + Optimizer states + Activation

- Low-rank projection only saves the memory of **Optimizer states**



$$X_{t+1} = X_t + P_t \rho(P_t^\top G_t)$$

- Sparse activation approachse saves the memory of **Activation**



$$M = \text{TopK}(G)$$
$$S = \text{SiLU}(G),$$
$$S' = S \odot M,$$
$$Z' = S' \odot U,$$

- How to save the **Parameter** and **Gradient**? **Parameter-efficient Methods!**
- Parameter-efficient methods can save the memory of parameter, gradient, and optimizer states at the same time.

Jiawei Zhao, et. al., GaLore: Memory-Efficient LLM Training by Gradient Low-Rank Projection, ICML 2024

Tong Wu, et. al., Mixture-of-Channels: Exploiting Sparse FFNs for Efficient LLMs Pre-Training and Inference, ArXiv 2025

Bottleneck of LoRA in LLMs pre-training

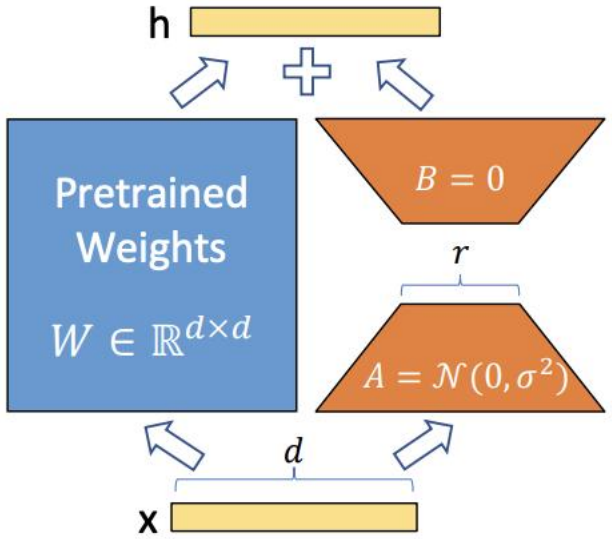
- Although LoRA enables fine-tuning of LLMs with fewer parameters, it is not applicable in pre-training.

$$\min_{W \in \mathbb{R}^{p \times q}} \quad \mathbb{E}_{\xi \sim \mathcal{D}}[F(W;\xi)] \qquad \text{(loss function for pre-training)}$$

$$\min_{A \in \mathbb{R}^{p \times r}, B \in \mathbb{R}^{r \times q}} \quad \mathbb{E}_{\xi \sim \mathcal{D}}[F(W + AB;\xi)] \qquad \text{(loss function for LoRA)}$$

Update A, B

Fix X



Pre-trian LLaMA on C4

Methods	60M		130M		350M		1B	
	Perplexity	Memory	Perplexity	Memory	Perplexity	Memory	Perplexity	Memory
AdamW♠	34.06	0.36G	25.08	0.76G	18.80	2.06G	15.56	7.80G
Low-Rank♠	78.18	0.26G	45.51	0.54G	37.41	1.08G	142.53	3.57G
LoRA♠	34.99	0.36G	33.92	0.80G	25.58	1.76G	19.21	6.17G

- The pre-training PPL of LoRA increases significantly.
- Calls for an effective algorithm for pre-training.

Low-Rank Activation Residuals Between Adjacent Layers in LLMs

- Parameters lack low-rank structure; low-rank approximation fails

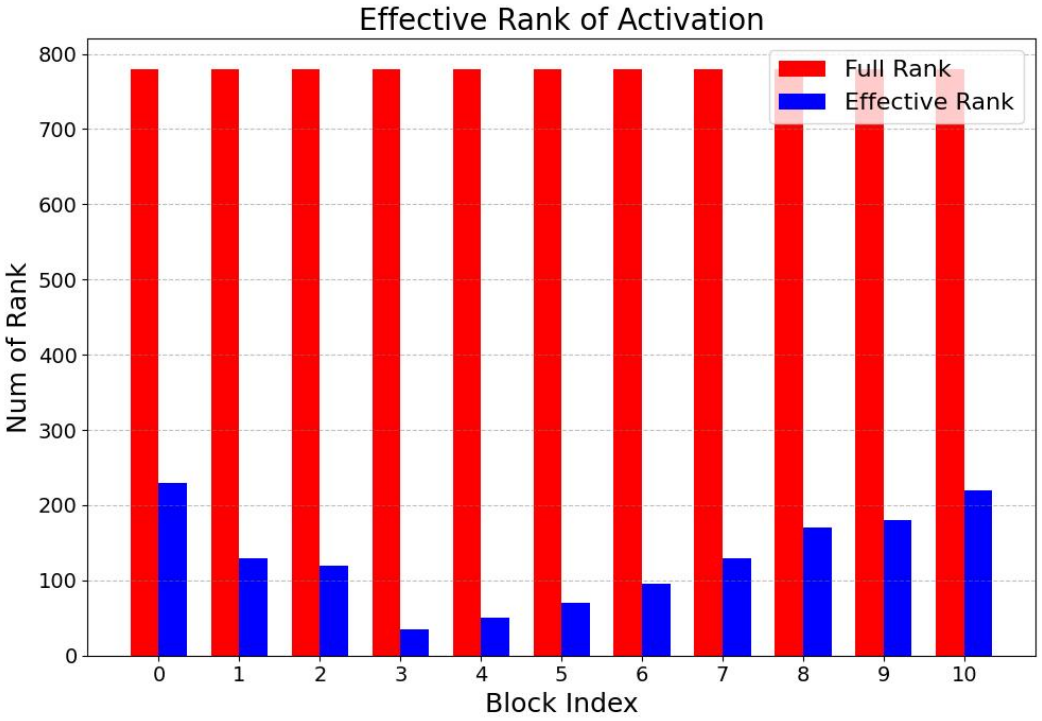
$$\underbrace{Q_\ell}_{\text{Activation}} = \underbrace{X}_{\text{Input}} \underbrace{W_\ell^Q}_{\text{Parameter}} \approx \underbrace{X}_{\text{Input}} \underbrace{A_\ell^Q}_{\text{Low-rank parameter}} \underbrace{B_\ell^Q}_{\text{Low-rank parameter}}$$

- **Core idea**: Compensate approximation error using **previous layer's activations**[1]

$$\underbrace{Q_\ell}_{\substack{\text{Activation} \\ \text{in layer } \ell}} = XW_\ell^Q \approx \underbrace{Q_{\ell-1}}_{\substack{\text{Activation in} \\ \text{layer } \ell-1}} + XA_\ell^Q B_\ell^Q$$

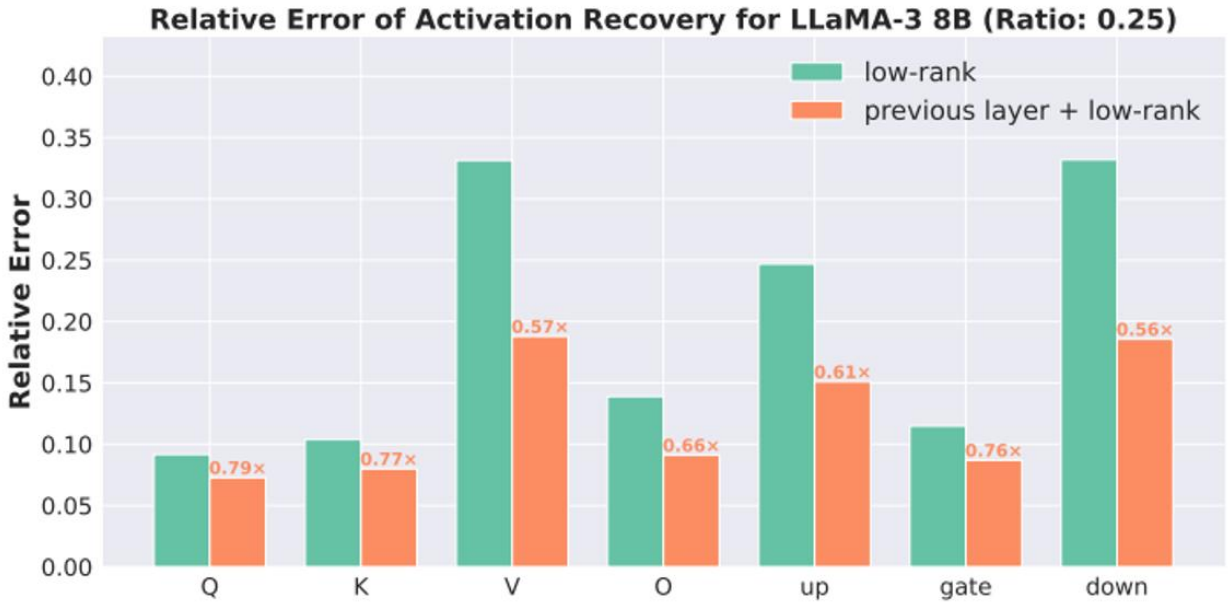
In other words, adjacent-layer activation residuals $Q_\ell - Q_{\ell-1}$ exhibit low-rank structure

Low-Rank Activation Residuals Between Adjacent Layers in LLMs



Activation residuals between layers exhibit low-rank property

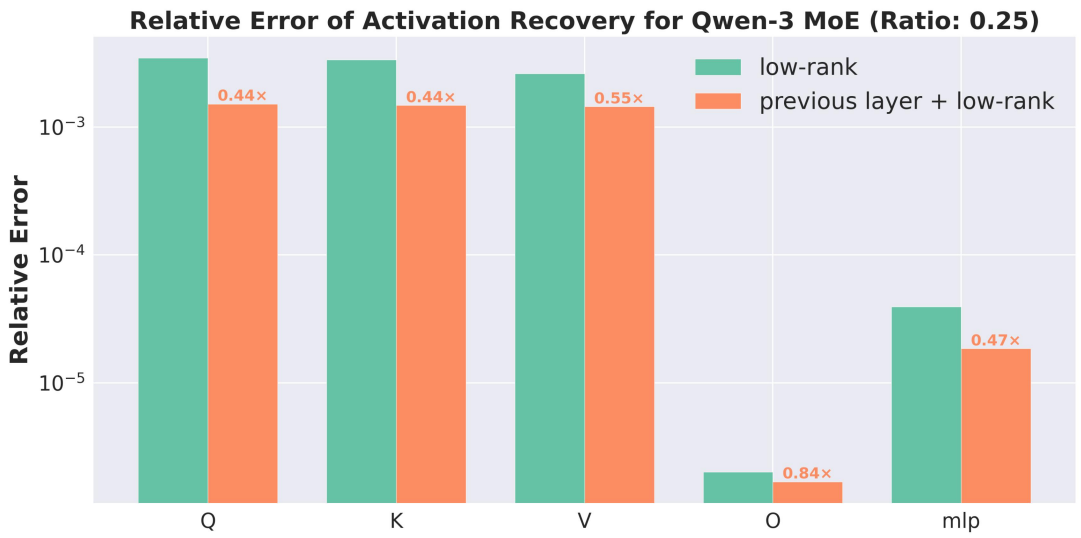
$$Q_\ell = XW_\ell^Q \approx XA_\ell^Q B_\ell^Q$$
$$Q_\ell = XW_\ell^Q \approx Q_{\ell-1} + XA_\ell^Q B_\ell^Q$$

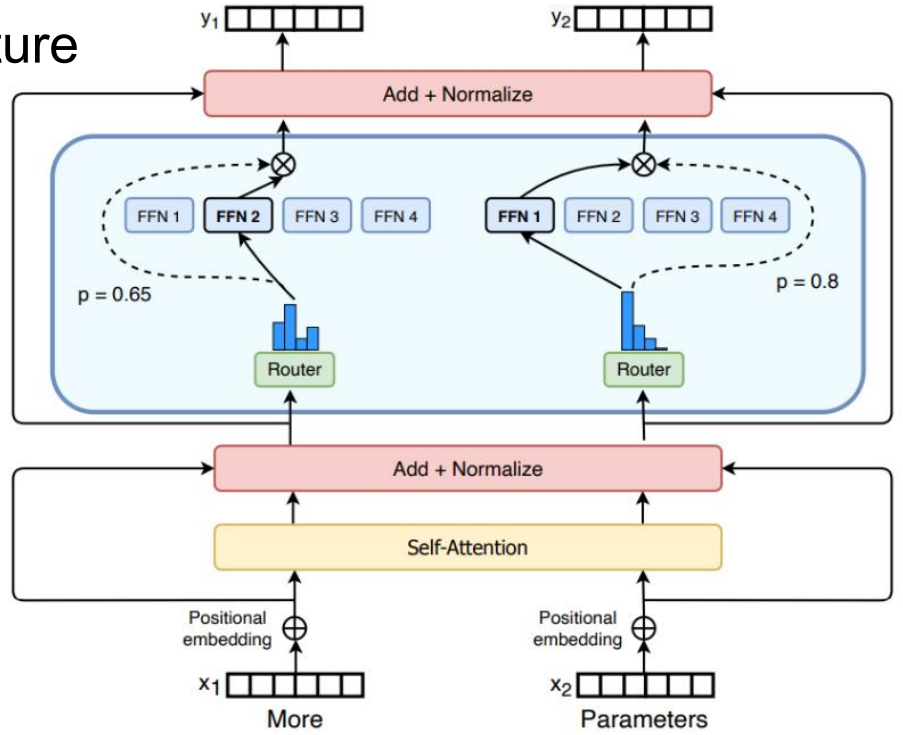


After error compensation, recovery error is reduced by **9%~54%**

Low-Rank Activation Residuals Between Adjacent Layers in LLMs

- Similar cross-layer low-rank structure in **MoE** architecture





- With the same rank, compensating with the previous layer's activations reduces the error by **16%~56%**

Theories behind low-rank structure in cross-layer activations

北京大学 PEKING UNIVERSITY

- The (high cosine) similarity between adjacent layer activations makes their residuals low-rank

Assumption 1 (Cosine similarity of adjacent attentions). *For $l = 2, 3, \cdots, L$, let $Y_l^P \in \mathbb{R}^{d\times d}$ as the activation of the linear of position P for the l-th layer. There exists a constant $\epsilon \in (0, 1)$ such that:*

$$\frac{\langle Y_l^P, Y_{l-1}^P \rangle_F}{\|Y_l^P\|_F \cdot \|Y_{l-1}^P\|_F} \geq 1 - \epsilon,$$

where $\langle \cdot, \cdot \rangle_F$ denotes the inner production of matrices induced by Frobenius norm.

Theorem 1. *Suppose Assumption 1 holds. Then there exists $r_0 > 0$ such that the approximation $\widetilde{Y}_{l,\beta}^P$ obtained by Eq. (3) has a lower error than the direct low-rank approximation $LR_r(Y_l^P)$ by a properly-selected β if $r < r_0$. Specifically, it holds that:*

$$\left\| Y_l^P - \widetilde{Y}_{l,\beta}^P \right\|_F^2 \leq \left\| Y_l^P - LR_r(Y_l^P) \right\|_F^2 .$$

Low-Rank Approximation with Error Compensation

Direct Low-Rank Approximation

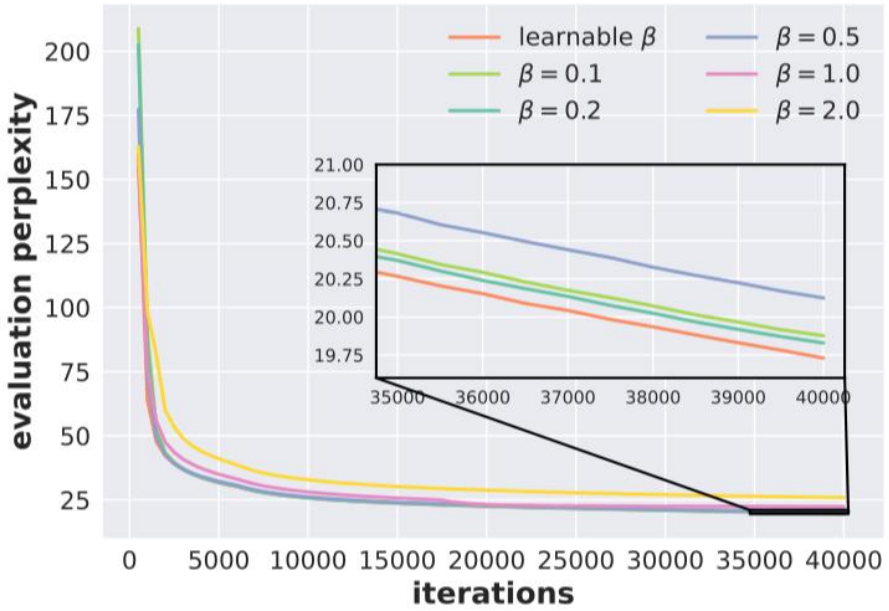
Cross-layer low-rank structure introduces smaller error than direct low-rank approximation of activations.

Learnable scaling factors

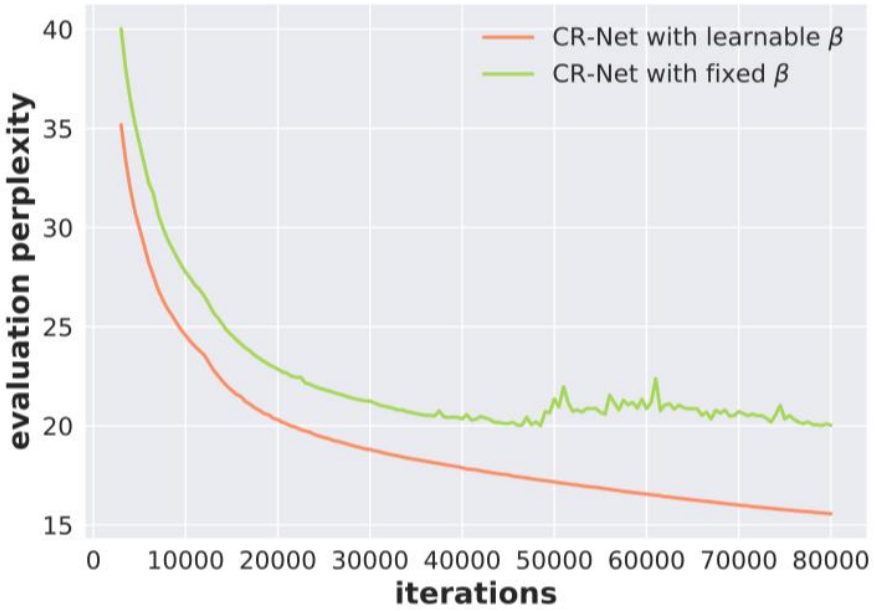
$$Y_\ell^{\mathrm{P}} = \beta_\ell^{\mathrm{P}} Y_{\ell-1}^{\mathrm{P}} + X_\ell^{\mathrm{P}} A_\ell^{\mathrm{P}} B_\ell^{\mathrm{P}}$$

- β_0 and β_l balance information between previous and current layers
- We make β_l **learnable** to **dynamically** adjust the influence of historical activations and low-rank output

LLaMA-2 350M



LLaMA-2 1B



Cross-layer Low-Rank Residual Network (CR-Net)

北京大学 PEKING UNIVERSITY

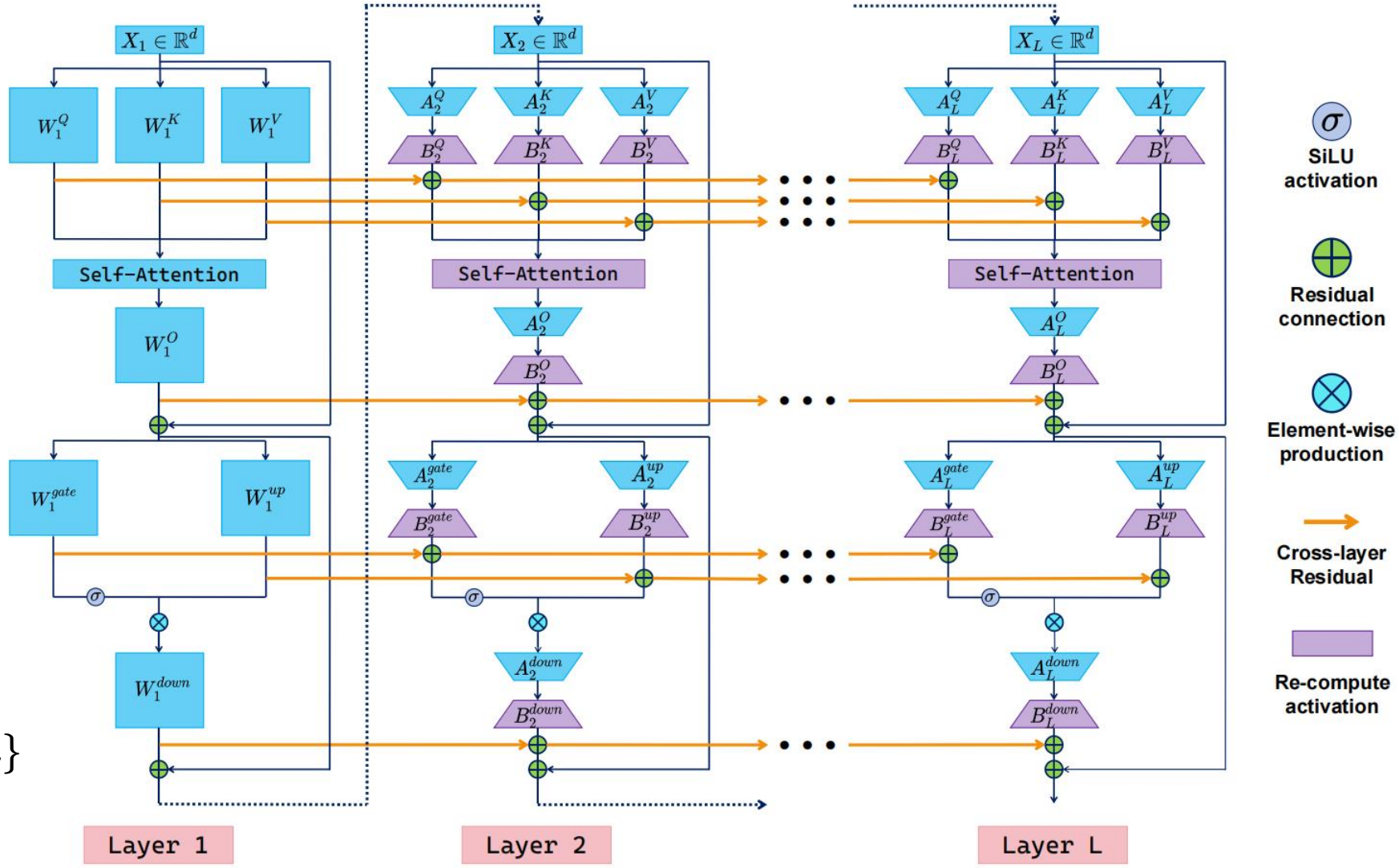
- Full-rank parameters in the firsr transformer layer
- For layer $\ell = 2, 3, \cdots, L$

$$Y_\ell^{\mathrm{P}} = \beta_\ell^{\mathrm{P}} Y_{\ell-1}^{\mathrm{P}} + X_\ell^{\mathrm{P}} A_\ell^{\mathrm{P}} B_\ell^{\mathrm{P}}$$

Learnable factor

Activation at position P

$$\mathrm{P} \in \{Q, K, V, O, gate, up, down\}$$



Cross-layer Low-Rank Residual Network (CR-Net)

北京大学 PEKING UNIVERSITY

- Full-size pre-training

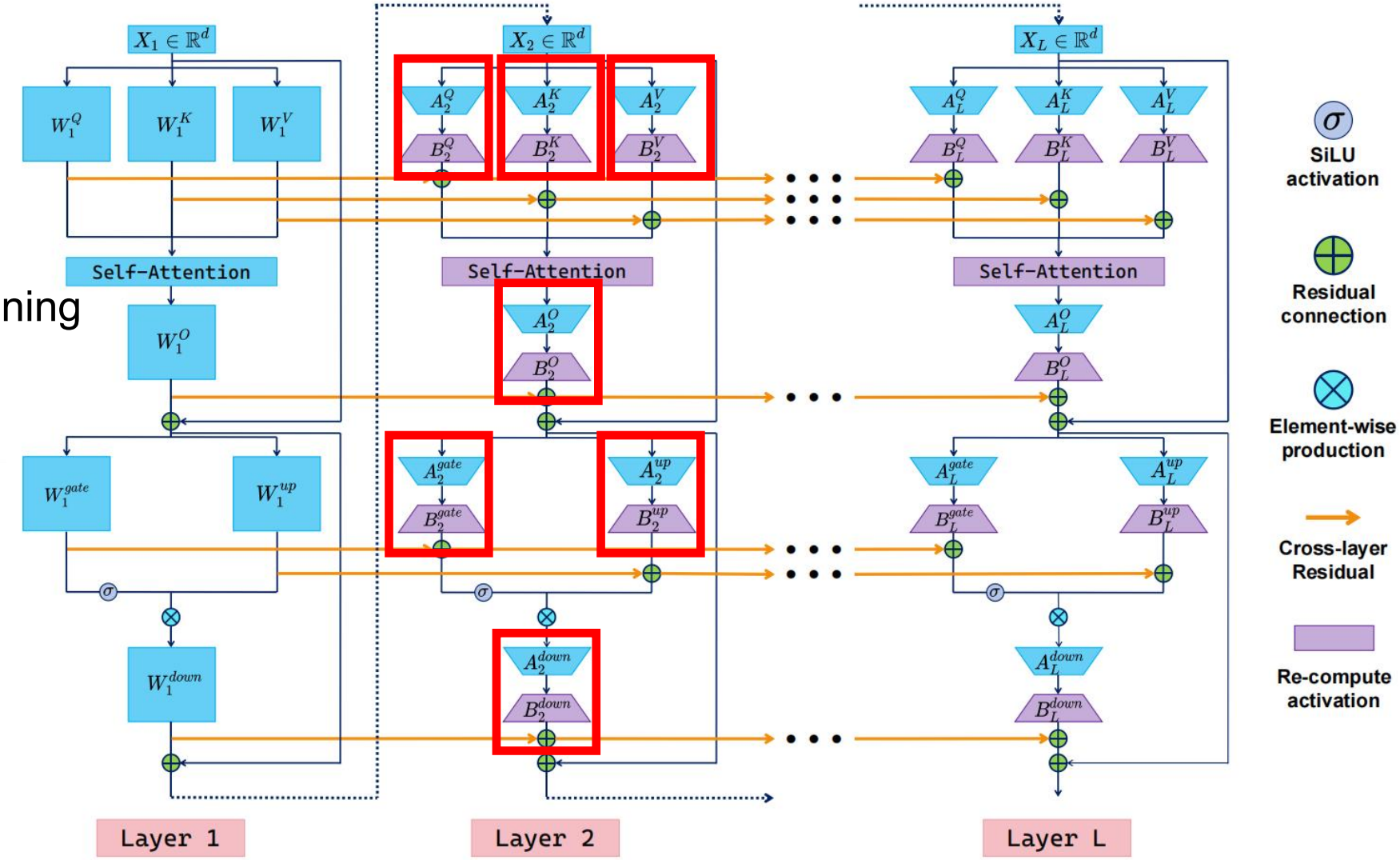
$$Y_1^{\mathrm{P}} = X_1^{\mathrm{P}} W_1^{\mathrm{P}} \quad \text{(m, n)}$$

- Parameter-efficient pre-training

$$Y_\ell^{\mathrm{P}} = \beta_\ell^{\mathrm{P}} Y_{\ell-1}^{\mathrm{P}} + X_\ell^{\mathrm{P}} A_\ell^{\mathrm{P}} B_\ell^{\mathrm{P}}$$

(m, r), (r, n)

- Reduce parameter from mn to $r(m+n)$
- **Leading to a smaller model, gradient, and optimizer states**



Boao Kong, Kun Yuan et al. *CR-Net: Scaling Parameter-Efficient Training with Cross-Layer Low-Rank Structure*, submitted to NeurIPS 2025

CR-Net: Experimental results

Pretrain LLaMA on C4

Approach	60M			130M			350M			1B		
	1.1B tokens			2.2B tokens			6.4B tokens			13.1B tokens		
	PPL	Para	Mem	PPL	Para	Mem	PPL	Para	Mem	PPL	Para	Mem
Full-rank	34.06	58	0.43	24.36	134	1.00	**18.80**	368	2.74	15.56	1339	9.98
LoRA	34.99	58	0.37	33.92	134	0.86	25.58	368	1.94	19.21	1339	6.79
ReLoRA	37.04	58	0.37	29.37	134	0.86	29.08	368	1.94	18.33	1339	6.79
SLTrain	34.15	44	0.32	26.04	97	0.72	19.42	194	1.45	16.14	646	4.81
CoLA	34.04	43	0.32	24.48	94	0.70	19.40	185	1.38	15.52	609	4.54
CR-Net ◊	**32.76**	43	0.32	**24.31**	90	0.67	18.95	183	1.36	**15.28**	583	4.35
GaLore	34.88	58	0.36	25.36	134	0.79	18.95	368	1.90	15.64	1339	6.60
RSO	34.55	58	0.36	25.34	134	0.79	18.87	368	1.90	15.86	1339	6.60
Apollo	**31.55**	58	0.36	**22.94**	134	0.79	**16.85**	368	1.90	14.20	1339	6.60
CR-Net †	32.76	43	0.32	23.74	106	0.79	17.08	250	1.86	**14.05**	870	6.48

- CR-Net achieves lower loss with fewer parameters (~43.6% in LLaMA-2 1B)
- At equal memory, CR-Net achieves lower loss at larger scales

The recomputation strategy of CR-Net

- CR-Net cannot directly save activation

(s, m) x (m, n) = (s, n) : Lsn

Vanilla: $Y_\ell^{\mathrm{P}} = X_\ell^{\mathrm{P}} W_\ell^{\mathrm{P}}$

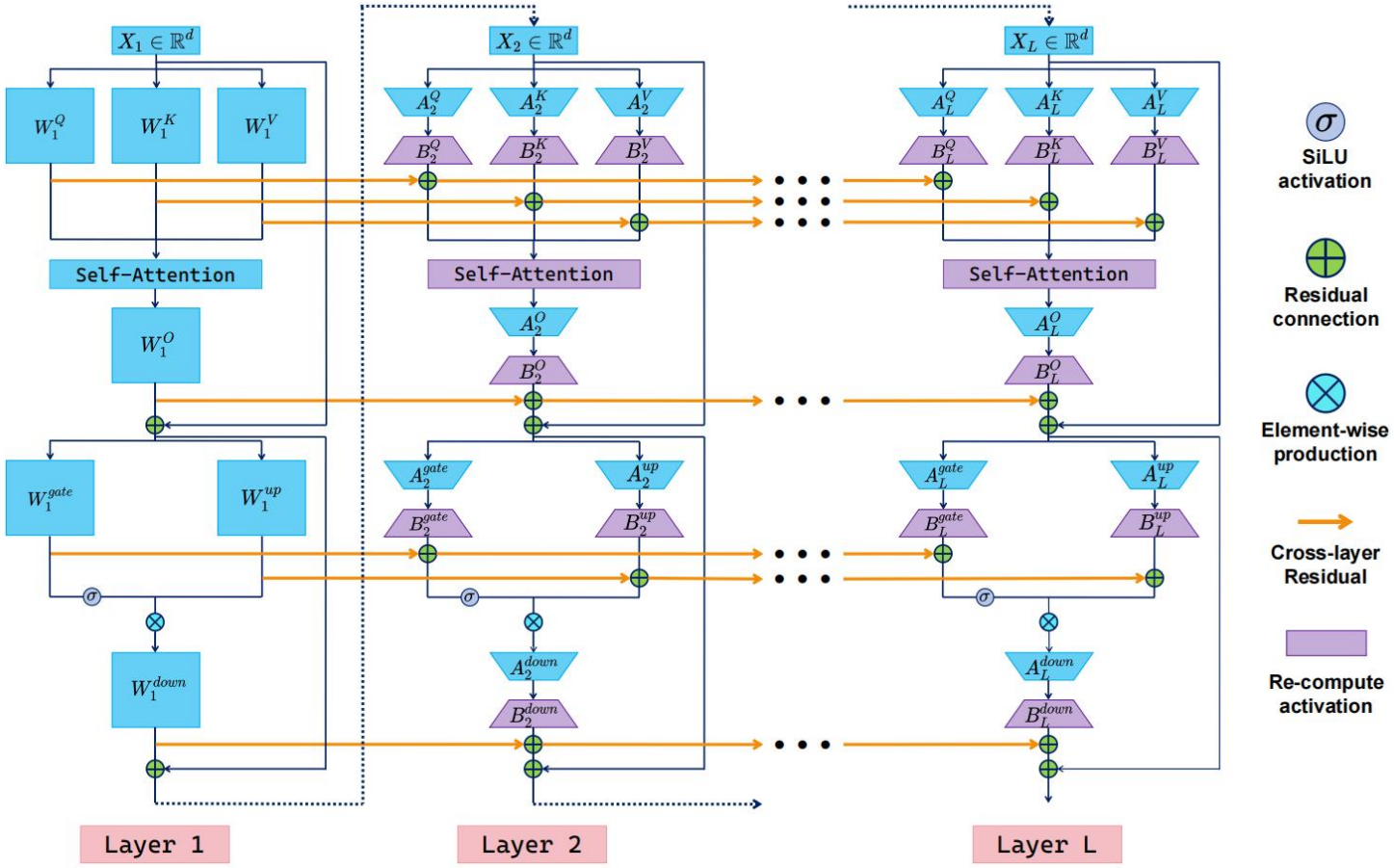
CR-Net: $Y_\ell^{\mathrm{P}} = \beta_\ell^{\mathrm{P}} Y_{\ell-1}^{\mathrm{P}} + X_\ell^{\mathrm{P}} A_\ell^{\mathrm{P}} B_\ell^{\mathrm{P}}$

(s, m) x (m, r) x (r, n) = (s, n): Lsn

- CR-Net + recomputation:

$$Y_\ell^{\mathrm{P}} = \frac{1}{\beta_\ell^{\mathrm{P}}}(Y_{\ell+1}^{\mathrm{P}} - X_{\ell+1}^{\mathrm{P}} A_{\ell+1}^{\mathrm{P}} B_{\ell+1}^{\mathrm{P}})$$

Only need the memory of: sn + Lsr



CR-Net: Memory saving

北京大学 PEKING UNIVERSITY

- Pre-trained on C4-en with LLaMA2-7B
- Rank: 1024 for CoLA-M, 896 for CR-Net

Table 4: Comparsion of validation perplexity (↓) and memory (↓) of different approaches in LLaMA-7B pre-training tasks. The results of compared methods are referred from [38, 57].

	Memory (GB)	10K	40K	65K	80K
8-bit Adam	72.59	N.A.	18.09	N.A.	15.47
8-bit GaLore	65.16	26.87	17.94	N.A.	15.39
Apollo	N.A.	N.A.	17.55	N.A.	14.39
CoLA-M	28.82	**22.76**	16.21	14.59	13.82
CR-Net w. re-computation	27.60	23.11	**16.01**	**14.47**	**13.72**
Training tokens (B)		1.3	5.2	8.5	10.5

- CR-Net achieves **62%** memory saving with better performance than baselines

CR-Net: Computation saving

北京大学 PEKING UNIVERSITY

Vanilla: $Y_\ell^{\mathrm{P}} = X_\ell^{\mathrm{P}} W_\ell^{\mathrm{P}}$

(s, m) x (m, n) = (s, n) : smn

CR-Net: $Y_\ell^{\mathrm{P}} = \beta_\ell^{\mathrm{P}} Y_{\ell-1}^{\mathrm{P}} + X_\ell^{\mathrm{P}} A_\ell^{\mathrm{P}} B_\ell^{\mathrm{P}}$

(s, n) + (s, m) x (m, r) x (r, n) = sn + s(m+n)r

- s: seq. length; h: hidden dimension; L: transformer layer; s=256, batch size=1
- Rank: 512 for other baselines, 448 for CR-Net (Ensures better validation perplexity of CR-Net.)

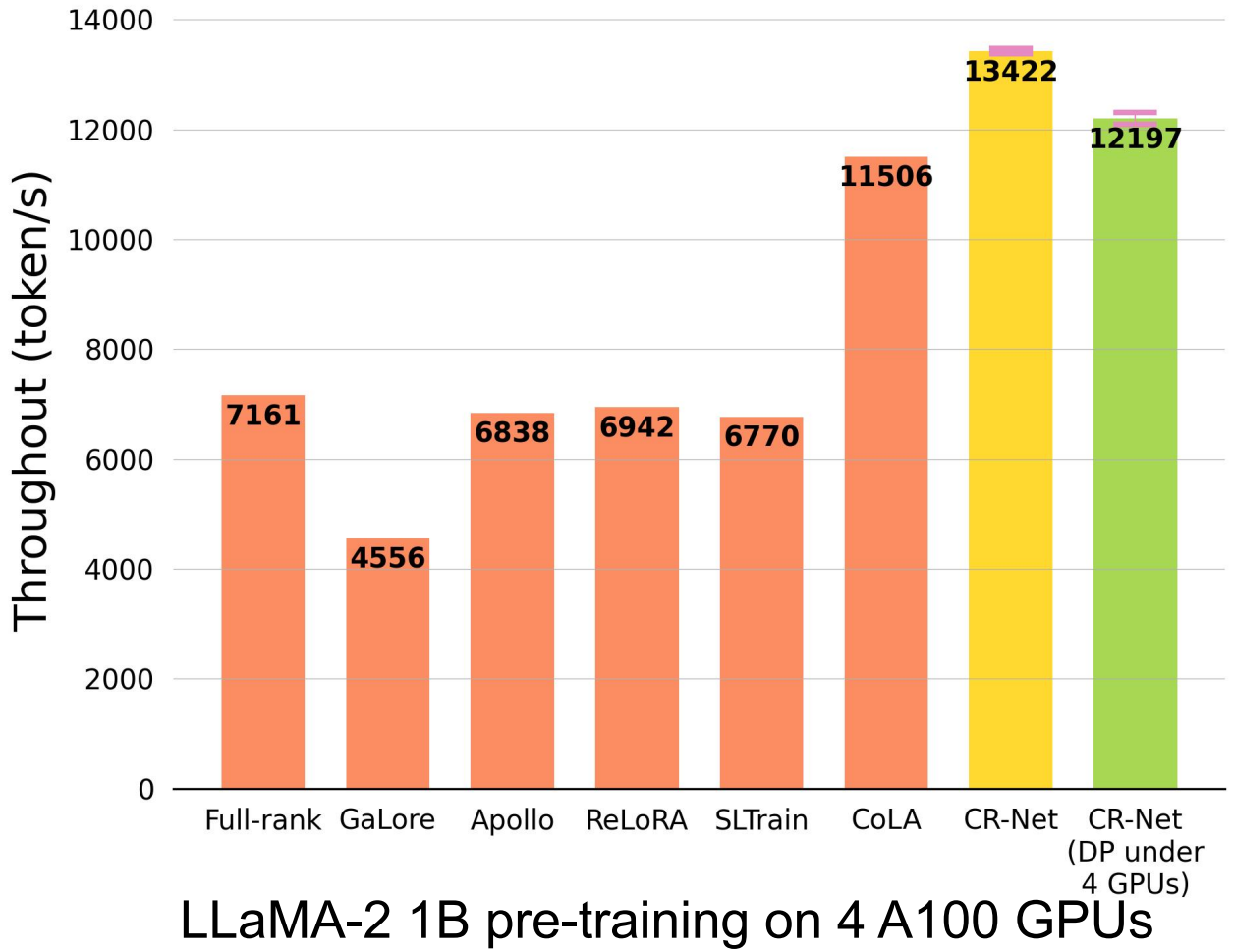
Approach	FLOPs	LLaMA-2 1B
Full-rank	$L(24sh^2 + 12s^2h + 18shh_{\mathrm{ff}})$	2.422×10^{12} (1.000×)
(Re)LoRA	$L(40sh^2 + 24s^2h + 30shh_{\mathrm{ff}})$	4.054×10^{12} (1.674×)
SLTrain	$L(24sh^2 + 12s^2h + 18shh_{\mathrm{ff}} + 24h^2r + 18hh_{\mathrm{ff}}r)$	7.164×10^{12} (2.958×)
GaLore	$L(24sh^2 + 12s^2h + 18shh_{\mathrm{ff}} + 16h^2r + 12hh_{\mathrm{ff}}r)$	5.583×10^{12} (2.305×)
CoLA	$L(48shr + 12s^2h + 18sr(h + h_{\mathrm{ff}}))$	1.005×10^{12} (0.415×)
CR-Net	$24sh^2 + 12s^2h + 18shh_{\mathrm{ff}} + (L-1)(48shr + 12s^2h + 18sr(h + h_{\mathrm{ff}}))$	0.934×10^{12} (0.385×)

- CR-Net uses **38.5%** of the FLOPs compared to the standard LLaMA-2 1B network.

CR-Net: Throughput

北京大学 PEKING UNIVERSITY

- CR-Net achieves **87%** higher throughput than the standard model.
- Even including communication overhead, CR-Net outperforms all baselines, with ~**66%** higher throughput than the standard model.



LLaMA-2 1B pre-training on 4 A100 GPUs

Hybrid between Vanilla and CR-Net

- Pre-training LLaMA-2 1B with sequence length s=256
- Rank: 512 for other baselines, 448 for CR-Net (Ensures better validation perplexity of CR-Net)

Algorithms	Memory (GB)	FLOPs ($\times 10^{14}$)
Vanilla GCP + Full-rank	11.98 (1.000×)	2.067 (1.000×)
CoLA-M	12.04 (1.005×)	0.764 (0.370×)
CR-Net‡	11.81(0.986×)	0.692 (0.334×)
CR-Net$^{\sharp}$	9.94(0.830×)	0.694 (0.335×)

Store full activation every 8 layers

Store full activation every 32 layers

- Compared to standard LLaMA-2 1B (vanilla GCP): At matched memory cost, CR-Net achieves **67%** faster computation.

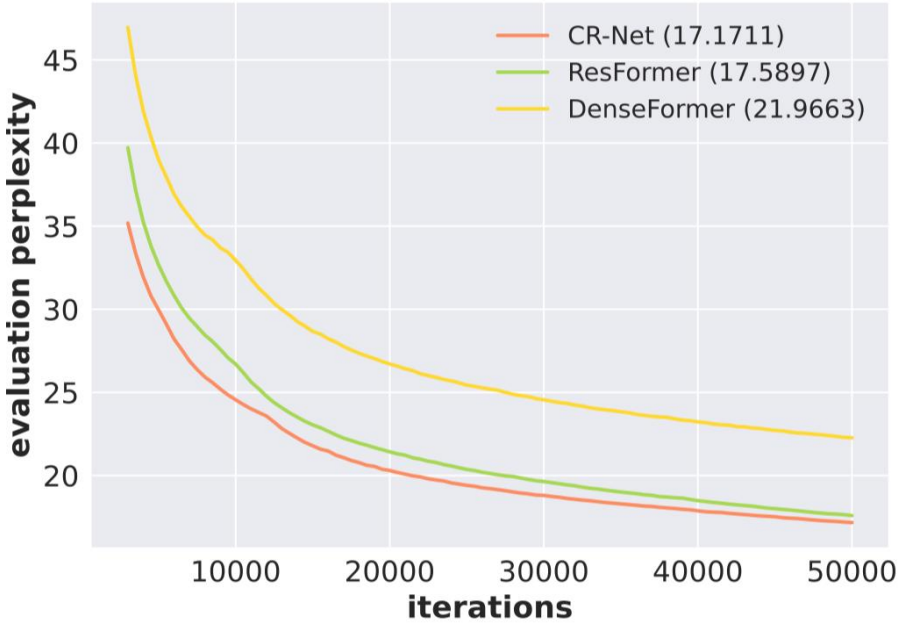
CR-Net: Efficient in both memory and compute

北京大学 PEKING UNIVERSITY

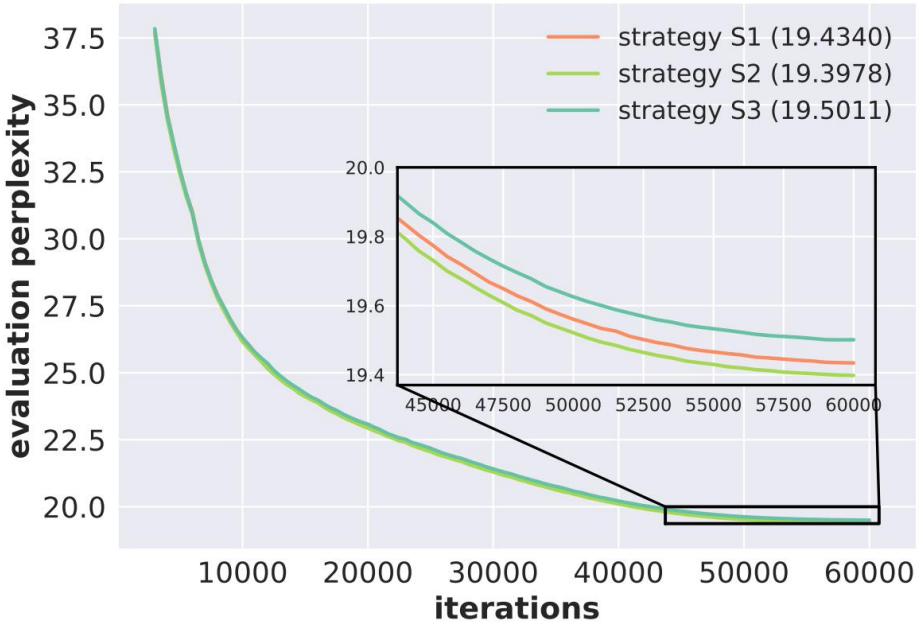
- Memory and compute complexity across methods (LLaMA-2 1B/7B, BF16)
- In CR-Net, b = number of stored full activation layers (‡: b=4, ♯: b=1)
- CoLA-M rank follows the literature; CR-Net rank r is tuned for best validation perplexity

Algorithms	LLaMA-2 1B		LLaMA-2 7B	
	Memory (GB)	FLOPs ($\times 10^{14}$)	Memory (GB)	FLOPs ($\times 10^{15}$)
Full-rank + Vanilla GCP	11.98 (1.000×)	2.133 (1.000×)	51.22 (1.000×)	2.119 (1.000×)
CoLA-M	12.04 (1.005×)	0.764 (0.358×)	24.78 (0.484×)	0.752 (0.355×)
CR-Net‡	11.81 (0.986×)	0.703 (0.330×)	23.35 (0.456×)	0.692 (0.326×)
CR-Net♯	9.94 (0.830×)	0.713 (0.334×)	22.42 (0.438×)	0.702 (0.331×)
Full-rank *w.o.* GCP	51.31 (4.283×)	0.764 (0.370×)	70.97 (1.386×)	1.608 (0.759×)

CR-Net: Ablations



Comparsion with ResFormer and DenseFormer



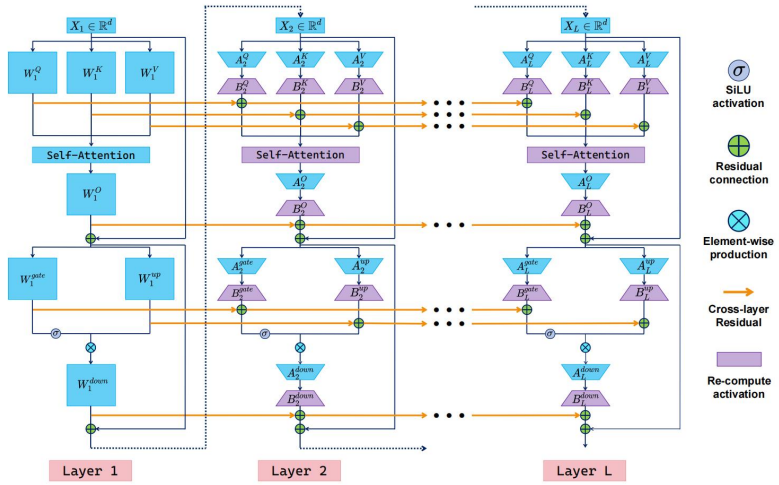
Comparsion with different low-rank strategy

R. Tian, et. al., ResFormer: Scaling ViTs with Multi-Resolution Training, CVPR 2023

Matteo Pagliardini, et. al., DenseFormer: Enhancing Information Flow in Transformers via Depth Weighted Averaging, NeurIPS 2024

Summary

- Parameter matrices are not low-rank; direct low-rank approximation leads to high error.
- **Core Idea**: Low-rank cross-layer residuals enable parameter-efficient architecture
- **Recomputation:** Tailored recomputation cuts activation memory, reducing compute by 66.6% at equal memory



- **Emprical performance**: Co-optimizes memory and compute: better performance with 43.6% fewer parameters (1B) and 38.5% less memory (7B)

北京大学
PEKING UNIVERSITY

Thanks!

ArXiv

Openreview