KVBENCH: AN EXTENSIBLE BENCHMARK TO EVALUATE APPROXIMATE KV CACHE REUSE

Anonymous authors
Paper under double-blind review

ABSTRACT

Approximate KV-cache reuse accelerates large language model inference by reusing cached states beyond identical request prefixes. However, these states depend on their original context, so transferring them can alter generation quality, and the resulting quality–latency trade-off may vary across models, tasks, and request structures. We present **KVBench**, an extensible benchmark that separates tasks, models, reuse methods, and request workflows to support controlled evaluation of these effects. KVBench measures task quality and online time-to-first-token relative to full-prefill references across RAG-style context reuse, multi-agent communication, and agentic skill reuse, covering five models with full- or hybrid-attention architectures. Our evaluation reveals substantial task–model variation in quality retention and recovery through recomputation. Controlled studies further show that instruction placement can strongly affect approximate reuse even when full-prefill quality remains stable, and that reusable-context length affects the latency benefit. We also study reuse of independently prepared skill caches on SkillsBench tasks. HYPIC TRR retains 83.0% of the reward improvement provided by skills, while reducing first-call time-to-first-token to roughly 20–30% of full prefill on executions with large reusable-context fractions. These findings establish cache preparation and request structure as important evaluation dimensions and provide a common framework for studying the benefits and limitations of approximate KV-cache reuse.

1 INTRODUCTION

Large language model (LLM) inference increasingly operates on workloads with persistent and recurring context. Retrieval-augmented generation (RAG) systems repeatedly process overlapping documents (Lewis et al., 2020), while agentic systems revisit instructions, tool descriptions, demonstrations, and intermediate results across executions (Yao et al., 2023). In these settings, prefill can dominate time-to-first-token (TTFT), even when much of the input has already been processed in an earlier request (Jin et al., 2025; Yao et al., 2025; Ye et al., 2025).

KV-cache reuse addresses this redundancy by retaining key–value states from previous computation. Exact prefix caching reuses states when requests share an identical token prefix (Kwon et al., 2023; Zheng et al., 2024). Recurring content, however, need not appear under the same prefix: retrieved documents may be reordered, agent messages may enter different conversation histories, and skill instructions may follow task-specific inputs. Because cached states depend on preceding context, identical text can have different states across requests. Approximate reuse methods address this mismatch by transferring, adapting, or selectively recomputing cached states (Yao et al., 2025; Hu et al., 2025; Ye et al., 2025). These methods can avoid substantial prefill computation, but may also change generation quality.

Evaluating approximate reuse therefore requires answering two coupled questions: how does transferring cached states affect task quality, and does the avoided computation outweigh the online costs of cache loading, adaptation, and repair? A task dataset alone does not specify this evaluation. The same example can induce different reuse conditions depending on how source caches are prepared, where reusable segments appear in the target request, and which states are shared across calls. Moreover, the consequences of these choices can vary across models and tasks. Meaningful comparison

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

thus requires explicit reuse workflows and consistent quality and latency measurements, together with sufficient coverage to test whether conclusions extend beyond individual settings.

We introduce **KVBench**, an extensible benchmark for controlled evaluation of approximate KV-cache reuse. KVBench separates four components: the *Task* defines inputs and quality evaluation; the *Model* provides a common execution interface; the *Method* implements cache preparation and reuse; and the *Workflow* specifies request construction, sequencing, and state sharing. This decomposition supports comparisons across compatible components while allowing new tasks, models, methods, and workflows to be incorporated without rewriting unrelated components. The current benchmark covers RAG-style context reuse, multi-agent communication, and agentic skill reuse, evaluating task quality and online TTFT against paired full-prefill references.

Our evaluation shows that quality retention and recovery through recomputation vary substantially across tasks and models. For example, with CacheBlend on Qwen3-8B, recomputing 5% of tokens raises relative quality from 0.041 to 0.988 on variable tracing, but only from 0.403 to 0.763 on MuSiQue. Controlled studies further show that instruction placement can substantially affect approximate-reuse quality while leaving full-prefill quality nearly unchanged, and that reusable-context length influences the latency benefit. Under the evaluated CacheBlend configurations, aggregate cross-document dependency measures explain little additional variation in quality loss after controlling for task identity, reinforcing the need for direct task-level evaluation.

We further study *skill reuse*, in which independently prepared caches of skill instructions, demonstrations, and tool guidance are reused in task-specific agent inputs. This workload exposes recurring procedural context as a reuse opportunity, while requiring evaluation of complete task outcomes because changes to the initial computation can affect subsequent tool-use trajectories. On 87 SkillsBench tasks, HYPIC TRR retains 83.0% of the reward improvement provided by full skill prefilling. For executions with large reusable-context fractions, its first-call TTFT falls to roughly 20–30% of the full-prefill reference, excluding offline cache preparation.

Our contributions are:

1. We present **KVBench**, a modular and extensible benchmark that evaluates approximate KV-cache reuse across tasks, models, methods, and explicitly specified request workflows using a common quality and latency measurement protocol.
2. We provide a systematic empirical study of reuse quality and efficiency, revealing task- and model-dependent behavior and identifying instruction placement and reusable-context length as important evaluation dimensions, which can affect performance significantly.
3. We extend evaluation to *agentic skill reuse* on 87 SkillsBench tasks, measuring task-level reward and first-call latency when reusing independently prepared skill caches. HYPIC TRR retains 83.0% of the improvement by full skill prefilling while reducing first-call TTFT by roughly 70–80% on executions with large reusable-context fractions, highlighting the practical potential of skill reuse and the remaining room for improving quality retention.

2 Related Work

Exact prefix caching is standard in LLM serving systems, including vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024), while RAGCache (Jin et al., 2025) extends this paradigm to RAG serving. For each request, exact prefix caching reuses the longest identical cached token prefix and prefills only the unmatched suffix. This preserves full-prefill outputs, but any token mismatch terminates the shared prefix.

Recent work explores approximate reuse, enabling cached chunks to be reused in new contexts. As summarized in Table 1, selective recomputation, exemplified by CacheBlend (Yao et al., 2025), repairs the resulting context mismatch. Other methods modify the reusable cache: KVCOMM adjusts shared KV caches online across agents, while KVPacket composes immutable caches through soft-token adapters (Ye et al., 2025; Chen et al., 2026). For hybrid models, HYPIC combines transition-operator state composition with seam-window repair, whereas LinearKV initializes each linear layer from a single cached state while retaining selective recomputation reuse methods in the full-attention layers (Liu et al., 2026a) (Liu et al., 2026b).

Table 1: Taxonomy of KV-cache reuse methods. Exact reuse requires an identical prefix, whereas approximate reuse relaxes this constraint and addresses the resulting context mismatch differently for full- and hybrid-attention architectures.

Reuse type	Attention architecture	Technique	Representative methods
Exact prefix reuse	-	Reuse the longest identical prefix and prefill only the unmatched suffix.	vLLM(APC), SGLang RadixAttention, and RAGCache
Approx. reuse	Full attention	Selectively recompute important tokens, using signals such as KV deviation, attention structure and query relevance.	CacheBlend, EPIC, Cache-Craft, KVShare(DHD), SamKV, CacheClip, A3, and ProphetKV
		Adapt cached states or redesign the reusable representation.	KVCOMM, KVPacket, KVLink, and SemPIC
	Hybrid attention	Extend position-independent reuse to hybrid models by reusing linear-layer states alongside full-attention KV.	HYPIC and LinearKV

Table 2: Comparison of KV-cache evaluation scope. KVBench evaluates approximate KV-cache reuse across tasks, models, methods, and workflows. RAG denotes RAG-style context reuse.

Evaluation	**Task**				**Model**	**Methods**	**Workflows**		
	Long-context retrieval& tracing	Multi-doc QA	Document summary	Knowledge math& code	Hybrid architecture	Multiple reuse methods	RAG	Multi-agent	Skill
SCBench	✓	×	✓	✓	✓	×	×	×	×
Agrawal et al.	✓	✓	✓	×	×	×	×	×	×
Cestola et al.	✓	✓	×	×	×	✓	✓	×	×
Liang et al.	×	×	×	✓	×	✓	×	✓	×
KVBench	✓	✓	✓	✓	✓	✓	✓	✓	✓

RAG and agentic applications naturally expose LLMs to recurring context. RAG systems repeatedly retrieve document chunks (Lewis et al., 2020), often for different queries or in different orders, creating opportunities for non-prefix reuse. RULER and LongBench tasks are commonly used to evaluate quality retention of KV-reuse methods on long-context workloads (Hsieh et al., 2024; Bai et al., 2024). Agentic systems (Yao et al., 2023) repeatedly invoke the same skills on different executions. SkillsBench evaluates utility across various tasks with execution-based verification, providing a natural workload to study skill reuse (Li et al., 2026).

Existing KV-cache evaluations address distinct questions. Broad KV-cache optimization benchmarks assess methods ranging from cache-lifecycle operations to specific optimizations such as quantization, pruning, and merging (Li et al., 2025; Agrawal & Mayer, 2026). Reuse-focused evaluations examine quality limitations of approximate reuse and the impact of cross-chunk interaction (Cestola et al., 2026; Liang et al., 2026). KVBench complements these studies by evaluating approximate reuse across tasks, models, methods, and workflows. See Table 2 for details.

3 Problem Formulation and Metrics

Approximate reuse. KVBench studies reuse of an identical token segment d under different preceding contexts. For a fixed model and positional-encoding configuration, let $C(d \mid p)$ denote its cache state after full-prefill of $[p, d]$. This includes key–value tensors and, for hybrid models, recurrent states or state-composition representations. Given $p^{\mathrm{src}} \neq p^{\mathrm{tgt}}$, the source cache and target reference are

$$C^{\mathrm{src}} = C(d \mid p^{\mathrm{src}}), \qquad C^{\mathrm{tgt}} = C(d \mid p^{\mathrm{tgt}}).$$

These states generally differ because preceding context affects token interactions and positional assignments. Exact prefix reuse avoids this mismatch by requiring an identical prefix. Approximate reuse instead uses a method $\mathcal{A}$ to construct $\widehat{C}^{\mathrm{tgt}}_{\mathcal{A}}$ from C^{src} through direct loading, state adaptation or composition, or selective recomputation. KVBench evaluates the resulting quality–latency trade-off.

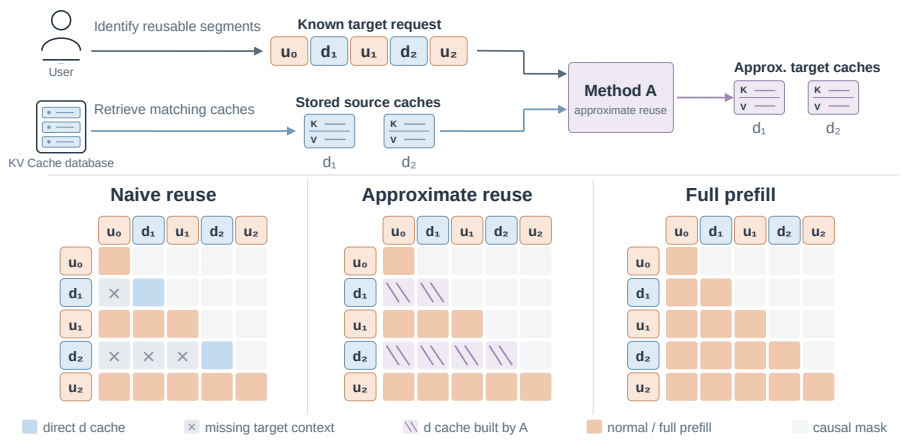


Figure 1: Interleaved KV reuse with independently prepared segments (rows attend to columns). Crosses mark target-prefix interactions absent under naive reuse. Purple hatching denotes states constructed by $\mathcal{A}$, not necessarily recomputed attention.

Requests and reuse workflows. A target request may interleave reusable and request-specific segments:

$$x^{\mathrm{tgt}} = [u_0, d_1, u_1, \ldots, d_m, u_m], \qquad \widehat{C}_{\mathcal{A}}(x^{\mathrm{tgt}}) = \mathcal{A}\big(x^{\mathrm{tgt}}; \{C_j^{\mathrm{src}}\}_{j=1}^{m}\big)\,,$$

where each u_j is processed online and each d_j has a stored cache C_j^{src}. A *workflow* specifies source-cache preparation, target-segment arrangement, and state sharing across calls, defining the reuse conditions beyond the task itself. Figure 1 illustrates independently prepared segments.

Static-task comparisons use identical target requests for reuse and full prefill. Interactive comparisons use the same task instances and agent configuration but allow trajectories to diverge, measuring quality through final task outcomes.

Metrics. KVBench measures quality with each task's native metric and online latency with TTFT. Let $Q_{\mathrm{reuse}}, Q_{\mathrm{full}}$ denote task scores and $T_{\mathrm{reuse}}, T_{\mathrm{full}}$ the corresponding TTFT statistics under reuse and full prefill. Alongside absolute scores, we report *relative quality* and *relative latency*:

$$R_Q = Q_{\mathrm{reuse}}/Q_{\mathrm{full}}, \qquad R_T = T_{\mathrm{reuse}}/T_{\mathrm{full}}.$$

A ratio of 1 matches the reference; $R_Q < 1$ indicates lower quality for our higher-is-better metrics, while $R_T < 1$ indicates reduced TTFT. When $Q_{\mathrm{full}} = 0$, we report only absolute quality scores.

TTFT includes online cache loading, adaptation, and recomputation. Whether source-cache preparation is included is specified per experiment. For skill reuse, offline preparation is excluded, and we report first-call TTFT and final task reward; the latency measurement therefore concerns initial response rather than total agent runtime. Exact prefix caching remains enabled in both skill-reuse and reference configurations.

4 KVBench: Modular Design for Controlled Reuse Evaluation

KVBench is designed to separate concerns that are often entangled in evaluations of KV-cache reuse. It distinguishes the task being evaluated, the language model being executed, the reuse method, and the multi-request workflow through which reusable states arise. This decomposition enables reuse methods and models to be evaluated consistently across application settings, while exposing interactions among methods, models, and tasks under a common execution and measurement framework.

KVBench is organized around four core abstractions. A *Task* supplies examples, constructs task-specific prompt content, specifies output constraints, and provides a quality evaluator. A *Model* exposes a language model through a common execution interface. A *Method* implements cache preparation and reuse. A *Workflow* specifies how one or more task examples are organized into a sequence of model requests and how reusable state is shared across them, including reordered RAG contexts, multi-agent interaction patterns, and agentic request sequences involving skill context.

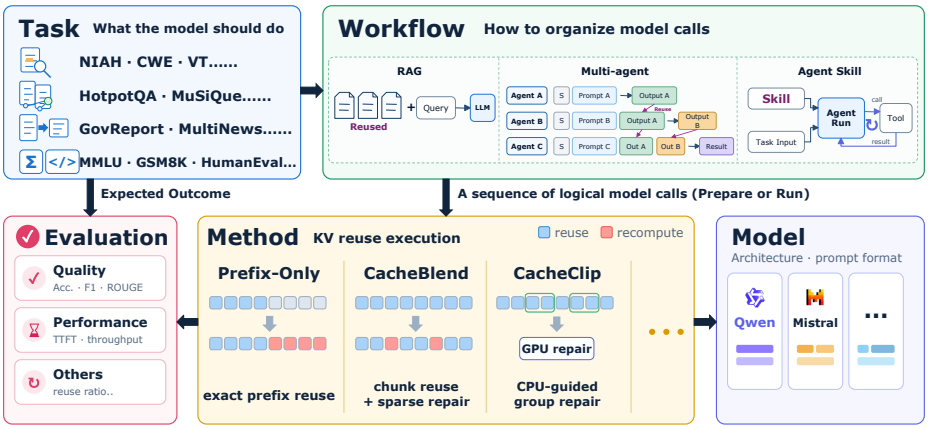


Figure 2: Overview of KVBench. Pluggable tasks, workflows, model adapters, and reuse methods are connected through unified interfaces, executed by the benchmark runner, and evaluated using both quality and systems metrics.

Table 3: Task and workflow coverage in KVBench.

Workflow	Tasks	Purpose
RAG-style context reuse	NIAH, CWE, VT, MuSiQue, 2WikiMQA, SAMSum, HotpotQA, TriviaQA, MultiNews, GovReport	Evaluates KV reuse across long-context workloads spanning retrieval, question answering, multi-hop reasoning, and generation.
Multi-agent communication	MMLU, GSM8K, HumanEval	Evaluates reuse of contexts produced and exchanged by multiple agents under a fully connected interaction workflow.
Agentic skill reuse	SkillsBench	Evaluates reuse of recurring skill context in tool-using agent executions with otherwise task-specific inputs and interactions.

A common execution and measurement layer coordinates these abstractions and handles model execution, timing, and metric collection. Modular interfaces allow for the incorporation of new tasks, models, reuse methods, and workflow patterns without rewriting unrelated components.

Figure 2 summarizes this organization. The four pluggable components are connected through unified interfaces, executed by a common runner, and evaluated jointly for task quality and systems efficiency.

This modularity supports both benchmark comparison and controlled analysis. Reuse methods can be compared under the same task, workflow, model, and evaluation setting, while models, reuse configurations, or properties of the reused context can be varied systematically in targeted studies. This design provides the basis for analyzing the method-model-application interactions in our evaluation.

4.1 RAG and Agentic Workflow Coverage

Table 3 summarizes the task and workflow coverage of KVBench. We group ten tasks into a *RAG-style context-reuse* family, covering settings where reusable long-context content is combined with request-specific queries or generation objectives. These tasks span retrieval, question answering, multi-hop reasoning, and long-context generation. Beyond RAG-style workloads, KVBench also covers two distinct agentic settings: multi-agent communication and agentic skill reuse.

The RAG-style suite combines synthetic long-context tasks from RULER (Hsieh et al., 2024) with question-answering and summarization workloads used in LongBench (Bai et al., 2024). NIAH primarily tests retrieval of localized information from long contexts, while CWE requires aggregating evidence distributed across the input and VT involves tracking dependencies among variables. NIAH, CWE, and VT are derived from RULER (Hsieh et al., 2024). TriviaQA provides a retrieval-oriented question-answering workload (Joshi et al., 2017). 2WikiMQA, MuSiQue, and HotpotQA

additionally require reasoning over multiple pieces of evidence (Ho et al., 2020; Trivedi et al., 2022; Yang et al., 2018). SAMSum, MultiNews, and GovReport extend the suite to generation, covering dialogue summarization, multi-document summarization, and long-document summarization, respectively (Gliwa et al., 2019; Fabbri et al., 2019; Huang et al., 2021).

The multi-agent workloads are adopted from KVCOMM (Ye et al., 2025). Multiple agents solve instances of MMLU, GSM8K, or HumanEval (Hendrycks et al., 2021; Cobbe et al., 2021; Chen et al., 2021) and exchange their generated contexts according to KVCOMM's fully connected interaction pattern. We incorporate these workflows into KVBench while preserving the original interaction structure, and evaluate them using the same quality and systems measurements.

Finally, SkillsBench (Li et al., 2026) provides realistic tool-use environments for evaluating *skill reuse*. A skill supplies recurring procedural context, such as task instructions, workflows, or tool guidance, that can be reused across different agent tasks. In our workflow, the skill context is prefilled independently, and its cached KV states are reused when processing the task-specific agent input. This setting evaluates KV reuse for skill-augmented agent workloads.

5 Evaluation of KV-Reuse Methods

5.1 Experimental Setup

We evaluate two groups of models under separate hardware and software environments. Experiments on Mistral-7B-Instruct-v0.3 (Jiang et al., 2023), Qwen3-4B, and Qwen3-8B (Yang et al., 2025a) are conducted on a single NVIDIA L40 GPU with 48 GB of memory, using CUDA 12.9 package builds. Experiments on the hybrid-architecture Qwen3.8-27B (Qwen Team, 2026b) and Qwen3.5-35B-A3B (Qwen Team, 2026a) models are conducted on two NVIDIA RTX PRO 6000 GPUs with 96 GB of memory each, using CUDA 13.0 package builds.

Unless otherwise stated, we evaluate 64 samples per task. All latency experiments use a request batch size of one. For standard full-attention models, we compare CacheBlend, A3, ProphetKV, CacheClip, KVPacket, EPIC, and a random-selection baseline. We denote CacheBlend configurations by their token-recomputation ratio; for example, CacheBlend(0.15) recomputes 15% of the tokens. EPIC uses the setting of recomputing 32 tokens, while the random-selection baseline recomputes 50% of the tokens by default. For hybrid-attention models, we evaluate HYPIC variants and LinearKV, which explicitly support reuse of hybrid-attention states. We report additional method-specific configurations and exceptions together with the corresponding results.

For every model–method–task configuration, the corresponding full-prefill baseline is evaluated with the same model, hardware environment, task examples, and decoding configuration.

5.2 Main Results

We now examine the quality–latency behavior of approximate KV-cache reuse across the tasks, models, and workflows covered by KVBench. Complete per-task results are reported in Appendix A. Unless otherwise specified, aggregate statistics below are medians over the 10 RAG-style tasks.

Because each task is evaluated on 64 examples and TTFT is subject to system-level variation, we focus on large and repeated trends rather than small differences between reuse configurations.

Table 4 brings the cross-method comparison on Qwen3-8B into the main text. CacheBlend and KVPacket generally preserve more task quality than more aggressive low-latency configurations, while still providing substantial TTFT reductions. In contrast, configurations such as EPIC often achieve lower TTFT ratios but can incur larger quality degradation on some tasks. A3 is competitive on some tasks but is not reported for the multi-agent workloads, where its chunk-interleaving implementation is unsupported. CacheClip shows the largest runtime variability: its CPU-side token selection can dominate latency on irregular prompts, even when its quality is reasonable. The task variation, especially on VT, MuSiQue, and TriviaQA, demonstrates why method rankings should be reported across diverse workloads rather than summarized by a single aggregate score.

For the hybrid architecture, Table 5 reports the Qwen3.5-35B-A3B comparison among HYPIC's Addition, Transition, Transition+RoPE, and Transition+RoPE+Recompute (TRR) variants, together

Table 4: Cross-method comparison on Qwen3-8B. R_Q is relative quality (higher is better) and R_T is relative TTFT (lower is better). For KVPacket, EPIC, and Rand(0.5), values are computed relative to the corresponding Qwen3-8B full-prefill result.

Task	Metric	Q_{full}	CacheBlend		A3		ProphetKV		CacheClip		KVPacket		EPIC		Rand	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	1.000	**1.000**	0.275	0.996	0.321	0.998	0.228	0.995	0.752	**1.000**	0.466	0.016	**0.127**	0.031	0.578
CWE	Acc.	1.000	**0.992**	0.288	0.937	0.316	0.949	**0.208**	0.958	0.788	0.947	0.249	0.141	0.213	0.227	0.388
VT	Acc.	1.000	**1.000**	0.267	0.357	0.327	0.984	0.470	0.823	0.691	0.619	0.146	0.013	**0.122**	0.006	0.423
MuSiQue	F1	0.287	0.710	0.265	0.658	0.329	0.904	0.231	0.688	1.093	**1.059**	0.514	0.828	**0.130**	0.554	0.591
SAMSum	ROUGE-L	0.338	0.944	0.286	0.800	0.354	0.911	**0.253**	0.819	2.809	0.863	0.358	**0.988**	0.405	0.879	0.721
2WikiMQA	F1	0.420	0.896	0.270	0.745	0.331	0.896	0.223	0.721	1.112	**0.964**	0.500	0.878	**0.121**	0.685	0.561
GovReport	ROUGE-L	0.199	**0.995**	0.242	0.969	0.335	0.988	0.230	0.959	0.385	0.980	0.263	0.960	**0.039**	0.943	0.267
HotpotQA	F1	0.524	**0.831**	0.262	0.647	0.489	0.774	0.199	0.267	0.757	0.716	0.181	0.612	**0.080**	0.442	0.270
MultiNews	ROUGE-L	0.148	1.003	0.297	1.006	0.374	1.010	0.290	1.010	0.970	1.002	0.322	1.017	**0.111**	**1.040**	0.245
TriviaQA	F1	0.870	0.967	0.330	0.778	0.406	**0.999**	0.456	0.950	10.560	0.896	0.189	0.993	**0.157**	0.857	0.337
MMLU	Acc.	0.672	0.977	0.634	–	–	0.973	2.200	0.912	8.400	**1.000**	**0.610**	0.957	0.937	0.913	0.953
GSM8K	Acc.	0.891	0.947	0.770	–	–	**1.084**	2.169	1.050	6.295	0.998	**0.648**	1.017	0.991	0.982	0.935
HumanEval	Acc.	0.875	**0.893**	**0.515**	–	–	0.934	2.055	0.872	9.865	**0.966**	0.590	0.915	0.935	0.881	0.873

Table 5: Qwen3.5-35B-A3B results with HYPIC variants and LinearKV. R_Q is relative quality (higher is better) and R_T is relative TTFT (lower is better).

Task	Metric	Q_{full}	Addition		Transition		Trans.+RoPE		TRR		LinearKV	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	1.0000	0.984	**0.686**	**1.000**	0.698	0.844	0.705	0.953	0.738	0.969	0.824
CWE	Acc.	1.0000	0.025	**0.637**	0.981	0.686	0.980	0.676	**0.989**	0.744	0.747	0.738
VT	Acc.	1.0000	0.153	**0.688**	0.081	0.699	0.078	0.704	0.594	0.752	**0.963**	0.824
MuSiQue	F1	0.3829	0.011	**0.545**	**0.812**	0.583	0.784	0.591	0.773	0.658	0.590	0.713
2WikiMQA	F1	0.4712	0.128	**0.497**	0.792	0.570	0.903	0.581	**0.974**	0.626	0.842	0.701
SAMSum	ROUGE-L	0.3164	0.163	**0.337**	0.971	0.480	0.992	0.466	**0.994**	0.659	0.941	0.661
GovReport	ROUGE-L	0.1934	0.918	**0.315**	0.925	0.344	0.932	0.345	0.946	0.338	**0.971**	0.590
HotpotQA	F1	0.7583	0.472	0.272	**0.688**	0.307	0.587	0.312	0.645	0.335	0.683	**0.254**
MultiNews	ROUGE-L	0.1428	**1.030**	**0.779**	1.014	0.844	1.003	0.859	1.012	0.870	0.997	0.956
TriviaQA	F1	0.9073	0.225	**0.349**	0.938	0.392	**0.945**	0.384	0.918	0.427	0.939	0.704

with LinearKV. The HYPIC variants generally preserve quality better than the simplest Addition configuration, while their latency remains substantially below full-prefill on long-context tasks. LinearKV provides a useful additional point in the design space, but does not uniformly dominate the HYPIC variants. These results reinforce that method comparisons should be reported jointly in terms of R_Q and R_T, rather than with a single reuse score.

The remaining results, the Mistral-7B cross-method comparison, the Qwen3-4B CacheBlend results, and the Qwen3.8-27B hybrid-model results are reported in Appendix A and Appendix B.

6 Findings and Analysis

6.1 Task–Model Interactions

Figure 3 makes the task–model dependence explicit. With CacheBlend(0), Qwen3-8B ranges from a quality ratio of 0.04 on VT to 1.02 on MultiNews; VT is likewise low for Mistral-7B (0.124) and Qwen3-4B (0.151). Model rankings are not consistent across tasks: the NIAH ratios are 0.812/0.304/0.719 (Qwen3-8B/Mistral-7B/Qwen3-4B), whereas MuSiQue gives 0.403/0.627/0.520. Recomputing 5% of tokens raises Qwen3-8B's VT ratio to 0.988, but raises MuSiQue only to 0.763. Thus, both reuse robustness and the benefit of recomputation depend on the task–model pair, so a small set of evaluated pairs can mischaracterize a reuse method.

6.2 Workload Structure and Reusable Context Length

Approximate KV reuse is sensitive not only to which context is reused, but also to where request-specific instructions are placed relative to the reusable context. We isolate this effect on HotpotQA with HYPIC by varying instruction placement while keeping the task and all other evaluation set-

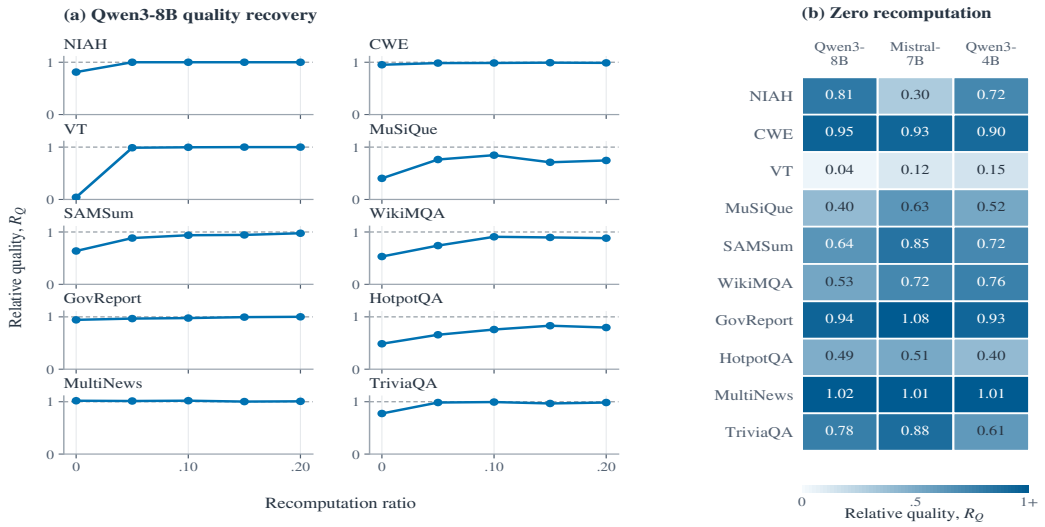


Figure 3: Task–model interactions under CacheBlend, using the RAG-style tasks in Appendix A. (a) Qwen3-8B quality recovery is task-dependent: 5% recomputation nearly restores VT but only partly recovers MuSiQue, while NIAH and CWE start from different baselines. (b) Per-task relative quality (R_Q) at zero recomputation across models, showing that model rankings vary by task.

tings fixed. As shown in Figure 4, the full-prefill reference is nearly unchanged across layouts, whereas approximate reuse can change substantially.

The effect is strongest for HYPIC Addition. Placing the instruction only before the reusable context gives an F1 of 0.463, compared with 0.222 when the instruction is placed only after the context—more than a twofold difference. When the leading instruction is included in the cached segment, F1 drops further to 0.088. In contrast, HYPIC TRR is much more stable across the four layouts (0.469–0.552), while full recomputation remains between 0.767 and 0.786. These results show that prompt layout is an independent evaluation dimension for approximate KV reuse: a method can appear substantially stronger or weaker under different placements of similar request-specific instructions.

6.3 Skill Reuse in Agentic Workloads

We evaluate skill reuse on all 87 SkillsBench 1.1 tasks (Li et al., 2026), with one agent rollout per task and configuration, using Qwen3.8-27B. Reusing independently prepared skill context preserves a substantial share of the task-level benefit of full skill prefilling: FullPrefill raises mean reward from 0.2215 under NoSkill to 0.4430, while HYPIC TRR, LinearKV, and HYPIC Addition retain 83.0%, 57.9%, and 46.7% of this mean-reward gain, respectively (Table 6). Across SkillsBench tasks, first-call TTFT for the HYPIC variants is generally lower for executions with a larger fraction of reusable context, reaching roughly 20–30% of FullPrefill on highly reusable executions. Overall, skill reuse provides substantial TTFT reductions when reusable skill context is large, while HYPIC TRR retains most of the reward gain obtained by full skill prefilling. Detailed setup, latency analysis, and complete per-task results are reported in Appendix C.

6.4 Cross-document Dependency is Weakly Predictive of Reuse Degradation

Approximate KV reuse may degrade quality because cached states no longer capture all interactions among independently processed context segments. So we ask whether cross-document dependency can explain the large variation in reuse quality across tasks and models. We use value-weighted dependency with skip-k (VWD-SkipK) as the primary attention-based measure and KV deviation as a complementary representation-level measure. Additional dependency metrics, including Cache Context Impact (CCI) from Cache-Craft (Agarwal et al., 2025), are reported in Appendix D.

The dependency measurements themselves are highly reproducible across models: task-level dependency scores from the attention-based metrics have Pearson correlations of approximately 0.99

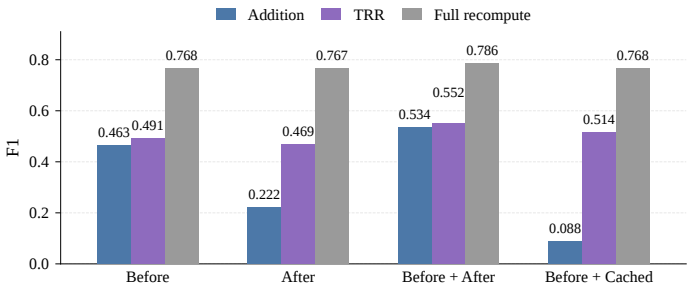


Figure 4: Effect of instruction placement on HotpotQA F1 under HYPIC. Addition is highly sensitive to prompt placement, whereas TRR and full recomputation exhibit smaller variations.

Setting	Mean Reward	Skill gain retained
NoSkill	0.2215	–
Addition	0.3249	46.7%
LinearKV	0.3498	57.9%
TRR	0.4054	83.0%
FullPrefill	0.4430	100.0%

Table 6: Mean reward on 87 SkillsBench tasks and fraction of the skill gain retained.

across the evaluated models, with similarly strong consistency at the sample level. However, under CacheBlend, dependency is only weakly associated with reuse-induced quality degradation. After controlling for task identity, the additional variance in quality loss explained by VWD-SKIPK is small; for example, on Qwen3-4B it accounts for only 0.399% of the variance in L_0. Per-task rank correlations are likewise generally close to zero and vary in sign across tasks and models. Alternative dependency measures yield the same qualitative conclusion.

These results suggest that cross-document dependency is a reproducible property of the workload, but its magnitude is only weakly associated with quality degradation even when the reuse method is fixed. Aggregate dependency alone is therefore unlikely to provide a method-agnostic indicator of reuse robustness, which remains strongly dependent on the task, model, and configuration.

7 CONCLUSION

We presented KVBench, an extensible benchmark for evaluating approximate KV-cache reuse across tasks, models, methods, and request workflows. Our results show that quality retention and latency benefits depend on the evaluated configuration, including instruction placement and the amount of reusable context. Long-context retrieval and reordering workloads can achieve substantial TTFT reductions, while short multi-agent messages may not amortize reuse overhead. The agentic extension demonstrates the potential of recurring skill context for reuse, while highlighting the need to evaluate task-level outcomes.

The central implication is methodological. Evaluating KV reuse requires more than measuring cache-hit or reuse ratios on a single benchmark. It requires a common protocol that jointly measures quality, latency, workload state, and the amount and structure of reused context. KVBench provides this foundation and supports future studies of reuse policies, repair strategies, hybrid architectures, and agentic inference.

LIMITATIONS

First, despite covering diverse tasks, models, methods, and workflows, our evaluation samples only part of the design space of approximate KV-cache reuse. The observed patterns should therefore be interpreted within the evaluated settings. KVBench's modular design supports extending this coverage as new models and methods emerge.

Second, limited evaluation budgets constrain the precision of quality estimates, particularly for stochastic agent trajectories. Small differences require further validation with repeated runs. Latency results also depend on hardware, serving configurations, and implementation choices, and may not transfer directly to other deployments.

Finally, our analysis characterizes variation in reuse robustness without fully explaining its causes. The weak predictive value of the evaluated cross-document dependency measures suggests that aggregate dependency alone is insufficient to explain quality degradation. Identifying which contextual interactions matter for downstream outcomes, and developing reliable predictors of reuse quality, remain open research directions.

AI USE STATEMENT

Generative AI tools were used as research and writing assistants in this work. They were used to provide feedback on research hypotheses, methodology, and experimental design; assist with the implementation and debugging of research code; provide preliminary analyses and feedback on the interpretation of experimental results; and assist with English drafting, reformulation, and editing.

Generative AI was not used to generate or clean research datasets, formulate or prove mathematical claims, or perform qualitative or thematic data analysis. All AI-assisted code, analyses, and manuscript content were reviewed and verified by the authors against the intended methodology, experimental outputs, and original sources where applicable. Final research decisions, interpretations, and conclusions were made by the authors, who take full responsibility for the content of this work.

ETHICS STATEMENT

This work studies the evaluation and efficiency of approximate KV cache reuse for large language model inference. Our experiments use existing publicly released research benchmarks and model artifacts. We do not recruit human participants, conduct user studies, or collect private user data, and we do not introduce a new dataset containing personal or sensitive information.

Approximate cache reuse may change model outputs while reducing inference cost, and efficiency improvements should therefore not be interpreted as evidence of reliability or safety in downstream deployment. In particular, systems used in high-stakes settings should independently validate the task-specific quality and safety effects of any reuse method. KVBench is intended to make such trade-offs more visible by reporting both efficiency and output-quality measurements across tasks, models, and reuse methods. More efficient inference is a general-purpose capability and may support both beneficial and harmful applications; our work does not introduce new model capabilities or mechanisms for bypassing existing safety controls.

REPRODUCIBILITY STATEMENT

We take several steps to facilitate reproduction of our results. The benchmark design, evaluation protocol, metrics, and experimental setup are described in the main paper. The appendix provides additional implementation details, complete task- and method-level results, and analyses omitted from the main text due to space constraints. The supplementary material provides the implementation and configurations used in our experiments, including model and runtime versions, task prompts, reuse and repair settings, and evaluation scripts. KVBench further exposes task, method, workflow, engine, and metric components through modular interfaces, allowing individual experimental components to be inspected and reproduced independently.

REFERENCES

Shubham Agarwal, Sai Sundaresan, Subrata Mitra, Debabrata Mahapatra, Archit Gupta, Rounak Sharma, Nirmal Joshua Kapu, Tong Yu, and Shiv Saini. Cache-craft: Managing chunk-caches for efficient retrieval-augmented generation. *Proc. ACM Manag. Data*, 3(3), June 2025. doi: 10.1145/3725273. URL https://doi.org/10.1145/3725273.

Nikita Agrawal and Ruben Mayer. Benchmarking kv-cache optimizations across task quality and system performance for long-context serving, 2026. URL https://arxiv.org/abs/2607.05399.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Ziyi Cao, Qingyi Si, Jingbin Zhang, and Bingquan Liu. Sparse attention across multiple-context kv cache. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30165–

30173, 2026. doi: 10.1609/aaai.v40i36.40266. URL https://ojs.aaai.org/index.php/AAAI/article/view/40266.

Samuel Cestola, Tianxiang Xia, Zheng Weiyan, Zheng Pengfei, and Diego Didona. An experimental study of kv cache reuse strategies in chunk-level caching systems, 2026. URL https://arxiv.org/abs/2603.20218.

Chuangtao Chen, Grace Li Zhang, Xunzhao Yin, Cheng Zhuo, Bing Li, and Ulf Schlichtmann. Kv packet: Recomputation-free context-independent kv caching for llms, 2026. URL https://arxiv.org/abs/2604.13226.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1074–1084, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1102. URL https://aclanthology.org/P19-1102/.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In Lu Wang, Jackie Chi Kit Cheung, Giuseppe Carenini, and Fei Liu (eds.), *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pp. 70–79, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5409. URL https://aclanthology.org/D19-5409/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580/.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Junhao Hu, Wenrui Huang, Weidong Wang, Haoyi Wang, Tiancheng Hu, Zhang Qin, Hao Feng, Xusheng Chen, Yizhou Shan, and Tao Xie. EPIC: Efficient position-independent caching for serving large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd*

International Conference on Machine Learning, volume 267 of *Proceedings of Machine Learning Research*, pp. 24391–24402. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/hu25j.html.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1419–1436, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.112. URL https://aclanthology.org/2021.naacl-main.112/.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Chao Jin, Zili Zhang, Xuanlin Jiang, Fangyue Liu, Shufan Liu, Xuanzhe Liu, and Xin Jin. Ragcache: Efficient knowledge caching for retrieval-augmented generation. *ACM Trans. Comput. Syst.*, 44 (1), November 2025. ISSN 0734-2071. doi: 10.1145/3768628. URL https://doi.org/10.1145/3768628.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, Yifeng He, Shenghan Zheng, Kyoung Whan Choe, Jiankai Sun, Shuyi Wang, Chujun Tao, Binxu Li, Xuandong Zhao, Hejia Geng, Xiaojun Wu, Junwei Zhou, Xiaokun Chen, Hanwen Xing, Yubo Li, Qunhong Zeng, Di Wang, Yuanli Wang, Roey Ben Chaim, Penghao Jiang, Haotian Shen, Luyang Kong, Xinyi Liu, Runhui Wang, Xuanqing Liu, Jiachen Li, Xin Lan, Yueqian Lin, Wengao Ye, Junwei He, Songlin Li, Yue Zhang, Yipeng Gao, Yijiang Li, Ze Ma, Liqiang Jing, Tianyu Wang, Kaixin Li, Yiqi Xue, Haoran Lyu, Yizhuo He, Yuchen Tian, Shutong Wu, Bowei Wang, Yixuan Gao, Bo Chen, Litong Liu, Sikai Cheng, Jiajun Bao, Shuaicheng Tong, Shuwen Xu, Terry Yue Zhuo, Tinghan Ye, Qi Qi, Miao Li, Longtai Liao, Zelin Tan, Chang Shi, Xilin Tang, Srinath Tankasala, Boqin Yuan, Yaoyao Qian, Jianhong Tu, Chenguang Wang, Yizhou Sun, Wei Wang, Aaron Taylor, Ziyue Yang, Changkun Guan, Zhikang Dong, Xinyu Zhang, Steven Dillmann, Han chung Lee, and Dawn Song. Skillsbench: Benchmarking how well agent skills work across diverse tasks, 2026. URL https://arxiv.org/abs/2602.12670.

Yucheng Li, Huiqiang Jiang, Qianhui Wu, Xufang Luo, Surin Ahn, Chengruidong Zhang, Amir Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. Scbench: A kv cache-centric analysis of long-context methods. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu

(eds.), *International Conference on Learning Representations*, volume 2025, pp. 66063–66093, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/a540b17fb2295c736d5afd6c507acf66-Paper-Conference.pdf.

Sichu Liang, Zhenglin Wang, Chujiajia, Pengfei Xia, Hui Zang, and Deyu Zhou. When KV cache reuse fails in multi-agent systems: Cross-candidate interaction is crucial for LLM judges. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7205–7230, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.327. URL https://aclanthology.org/2026.acl-long.327/.

Yifei Liu, Juntong Wu, Yang Liu, Junhao Hu, Minghao Li, Xiaoxu Chen, and Weihang Chen. Hypic: Accelerating hybrid-attention llm serving with position-independent caching, 2026a. URL https://arxiv.org/abs/2607.01299.

Yirui Liu, Ruoling Qi, Longwen Wang, Xuaner Wu, Jian Chen, Yuxin Jin, Jiawei Shao, and Xuelong Li. Linearkv: One cached state suffices for position-independent caching in hybrid llms, 2026b. URL https://arxiv.org/abs/2608.11231.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 05 2022. ISSN 2307-387X. doi: 10.1162/tacl_a_00475. URL https://doi.org/10.1162/tacl_a_00475.

Shihao Wang, Jiahao Chen, Yanqi Pan, Hao Huang, Yichen Hao, Xiangyu Zou, Wen Xia, Chongyang Qiu, Wentao Zhang, and PengFei Wang. ProphetKV: User-query-driven selective recomputation for efficient KV cache reuse in retrieval-augmented generation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SwGwmKIgUb.

Hui Xie, Peng Xiao, Yutong Deng, Shuoran Dou, Jian Yang, and Jinyang Guo. Sempic: Learning semantic position-independent kv caches, 2026. URL https://arxiv.org/abs/2607.28069.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

Bin Yang, Qiuyu Leng, Jun Zeng, and Zhenhua Wu. Cacheclip: Accelerating rag with effective kv cache reuse, 2026. URL https://arxiv.org/abs/2510.10129.

Huan Yang, Renji Zhang, Mingzhe Huang, Weijun Wang, Yin Tang, Yuanchun Li, Yunxin Liu, and Deyu Zhang. Kvshare: An llm service system with efficient and effective multi-tenant kv cache reuse, 2025b. URL https://arxiv.org/abs/2503.16525.

Jingbo Yang, Bairu Hou, Wei Wei, Yujia Bao, and Shiyu Chang. KVLink: Accelerating large language models via efficient KV cache reuse. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025c. URL https://openreview.net/forum?id=oDcAGSXZZP.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. Cacheblend: Fast large language model serving for rag with cached knowledge fusion. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 94–109, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400711961. doi: 10.1145/3689031.3696098. URL https://doi.org/10.1145/3689031.3696098.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Hancheng Ye, Zhengqi Gao, Mingyuan Ma, Qinsi Wang, Yuzhe Fu, Ming-Yu Chung, Yueqian Lin, Zhijian Liu, Jianyi Zhang, Danyang Zhuo, and Yiran Chen. Kvcomm: Online cross-context kv-cache communication for efficient llm-based multi-agent systems. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 17882–17928. Curran Associates, Inc., 2025. doi: 10.52202/085713-0605. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/1a074a28c3a6f2056562d00649ae6416-Paper-Conference.pdf.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. Sglang: Efficient execution of structured language model programs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583. Curran Associates, Inc., 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/724be4472168f31ba1c9ac630f15dec8-Paper-Conference.pdf.

Yuechi Zhou, Yi Su, Jianxin Zhang, Juntao Li, Qingrong Xia, Zhefeng Wang, Xinyu Duan, and Baoxing Huai. a^3: Attention-aware accurate kv cache fusion for fast large language model serving, 2025. URL https://arxiv.org/abs/2511.17560.

Appendix

A Detailed Results for the Main KVBench Experiments

Q_{full} is the absolute full-prefill quality; $R_Q = Q/Q_{\text{full}}$ and $R_T = \text{TTFT}/\text{TTFT}_{\text{full}}$. Higher R_Q and lower R_T are better. Bold independently marks the highest R_Q and the lowest R_T in each row; the two bold entries may correspond to different configurations.

Table 7: Results for Qwen3-8B with CacheBlend.

Task	Metric	Q_{full}	CB(0)		CB(.05)		CB(.10)		CB(.15)		CB(.20)	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	1.000	0.812	**0.173**	**1.000**	0.184	**1.000**	0.228	**1.000**	0.275	**1.000**	0.335
CWE	Acc.	1.000	0.952	**0.171**	0.984	0.185	0.986	0.227	**0.992**	0.288	0.988	0.344
VT	Acc.	1.000	0.041	**0.150**	0.988	0.164	0.997	0.215	**1.000**	0.267	**1.000**	0.315
MuSiQue	F1	0.287	0.403	**0.129**	0.763	0.165	**0.846**	0.223	0.710	0.265	0.745	0.315
SAMSum	ROUGE-L	0.338	0.637	**0.117**	0.884	0.168	0.938	0.229	0.944	0.286	**0.976**	0.349
2WikiMQA	F1	0.420	0.531	**0.134**	0.740	0.163	**0.908**	0.230	0.896	0.270	0.882	0.320
GovReport	ROUGE-L	0.199	0.943	**0.081**	0.967	0.128	0.977	0.190	0.995	0.242	**1.001**	0.313
HotpotQA	F1	0.524	0.486	**0.080**	0.658	0.131	0.758	0.193	**0.831**	0.262	0.796	0.317
MultiNews	ROUGE-L	0.148	1.019	**0.216**	1.014	0.247	**1.020**	0.263	1.003	0.297	1.008	0.347
TriviaQA	F1	0.870	0.775	**0.140**	0.985	0.201	**0.994**	0.262	0.967	0.330	0.985	0.384
MMLU	Acc.	0.672	1.000	**0.604**	1.046	0.715	**1.070**	0.669	0.977	0.634	1.000	0.618
GSM8K	Acc.	0.891	0.912	**0.770**	**0.965**	0.934	0.860	0.815	0.947	0.770	0.912	0.780
HumanEval	Acc.	0.875	0.786	**0.471**	0.839	0.535	0.893	0.506	0.893	0.515	**0.929**	0.524

Table 8: Results for Mistral-7B with CacheBlend.

Task	Metric	Q_{full}	CB(0)		CB(.05)		CB(.10)		CB(.15)		CB(.20)	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	0.875	0.304	**0.173**	**0.964**	0.197	0.893	0.239	0.911	0.264	0.911	0.319
CWE	Acc.	0.991	**0.932**	**0.158**	0.834	0.185	0.877	0.228	0.875	0.266	0.899	0.329
VT	Acc.	0.828	0.124	**0.144**	**1.011**	0.169	0.928	0.220	0.940	0.252	0.962	0.316
MuSiQue	F1	0.224	0.627	**0.119**	**1.013**	0.159	0.990	0.214	0.976	0.264	0.927	0.323
SAMSum	ROUGE-L	0.325	0.846	**0.113**	**0.990**	0.169	0.979	0.230	0.990	0.291	0.990	0.365
2WikiMQA	F1	0.339	0.722	**0.139**	0.771	0.162	0.800	0.215	**0.803**	0.263	0.687	0.319
GovReport	ROUGE-L	0.206	1.077	**0.080**	1.066	0.132	**1.085**	0.201	1.049	0.266	1.068	0.345
HotpotQA	F1	0.489	0.513	**0.078**	0.800	0.134	0.902	0.200	**0.978**	0.269	0.962	0.350
MultiNews	ROUGE-L	0.163	**1.012**	**0.158**	0.991	0.185	0.992	0.247	0.986	0.273	0.982	0.332
TriviaQA	F1	0.841	0.884	**0.136**	1.057	0.200	1.089	0.270	**1.093**	0.336	1.071	0.416

Table 9: Results for Qwen3-4B with CacheBlend.

Task	Metric	Q_{full}	CB(0)		CB(.05)		CB(.10)		CB(.15)		CB(.20)	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	1.000	0.719	**0.268**	**1.000**	0.306	**1.000**	0.287	**1.000**	0.310	**1.000**	0.385
CWE	Acc.	0.992	0.904	**0.281**	1.000	0.288	**1.002**	0.288	0.978	0.327	0.986	0.409
VT	Acc.	0.991	0.151	0.240	0.994	**0.237**	1.000	0.272	**1.003**	0.300	0.994	0.361
MuSiQue	F1	0.217	0.520	**0.178**	**1.087**	0.210	1.033	0.263	0.995	0.319	0.841	0.373
SAMSum	ROUGE-L	0.327	0.718	**0.157**	0.902	0.209	0.961	0.270	1.002	0.341	**1.003**	0.412
2WikiMQA	F1	0.305	0.763	**0.186**	0.912	0.207	**0.930**	0.265	0.788	0.319	0.799	0.378
GovReport	ROUGE-L	0.195	0.929	**0.104**	0.975	0.162	0.969	0.228	0.968	0.305	**0.976**	0.373
HotpotQA	F1	0.474	0.404	**0.101**	0.714	0.159	0.878	0.227	0.872	0.309	**0.955**	0.381
MultiNews	ROUGE-L	0.152	1.006	**0.298**	1.007	0.323	**1.010**	0.338	0.989	0.375	0.995	0.427
TriviaQA	F1	0.855	0.613	**0.169**	0.963	0.238	**0.975**	0.311	0.948	0.387	0.960	0.465

Table 10: Results for Qwen3.8-27B with HYPIC variants and LinearKV.

Task	Metric	Q_{full}	Addition		Transition		Trans.+RoPE		TRR		LinearKV	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	1.0000	**1.000**	**0.266**	**1.000**	0.272	0.953	0.283	0.984	0.298	0.984	0.409
CWE	Acc.	1.0000	0.000	**0.251**	**0.828**	0.264	0.577	0.276	0.630	0.307	0.586	0.430
VT	Acc.	1.0000	0.000	**0.250**	0.559	0.253	0.391	0.268	0.919	0.281	**0.984**	0.395
MuSiQue	F1	0.5519	0.512	**0.159**	**0.637**	0.176	0.622	0.194	0.635	0.211	0.583	0.367
2WikiMQA	F1	0.4786	0.632	**0.154**	0.779	0.175	0.744	0.186	0.824	0.202	**0.834**	0.372
SAMSum	ROUGE-L	0.4181	0.072	**0.116**	0.870	0.161	0.844	0.170	**0.931**	0.269	0.779	0.376
GovReport	ROUGE-L	0.2052	0.873	0.103	0.921	**0.100**	0.933	0.103	0.923	0.112	**0.944**	0.343
HotpotQA	F1	0.6500	0.024	**0.080**	0.744	0.087	0.650	0.094	**0.805**	0.101	0.781	0.086
MultiNews	ROUGE-L	0.1496	1.001	**0.332**	**1.010**	0.358	0.984	0.385	0.990	0.394	0.975	0.498
TriviaQA	F1	0.8786	0.972	**0.131**	1.000	0.142	1.000	0.141	**1.009**	0.160	0.887	0.353

Table 11: Cross-method comparison on Mistral-7B.

Task	Metric	Q_{full}	CacheBlend		A3		ProphetKV		CacheClip		EPIC		KVPacket		Rand	
			$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$	$R_Q \uparrow$	$R_T \downarrow$
NIAH	Acc.	0.875	**1.438**	0.249	1.394	0.441	1.206	0.212	1.324	0.796	0.052	**0.122**	0.931	0.337	0.000	0.267
CWE	Acc.	0.991	0.998	0.247	0.923	0.480	**1.002**	0.208	0.996	0.873	0.121	0.213	0.939	0.436	0.307	**0.175**
VT	Acc.	0.828	0.948	0.255	0.338	0.471	1.004	0.196	0.914	0.748	0.067	**0.127**	**1.017**	0.497	0.057	0.201
MuSiQue	F1	0.224	**0.982**	0.283	0.452	0.479	0.974	0.222	0.785	1.061	0.889	**0.130**	0.930	0.651	0.785	0.255
SAMSum	ROUGE-L	0.325	1.442	0.290	1.165	0.496	1.291	**0.236**	1.404	2.523	**1.484**	0.379	1.292	0.895	1.425	0.310
2WikiMQA	F1	0.339	0.961	0.295	0.544	0.473	0.894	0.221	**1.055**	1.109	0.793	**0.127**	0.902	0.651	0.820	0.256
GovReport	ROUGE-L	0.206	1.017	0.277	1.016	0.483	**1.081**	0.353	1.062	0.341	1.080	**0.040**	1.006	0.277	1.048	0.315
HotpotQA	F1	0.489	0.976	0.278	0.492	0.483	0.955	0.210	0.626	0.703	0.836	**0.061**	**1.007**	0.778	0.679	0.299
MultiNews	ROUGE-L	0.163	0.982	0.298	**1.033**	0.468	0.980	0.280	1.010	0.866	0.986	**0.112**	1.007	0.580	1.013	0.287
TriviaQA	F1	0.841	0.951	0.343	0.800	0.617	0.997	0.451	0.992	10.930	**1.000**	**0.165**	0.996	0.906	0.989	0.386
MMLU	Acc.	0.5067	0.308	1.120	–	–	1.000	1.792	0.880	3.806	0.906	1.066	**1.125**	**0.710**	0.969	1.025
GSM8K	Acc.	0.3267	0.733	0.904	–	–	**1.176**	1.988	1.118	4.228	0.880	1.120	0.840	**0.643**	1.000	1.027
HumanEval	Acc.	0.0933	0.833	**0.563**	–	–	1.250	2.010	1.000	6.339	**1.714**	0.998	0.714	0.579	0.286	0.969

Tables 7–9 report the complete CacheBlend results for Qwen3-8B, Mistral-7B, and Qwen3-4B, respectively. Table 10 reports results for Qwen3.8-27B with HYPIC variants and LinearKV.

In the CacheBlend tables, $\mathrm{CB}(x)$ denotes CacheBlend with a token recomputation ratio of x: $\mathrm{CB}(0)$ directly reuses the cached states without token recomputation, whereas $\mathrm{CB}(0.05)$ recomputes 5% of the tokens to repair the reused cache. Because R_Q is estimated from finite evaluation samples, occasional values of $R_Q > 1$ are possible. They indicate that the measured reuse score exceeded the paired full-prefill score on those samples, rather than a systematic improvement in the model's underlying task capability.

In Table 11, each method is normalized by its paired Full run. Several A3 entries are unavailable because the A3 implementation does not support reusable chunks interleaved with other tokens; these entries therefore indicate unsupported configurations rather than failed runs.

Tables 4 and 11 show that CacheClip's R_T varies sharply across tasks. In our reproduction, its CPU auxiliary-model forward pass can dominate online latency, which also includes window grouping, token mapping, and selective recomputation. These timings depend on our runtime and kernels and should not be read as a direct reproduction of the original system's efficiency. GovReport is a lower-R_T exception, plausibly because four nearly equal contiguous segments make CPU selection and batching more regular; longer, more heterogeneous query-side inputs such as TriviaQA can increase that cost. These explanations are prompt- and implementation-dependent, not general properties of CacheClip.

B Reusable-Context Length Scaling

We additionally examine how reusable-context length affects the latency benefit of KV reuse. Using the length-controlled COPY workload, we vary only the reusable segment length while holding the model, reuse configuration, and evaluation protocol fixed. This controlled setting separates the effect

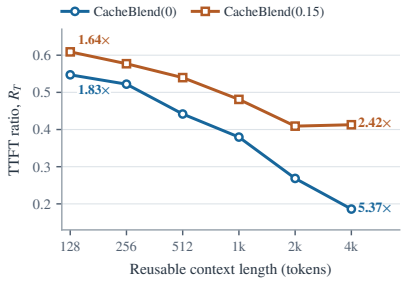


Figure 5: TTFT ratio by reusable length on COPY (Qwen3-8B). Lower is better.

Reusable tokens	Full TTFT	CB(0.15)			CB(0)		
		TTFT	Ratio	Speedup	TTFT	Ratio	Speedup
128	0.3465	0.2110	0.609	1.642×	0.1896	0.547	1.828×
256	0.3764	0.2172	0.577	1.733×	0.1965	0.522	1.915×
512	0.4577	0.2470	0.540	1.853×	0.2022	0.442	2.263×
1024	0.5997	0.2885	0.481	2.078×	0.2277	0.380	2.634×
2048	0.9483	0.3880	0.409	2.444×	0.2546	0.268	3.724×
4096	1.5592	0.6438	0.413	2.422×	0.2901	0.186	5.374×

Table 12: TTFT in seconds on COPY (Qwen3-8B). Ratio and speedup use FULLPREFILL as baseline. Full denotes FULLPREFILL; CB denotes CacheBlend.

of context length from the semantic and structural differences across benchmark tasks. Figure 5 summarizes the resulting TTFT ratios, and Table 12 provides the corresponding raw measurements for Qwen3-8B under two CacheBlend configurations.

As reusable context grows, the latency benefit of reuse becomes larger. For CacheBlend(0), the TTFT ratio decreases from 0.547 at 128 reusable tokens to 0.186 at 4096 tokens, corresponding to a speedup increase from $1.83\times$ to $5.37\times$. CacheBlend(0.15) improves from 0.609 to approximately 0.41 over the same range, with most of the gain realized by 2048 tokens and little further improvement thereafter.

This behavior is consistent with two simple effects. Fixed reuse overheads are better amortized when more prefill computation can be avoided, whereas residual work that scales with context length limits the benefit at longer lengths. Thus, reuse ratio alone does not determine latency reduction: the absolute amount of reusable context and the residual computation of the reuse method also matter.

C SKILL-REUSE EVALUATION DETAILS

Setup. We evaluate skill reuse on all 87 tasks in SkillsBench 1.1 (Li et al., 2026), using one agent rollout per task and configuration. Experiments use Qwen3.8-27B with pi (v0.73.1) via `pi-acp` (v0.0.32), running on $2\times$ NVIDIA RTX PRO 6000 96 GB GPUs. Qwen3.8-27B interleaves recurrent Gated DeltaNet layers with full-attention layers (Qwen Team, 2026b); we therefore evaluate reuse methods that support this hybrid-state setting.

We follow the recommended Qwen3.8 thinking-mode sampling configuration, with temperature 1.0, top-p 0.95, top-k 20, and minimum-p 0. We set the maximum generation length to 40,960 tokens and the maximum model length to 256,000 tokens. Each agent execution has a timeout of 5 hours.

For each task, the provided skill documents are prepared independently before the agent run, and their cached states are reused when processing the first model input. This offline preparation cost is excluded from the measured first-call TTFT. We compare against NOSKILL, which runs the same task without the skill context, and FULLPREFILL, which processes the skill-conditioned input through standard model computation. Exact prefix caching remains enabled across all configurations.

For HYPIC (Liu et al., 2026a), we evaluate Addition (A), Transition (T), Transition+RoPE (TR), and Transition+RoPE+Recompute (TRR). TRR uses the default seam-recomputation width of 8 tokens. T and TR are used in the latency analysis, while the quality comparison reports Addition and TRR. We additionally evaluate LINEARKV (Liu et al., 2026b), using EPIC as the token-recomputation selector with a 20% recomputation budget.

Latency versus reusable skill context. Figure 6 shows the four internal HYPIC variants. When only a small fraction of the first-call context is reusable, reuse provides limited benefit and its additional processing can offset saved prefill computation. As the reusable fraction increases, normalized TTFT falls rapidly; for highly reusable executions, the HYPIC variants reach roughly 20–30% of

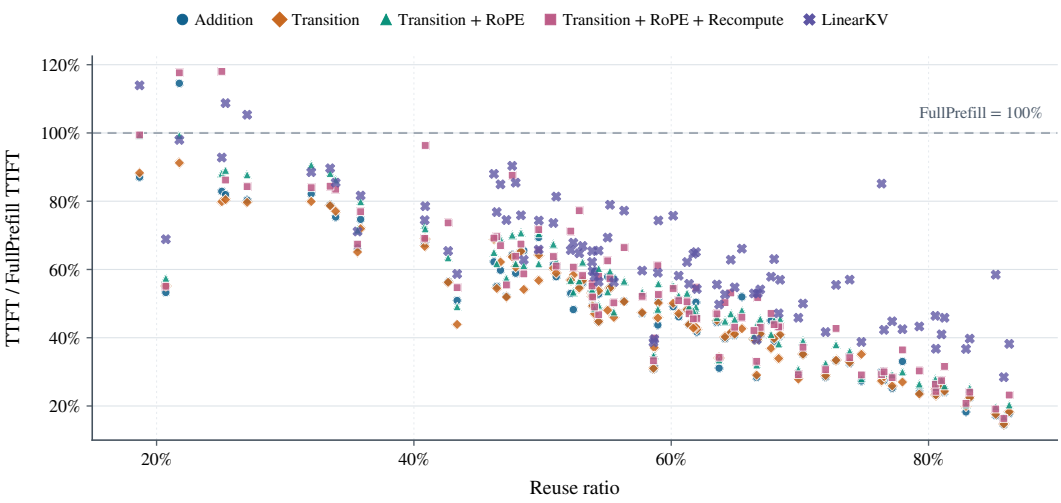


Figure 6: First-call TTFT versus reuse ratio across SkillsBench tasks, for four internal HYPIC variants and LINEARKV. TTFT is normalized to FULLPREFILL; lower is better.

FULLPREFILL. The variants follow a similar overall trend, indicating that the amount of recurring skill context available for reuse is a strong predictor of the attainable systems benefit in these runs.

C.1 FULL PER-TASK SKILLSBENCH RESULTS

Tables 13 and 14 report the complete per-task results used in the skill-reuse study. The reward table includes FULLPREFILL, TRR, Addition, NOSKILL, and LINEARKV. The TTFT table additionally reports the Transition and Transition+RoPE variants. TTFT is reported in seconds. Prompt length is measured in tokens, and reuse ratio is reported as a fraction. Long task identifiers are abbreviated consistently across both tables.

Table 13: Complete per-task SkillsBench reward results. Higher is better.

Task	Full Prefill	TRR	Addition	NoSkill	Linear KV
3d-scan-calc	1	1	1	1	1
ada-bathroom-plan-repair	0	0	0	0	0
adaptive-cruise-control	0	0	0	0	0
azure-bgp-route-leak	0	0	0	0	0
bike-rebalance	0	0	0	0	0
citation-check	1	1	1	1	1
civ6-adjacency-optimizer	0	0	0	0	0
court-form-filling	1	1	0	0	1
wyckoff-pos-analysis	0	0	0	0	0
dapt-intrusion-detection	1	1	1	0	1
data-to-d3	0	0	0	0	0
debug-trl-grpo	0.6	0.6	0.6	0.6	0.6
dialogue-parser	0.833	0.667	0.667	0.667	0.833
drone-planning-control	0	0	0	0	0
object-aware-egomotion	0	0	0	0	0
quake-phase-assoc	1	1	1	1	1
quake-plate-calc	1	1	1	1	1
econ-detrend-corr	1	1	1	1	1
edit-pdf	1	1	1	1	1
ac-optimal-power-flow	0	0	0	0	0
energy-market-pricing	1	1	0	0	0
energy-unit-commitment	0	0	0	0	0

Continued on next page

Table 13 (continued)

Task	Full Prefill	TRR	Addition	NoSkill	Linear KV
enterprise-info-search	0	0	0	0	0
exam-block-sequencing	0	0	0	0	0
exceltable-in-ppt	1	1	1	1	1
exoplanet-period	0	0	0	0	0
financial-modeling-qa	0	0	0	0	0
fix-build-agentops	0	0	0	0	0
fix-build-google-auto	0	0	0	0	0
fix-druid-loophole-cve	0	0	0	0	0
fix-erlang-ssh-cve	0	0	0	0	0
fix-visual-stability	1	1	0	0	0
flink-query	0	0	0	0	0
flood-risk-analysis	1	1	1	0	1
glm-lake-mendota	1	1	1	1	1
grav-wave-detection	1	1	0	1	1
grid-dispatch-operator	1	1	1	0	1
hvac-control	1	1	1	1	1
invoice-fraud-detection	0	1	1	0	0
jax-computing-basics	1	1	1	1	1
jpg-ocr-stat	0	0	0	0	0
lab-unit-harmonization	1	0	0	0	0
lake-warming-attribution	1	1	1	0	1
latex-formula-extraction	0	0	0	0	0
lean4-proof	0	0	0	0	0
llm-prefix-cache-replay	1	1	1	0	1
mfg-codebook-norm	0	0	0	0	0
mfg-equipment-maint	1	1	1	0	0
mfg-fjsp-opt	1	0	0	0	0
mario-coin-counting	1	1	0	0	1
mars-clouds-clustering	1	1	1	1	1
multilingual-dubbing	0	0	0	0	0
offer-letter-generator	1	1	1	1	1
organize-messy-files	0	0	0	0	0
paper-anonymizer	1	1	0	1	1
parallel-tfidf-search	0	0	0	0	0
paratransit-routing	0	0	0	0	0
pddl-airport-planning	1	1	1	0	1
pddl-tpp-planning	1	1	1	0	1
pdf-excel-diff	1	1	1	1	1
powerlifting-coef-calc	1	0	0	0	0
pptx-ref-formatting	1	1	1	1	1
protein-expr-analysis	0	1	1	0	0
python-scala-translation	0	1	1	0	0
quantum-numerical-sim	0	0	0	0	0
r2r-mpc-control	1	0	0	0	0
radar-vital-signs	0	0	0	0	0
react-perf-debugging	0	0	0	0	0
reserves-at-risk-calc	0	0	0	0	0
sales-pivot-analysis	0	0	0	0	0
sec-financial-report	1	0	0	0	1
seismic-phase-picking	0	0	0	0	0
setup-fuzzing-py	0.66	0	0	0	0
shock-analysis-demand	0	0	0	0	0
shock-analysis-supply	0	0	0	0	0
simpo-code-reproduction	0	0	0	0	0
software-dep-audit	0	0	0	0	0

Continued on next page

Table 13 (continued)

Task	Full Prefill	TRR	Addition	NoSkill	Linear KV
spring-jakarta-migration	0	0	0	0	0
suricata-custom-exfil	0	0	0	0	0
syzkaller-ppdev-syzlang	0	0	0	0	0
threejs-structure-parser	1	1	1	1	1
threejs-to-obj	1	1	1	1	1
tictoc-abort-detection	0.45	0	0	0	0
travel-planning	1	1	0	0	1
video-silence-remover	0	0	0	0	0
weighted-gdp-calc	0	0	0	0	0
xlsx-recover-data	0	0	0	0	0

Table 14: Complete per-task SkillsBench TTFT results. Values are raw TTFT in seconds; prompt length is in tokens, and reuse ratio is reported as a fraction.

Task	Prompt len.	Reuse ratio	FullPrefill	Addition	Transition	Trans. +RoPE	TRR	LinearKV
3d-scan-calc	2596	0.207	0.618	0.329	0.341	0.355	0.340	0.425
ada-bathroom-plan-repair	7142	0.434	1.230	0.626	0.540	0.605	0.673	0.722
adaptive-cruise-control	5643	0.485	0.773	0.507	0.419	0.473	0.454	0.486
azure-bgp-route-leak	4535	0.497	0.601	0.417	0.341	0.371	0.431	0.395
bike-rebalance	12356	0.720	1.719	0.489	0.497	0.559	0.527	0.716
citation-check	10606	0.806	1.456	0.339	0.338	0.371	0.353	0.535
civ6-adjacency-optimizer	7635	0.655	0.917	0.476	0.391	0.440	0.422	0.606
court-form-filling	4109	0.472	0.624	0.326	0.324	0.359	0.346	0.465
wyckoff-pos-analysis	12068	0.805	1.476	0.377	0.372	0.412	0.389	0.685
dapt-intrusion-detection	8049	0.678	1.041	0.469	0.384	0.427	0.491	0.603
data-to-d3	3969	0.408	0.529	0.358	0.353	0.386	0.366	0.394
debug-trl-grpo	5772	0.602	0.696	0.342	0.349	0.385	0.379	0.527
dialogue-parser	2829	0.218	0.366	0.420	0.334	0.364	0.431	0.359
drone-planning-control	9113	0.646	1.109	0.456	0.461	0.522	0.589	0.697
object-aware-egomotion	4400	0.467	0.581	0.347	0.361	0.401	0.389	0.493
quake-phase-assoc	9704	0.728	1.182	0.394	0.395	0.449	0.505	0.656
quake-plate-calc	4108	0.497	0.506	0.329	0.325	0.357	0.333	0.376
econ-detrend-corr	3507	0.320	0.438	0.360	0.350	0.397	0.368	0.388
edit-pdf	5282	0.552	0.651	0.356	0.355	0.387	0.373	0.514
ac-optimal-power-flow	7008	0.614	0.897	0.393	0.393	0.442	0.422	0.500
energy-market-pricing	7850	0.620	1.024	0.427	0.434	0.482	0.468	0.556
energy-unit-commitment	8227	0.642	1.065	0.424	0.429	0.478	0.536	0.561
enterprise-info-search	4410	0.531	0.576	0.323	0.327	0.358	0.335	0.385
exam-block-sequencing	7379	0.649	0.949	0.386	0.390	0.431	0.409	0.519
exceltable-in-ppt	11506	0.810	1.401	0.355	0.357	0.388	0.384	0.574
exoplanet-period	11040	0.772	1.522	0.384	0.394	0.445	0.431	0.682
financial-modeling-qa	6874	0.685	0.826	0.335	0.338	0.376	0.358	0.471
fix-build-agentops	9204	0.748	1.387	0.379	0.487	0.389	0.404	0.537
fix-build-google-auto	13207	0.832	1.607	0.359	0.360	0.407	0.387	0.639
fix-druid-loophole-cve	7796	0.699	1.315	0.379	0.367	0.405	0.384	0.603
fix-erlang-ssh-cve	18033	0.859	2.831	0.410	0.417	0.478	0.463	0.805
fix-visual-stability	5940	0.637	1.064	0.331	0.362	0.357	0.364	0.529
flink-query	9888	0.739	1.205	0.391	0.395	0.435	0.412	0.687
flood-risk-analysis	5095	0.577	0.700	0.329	0.331	0.374	0.365	0.418
glm-lake-mendota	4801	0.544	0.631	0.333	0.340	0.381	0.366	0.414
grav-wave-detection	5082	0.563	0.663	0.335	0.335	0.375	0.440	0.512
grid-dispatch-operator	5931	0.606	0.768	0.354	0.362	0.402	0.391	0.447
hvac-control	7109	0.619	0.943	0.475	0.407	0.463	0.515	0.614
invoice-fraud-detection	8098	0.684	1.137	0.458	0.386	0.435	0.491	0.536
jax-computing-basics	3209	0.356	0.492	0.327	0.320	0.352	0.332	0.350
jpg-ocr-stat	18546	0.852	2.466	0.422	0.433	0.489	0.472	1.442
lab-unit-harmonization	5250	0.544	0.795	0.356	0.356	0.394	0.372	0.448
lake-warming-attribution	5183	0.551	0.761	0.440	0.365	0.407	0.476	0.528
latex-formula-extraction	4793	0.508	0.586	0.360	0.354	0.396	0.374	0.432
lean4-proof	7548	0.702	0.987	0.347	0.347	0.386	0.367	0.494
llm-prefix-cache-replay	6056	0.617	0.828	0.358	0.355	0.397	0.377	0.535
mfg-codebook-norm	3184	0.270	0.438	0.352	0.349	0.384	0.369	0.461
mfg-equipment-maint	6259	0.539	0.840	0.416	0.414	0.459	0.436	0.500
mfg-fjsp-opt	3318	0.359	0.446	0.333	0.321	0.356	0.343	0.364

Continued on next page

Table 14 (continued)

Task	Prompt len.	Reuse ratio	FullPrefill	Addition	Transition	Trans. +RoPE	TRR	LinearKV
mario-coin-counting	4149	0.427	0.631	0.356	0.355	0.401	0.465	0.413
mars-clouds-clustering	5133	0.464	0.720	0.396	0.392	0.445	0.501	0.553
multilingual-dubbing	8438	0.680	1.023	0.399	0.407	0.462	0.448	0.645
offer-letter-generator	3808	0.483	0.480	0.314	0.313	0.339	0.324	0.364
organize-messy-files	18038	0.863	2.281	0.406	0.418	0.464	0.529	0.871
paper-anonymizer	5034	0.590	0.656	0.333	0.329	0.367	0.346	0.488
parallel-tfidf-search	7101	0.667	0.916	0.359	0.362	0.407	0.474	0.485
paratransit-routing	6203	0.669	0.806	0.330	0.332	0.367	0.347	0.437
pddl-airport-planning	3052	0.251	0.419	0.347	0.335	0.370	0.494	0.389
pddl-tpp-planning	3017	0.254	0.412	0.338	0.332	0.367	0.355	0.448
pdf-excel-diff	7084	0.664	0.913	0.362	0.362	0.372	0.385	0.484
powerlifting-coef-calc	11565	0.793	1.586	0.379	0.373	0.420	0.480	0.687
pptx-ref-formatting	8584	0.764	1.170	0.351	0.320	0.353	0.342	0.996
protein-expr-analysis	5120	0.540	0.746	0.443	0.351	0.363	0.366	0.432
python-scala-translation	15573	0.829	2.241	0.409	0.446	0.457	0.466	0.823
quantum-numerical-sim	5198	0.524	0.762	0.368	0.445	0.407	0.462	0.517
r2r-mpc-control	4204	0.339	0.515	0.388	0.397	0.444	0.430	0.440
radar-vital-signs	6358	0.587	1.250	0.394	0.464	0.436	0.499	0.495
react-perf-debugging	5229	0.586	1.074	0.331	0.333	0.373	0.357	0.415
reserves-at-risk-calc	5138	0.539	0.669	0.360	0.355	0.363	0.374	0.438
sales-pivot-analysis	6056	0.612	0.744	0.357	0.359	0.395	0.376	0.463
sec-financial-report	3583	0.335	0.446	0.352	0.351	0.394	0.377	0.400
seismic-phase-picking	7980	0.666	1.381	0.392	0.401	0.443	0.457	0.545
setup-fuzzing-py	10370	0.780	1.337	0.442	0.361	0.402	0.487	0.569
shock-analysis-demand	5138	0.539	0.672	0.351	0.351	0.364	0.371	0.418
shock-analysis-supply	5419	0.511	0.662	0.383	0.390	0.414	0.404	0.538
simpo-code-reproduction	4621	0.529	0.609	0.336	0.332	0.346	0.470	0.394
software-dep-audit	9703	0.765	1.264	0.348	0.362	0.373	0.379	0.535
spring-jakarta-migration	13598	0.812	1.662	0.399	0.405	0.432	0.525	0.762
suricata-custom-exfil	4083	0.462	0.535	0.333	0.369	0.348	0.371	0.471
syzkaller-ppdev-syzlang	4261	0.476	0.531	0.341	0.338	0.373	0.465	0.480
threejs-structure-parser	4277	0.479	0.565	0.333	0.343	0.349	0.361	0.483
threejs-to-obj	3520	0.409	0.466	0.319	0.321	0.335	0.449	0.366
tictoc-abort-detection	5606	0.522	0.725	0.384	0.412	0.412	0.516	0.476
travel-planning	3475	0.187	0.468	0.407	0.413	0.465	0.465	0.533
video-silence-remover	6791	0.590	0.902	0.394	0.413	0.436	0.551	0.532
weighted-gdp-calc	4982	0.555	0.731	0.419	0.336	0.348	0.367	0.412
xlsx-recover-data	5630	0.636	0.730	0.325	0.329	0.336	0.343	0.406

D Cross-Document Dependency Analysis

This appendix analyzes the relationship between cross-document dependency and quality degradation under approximate KV reuse on the RAG-style context-reuse workload under CacheBlend. The main text summarizes the central observation: dependency measurements are highly consistent across models at the task level, but dependency magnitude alone has limited explanatory power for reuse-induced quality loss.

D.1 Dependency Metrics

We evaluate six measures of cross-document dependency. All metrics are computed at the sample level and are aggregated only when task-level analyses are required.

Let the reusable context consist of chunks $C_1, \ldots, C_m$, and let $\alpha_{t,s}^{(\ell,h)}$ denote the full-prefill attention weight at layer ℓ and attention head h from query token t to key token s. For a token $t \in C_i$, let b_i denote the first aligned token position of that chunk in the full prompt.

Missing Attention. When a reusable chunk is encoded independently, tokens in preceding context are unavailable. Missing Attention therefore measures the attention mass that a token assigns, under full-prefill, to keys preceding its own chunk:

$$d_{\mathrm{MA}}^{(\ell,h)}(t) = \sum_{s<b_i} \alpha_{t,s}^{(\ell,h)}, \qquad t \in C_i. \tag{1}$$

The sample-level Missing Attention score is the mean of $d_{\mathrm{MA}}^{(\ell,h)}(t)$ over all evaluated reusable tokens, layers, and attention heads. A larger value indicates that, under full-prefill, a greater fraction of the token's attention mass is assigned to context unavailable during independent chunk encoding.

Value-Weighted Dependency. Attention mass alone does not account for the magnitude of the value representations being aggregated. We therefore additionally compute Value-Weighted Dependency (VWD). Let $V_s^{(\ell,h)}$ denote the value vector associated with key token s. For each reusable token,

$$d_{\mathrm{VWD}}^{(\ell,h)}(t) = \frac{\sum_{s<b_i} \alpha_{t,s}^{(\ell,h)} \left\| V_s^{(\ell,h)} \right\|_2}{\sum_{s\le t} \alpha_{t,s}^{(\ell,h)} \left\| V_s^{(\ell,h)} \right\|_2}. \tag{2}$$

As with Missing Attention, the final sample-level score is averaged over reusable tokens, layers, and attention heads. VWD thus measures the fraction of the value-weighted attention contribution that originates from context unavailable under independent chunk encoding.

Skip-k variants. Tokens near the beginning of a reusable chunk are expected to exhibit particularly strong dependence on preceding context. To distinguish this boundary-local effect from dependency deeper inside the chunk, we additionally compute Skip-k variants of Missing Attention and VWD. Specifically, we exclude the first k token positions of each independently encoded chunk from the query-token average, while leaving the metric definitions otherwise unchanged. Throughout this study, we use $k = 16$. We denote the resulting metrics as Missing Attention Skip-k and VWD-SkipK.

KV Deviation. KV Deviation directly measures how much the cached representations themselves change when a chunk is encoded independently rather than in its full context. Tokens in independently encoded chunks are aligned to their corresponding full-prompt tokens, and the independent prefill uses the same absolute positional indices as the full-prefill. This avoids attributing differences caused purely by positional encoding to contextual dependency.

For an aligned token pair $(t, \tilde{t})$, let $K_{\ell,h,t}^{\mathrm{full}}$ and $V_{\ell,h,t}^{\mathrm{full}}$ denote its full-prefill KV states, and let $K_{\ell,h,\tilde{t}}^{\mathrm{ind}}$ and $V_{\ell,h,\tilde{t}}^{\mathrm{ind}}$ denote the corresponding states produced by independent chunk encoding. We define

$$d_K(\ell,h,t) = 1 - \cos\left(K_{\ell,h,t}^{\mathrm{full}}, K_{\ell,h,\tilde{t}}^{\mathrm{ind}}\right), \tag{3}$$

$$d_V(\ell,h,t) = 1 - \cos\left(V_{\ell,h,t}^{\mathrm{full}}, V_{\ell,h,\tilde{t}}^{\mathrm{ind}}\right), \tag{4}$$

and compute

$$D_{\mathrm{KV}} = \frac{1}{|\Omega|} \sum_{(\ell,h,t)\in\Omega} \frac{d_K(\ell,h,t) + d_V(\ell,h,t)}{2}, \tag{5}$$

where Ω contains all valid aligned token, layer, and KV-head combinations. KV Deviation therefore provides a representation-level measure that is complementary to the attention-based dependency metrics.

CCI. We additionally implement Cache Context Impact (CCI), introduced by Cache-Craft (Agarwal et al., 2025). CCI compares the attention received from preceding reusable chunks with self-contextualization within the current chunk. Following Cache-Craft, for each layer ℓ we define

$$\mathrm{inter}_\ell(C_j, C_i) = \frac{1}{H} \sum_{h=1}^{H} \sum_{t\in C_i} \sum_{s\in C_j} \alpha_{t,s}^{(\ell,h)}, \qquad j < i, \tag{6}$$

$$\mathrm{intra}_\ell(C_i) = \frac{1}{H} \sum_{h=1}^{H} \sum_{t\in C_i} \sum_{\substack{s\in C_i \\ s<t}} \alpha_{t,s}^{(\ell,h)}. \tag{7}$$

The normalized inter- and intra-chunk quantities are

$$a_\ell(C_i) = \sum_{j<i} \frac{\mathrm{inter}_\ell(C_j, C_i)}{|C_i|\,|C_j|}, \qquad b_\ell(C_i) = \frac{\mathrm{intra}_\ell(C_i)}{|C_i|^2}. \tag{8}$$

Table 15: Task-level relationships among dependency metrics. All metrics are first averaged within each model–task pair. Spearman correlations are then computed across tasks separately for each model; each entry reports the median correlation across the three models.

	MA	VWD	MA-Skip-k	VWD-SkipK	KVD	CCI
MA	1.000	0.978	1.000	0.978	0.533	0.524
VWD		1.000	0.978	1.000	0.588	0.441
MA-Skip-k			1.000	0.978	0.533	0.524
VWD-SkipK				1.000	0.588	0.441
KVD					1.000	0.182
CCI						1.000

They are averaged across the L transformer layers,

$$\bar{a}(C_i) = \frac{1}{L}\sum_{\ell=1}^{L} a_\ell(C_i), \qquad \bar{b}(C_i) = \frac{1}{L}\sum_{\ell=1}^{L} b_\ell(C_i), \tag{9}$$

and Cache-Craft defines

$$\mathrm{CCI}(C_i) = \frac{1}{1+\exp\big(-\bar{a}(C_i)/\bar{b}(C_i)\big)}. \tag{10}$$

A larger CCI indicates that the chunk is more strongly contextualized by its preceding chunks relative to its own internal context. In our implementation, CCI is computed for every reusable chunk with a preceding reusable chunk, and the sample-level score used in the analyses below is the token-count-weighted mean

$$\mathrm{CCI}_{\mathrm{sample}} = \frac{\sum_{i=2}^{m} |C_i|\ \mathrm{CCI}(C_i)}{\sum_{i=2}^{m} |C_i|}. \tag{11}$$

D.2 Consistency Across Metrics and Models

We first examine the relationships among the dependency metrics themselves. To characterize task-level dependency while placing all six metrics on the same footing, we average each metric within every model–task pair. We then compute Spearman correlations across tasks separately for each model and report the median correlation across the three models in Table 15.

The four attention-based metrics are nearly redundant at the task level. Their pairwise median correlations range from 0.978 to 1.000, indicating that Missing Attention, VWD, and their Skip-k variants largely rank task dependency in the same way. KV Deviation is substantially less correlated with this family, with $\rho = 0.533$–0.588. CCI likewise shows only moderate correlation with the attention-based metrics ($\rho = 0.441$–0.524) and is only weakly correlated with KV Deviation ($\rho = 0.182$).

These relationships motivate the compact representative subset emphasized in the main text. We use VWD-SkipK as a representative of the highly redundant attention-based family because it combines attention mass with the magnitude of the corresponding value representations while excluding local cross-segment interactions within the Skip-k window. We additionally retain KV Deviation and CCI because neither is well approximated by the attention-based family, and they are themselves only weakly correlated. Together, these three metrics provide complementary views of cross-document dependency. For completeness, all six metrics are retained in the analyses reported in this appendix.

We next examine whether these dependency measurements are stable across models. Table 16 reports task-level correlations obtained by averaging sample-level dependency within each task. For the four attention-based metrics, all pairwise Pearson correlations across Qwen3-4B, Qwen3-8B, and Mistral-7B exceed 0.99. KV Deviation is also highly consistent between Qwen3-4B and Qwen3-8B, although its agreement is weaker for comparisons involving Mistral-7B. CCI exhibits similarly high task-level consistency, with pairwise correlations between 0.988 and 0.995.

The sample-level results provide a more fine-grained view. For the four attention-based metrics, the median per-task Spearman correlation across model pairs ranges from 0.885 to 0.968. The corresponding values for KV Deviation range from 0.884 to 0.989. CCI is less stable at the individual sample level: its median per-task correlations are 0.862 for Qwen3-4B versus Qwen3-8B, 0.652

Table 16: Cross-model consistency of dependency metrics at the task level. Values are pairwise Pearson correlations between task-level dependency scores.

Metric	Qwen3-4B vs. Qwen3-8B	Qwen3-4B vs. Mistral-7B	Qwen3-8B vs. Mistral-7B
Missing Attention	0.999	0.997	0.995
VWD	0.999	0.997	0.994
Missing Attention Skip-k	0.999	0.996	0.995
VWD-SkipK	0.999	0.996	0.993
KV Deviation	0.994	0.902	0.855
CCI	0.995	0.988	0.993

for Qwen3-4B versus Mistral-7B, and 0.822 for Qwen3-8B versus Mistral-7B. Thus, the aggregate dependency structure of a task is highly reproducible across models for all six metrics, while the precise ordering of individual samples is somewhat more model-dependent for CCI.

Taken together with Table 15, these results distinguish two properties of the measurements: the four attention-based metrics are highly redundant with one another, whereas KV Deviation and CCI capture more distinct dependency signals. At the same time, the task-level structure measured by each metric is largely preserved across model choice.

D.3 Dependency Versus Reuse Quality

We next test whether samples with stronger cross-document dependency suffer larger quality degradation under approximate KV reuse. We consider two summaries of reuse degradation. Let the signed quality loss at recomputation ratio r be the full-prefill quality minus the quality obtained with CacheBlend at ratio r. L_0 denotes the loss under the most aggressive reuse configuration, with recomputation ratio 0, while *LossAUC* summarizes the signed loss over recomputation ratios $\{0, 0.05, 0.10, 0.15, 0.20\}$ using the trapezoidal area normalized by the interval width (0.20). Negative losses are retained rather than clipped, so that occasional quality improvements under reuse do not introduce a positive bias.

A naive correlation across all samples is difficult to interpret because both dependency and task difficulty vary substantially across tasks. We therefore analyze the relationship primarily within tasks. For each task, we compute a Spearman correlation between sample-level dependency and quality loss. We additionally fit task-controlled regressions in which dependency is standardized within each task and task fixed effects absorb differences in average task difficulty and dependency level. We report the incremental variance explained by dependency, denoted ΔR^2, after accounting for task identity. Tasks receive equal weight in this analysis.

Table 17 reports the complete results for all six dependency metrics. Although the main text emphasizes VWD-SkipK, KV Deviation, and CCI as a compact representative subset, we report the remaining attention-based variants here to verify that the conclusion does not depend on the particular representative chosen from that highly correlated family.

The relationship is consistently weak. Across all six metrics, three models, and both quality-loss summaries, the median within-task Spearman correlation never exceeds 0.143 in magnitude. The largest incremental variance explained is $\Delta R^2 = 0.00435$, corresponding to only 0.435% of additional variance, for Missing Attention on Qwen3-4B under L_0. CCI follows the same overall pattern: its within-task median correlations remain close to zero, and its largest ΔR^2 in this analysis is 0.00340. Thus, the conclusion does not depend on whether dependency is quantified using attention mass, value-weighted attention, representation deviation, or CCI.

D.4 Robustness Checks

We perform several additional checks to ensure that the weak relationship is not an artifact of a particular dependency metric, subset of the data, or reuse configuration.

Alternative dependency metrics. The weak relationship is not specific to VWD-SkipK. As shown in Table 17, Missing Attention, VWD, their Skip-k variants, KV Deviation, and CCI all produce small and sign-inconsistent within-task correlations. This comparison spans metrics based

Table 17: Relationship between sample-level dependency and reuse-induced quality degradation. Each entry reports $\tilde{\rho}/\Delta R^2$, where $\tilde{\rho}$ is the median per-task Spearman correlation and ΔR^2 is the additional proportion of variance explained after controlling for task identity.

Model	Outcome	MA	VWD	MA-Skip-k	VWD-SkipK	KVD	CCI
Qwen3-4B	L_0	$-0.031/0.00435$	$-0.011/0.00417$	$-0.028/0.00421$	$-0.027/0.00399$	$0.009/0.00021$	$0.011/0.00340$
	LossAUC	$-0.021/0.00013$	$-0.039/0.00001$	$-0.024/0.00006$	$-0.043/0.00000$	$0.036/0.00024$	$-0.037/0.00164$
Qwen3-8B	L_0	$-0.053/0.00003$	$-0.022/0.00004$	$-0.056/0.00003$	$-0.034/0.00005$	$-0.023/0.00002$	$0.005/0.00051$
	LossAUC	$0.054/0.00084$	$0.082/0.00042$	$0.044/0.00081$	$0.081/0.00038$	$-0.070/0.00023$	$-0.031/0.00128$
Mistral-7B	L_0	$0.138/0.00050$	$0.046/0.00074$	$0.143/0.00049$	$0.040/0.00078$	$0.071/0.00002$	$0.038/0.00078$
	LossAUC	$0.098/0.00048$	$0.083/0.00037$	$0.095/0.00042$	$0.083/0.00035$	$0.082/0.00038$	$0.005/0.00003$

on attention mass, value-weighted attention, representation deviation, and cache-context impact, yet none provides substantial predictive power for reuse-induced quality degradation.

Ceiling effects. Some tasks contain many samples on which both full-prefill and reuse achieve perfect quality, leaving no observable degradation for those samples and potentially attenuating correlation estimates. We therefore repeat the analysis after removing samples that achieve perfect quality under both full prefill and the most aggressive CacheBlend configuration. The relationship does not become substantially stronger. Across all six metrics, three models, and both L_0 and LossAUC, the largest task-controlled ΔR^2 is 0.00209, corresponding to only 0.209% of additional variance. This maximum occurs for CCI on Qwen3-8B under LossAUC.

Across reuse levels. The conclusion is also robust to the choice of reuse severity. We repeat the task-controlled analysis separately using the signed quality loss at each CacheBlend recomputation ratio from 0 to 0.20. Across all six dependency metrics and all three models, the largest ΔR^2 observed at any recomputation ratio is 0.00512, corresponding to 0.512% of additional variance. This maximum occurs for CCI on Qwen3-8B at recomputation ratio 0.20. Even in this most favorable case, dependency accounts for only about half of one percent of additional variance after controlling for task identity. We therefore find no evidence that dependency becomes a substantially stronger predictor at either more aggressive or more conservative reuse levels.

Across models. The weak predictive relationship appears across Qwen3-4B, Qwen3-8B, and Mistral-7B. This is particularly informative because the task-level dependency patterns are highly stable across models for all six metrics, while the attention-based metrics and KV Deviation also show strong sample-level agreement. CCI exhibits somewhat lower sample-level cross-model consistency, yet its relationship with quality degradation remains similarly weak. Thus, neither dependency measures that are highly model-stable nor one whose fine-grained values are more model-dependent provides strong predictive power for reuse-induced quality loss.

Taken together, these results indicate that cross-document dependency is a measurable and highly reproducible workload property at the task level, but its scalar magnitude alone has limited predictive power for robustness to approximate KV reuse. Whether discarding or approximating cross-segment interactions affects downstream quality must therefore depend on additional properties of the model, task, reuse method, or the particular interactions being altered, rather than only on the aggregate amount of cross-document dependency.