Encoder-Decoder or Decoder-Only? Revisiting Encoder-Decoder Large Language Model

Biao Zhang∗ , Yong Cheng, Siamak Shakeri, Xinyi Wang† , Min Ma, Orhan Firat

Google Deepmind

2025.10.30

Paper: https://arxiv.org/pdf/2510.26622

채진영

Contributions

- RedLLM and DecLLM show similar scaling exponents, **while DecLLM almost dominates the compute-optimal frontier**.
- During pretraining, **RedLLM performs badly at zero-shot learning**; its few-shot capability scales slightly with model sizes **but still lags far behind DecLLM**.
- After instruction tuning, **RedLLM achieves comparable zero- and few-shot performance** to DecLLM across scales while enjoying **significantly better inference efficiency**.
- At finetuning, **RedLLM benefits from the bidirectional attention in its encoder**; adapting DecLLM with this structure also yields significant improvements. Still, RedLLM provides **the overall best quality-efficiency trade-off**.
- RedLLM also shows promising **context-length extrapolation capability**.

Background - Why revisit Encoder-Decoder vs. Decoder-Only Architectures?

- Large Language Models (LLMs) have succeeded due to scalable, general-purpose architectures that learn from massive unlabeled data.

- Two major architectures in language modeling:
    a. **Encoder-Decoder:** Separate encoder for understanding input, decoder for generating output (e.g., T5).
    b. **Decoder-Only:** Single module for both understanding and generation (e.g., GPT).

- Recent trend **strongly favors decoder-only models** (e.g., LLaMA, Gemma, Mistral), mostly due to GPT's success — **not because encoder-decoder is inferior.**

- Prior research suggests encoder-decoder can outperform decoder-only when given **comparable compute and proper objectives** (e.g., UL2), but these comparisons ignored **scaling behavior**, which is central to modern LLM development.

Background - Revisiting Encoder-Decoder LLMs at Scale (RedLLM)

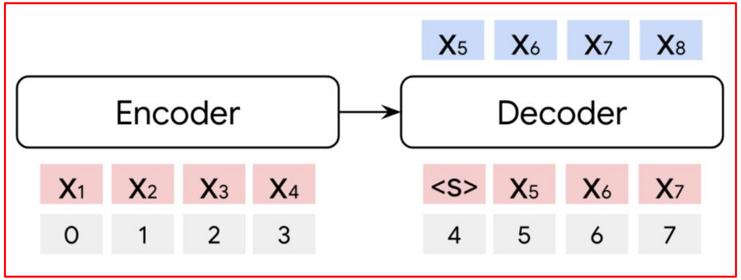
- This work re-evaluates encoder-decoder LLMs (RedLLM) vs decoder-only LLMs (DecLLM) **through scaling analysis**.
- **Approach:**
    a. Improve encoder-decoder modeling using recent techniques:
        i. Rotary positional embeddings with continuous positions
        ii. Prefix language modeling objective
    b. Pretrain models (150M–8B parameters) on **1.6T tokens (RedPajama V1)**.
    c. Instruction-tune models on **FLAN**.
- **Evaluated on:**
    a. Zero-shot and few-shot learning across 13 tasks
    b. In-domain and out-of-domain scaling performance
    c. Finetuning and inference efficiency trade-offs

- **Goal:** Understand **quality vs. efficiency trade-offs** to determine when encoder-decoder or decoder-only architectures are preferable.

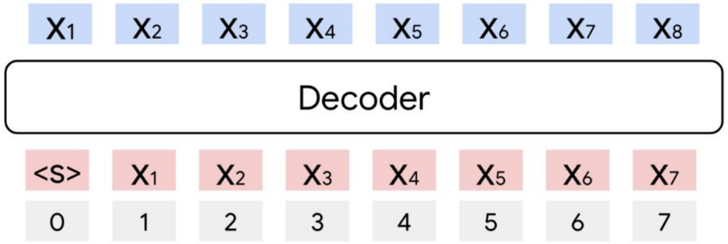
RedLLM vs. DecLLM

- Revisiting Encoder-Decoder LLMs at Scale vs. Decoder-Only LLM

rotary embedding



(a) RedLLM



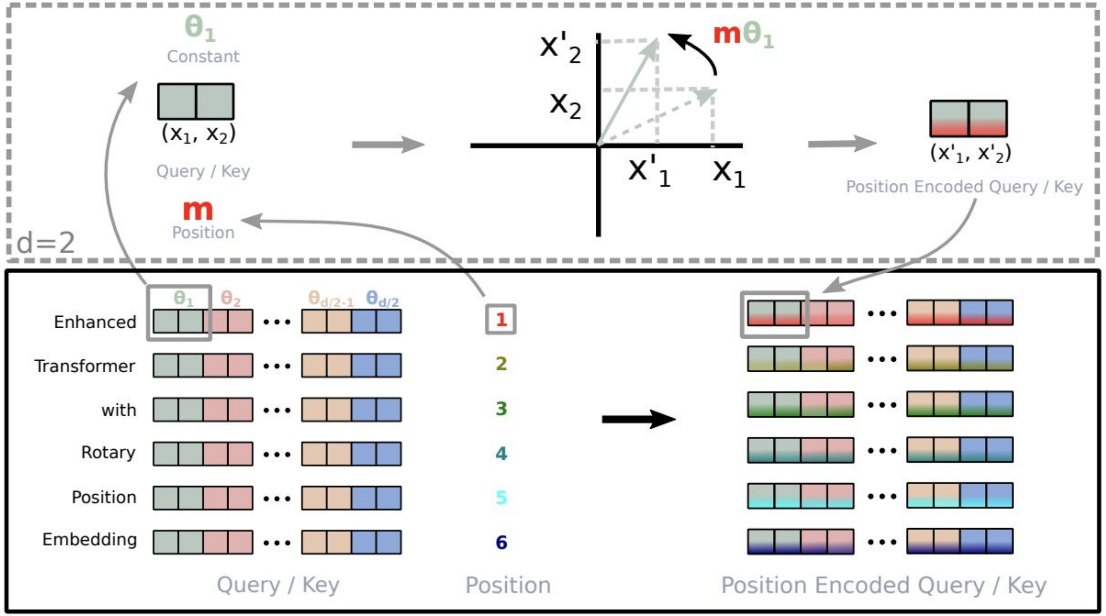
(b) DecLLM

	DecLLM	RedLLM
Attention	Multi-Head Dot-Product Attention	
FFN Activation	SwiGLU	
LayerNorm	RMSNorm (Pre-Normalization)	
Position		
Modeling	Rotary Embedding	
Type	Continuous Position	
Embeddings	All Tied	
Extra Norm	Q, K, V	Q, K, V, Attn Output
Rotary Usage	Self-Attention	Self&Cross-Attention
Loss	Causal LM	Prefix LM

(c) Model specification

Appendix.

1. absolution position embedding
2. relative position embedding
3. rotary embedding



RedLLM vs. DecLLM

Model Size	d	d_{ffn}	h	d_h	L_{dec}	L_{red}
150M	1024	4096	8	128	8	3/3
1B	2048	8192	16	128	16	7/7
2B	2560	10240	20	128	20	9/9
4B	3072	12288	24	128	24	10/10
8B	4096	16384	32	128	32	14/14

(a) Configurations for different-sized LLMs.

	Training Flops		#Params	
	Dec	Red	Dec	Red
RedPajama	0.20	0.24	0.17	0.18
Paloma	0.24	0.27	0.20	0.20

(b) Fitted scaling exponents.

	Pretraining	Finetuning
Vocabulary	32768	
Dataset	RedPajama V1	FLAN
Steps	400K	190K
Batch Size	2048	1024
Sequence Length	DecLLM: 2048 RedLLM: 1024/1024	2048/512
Optimizer	Adafactor(decay=0.8)	
LR Schedule	2k-step warmup to 0.01 + cosine decay by 0.1	fixed, 0.001
Gradient Clip	1.0	
Dropout	0.0	0.05
Z-Loss	0.0001	
Precision	bfloat16	

(c) Hyperparameters for pretraining and finetuning.

Setup

- **Models & Training**
  a. Train both RedLLM (encoder-decoder) and DecLLM (decoder-only) from **150M → 8B parameters**.
  b. Pretrained for **400K steps** (~1.6T tokens total).

- **Pretraining**
  a. Data: **RedPajama V1** (open reproduction of LLaMA corpus).
  b. **32K subword tokenizer**.
  c. **DecLLM:** contiguous 2048-token sequences, **RedLLM:** split into **1024 input / 1024 target** tokens.

Setup

- **Finetuning (Instruction Tuning)**
  a. Dataset: **FLAN** (1800+ diverse tasks).
  b. Max input/output: **2048 / 512 tokens**.
  c. Full-parameter tuning; loss on target output only.

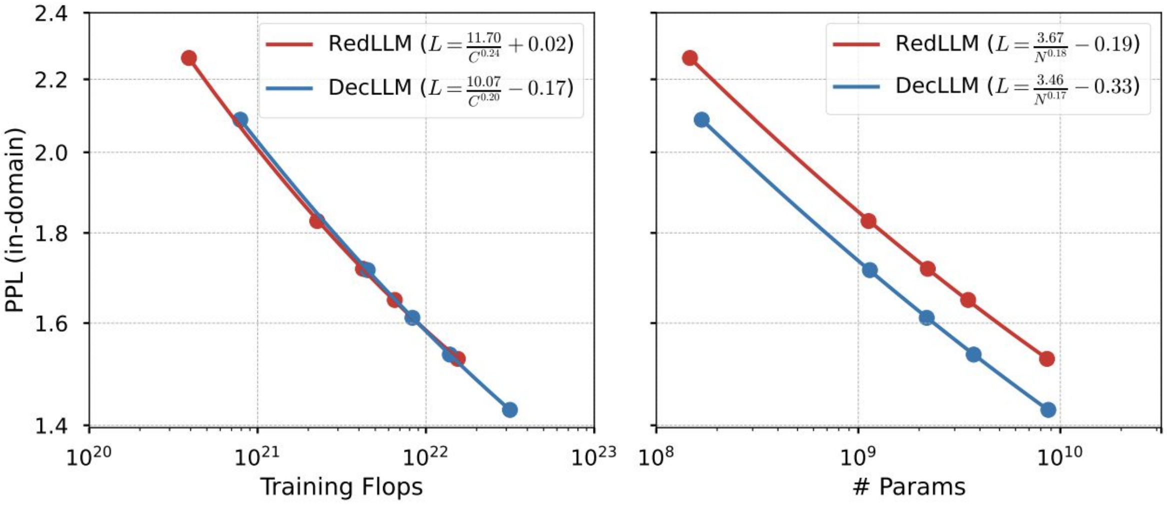
- **Evaluation**
  a. **Scaling analysis:** Perplexity on **RedPajama** (in-domain) and **Paloma** (out-of-domain)
  b. **Task performance:** Zero-shot & few-shot on **13 benchmark tasks** (e.g., BoolQ, ANLI, MMLU, GSM8K, WMT).
  c. Report **accuracy** (or **ChrF** for WMT), using **greedy decoding**

Results - Pretraining

- **RedLLM and DecLLM scale at similar rates when increasing the compute (training Flops) and model parameters.**

	Training Flops		#Params	
	Dec	Red	Dec	Red
RedPajama	0.20	0.24	0.17	0.18
Paloma	0.24	0.27	0.20	0.20

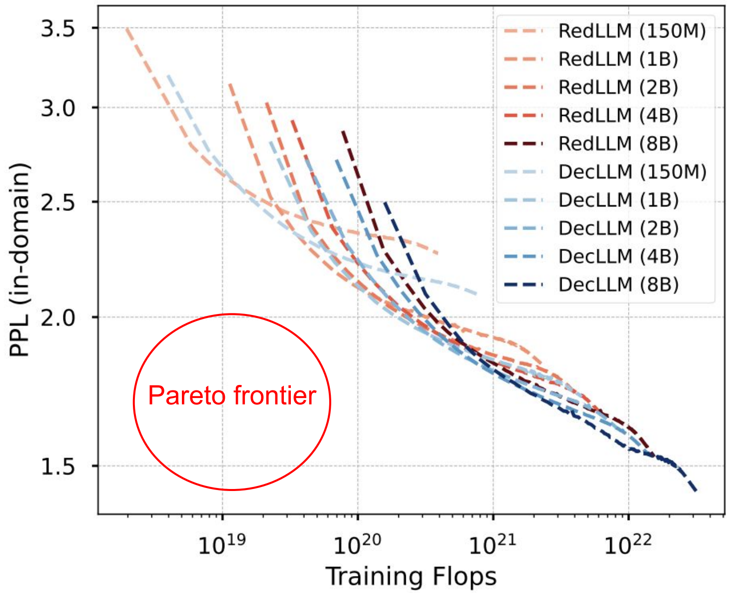
scaling exponents

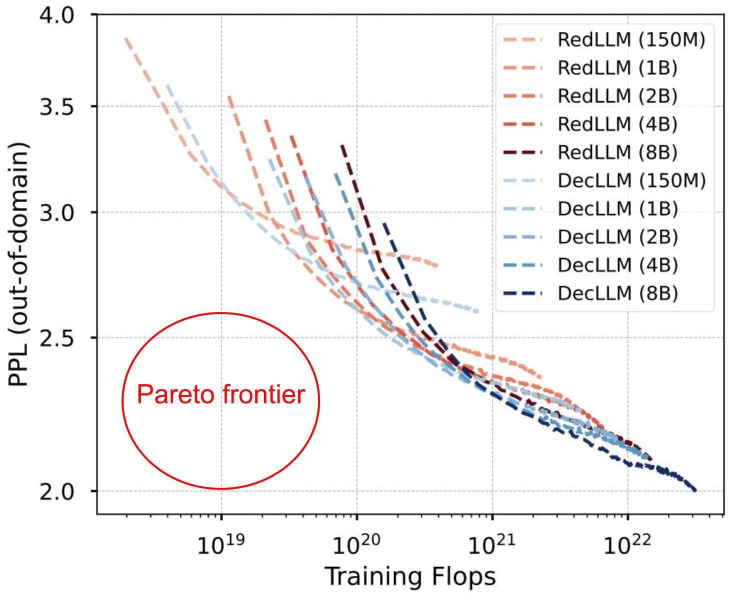


Evaluation on in-domain set (RedPajama)

Results - Pretraining

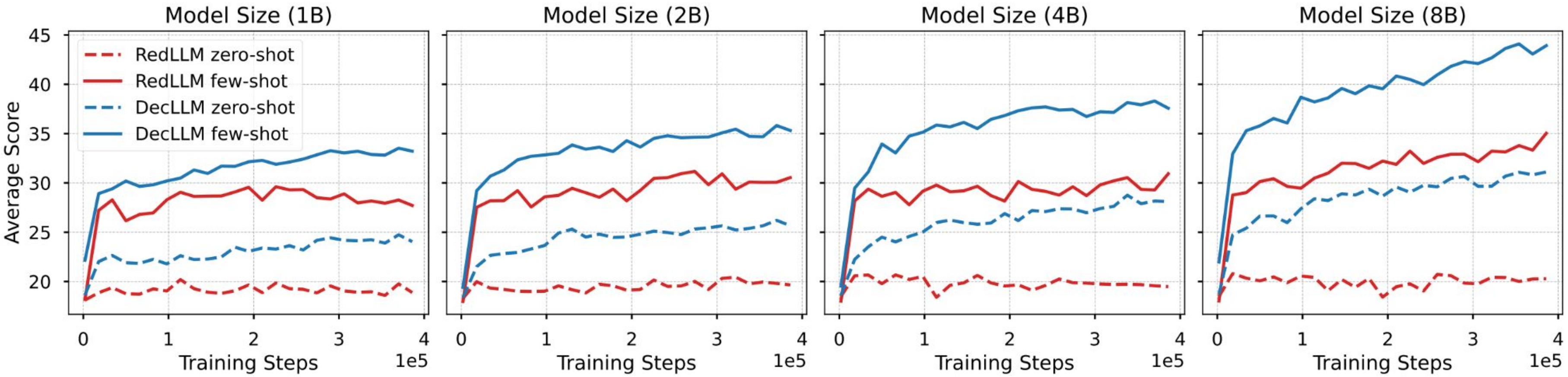
- **RedLLM lags behind DecLLM for compute-optimal training.**





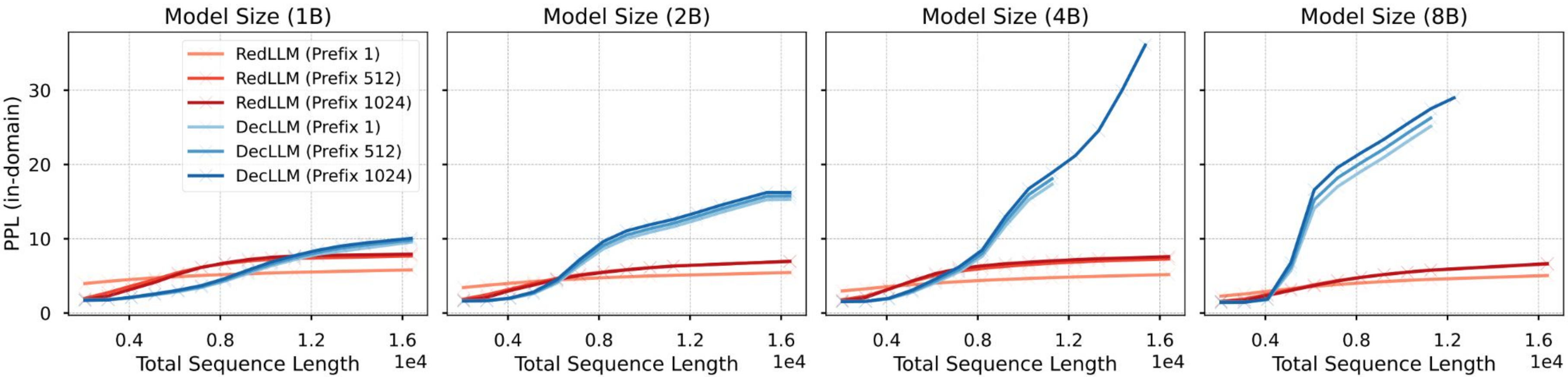
Results - Pretraining

- **RedLLM scales slightly for few-shot learning but is poor at zero-shot; both largely underperform DecLLM.**



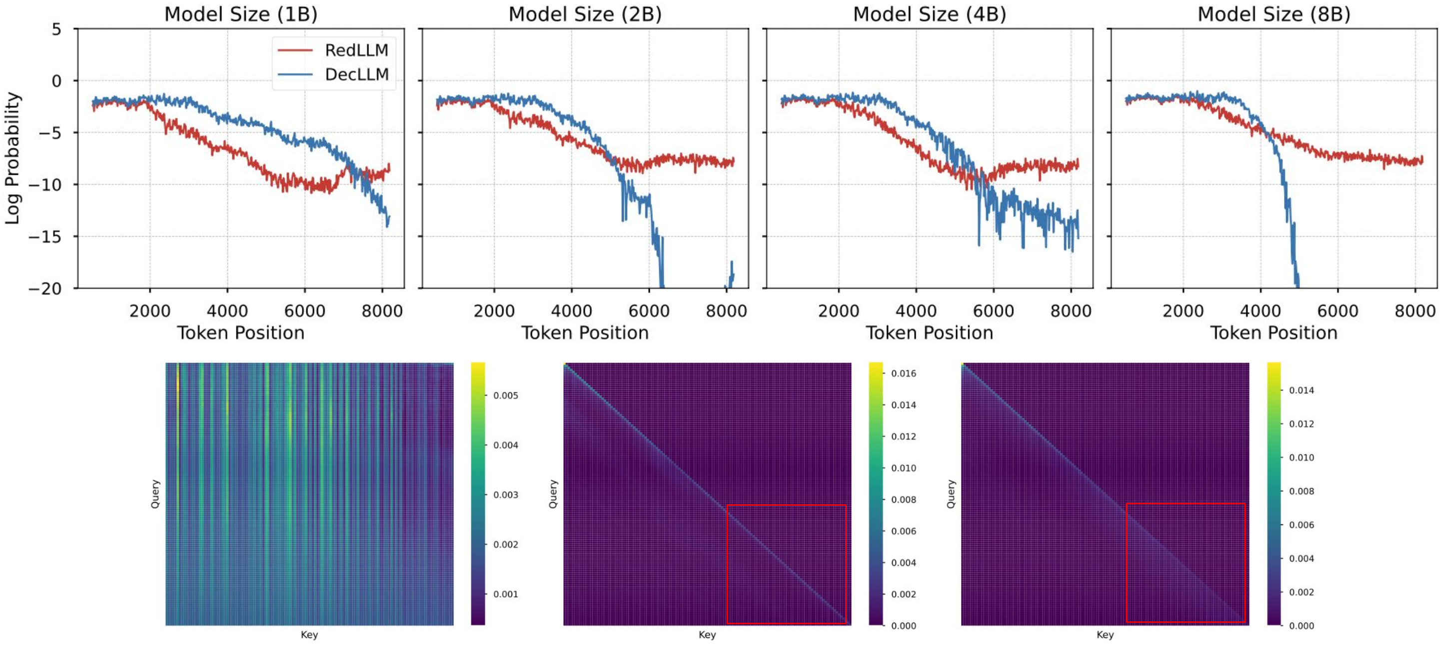
Results - Pretraining

- **RedLLM shows comparable and even better length extrapolation capability than DecLLM along model scaling.**



Results - Pretraining

- **The decoder self- and cross-attention in RedLLM show intriguing patterns under long context.**

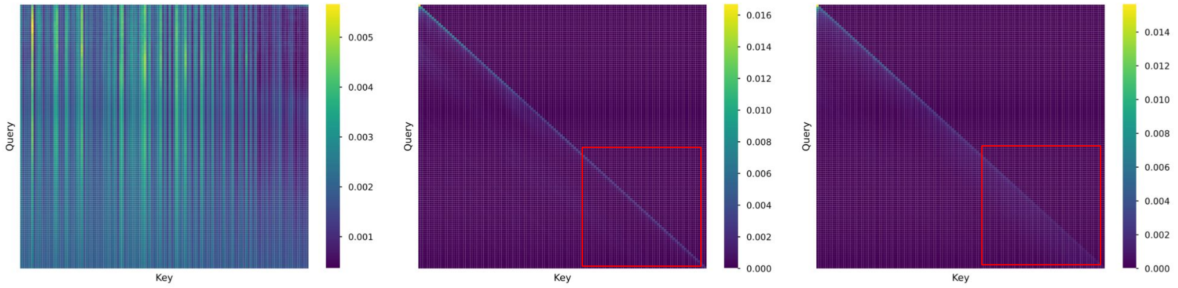


(a) RedLLM: cross-attention. (b) RedLLM: self-attention. (c) DecLLM: self-attention.

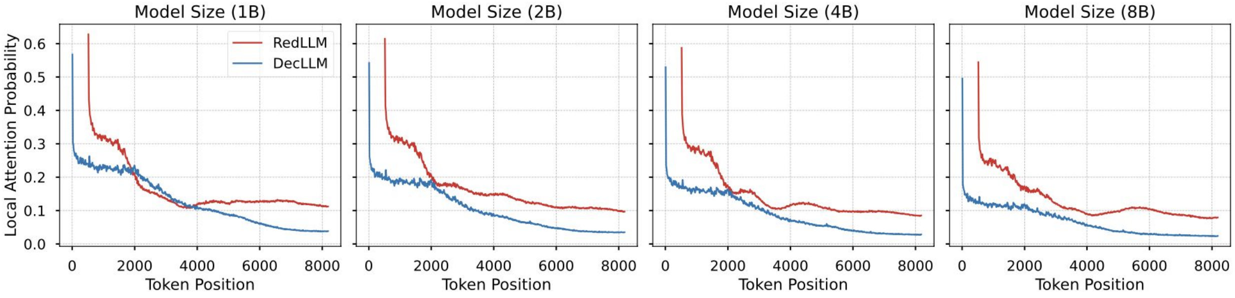
→ strength of locality weakens with the increase of token position

Results - Pretraining

- **The decoder self- and cross-attention in RedLLM show intriguing patterns under long context.**



(a) RedLLM: cross-attention. (b) RedLLM: self-attention. (c) DecLLM: self-attention.



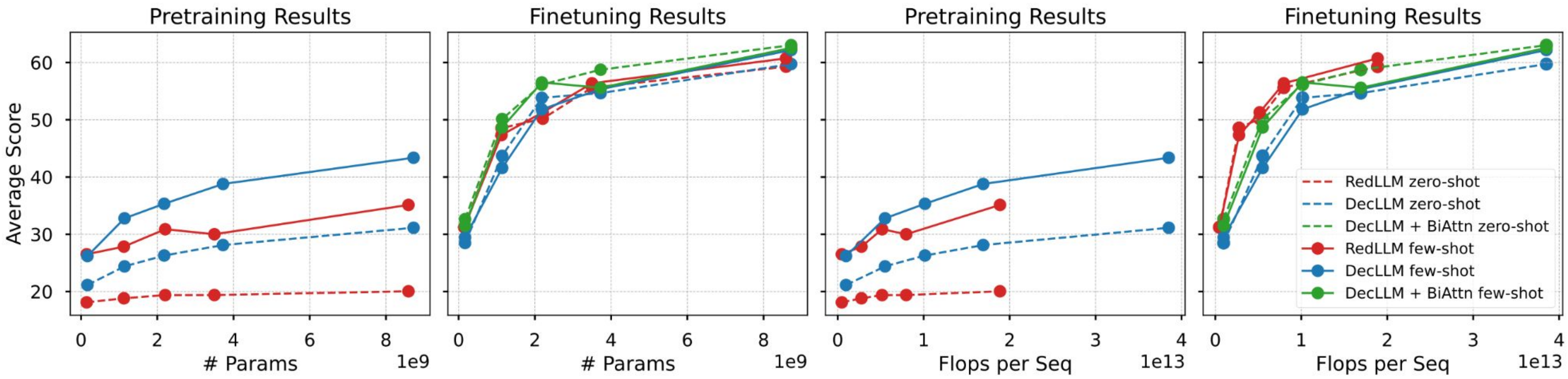
Results - Finetuning

- **RedLLM shows high adaptability: matching and even surpassing DecLLM across scales after finetuning.**

Setup			150M	1B	2B	4B	8B
Pretraining	Zero-Shot	RedLLM	18.11	18.82	19.38	19.39	20.04
		DecLLM	21.14	24.39	26.29	28.12	31.13
	Few-Shot	RedLLM	26.51	27.84	30.88	30.01	35.13
		DecLLM	26.21	32.79	35.33	38.79	43.37
Finetuning	Zero-Shot	RedLLM	31.23	48.55	50.19	55.61	59.69
		DecLLM	29.97	43.70	53.84	54.63	58.26
		+ BiAttn	33.73	50.12	56.15	58.07	63.03
	Few-Shot	RedLLM	31.24	47.32	51.30	56.37	61.32
		DecLLM	30.14	41.58	51.82	57.22	59.02
		+ BiAttn	31.50	48.13	56.52	55.95	62.54

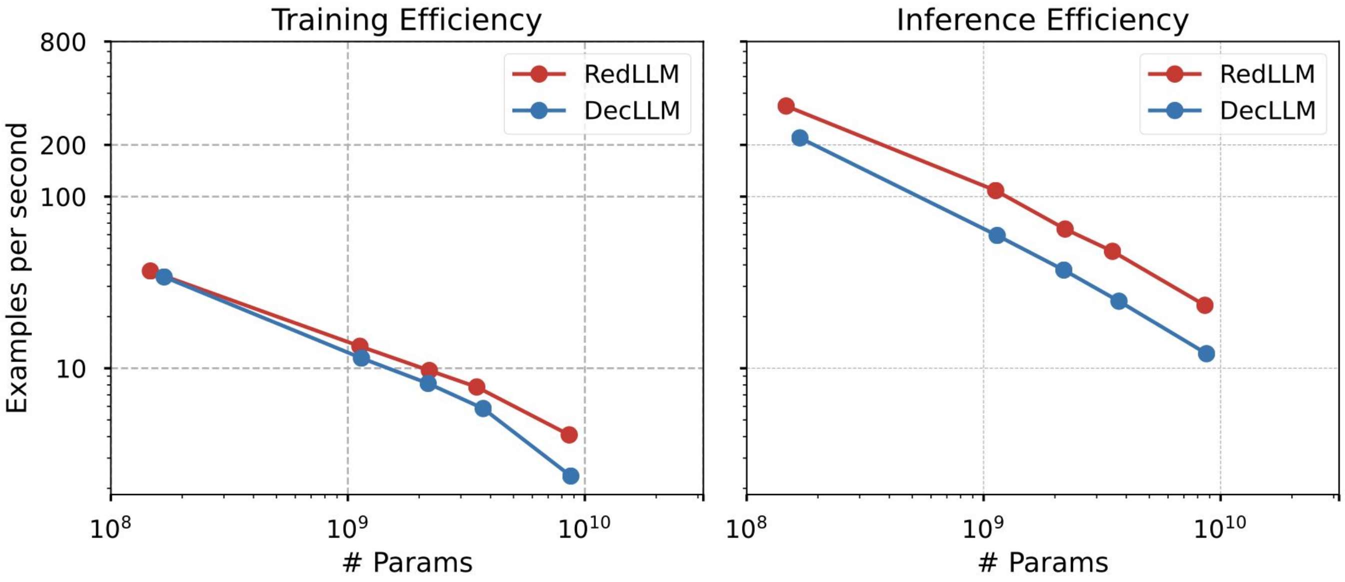
Results - Finetuning

- **Bidirectional input attention improves DecLLM greatly but doesn't change the quality compute frontier.**



Results - Finetuning

- **RedLLM has clear advantage over DecLLM on training and inference efficiency.**



Discussion

-