Qeios

Peer Review

Review of: "vPID: Virtual Processing-In-Disk Architecture for Sub-Millisecond Multi-Model LLM Inference Switching"

Kishor Kunal[1]

1. University of Minnesota, Minneapolis, United States

This paper presents a clever VPID (Virtual Processing-In-Disk). By treating NVMe storage as persistent memory via memory-mapped I/O, the system bypasses traditional model reloading, enabling ultra-low latency switches and $O(1)$ context queries through persistent KV caching. This is a highly specialized engineering workaround, but it is not a general-purpose breakthrough for LLM serving. The methodology fundamentally ignores broader industry trends. The current frontier of AI inference is moving toward aggressive quantization (e.g., INT4, FP8) and leveraging specialized hardware accelerators (like Tensor Cores) to maximize throughput.

Strengths

1. <1ms switching: Atomic pointer swaps eliminate reload lag.
2. No dequantization cost: Pre-computed FP32 weights speed up token generation.
3. O(1) context queries: Cached KV states enable instant follow-up questions.
4. Smart memory tiering: Auto-shifts active data to the GPU and inactive data to the SSD.

Weaknesses

1. Massive disk usage: Storing uncompressed FP32 weights requires huge amounts of storage. Instead of expanding weights to FP32, the system should pre-compute weights into FP16 or BF16, reducing storage requirements by 50%.
2. I/O bottlenecks: Reading uncompressed weights directly from an SSD hits PCIe speed limits. Rather than routing memory-mapped I/O through the CPU's system RAM, implementing Direct Memory Access (DMA) could allow direct streaming.

doi.org/10.32388/0M9ZKW

3. Slow first query: The upfront O(n^2) prefill makes initial ingestion computationally expensive.
4. Ignores modern hardware: Forcing FP32 avoids highly efficient, lower-precision accelerators (like INT8/FP8 Tensor Cores).

Declarations

Potential competing interests: No potential competing interests to declare.