EPIC: Efficient Position-Independent Caching for Serving Large Language Models

Junhao Hu[1], Wenrui Huang[2], Weidong Wang[2], Haoyi Wang[1], Tiancheng Hu[1], Qin Zhang[3]
Hao Feng[3], Xusheng Chen[3], Yizhou Shan[3], Tao Xie[1]

Peking University[1], Nanjing University[2], Huawei Cloud[3]

LLM Context Caching Challenge

- In LLM serving, immutable chunks (like system messages, few-shot examples, and documents) are frequently repeated across requests.
- Traditional context caching **(Prefix-Based Context Caching)** reuses Key-Value (KV) vectors but requires exact prefix matches, limiting reuse cases.

Key Challenge
Existing context caching methods fail to efficiently reuse immutable chunks when preceded by varying prefixes.

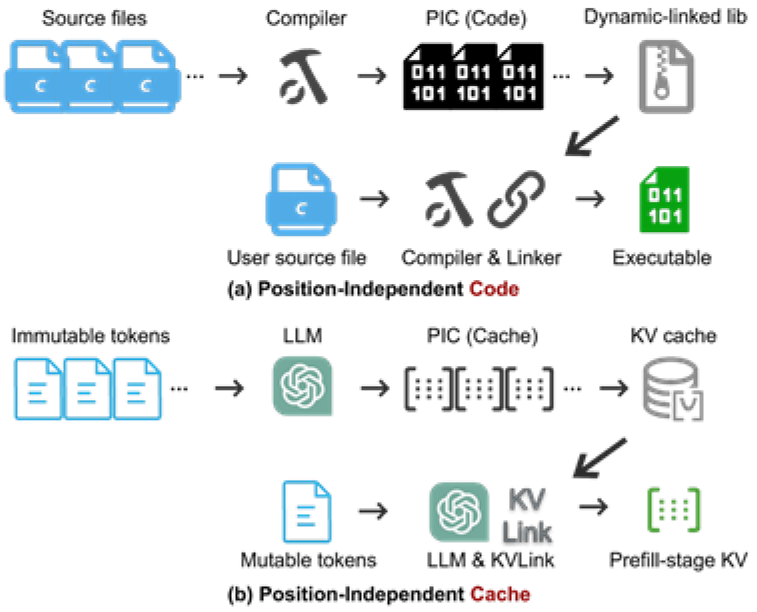
Position-Independent Caching (PIC)

Key Challenge
Existing context caching methods fail to efficiently reuse immutable chunks when preceded by varying prefixes.

Position-Independent Caching (PIC) enables modular reuse of KV vectors regardless of prefixes

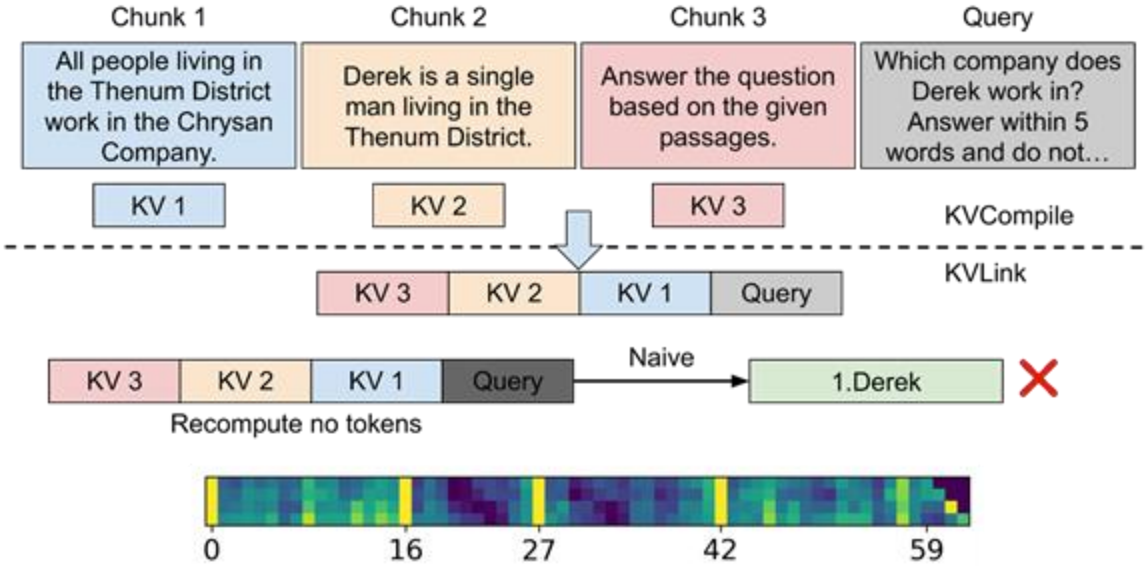
Two-Step Framework: Compile + Link

Next: Some approaches for PIC



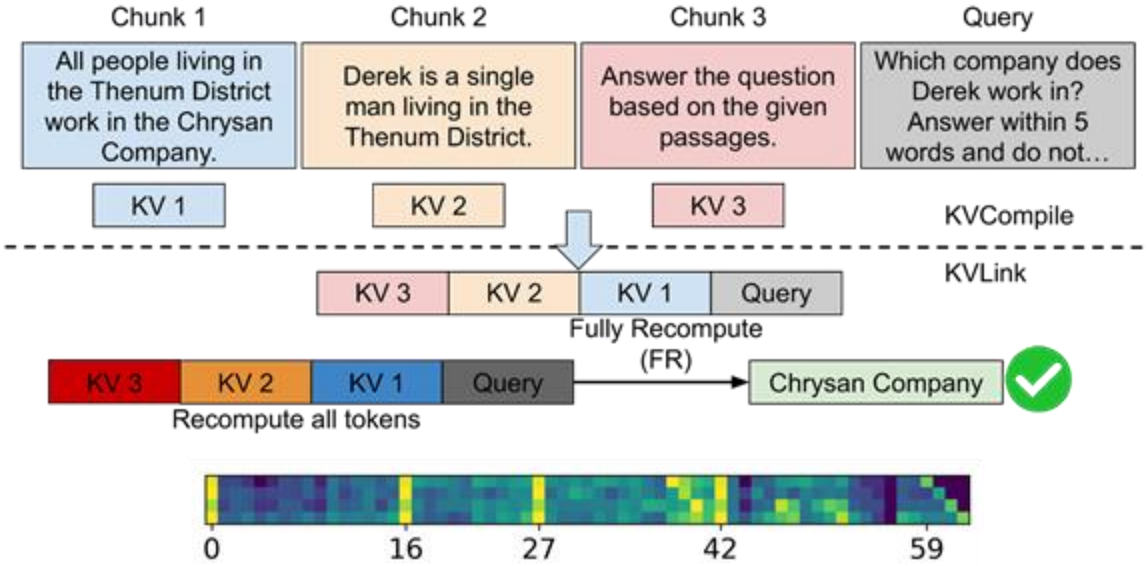
Approaches for PIC — *Naive*

Naive: $O(1)$ link time; low accuracy. Most attention scores concentrate on each chunk's initial tokens, exhibiting the "attention sink" phenomenon.



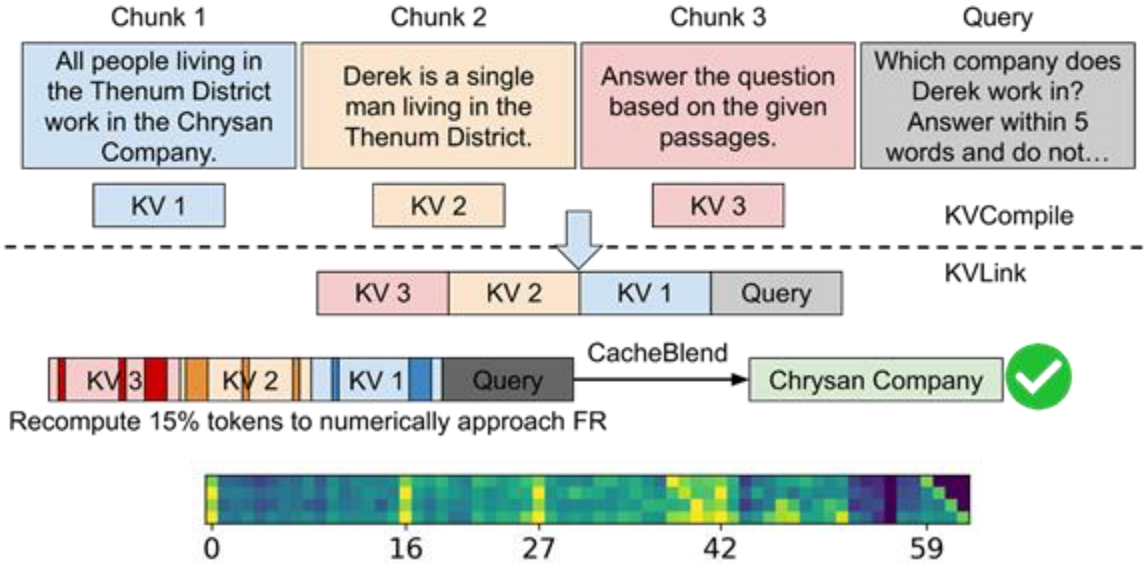
Approaches for PIC — *Fully Recompute*

Fully Recompute (FR): $O(N^2)$ link time; full accuracy. Each chunk's initial tokens release part of their attention to more relevant positions.



Approaches for PIC — *CacheBlend*

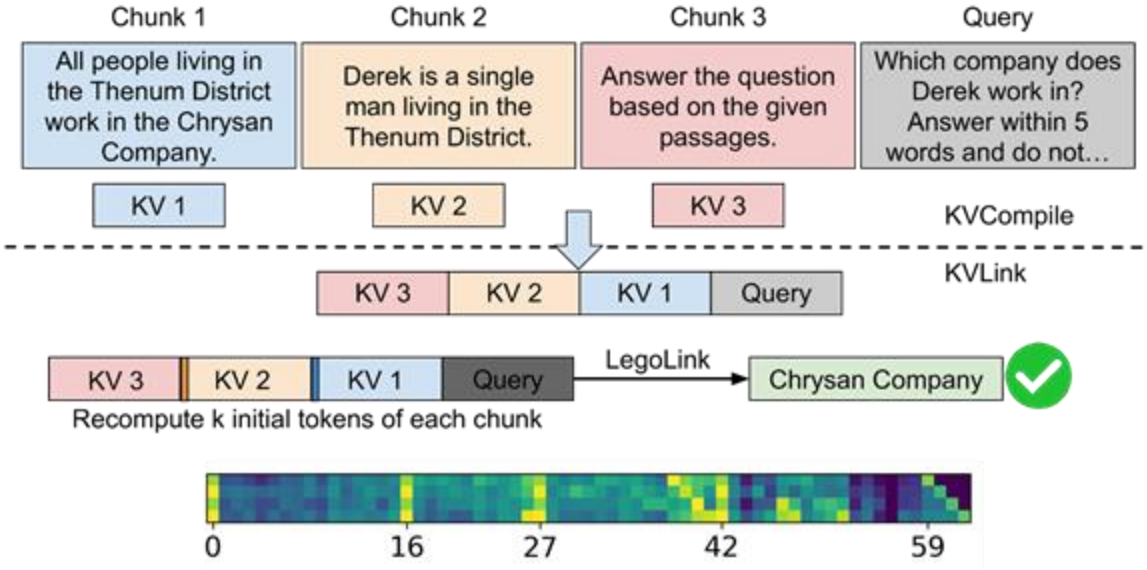
CacheBlend: $O(15\% N^2)$ link time; ~full accuracy. *CacheBlend* approximates FR's attention map by selectively recomputing only 15% of tokens with the largest deviation from the FR. These selected tokens often include initial tokens of each chunk



Our Approach – *LegoLink*

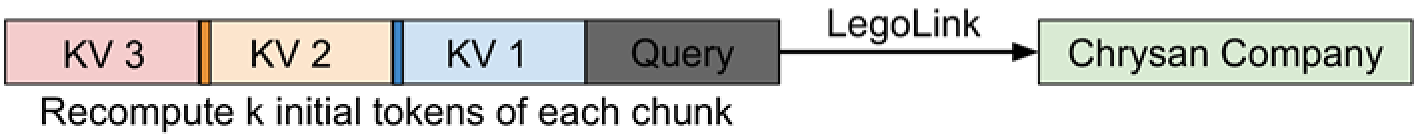
LegoLink: $O(kN)$ link time; ~full accuracy. *LegoLink* allows initial tokens of latter chunks to recognize their non-initial positions and crippling their attention-sink ability

Next: More details for *LegoLink*



LegoLink Details

- Attention Sink (Xiao et al, 2024): Initial tokens of each chunk disproportionately absorb attention
- Recomputing k initial tokens of each chunk (except the first chunk) allows these tokens to recognize their non-initial positions
- EPIC — Our serving system based on *LegoLink* (More details in appendix)



Benefits of *LegoLink*

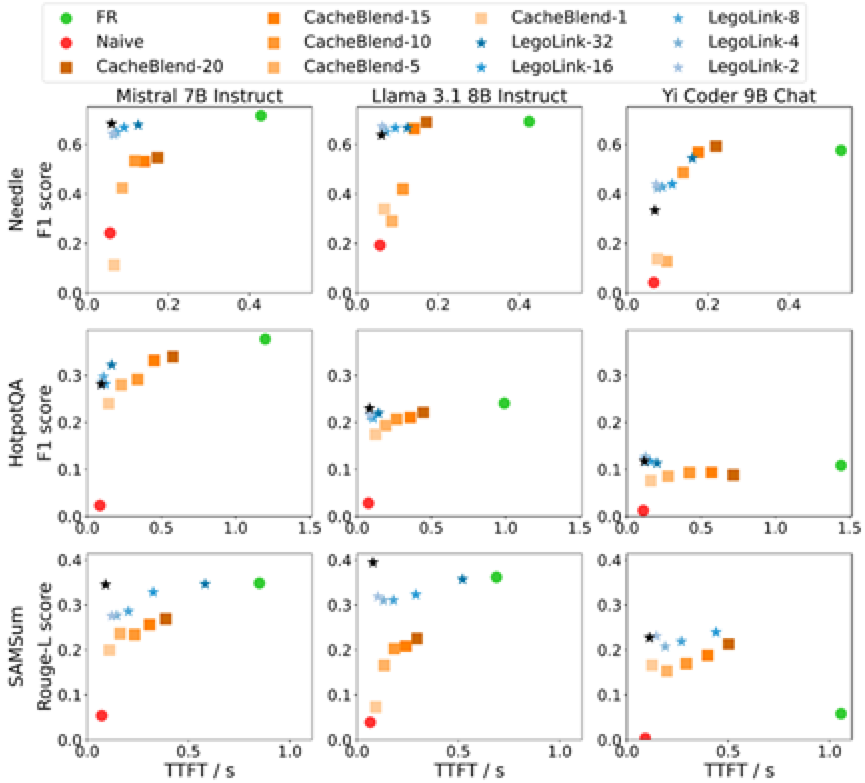
1. Linear link complexity, $O(kN)$, with negligible accuracy loss
2. Static token selection (compared with *CacheBlend*)

How Was EPIC Evaluated?

- Implemented based on vLLM 0.4.1 with 2K lines of Python code
- Evaluated on six datasets: 2WikiMQA, MuSiQue, SAMSum, MultiNews, HotpotQA, Needle in a Haystack
- Used three state-of-the-art open-source LLMs: Mistral 7B Instruct, Llama 3.1 8B Instruct, Yi Coder 9B Chat
- Compared against FR, Naive, and CacheBlend algorithms

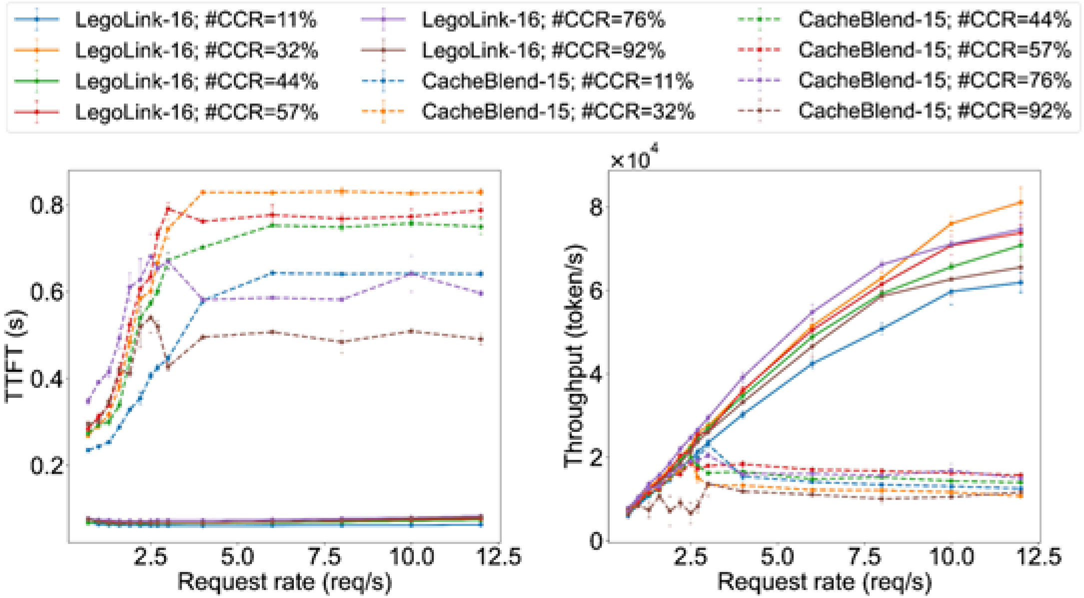
Evaluation — Accuracy vs TTFT

- *LegoLink* variants establish a new Pareto frontier, outperforming *CacheBlend* in most cases
- *LegoLink*-2 limits accuracy drops within 0-7% and reduces TTFT by up to 300% compared to *CacheBlend*-15
- Increasing recomputed tokens in *LegoLink* yields diminishing accuracy gains



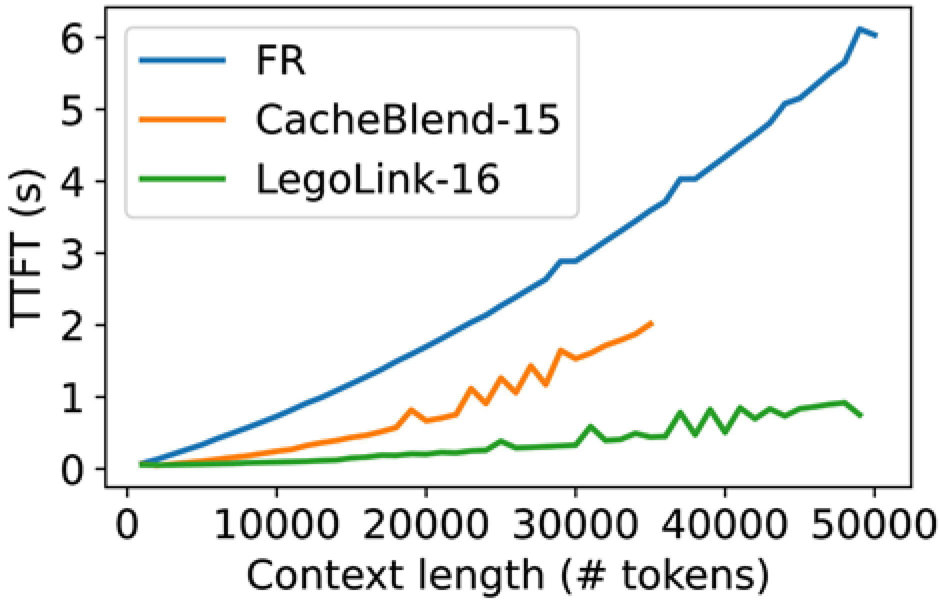
Evaluation — Latency and Throughput

- EPIC achieves up to 8× reduction in TTFT and 7× increase in throughput compared to existing systems
- Under asynchronous workloads, EPIC maintains stable performance as Context Cache Ratio (CCR) increases



Evaluation — Latency Under Long Context

- EPIC supports longer context lengths with smaller latency, without out-of-memory errors



What's the Significance of Our Work?

- Formalizes the PIC framework and advances the state of the art in this emerging area
- *LegoLink* significantly reduces recomputation complexity while maintaining accuracy
- EPIC demonstrates substantial improvements in serving performance for LLMs

Thank You!

Appendix: The EPIC Serving System