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

arXiv:2510.26622

Zhang et al.

November 27, 2025

Agenda

1. Introduction & Context
2. Architecture (RedLLM)
3. Experiments
4. Analysis
5. Conclusion

The Status Quo: Decoder-Only Dominance

- **Current Landscape:** The vast majority of modern LLMs (LLaMA, Mistral, GPT-4) utilize **Decoder-Only (DecLLM)** architectures.
- **The Question:** Is this dominance due to inherent architectural superiority, or is it an evolutionary artifact?
- **Historical Context:**
  - *Encoder-Only:* Resurfacing with ModernBERT (GeGLU, RoPE, Unpadding).
  - *Encoder-Decoder:* T5, UL2, and recently T5Gemma.
- **Goal:** A rigorous scaling analysis (150M to 8B) comparing modernized Encoder-Decoder (**RedLLM**) vs. Decoder-Only (**DecLLM**).

Modernizing the Architecture: RedLLM

- **RoPE (Rotary Positional Embeddings):** Applied everywhere (Encoder Self-Attn, Decoder Self-Attn, Cross-Attn).
- **RMSNorm** and **SwiGLU** activations.
- **Stabilization:** Add. normalization for RedLLM due to training instability.

	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

Experimental Setup

Scales:

- 5 scales: 150M $\rightarrow$ 8B parameters.
- **Constraint:** RedLLM maintains a balanced architecture (equal encoder/decoder layers).

Data:

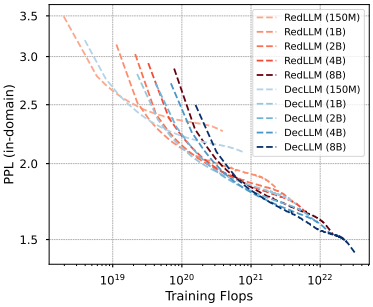
- **Pre-training:** RedPajama V1 (1.6T tokens).
- **Fine-tuning:** FLAN (Instruction Tuning).

Pre-training Objectives:

- **DecLLM:** Standard Causal Language Modeling (Next token prediction).
- **RedLLM:** Prefix Language Modeling (Prefix LM).
  - First half: Encoder input (Bidirectional attention).
  - Second half: Decoder target (Causal attention).
  - *Note:* Effective target tokens for RedLLM are half that of DecLLM (0.8T vs 1.6T).

Phase 1: DecLLM Dominates Pre-training

1. **Compute Optimality:** DecLLM achieves lower perplexity for the same compute budget.
   - RedLLM requires approx. 2× FLOPs to match DecLLM perplexity.
2. **Zero/Few-Shot Capabilities:**
   - DecLLM is significantly stronger immediately after pre-training.
   - *Example (8B scale):* DecLLM few-shot score **43.37** vs. RedLLM **35.13**.



Phase 2: Post-finetuning Turnaround

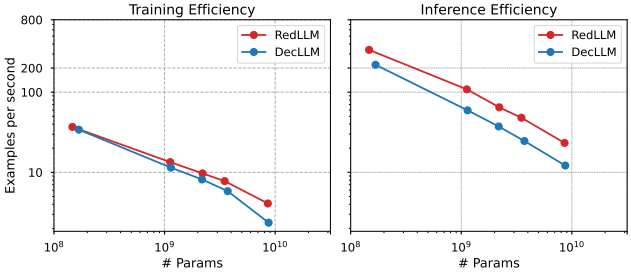
Setup			150M	1B	2B	4B	8B
Pretraining	Zero-Shot	RedLLM	18.11	18.82	19.38	19.39	20.04
		DecLLM	21.14	24.39	26.29	28.12	31.13
	Few-Shot	RedLLM	26.51	27.84	30.88	30.01	35.13
		DecLLM	26.21	32.79	35.33	38.79	43.37
Finetuning	Zero-Shot	RedLLM	31.23	48.55	50.19	55.61	59.69
		DecLLM	29.97	43.70	53.84	54.63	58.26
		+ 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

- **Performance Catch-up:** RedLLM closes the gap and often surpasses DecLLM.

The Pareto Frontier: Quality vs. Efficiency

The strongest argument for RedLLM is **Inference Efficiency**.

- For a given latency/throughput budget, RedLLM provides higher quality.
- **Throughput:** RedLLM shows higher tokens/sec during both training and inference compared to parameter-matched DecLLM.



Throughput comparison

Empirically, RedLLM achieves higher throughput.

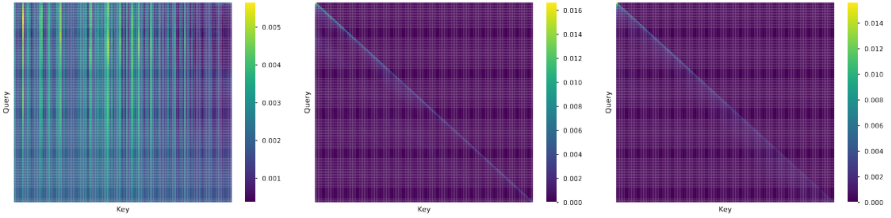
Feature	DecLLM (8B)	RedLLM (8B)
Structure	32 Layers	14 Enc + 14 Dec Layers
Block components	Self-Attn + MLP	(Self + Cross) + MLP
Params per block	Fewer	More (due to Cross-Attn)
Total Blocks	**32**	**28**

- To match parameter counts with "heavier" decoder blocks (Self+Cross+MLP), RedLLM uses fewer total layers.
- **Prompt Processing:** Encoder processes prompt in parallel (Bi-directional).
- **Generation:** Decoder Self-Attn context grows from 0 to N_{gen} (not $N_{prompt} + N_{gen}$), potentially reducing quadratic overhead.

Long Context Extrapolation

The Problem: Locality Decay

On long contexts, self-attention in decoder deterioriates with token position ("Locality Decay") $\rightarrow$ alleviated by cross-attention in RedLLM.

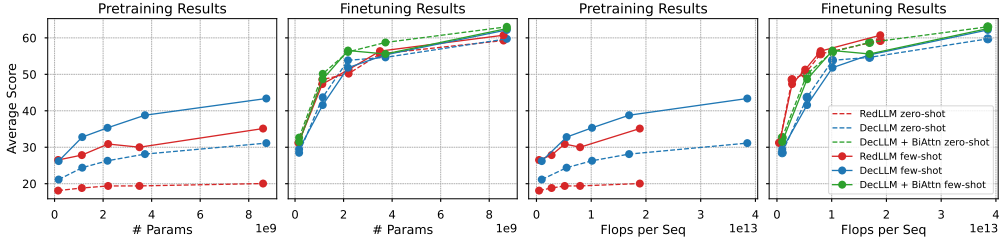


(a) RedLLM: cross-attention. (b) RedLLM: self-attention. (c) DecLLM: self-attention.

DecLLM + BiAttn

Hypothesis: The Encoder's advantage is Bidirectional Attention on the prompt.
Experiment: Enable Bidirectional Attention for prefix tokens in DecLLM.

- **Result:** Significant performance improvement for DecLLM.
- RedLLM still retains the edge in the Quality/Efficiency trade-off.



Summary & Takeaways

1. **Pre-training:** DecLLMs look better in pre-training (Compute-optimal, better Zero-shot).
2. **Fine-tuning:** After Instruction Tuning, Encoder-Decoder architectures match or beat Decoder-only models.
3. **Efficiency:** RedLLM turns out superior in Inference FLOPs vs. Quality.
4. **Mechanism:**
   - Higher throughput due to bidirectional prompt encoding.
   - Cross-attention enables better long-context robustness.