RECEIVER-CONDITIONED SELECTION AND RECONSTRUCTION FOR LATENT COMMUNICATION BETWEEN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

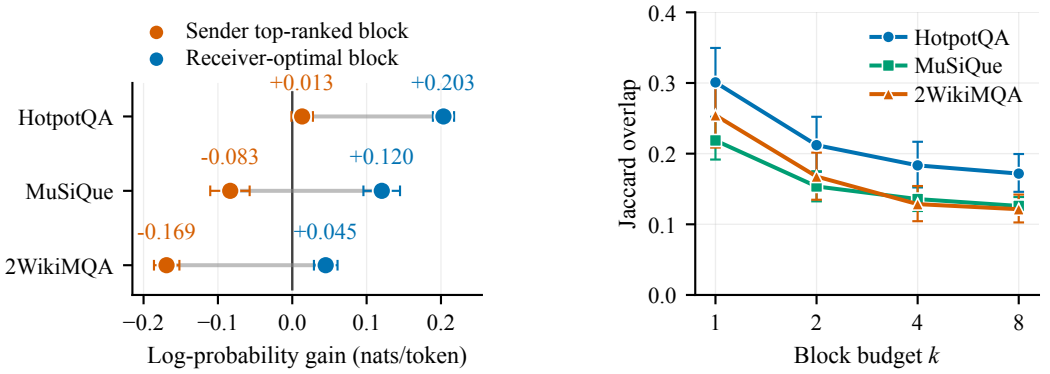
Latent communication through key-value (KV) caches can avoid natural-language decoding and repeated prefilling by transferring internal states between large language model (LLM) agents. Existing approaches fix transmission choices before the current query is known or reuse latent states formed outside the receiver context. We find that the content a sender ranks first can hurt the receiver and that different receivers need different content. We therefore condition what to retrieve and how to read it on the receiver. Each document is written once, before any receiver or query exists, as an INT8 snapshot of its early-layer state. At read time, the receiver uses its state to select a document and reruns the upper layers under its private context, so document states are formed for that receiver rather than transplanted. Across Qwen3-4B, Qwen3-8B, and Llama-3.1-8B-Instruct, receiver-aware selection with BGE-M3 improves F1 by 15.5–17.2 points over query-only selection for both text and checkpoint delivery. With selection fixed, read-time reconstruction improves F1 by 5.1–13.5 points over full static KV reuse with payloads $32\times$ to $57\times$ smaller. Under controlled TCP networking, the complete system reduces time to first token (TTFT) by 6.91%, 5.81%, and 4.23% on Qwen3-4B, Qwen3-8B, and Llama-3.1-8B-Instruct, respectively. In the end-to-end evaluation, answer quality remains comparable to native text, with F1 differences ranging from -0.82 to $+1.88$ points. Code is available at `anonymous.4open.science/r/rcsr-latent-comm-B184`.

1 INTRODUCTION

Multi-agent large language model (LLM) systems increasingly divide work across specialized agents that exchange intermediate information before producing a final response (Wu et al., 2024; Li et al., 2023; Hong et al., 2024; Chen et al., 2024b). Most communicate through text, which requires the sender to decode its internal state into tokens and the receiver to prefill those tokens again. Latent communication can avoid part of this repeated work by transferring hidden states or key-value (KV) caches directly (Shi et al., 2026; Fu et al., 2026; Liu et al., 2026b; Zou et al., 2026). The saving compounds when the same content is read more than once, because every reader of text would prefill the same document again. This naturally leads to a write-once, query-later store, where documents are encoded before the reader and question are known and their latent representations are stored for later access (Chen et al., 2026; Baquero et al., 2026). In this setting the eventual receiver must make two decisions: ① which stored document to retrieve; ② how to form that document's keys and values under the receiver's own prefix, which contains its question and private prompt. The first determines what information reaches the receiver. The second is specific to latent reuse, because keys and values formed under the write prefix are not the ones the receiver would form after its own prefix. However, text sidesteps this mismatch by prefilling the retrieved document anew, at the cost of restoring exactly the prefill that latent communication seeks to avoid. Poor selection or mismatched states can make the transfer unhelpful, so both decisions are central to the benefit of latent communication. We call them receiver-conditioned selection and receiver-conditioned reconstruction.

Prior work addresses different parts of this setting. Multi-hop retrieval uses accumulated evidence to guide further retrieval (Xiong et al., 2021; Khattab et al., 2021; Trivedi et al., 2023), while KVComm by Shi et al. (2026) selects layers of KV pairs using receiver attention on calibration data. Cross-

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.



(a) Sender-side rankings can hurt the receiver.

(b) Different receivers select different subsets.

Figure 1: Diagnostics of receiver context on Qwen3-4B. (a) Gains in mean gold answer log probability per token from blocks preferred by the receiver and blocks ranked highest by the sender, relative to no communication. (b) Mean Jaccard overlap between block subsets selected for two receivers by approximate beam search on HotpotQA, MuSiQue, and 2WikiMultihopQA. Error bars show 95% bootstrap intervals over questions. 2WikiMQA abbreviates 2WikiMultihopQA. Oracle selection is used only for this diagnostic. Details are in Appendix A.

context reuse methods recompute selected states or correct existing caches (Yao et al., 2025; Ye et al., 2025). KV Packet and SemPIC learn reusable document representations for an unchanged reader (Chen et al., 2026; Xie et al., 2026), and KV-Direct reconstructs KV from residual checkpoints to reduce inference memory (Qasim et al., 2026). Our focus is the combination of receiver-conditioned document selection and read-time reconstruction in a query-later store.

We ran two block-level diagnostics on Qwen3-4B to measure how much the receiver matters (Figure 1). For each question, the two receivers each hold a different supporting document, while the sender has access to both. First, the block ranked highest by a sender-side attention score lowers the receiver's gold-answer log probability on two of three datasets, whereas the receiver-optimal block helps on all three. Second, across HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), and 2WikiMultihopQA (Ho et al., 2020), the mean Jaccard overlap between the subsets selected by beam search for two receivers ranges from 0.12 to 0.30 for block budgets $k \in \{1, 2, 4, 8\}$. These results motivate using receiver-side information for selection. Because receivers differ while the eventual receiver is unknown at write time, we defer receiver-specific contextualization until read time.

Motivated by these findings, we separate what to retrieve from how to read it and condition both decisions on the receiver. Each document is written once, before any receiver or query exists, as a compact snapshot of its early-layer state. At read time, the receiver uses its own context to select a document and completes the remaining layers of the frozen backbone under its private prefix, so the document's upper-layer states are formed for that receiver rather than transplanted from the writer. No parameters are trained. We evaluate selection, reconstruction, and the complete system on three backbones through matched comparisons, mismatched-content controls (Cheng et al., 2026), and end-to-end measurements over controlled TCP.

Our contributions are as follows:

- We propose a write-once latent communication system that separates receiver-conditioned selection from read-time reconstruction. In a selection study using BGE-M3 (Chen et al., 2024a) across three backbones, receiver conditioning improves F1 by 15.5–17.2 points over query-only selection for both text and checkpoint delivery.
- We reconstruct upper-layer document states from a stored INT8 residual checkpoint under the receiver's prefix. With selection fixed, read-time reconstruction improves F1 by 5.1–13.5 points over full static KV reuse with payloads 32× to 57× smaller, and it also exceeds KVCOMM (Ye et al., 2025) and CacheBlend (Yao et al., 2025) applied to the full static payload.

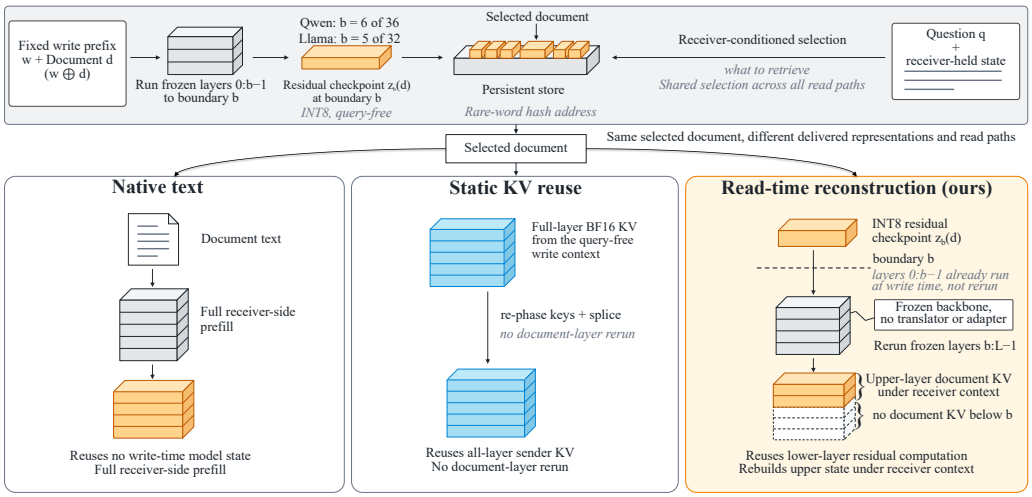


Figure 2: Three read paths for the same selected document. Native text is prefilled under the receiver's prefix. Static KV reuse transfers write-time caches with positional correction. Our method stores an INT8 residual checkpoint and reconstructs upper-layer states under the receiver's prefix using the frozen backbone.

- We evaluate the complete pipeline on three backbones under controlled TCP networking. It reduces time to first token (TTFT) by 4.23–6.91% and improves F1 over no communication by 19.17–22.56 points, while answer quality remains comparable to native text, with F1 differences from -0.82 to $+1.88$ points.

2 Related Work

Context-aware retrieval. MDR conditions subsequent retrieval on the question and previously retrieved passages, Baleen condenses accumulated evidence for further retrieval, and IRCoT interleaves retrieval with reasoning (Xiong et al., 2021; Khattab et al., 2021; Trivedi et al., 2023). These methods establish the value of using available context to find missing information. We study this principle in a shared latent store. The selection study tests whether receiver conditioning improves answer quality under both text and checkpoint delivery; read-time reconstruction addresses how the selected checkpoint is consumed.

Reusable representations and KV reconstruction. CacheBlend selectively recomputes cached tokens, APE aligns parallel encoding through a shared prefix and attention scaling, and block-structured and position-independent methods enable reuse across layouts (Yao et al., 2025; Yang et al., 2025b; Ma et al., 2025; Hu et al., 2025). KV Packet trains soft-token boundary adapters for reusable document packets (Chen et al., 2026). SemPIC trains a LoRA Writer through behavioral distillation to produce document KV for an unchanged Reader, avoiding document recomputation on cache hits (Xie et al., 2026). MaSRead retrieves stored latent fragments through content-addressed routing and reads them under hard attention masks (Baquero et al., 2026). KV-Direct studies residual patching across tasks and reconstructs KV from checkpoints to reduce inference memory (Qasim et al., 2026). KV quantization and streaming reduce the size and delivery cost of existing caches (Liu et al., 2024b;a). Our method trains no parameters and instead recomputes the document's upper layers at read time.

Latent communication between agents. Cache-to-Cache and XKV learn modules for transferring caches between models (Fu et al., 2026; Liu et al., 2026a). DroidSpeak reuses states between agents sharing a base model with selective layer recomputation, while LatentMAS communicates hidden representations through shared latent memory (Liu et al., 2026b; Zou et al., 2026). KVComm by Shi et al. (2026) selects KV layers by receiver attention on calibration data, whereas KVCOMM by Ye et al. (2025) estimates cache offsets from online anchors to adapt shared content

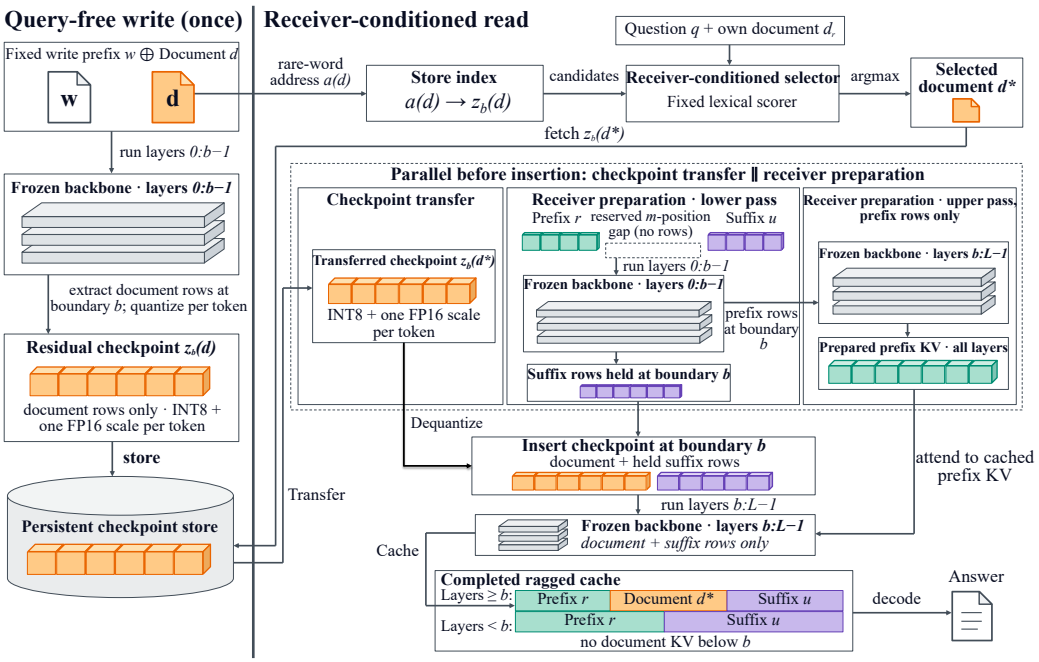


Figure 3: Overview of the shared latent communication system. Documents are written once as INT8 checkpoints with retrieval addresses. At read time, the receiver selects a document and prepares its own states during checkpoint transfer. The checkpoint is then inserted at layer b, and the frozen upper layers reconstruct document states under the receiver's context.

across prefixes. In contrast, our stored checkpoint is fixed before the receiver and query are known, and selection operates on whole documents, conditioned on the receiver's question and existing document.

3 METHOD

3.1 PROBLEM FORMULATION

We consider a write-once, query-later store. A sender stores each document d once, together with an address for lookup, before any question is known. The stored representation is fixed at write time and cannot depend on the eventual receiver. Later a receiver holds a private prefix r containing its question q and a document d_r it already owns, and needs stored documents to answer q. It sees the addresses of all stored documents, fetches the representations of the documents it selects, and generates an answer. All agents run the same frozen L-layer decoder, so model state written by one agent can be resumed by another.

A read therefore involves two choices: which stored documents to fetch, and how each fetched representation becomes key–value states under r. We seek choices that preserve answer quality while reducing TTFT relative to re-prefilling the fetched documents as text. Figure 2 shows the three read paths compared in this paper. Native text stores the document text and re-prefills it under r. Static KV reuse stores the write-time keys and values of every layer and transplants them after positional correction. Read-time reconstruction stores only the residual checkpoint at a boundary layer b and forms every upper-layer document state under r.

3.2 QUERY-FREE CHECKPOINT WRITE

Figure 3 summarizes the complete path from write to answer. Each stored document d is encoded under a fixed write prefix w with no question, using the backbone's native chat template. Appendix B specifies the template, quantization, and address construction. Let m denote the token count of d

under the backbone's tokenizer. The source windows contain 64 Qwen tokens. Llama tokenizes the same text, so its checkpoint length can vary. We run only the layers below the boundary, take the residual stream entering layer b, and keep the rows belonging to d as the residual checkpoint

$$z_b(d) \;=\; \mathcal{Q}\big(\, f^{0:b-1}(w \oplus d)\big|_d\big)\,, \tag{1}$$

where $f^{0:b-1}$ is the composition of decoder layers 0 through $b-1$, $\cdot|_d$ keeps the rows at the positions of d, and $\mathcal{Q}$ applies symmetric INT8 quantization with one FP16 scale per token. The checkpoint is written once to a persistent store and indexed by a 96-byte address $a(d)$ that holds up to 24 distinct unsigned 32-bit hashes of rare terms from the document window. Windows with fewer distinct hashes leave the remaining slots empty, so every address has the same size. A shared table of document frequencies supplies the inverse document frequency (IDF) weights and is stored separately from these addresses. For variable-length entries, the directory also records the token count in a two-byte field used for receiver preparation.

3.3 Selector interface

The selector at the top of Figure 3 uses the receiver's question q and its own document d_r to choose which stored document to retrieve. In the end-to-end system, candidate documents d are represented by their 96-byte addresses $a(d)$. The receiver uses shared term statistics to weight and hash terms from q and d_r. A fixed lexical scorer matches these local features against each address. The score $s(a)$ ranks candidates by combining question relevance and connections to the receiver's context while penalizing lexical redundancy

$$s(a) \;=\; \big(\rho(a,q) + \tfrac{1}{2}\,\beta(a,d_r)\big)\,\big(1-(1-\nu(a))^4\big)\,. \tag{2}$$

Then the receiver selects $d^\star = \arg\max_{d\in\mathcal{D}} s(a(d))$ and fetches its stored checkpoint $z_b(d^\star)$. Here, $\mathcal{D}$ is the set of stored candidates excluding the receiver's own document. The terms $\rho(a,q)$ and $\beta(a,d_r)$ sum the local IDF weights of hashes matching the question and receiver document, respectively. Novelty $\nu(a)$ measures how little of the address is covered by the receiver's features, using local weights and a default weight of one for unmatched hashes. The exact normalization is given in Appendix B. The factor $1-(1-\nu(a))^4$ downweights redundant addresses.

3.4 Read-time upper-state reconstruction

On the receiver-preparation lanes of Figure 3, the receiver inserts the stored document checkpoint at layer b and runs the frozen upper layers under its own prefix. The receiver first processes its private prefix r and prompt suffix u. It reserves m positions between them for the document. We denote this layout by $\mathcal{L}_m(r,u)$. The prefix and suffix retain their final position indices, but the reserved positions contain no token rows. The prefix and suffix are processed through layers 0 to $b-1$. Then, the dequantized checkpoint is inserted at layer b

$$\begin{aligned} H^{(b)} &= \mathrm{insert}_m\big(f^{0:b-1}(\mathcal{L}_m(r,u))\,,\,\mathcal{Q}^{-1}(z_b(d^\star))\big)\,,\\ H^{(L)} &= f^{b:L-1}\Big(H^{(b)}\Big)\,, \end{aligned} \tag{3}$$

where $\mathcal{Q}^{-1}$ dequantizes the checkpoint, $\mathrm{insert}_m(X,Z)$ inserts the m document rows of Z at the reserved positions without changing the other rows, and $f^{b:L-1}$ applies frozen decoder layers b through $L-1$.

In execution, the receiver computes and caches the prefix KV at every layer before inserting the checkpoint, while the suffix states wait at layer b. After insertion, only the document and suffix rows run through the upper layers, attending to the cached prefix KV. The resulting cache contains prefix and suffix KV below b, and prefix, document, and suffix KV from b upward. Position indices retain the reserved document gap, while attention masks operate over the rows actually stored at each layer (Appendix B).

3.5 Overlapped read-time execution

After selection, the receiver obtains the checkpoint length m from the directory and reserves the corresponding document positions. This fixes the suffix positions before the checkpoint arrives. Receiver preparation consists of the lower pass over $\mathcal{L}_m(r,u)$ and the upper pass over the prefix rows

described in Section 3.4. It does not depend on checkpoint contents and can run during checkpoint transfer, as shown by the parallel region in Figure 3. Once transfer and preparation are both complete, the receiver dequantizes the checkpoint and inserts it at layer b. The document and suffix rows then pass through layers b to $L-1$, attending to the cached prefix keys and values. The native-text baseline also overlaps transfer with prefix preparation. Once both finish, it processes the document and suffix through all layers.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. The selection and read-path studies use the same 800 questions from the official training split of 2WikiMultihopQA, excluding question and supporting-document identities recorded in the available history of earlier diagnostic, retrieval, reconstruction, and systems experiments (Appendix C). The end-to-end evaluation uses the same 400 2WikiMultihopQA questions across all three backbones. These questions were initially held out for Qwen3-4B and later reused for the two 8B evaluations. Each question has a fixed directory of 62 documents, including the receiver's own document and the target document. The selector ranks the remaining 61 candidates, and each receiver retrieves one of them. Documents are 64-token windows under the Qwen3 tokenizer, centered using annotated supporting sentences. Each directory includes 60 distractors sampled using a mixture of lexical relevance and random sampling.

Models and configuration. We evaluate selection, read-time reconstruction, and end-to-end performance on Qwen3-4B, Qwen3-8B (Yang et al., 2025a), and Llama-3.1-8B-Instruct (Grattafiori et al., 2024). Within each experiment, the sender and receiver share the same frozen backbone. The reconstruction boundary is $b = 6$ of 36 layers for Qwen and $b = 5$ of 32 layers for Llama. The Qwen boundary was selected during development on Qwen3-4B and transferred to Llama at the nearest relative depth. It was fixed before constructing the stored representations and was held fixed for evaluation (Appendix D).

Evaluation. Answers are generated by greedy decoding with at most 24 tokens and scored by token F1 and exact match (EM). Confidence intervals use paired bootstrap resampling over groups of questions linked by shared supporting-document titles. For end-to-end measurements, the sender and receiver run on separate RTX 4090 GPUs and communicate over TCP through a controlled network with 1 Gbps bandwidth and 5 ms round-trip latency. TTFT is measured at the receiver from the start of selection to the first generated token, and each latency observation is the median of 21 timings from three process restarts (Appendix H). The samples, comparisons, and analysis criteria were specified before evaluation (Appendix C).

4.2 EFFECT OF RECEIVER-CONDITIONED SELECTION

We compare receiver-aware and query-only selection using BGE-M3 over the same 61 candidates, with a dense index built from candidate document text rather than the 96-byte addresses. The two settings differ only in the retrieval query: receiver-aware selection uses $q \oplus d_r$, while query-only selection uses q alone. Both are evaluated with text and checkpoint delivery on the same 800 questions across three backbones. This comparison tests the value of receiver context relative to a query-only retriever; it does not compare against sender-side selection methods. We include no communication, random document selection, and a deranged checkpoint from a different question as controls. Random selection is averaged over three seeds. We also compare INT8 and BF16 checkpoints to assess the effect of quantization. Selection statistics are reported in Appendix E.

Receiver-aware selection retrieves the target document on 720 of 800 questions (90.00%), compared with 518 (64.75%) for query-only selection. Table 1 shows that receiver-aware selection improves F1 by 15.52–17.19 percentage points across the three backbones and two delivery modes. The gains are similar in magnitude for text and checkpoint delivery. Receiver conditioning changes the selected document on 286 of the 800 questions. Within these 286 questions, per-question F1 improves on 148–163 and decreases on 12–18, depending on the backbone and delivery mode. Deranged checkpoints score below no communication on all three backbones. Random selection

Table 1: Selection results on three backbones using the same 800 questions. F1 is reported in percent. F1 gain denotes receiver-aware minus query-only performance in percentage points. Random selection is averaged over three seeds. Ckpt. denotes checkpoint delivery.

	Qwen3-4B		Qwen3-8B		Llama-3.1-8B-Instruct	
Selection	Text	Ckpt.	Text	Ckpt.	Text	Ckpt.
Receiver-aware	**66.64**	**67.26**	**71.50**	**71.71**	**71.24**	**69.47**
Query-only	50.37	51.52	54.31	54.60	55.72	53.00
Random selection	38.39	40.55	42.53	42.71	41.63	40.76
No communication	40.19	—	46.14	—	45.73	—
Deranged checkpoint	—	39.36	—	42.66	—	39.63
F1 gain	+16.27	+15.74	+17.19	+17.12	+15.52	+16.47

Table 2: Read-path comparison on 800 questions per backbone, with the selected document fixed. F1 is in percent. Gains are reconstruction minus static reuse in percentage points. Payloads are mean serialized bytes per question, including headers. KVCOMM (Ye et al., 2025) and CacheBlend (Yao et al., 2025), which recomputes 15% of document tokens, reuse the full static KV payload. †Exploratory comparisons added after the preregistered analyses (Appendix J).

	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
Native text	66.64	71.50	**71.24**
Full static KV reuse	53.72	65.18	64.35
KVCOMM†	52.51	66.24	65.24
CacheBlend (15%)†	56.26	66.74	64.84
Reconstruction (ours)	**67.26**	**71.71**	69.47
F1 gain	+13.54	+6.53	+5.13
Checkpoint payload (bytes)	163,987	262,286	244,531
Static KV payload (bytes)	9,436,664	9,436,664	7,820,280

performs substantially worse than receiver-aware selection. Additional comparisons of text and checkpoint delivery and the effect of checkpoint quantization are reported in Appendix F.

4.3 Effect of read-time reconstruction

We compare native text, full static KV reuse, and read-time reconstruction on the same 800 questions and three backbones as the selection study. All three paths use the same receiver-selected document.

Table 2 shows that static reuse lowers F1 by 6.32–12.92 percentage points relative to native text. Read-time reconstruction improves over static reuse by 13.54 points on Qwen3-4B, 6.53 on Qwen3-8B, and 5.13 on Llama-3.1-8B-Instruct. Its mean F1 is slightly above native text on both Qwen models, but remains about 1.8 points lower on Llama-3.1-8B-Instruct. The checkpoint payload is approximately $1/57$ to $1/32$ the size of full static KV, combining higher answer quality with substantially less data transfer than static reuse. Two exploratory baselines that adapt the full static KV payload to the receiver's prefix remain below reconstruction on every backbone. Anchor-based offset correction (KVCOMM) changes F1 by -1.21 to $+1.06$ points relative to static reuse, and recomputing 15% or 30% of document tokens (CacheBlend) adds 0.49–2.54 or 1.88–7.20 points (Appendix J). Reconstruction performs more read-time document computation than CacheBlend, 83–84% of document layer–token computations against 19–34%, while transferring 32–57 times fewer bytes.

Figure 4 shows the results for individual questions. Most questions have the same F1 under all three read paths. Among questions where static reuse lowers F1, reconstruction often restores the native-text score on the Qwen models, while some Llama-3.1-8B-Instruct questions remain at the static-reuse score. Additional analysis shows that 91–94% of the aggregate F1 loss from static reuse occurs

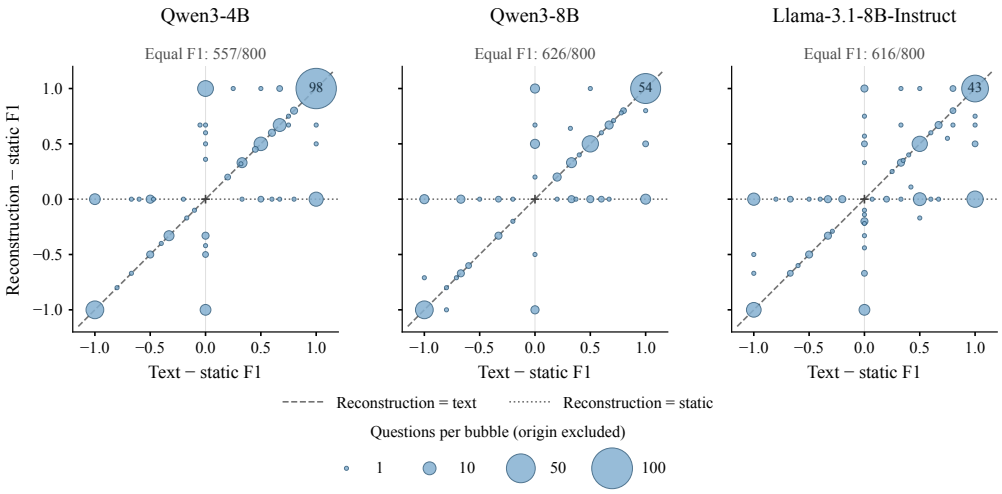


Figure 4: Per-question read-path comparisons on 800 questions per backbone. Axes use F1 differences on the unit scale: 0.01 equals one percentage point. Bubble area is proportional to the number of questions sharing differences rounded to 0.01. The cross marks the origin; counts above each panel report questions with equal F1 across all three paths. Dashed and dotted lines indicate reconstruction matching text and static-reuse F1, respectively.

Table 3: End-to-end results on the same 400 questions. TTFT reduction is the mean of per-question relative reductions. Completion latency change uses the ratio of means. F1 and EM differences are in percentage points. Differences and ratios compare checkpoint delivery with native text unless otherwise specified. Confidence intervals are reported in Appendix H.

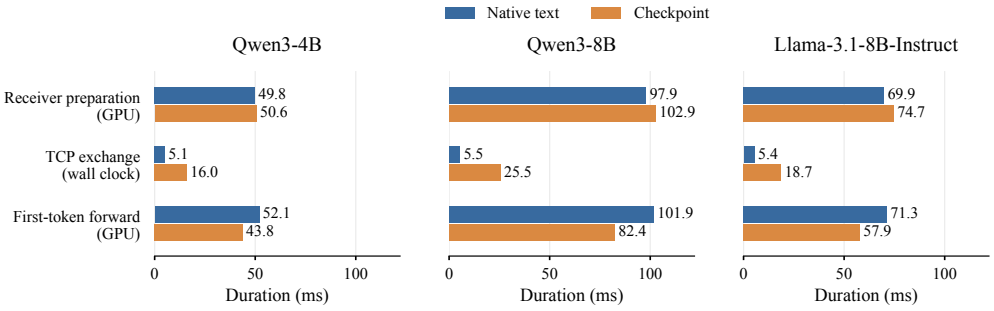
Metric	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
Native-text TTFT (ms)	104.3	207.2	148.9
Checkpoint TTFT (ms)	97.1	195.2	142.6
TTFT difference (ms)	−7.21	−12.05	−6.30
TTFT reduction	6.91%	5.81%	4.23%
Completion latency change	−0.13%	+0.44%	−1.72%
Checkpoint − text F1	+1.88	+1.15	−0.82
Checkpoint − text EM	+0.75	+1.75	−0.75
Checkpoint − no communication F1	+22.56	+21.18	+19.17
Checkpoint − deranged F1	+22.34	+25.36	+25.81

on questions where its first generated token differs from that of native text. Per-question counts and statistical details are reported in Appendix F. A development-stage study on 500 MuSiQue questions also found that reconstruction improves over static reuse (Appendix G).

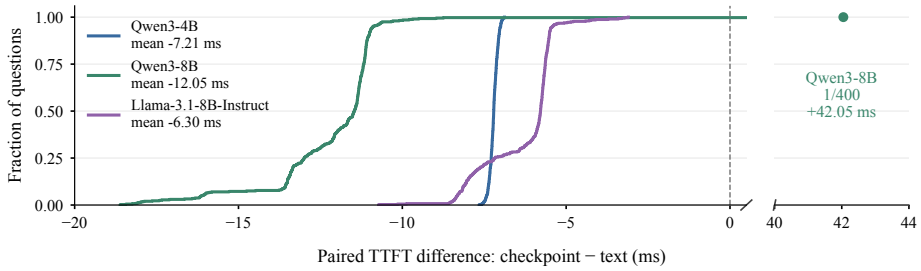
4.4 End-to-end performance

We compare checkpoint and native-text delivery using the lexical selector of Section 3.3 and a store built before the session. Both methods retrieve the same selected document. Table 3 reports results on the same 400 questions across all three backbones.

Checkpoint delivery reduces TTFT on all three backbones, by 6.91% on Qwen3-4B, 5.81% on Qwen3-8B, and 4.23% on Llama-3.1-8B-Instruct. Mean completion latency changes by less than 2% in magnitude on each backbone. The quality comparison differs across backbones. Checkpoint delivery achieves slightly higher F1 and EM scores than native text on both Qwen models. On Llama-3.1-8B-Instruct, F1 and EM point estimates are lower by 0.82 and 0.75 points. Both intervals



(a) Recorded stage durations.



(b) Per-question TTFT differences.

Figure 5: End-to-end timing on 400 questions per backbone, using per-question medians over 21 measurements. (a) Mean stage durations, measured on the GPU for preparation and forward computation and by wall clock for TCP exchange. Transfer overlaps preparation, so durations are not additive. (b) Cumulative distributions of checkpoint-minus-text TTFT. Negative values indicate faster checkpoint delivery. The isolated point marks the single positive Qwen3-8B difference.

include zero, but their lower bounds do not rule out losses beyond the specified tolerances of one F1 point and two EM points. Its 4.23% TTFT reduction also falls below the specified 5% target. Across all three backbones, checkpoint delivery improves F1 over no communication by 19.17–22.56 points and over a deranged checkpoint by 22.34–25.81 points.

Figure 5 compares stage durations and per-question TTFT differences across the three backbones. Checkpoint delivery shortens the final first-token forward pass by 8.3–19.5 ms. Receiver preparation takes slightly longer, while TCP exchange runs concurrently with preparation. Checkpoint delivery is faster on all 400 questions for Qwen3-4B and Llama-3.1-8B-Instruct, and on 399 of 400 for Qwen3-8B. Additional stage timings and statistical details are reported in Appendix H.

5 Conclusion

We presented receiver-conditioned selection and reconstruction for latent communication, with documents written once as INT8 residual checkpoints and frozen upper layers rerun under each receiver's private context. In a selection study using BGE-M3 across three backbones, receiver conditioning improves F1 by 15.5–17.2 points over query-only selection for text and checkpoint delivery. With selection fixed, read-time reconstruction improves F1 by 5.1–13.5 points over full static KV reuse with payloads 32× to 57× smaller, and it also exceeds KVCOMM and CacheBlend applied to the full static payload. Under controlled TCP networking, checkpoint delivery reduces TTFT by 4.23–6.91% while checkpoint transactions carry 413–661 times the application-layer bytes of native text. End-to-end answer quality remains comparable to native text. Reconstruction still reruns 83–84% of each document's layer computation at read time. Retrieval of several documents per question, longer documents, other network conditions, and cross-model communication remain to be evaluated.

AI USE STATEMENT

We used large language models to improve the clarity and organization of the manuscript, identify and summarize related work, and assist with research discussions, code development, and analysis of experimental results. The authors reviewed and revised the assisted content and take responsibility for the final manuscript, implementation, and reported findings.

ETHICS STATEMENT

This work uses public question-answering datasets and open-weight language models, involves no human subjects, collects no new personal data, and studies communication between cooperating agents that share one frozen backbone. Transferring internal states between agents inherits the privacy considerations of sharing the underlying documents. The deranged checkpoint is used only as an evaluation control for example-specific transfer.

REPRODUCIBILITY STATEMENT

Every confirmatory study was preregistered. The code executed by the reported runs, including data preparation, the method and baselines, experiment runners, scoring, and statistics, is available at `anonymous.4open.science/r/rcsr-latent-comm-B184`. The preregistration documents, the dated protocol amendment, the sealed execution plans, prepared inputs, and raw outputs will be released upon publication. Section 3 and Appendix B specify the checkpoint write, selector, and read path. Appendix I describes the read-path implementation checks, and the backbones are used with their public weights and fixed revisions.

REFERENCES

Carlos Baquero, Luís Brito, and João Resende. MaSRead: Content-addressed reading of replicated latent stores. *arXiv preprint arXiv:2608.11218*, 2026.

Chuangtao Chen, Grace Li Zhang, Xunzhao Yin, Cheng Zhuo, Bing Li, and Ulf Schlichtmann. KV Packet: Recomputation-free context-independent KV caching for LLMs. *arXiv preprint arXiv:2604.13226*, 2026.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-Embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2318–2335, 2024a. doi: 10.18653/v1/2024.findings-acl.137.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *International Conference on Learning Representations (ICLR)*, 2024b. arXiv:2308.10848.

Jiaming Cheng, Subhransu Das, and Rajiv Ramnath. When does latent communication pay? A causal audit of relayed KV caches in multi-agent LLMs. *arXiv preprint arXiv:2608.04893*, 2026.

Tianyu Fu, Zihan Min, Hanling Zhang, Jichao Yan, Guohao Dai, Wanli Ouyang, and Yu Wang. Cache-to-Cache: Direct semantic communication between large language models. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.03215.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pp. 6609–6625, 2020. doi: 10.18653/v1/2020.coling-main.580.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2308.00352.

Junhao Hu, Wenrui Huang, Weidong Wang, Haoyi Wang, Tiancheng Hu, Qin Zhang, Hao Feng, Xusheng Chen, Yizhou Shan, and Tao Xie. EPIC: Efficient position-independent caching for serving large language models. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2410.15332.

Omar Khattab, Christopher Potts, and Matei Zaharia. Baleen: Robust multi-hop reasoning at scale via condensed retrieval. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2101.00436.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative agents for "mind" exploration of large language model society. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2303.17760.

Jiyao Liu, Qi Zhang, Yaoyi Jia, Ziwen Kan, and Song Wang. Dual-cache latent space communication between heterogeneous language models. *arXiv preprint arXiv:2608.20617*, 2026a.

Yuhan Liu, Hanchen Li, Yihua Cheng, Siddhant Ray, Yuyang Huang, Qizheng Zhang, Kuntai Du, Jiayi Yao, Shan Lu, Ganesh Ananthanarayanan, Michael Maire, Henry Hoffmann, Ari Holtzman, and Junchen Jiang. CacheGen: KV cache compression and streaming for fast large language model serving. In *Proceedings of the ACM SIGCOMM 2024 Conference*, 2024a. doi: 10.1145/3651890.3672274.

Yuhan Liu, Yuyang Huang, Jiayi Yao, Shaoting Feng, Zhuohan Gu, Kuntai Du, Hanchen Li, Yihua Cheng, Junchen Jiang, Shan Lu, Madan Musuvathi, and Esha Choukse. DroidSpeak: KV cache sharing across fine-tuned model variants. In *23rd USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2026b. arXiv:2411.02820.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *International Conference on Machine Learning (ICML)*, 2024b. arXiv:2402.02750.

Dongyang Ma, Yan Wang, and Tian Lan. Block-Attention for efficient prefilling. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2409.15355.

Kaleem Ullah Qasim, Jiashu Zhang, Muhammad Kafeel Shaheen, Razan Alharith, and Heying Zhang. The residual stream is all you need: On the redundancy of the KV cache in transformer inference. *arXiv preprint arXiv:2603.19664*, 2026.

Xiangyu Shi, Marco Chiesa, Gerald Q. Maguire, Jr., and Dejan Kostić. KVComm: Enabling efficient LLM communication through selective KV sharing. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.03346.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, 2023. doi: 10.18653/v1/2023.acl-long.557.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversations. In *Conference on Language Modeling (COLM)*, 2024. arXiv:2308.08155.

Hui Xie, Peng Xiao, Yutong Deng, Shuoran Dou, Jian Yang, and Jinyang Guo. SemPIC: Learning semantic position-independent KV caches. *arXiv preprint arXiv:2607.28069*, 2026.

Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2009.12756.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Xinyu Yang, Tianqi Chen, and Beidi Chen. APE: Faster and longer context-augmented generation via adaptive parallel encoding. In *International Conference on Learning Representations (ICLR)*, 2025b. arXiv:2502.05431.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2369–2380, 2018. doi: 10.18653/v1/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)*, 2025. arXiv:2405.16444.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2510.12872.

Jiaru Zou, Xiyuan Yang, Ruizhong Qiu, Gaotang Li, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. Latent collaboration in multi-agent systems. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2511.20639.

A RECEIVER-CONTEXT DIAGNOSTIC DETAILS

Figure 1 uses Qwen3-4B to compare block preferences under different receiver contexts. For each question, the sender receives the question and two 64-token supporting-document windows. The two receivers receive the same question but each holds only one of these windows. At each of the zero-indexed layers 8, 17, 26, and 35, each window is divided into eight nonoverlapping 8-token blocks. A candidate comprises the keys and values of one block at one layer, giving 64 candidates per question.

Block preferences. In Figure 1(a), the sender-side attention score is the sum of attention weights to a candidate's tokens, over all heads at its layer and all positions predicting gold answer tokens under teacher forcing. The sender selects the candidate with the largest score. For each receiver, all 64 singleton candidates are evaluated by their change in mean gold answer token log probability relative to no communication, measured in nats per answer token. The receiver-optimal block maximizes this gain. The plotted means average the two receivers within each question, then average across questions.

Subset overlap. In Figure 1(b), each receiver independently uses beam search with width 32 to maximize the same gain over subsets of at most $k \in \{1, 2, 4, 8\}$ candidates. The search retains the best subset found at any depth up to k, including the empty subset. These are approximate oracle selections, rather than globally optimal subsets. For each question, overlap is the Jaccard index of the two selected sets of layer–block candidates, with $J(\emptyset, \emptyset) = 1$. Sample sizes are listed in Table 4. Gold answers are used only for these diagnostics, not for the system's document selection.

Uncertainty. Error bars show 95% percentile bootstrap intervals from 10,000 resamples of questions within each dataset (random seed 20260924). Each resampled question retains both receivers and all compared conditions; in panel (a), the two receiver gains are averaged before resampling. These intervals describe uncertainty in the reported means on the diagnostic samples.

Table 4: Number of questions in the receiver-context diagnostics.

Dataset	Block preferences (a)	Subset overlap (b)
HotpotQA	3,658	349
MuSiQue	814	814
2WikiMultihopQA	4,626	350

B IMPLEMENTATION DETAILS

Write prefix. The write prefix w uses the backbone's native chat template with one user message beginning with the following text:

```
Answer the question with a short span only.

Context:
```

The literal separator before `Context:` is two newline characters, followed by one newline before the document. No question is included. The document's token IDs are inserted as a contiguous span after this prefix, without tokenizing them again. Chat formatting is enabled and thinking is disabled where supported. Thus, the instruction is shared across backbones, while template tokens and the length of w depend on the tokenizer. Only the document rows entering layer b are stored.

Quantization. For a document row $x_i \in \mathbb{R}^h$, the stored scale and integer values are

$$\sigma_i = \mathrm{FP16}\left(\max\left\{\frac{\max_j |x_{ij}|}{127}, 2^{-14}\right\}\right),$$

$$c_{ij} = \mathrm{INT8}\left(\mathrm{clip}_{[-127,127]}\left(\mathrm{round}\left(\frac{x_{ij}}{\sigma_i}\right)\right)\right).$$

The maximum and division are computed in FP32. The scale is rounded to FP16 before computing the integer values, so encoding uses the same scale as decoding. Rounding is to the nearest integer, with ties to even; the zero point is zero. Reconstruction multiplies c_{ij} by σ_i in FP32 and casts the result to the backbone's activation dtype. Each row therefore contains h INT8 values and one FP16 scale, or $h + 2$ bytes before the checkpoint header.

Address construction. The lexical directory uses the text of each stored document window; no separate title terms are added. Terms are Unicode letters or numbers, with an optional internal straight or curly apostrophe. Underscores and other punctuation split terms. Terms are lowercased, curly apostrophes become straight apostrophes, and terms of length one and the fixed stopwords listed below are removed. No stemming is applied.

Document frequencies are computed over all document windows in the prepared store, counting each term once per window. For N windows and document frequency $\mathrm{df}(t)$, the shared weight is

$$I(t) = \log\left(1 + \frac{N - \mathrm{df}(t) + 0.5}{\mathrm{df}(t) + 0.5}\right).$$

Unseen terms use $\mathrm{df}(t) = 0$. Unique terms are sorted by decreasing $I(t)$, with lexical order breaking ties. Each term is hashed from UTF-8 using BLAKE2s with a four-byte digest and personalization string `raskv-a1`; the digest is interpreted as an unsigned integer in little endian order. Construction examines at most the first 48 terms and retains up to 24 distinct hashes. Colliding hashes occupy one slot. The address budget is 24 unsigned 32-bit slots, or 96 bytes, including unused capacity.

The receiver also loads the shared term-to-frequency table and N. These statistics are additional directory metadata, not part of the 96 bytes allocated to each address or a per-request payload. The implementation keeps the table keyed by normalized terms; it does not recover a term's IDF from a candidate hash.

Lexical scoring. The receiver applies the same normalization and weighting to its question and document. Let $Q(h)$ be the weight at hash h from all unique question terms, and $R(h)$ the weight from the 24 highest weighted unique terms in the receiver's document. If local terms collide, their maximum weight is retained; missing hashes have weight zero. For the retained hashes $A(a)$ of candidate address a,

$$\rho(a, q) = \sum_{h \in A(a)} Q(h), \qquad \beta(a, d_r) = \sum_{h \in A(a)} R(h),$$

$$M(a) = \sum_{h \in A(a)} \max\{Q(h), R(h), 1\}, \qquad \nu(a) = 1 - \frac{\beta(a, d_r)}{M(a)}.$$

The implementation clips ν to $[0, 1]$ and sets it to 1 if $M(a) = 0$. This normalization uses local weights and a default weight of 1, rather than the unknown IDF of every candidate term. Hash matches are not verified against candidate text, so collisions can produce false matches. The receiver weight $1/2$ and exponent 4 in Equation 2 are fixed settings used throughout the reported end-to-end experiments. The receiver's own entry is excluded; ties are broken by ascending entry ID.

The fixed stopwords are:

> a about above after again against all am an and any are as at be because been before being below between both but by can could did do does doing down during each few for from further had has have having he her here hers herself him himself his how i if in into is it its itself just me more most my myself no nor not now of off on once only or other our ours ourselves out over own same she should so some such than that the their theirs them themselves then there these they this those through to too under until up very was we were what when where which while who whom why will with would you your yours yourself yourselves.

Positions and attention masks. Let p and s be the prefix and suffix lengths. During preparation, the lower layers process $p + s$ rows with logical position IDs $0, \ldots, p-1, p+m, \ldots, p+m+s-1$ and physical cache indices $0, \ldots, p + s - 1$. Rotary position embeddings (RoPE) use the logical positions; causal masks use the rows actually present in each layer's cache. No zero rows or masked

placeholders are allocated for the document below b. The upper layers first process the prefix, then the inserted document and suffix at consecutive logical positions starting at p. Generated tokens retain their positions in the full sequence, while their lower caches remain shorter by m rows. For cached attention, masks align each query to the corresponding final cache positions; the scaled dot-product attention (SDPA) implementation uses a lower right causal bias, with an equivalent additive causal mask for the eager path.

Overlapped execution. Both delivery modes start the TCP exchange before receiver preparation. Checkpoint delivery prepares the prefix and suffix below b and the prefix above it. Native text prepares the prefix through all layers. Once preparation and transfer finish, checkpoint delivery reconstructs the document and processes the suffix above b; native text processes the received document and suffix through all layers. Transfer is included in TTFT, even when it finishes before preparation and therefore does not extend the critical path.

C Experimental data and analysis

Separation from earlier experiments. The 800 questions exclude sample IDs, normalized questions, and supporting document or window identities recorded in the available history of earlier diagnostic, retrieval, reconstruction, and systems runs, including earlier confirmation attempts. The inventory includes block and oracle scoring, objective probes, retrieval budget studies, hybrid and open-store development, and reconstruction development. Matching uses normalized titles, full documents, and windows, including legacy aliases. The questions are also disjoint from the 64 engineering cases in the source construction. This establishes separation from the inventoried experiments, not from model pretraining data or unavailable historical records.

Windows and directories. The source uses compositional and inference questions with two annotated supporting documents. Each window is centered on the midpoint of the span from the first to the last annotated supporting sentence, then shifted as needed to remain within the document. Documents shorter than 64 Qwen tokens are excluded. Supporting-fact annotations determine the window; the answer string is not used to select or filter it. The window need not contain the entire supporting span or the answer string. The receiver owns the first supporting document in context order, and the second is the designated target.

Each directory contains these two entries and 60 distractors. Four distractors are sampled from the 100 highest scoring documents under lexical question matching; 56 are sampled from the remaining documents. The latter may also include other entries from the top 100. Sampling and final shuffling use a seed derived from the sample. The selection study samples from its partition's 1,600 supporting windows. The end-to-end store contains 816 windows from 400 evaluation questions and eight smoke-test questions, so smoke-test windows can appear as distractors. Directories remain fixed across compared methods.

Analysis plan. Samples, comparisons, estimators, and decision criteria were specified before evaluation generation. Additional retrieval diagnostics, storage estimates, and summaries by restart are descriptive analyses. Bootstrap resampling groups questions connected by either normalized supporting-document title and preserves the pairing of methods. Shared distractors are part of the fixed candidate library, not links between question groups. The selection and reconstruction samples have 800 distinct groups; the end-to-end sample has 400.

The original selection family contains four effects: two 8B backbones and two delivery modes. Its intervals use confidence level $1 - 0.05/4 = 98.75\%$; the later Qwen3-4B replication retains this level. The reconstruction family contains six effects: static-reuse loss and recovery beyond half that loss on each of three backbones. Its confidence level is $1 - 0.05/6 = 99.1667\%$. Both analyses use 100,000 paired bootstrap resamples. Descriptive reconstruction intervals and end-to-end intervals use 95% confidence and 20,000 resamples. The detailed estimates and decision criteria appear in the corresponding analysis appendices.

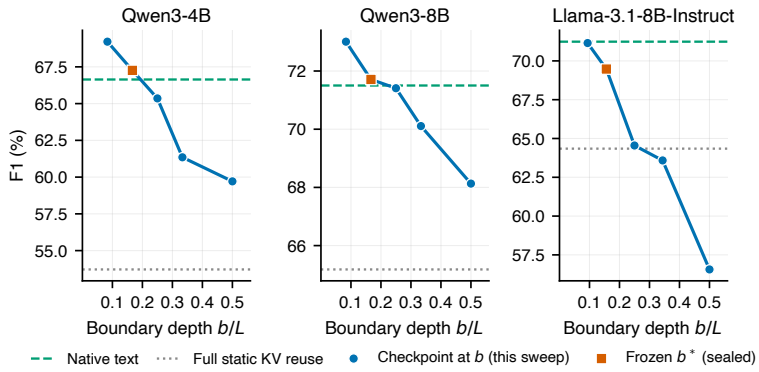


Figure 6: Boundary sensitivity on 800 questions per backbone. Circles mark additional boundaries and squares the fixed boundary b^*. Dashed and dotted lines show native-text and static-reuse F1.

D BOUNDARY SELECTION AND SENSITIVITY

Development-stage selection. The boundary $b = 6$ of 36 layers was selected on Qwen3-4B before constructing the stored representations used in the main experiments. The development sweep used 30 questions, ten each from 2WikiMultihopQA, HotpotQA, and MuSiQue. Its earlier prototype stored a BF16 boundary residual and reused the sender's lower-layer KV. Thus, it differs from the final method, which stores only the residual checkpoint. The endpoints reproduced native text at $b = 0$ and full static KV reuse at $b = 36$.

Table 5 reports dataset-balanced mean F1. The early boundary $b = 6$ achieved 42.8 F1, close to 42.6 at $b = 0$; deeper boundaries scored 35.5–36.2. The selected boundary was then fixed for both Qwen models. Llama uses the nearest relative depth, $b = 5$ of 32, without a separate development sweep.

Table 5: Development-only boundary sweep of an earlier BF16 prototype on Qwen3-4B. Mean F1 is reported in percent over 30 questions, with 10 from each of three datasets.

Boundary b	0	6	12	18	24	30	36
Layers recomputed	36	30	24	18	12	6	0
Mean F1	42.6	42.8	36.2	36.0	35.5	35.5	35.9

Sensitivity of the final representation. After completing the 800-question read-path evaluation, an exploratory sweep evaluated the final INT8 representation on the same 800 questions and selected documents as the read-path study. It added four boundaries per backbone: $b \in \{3, 9, 12, 18\}$ for Qwen and $b \in \{3, 8, 11, 16\}$ for Llama, yielding 9,600 additional generations. Native-text, static-reuse, and fixed-boundary (b^*) outputs were reused. A 16-question check per backbone reproduced the existing b^* outputs before the sweep. The sweep was not used to reselect b^*. Intervals use 20,000 paired cluster-bootstrap resamples with seed 20260916 and 95% confidence, without multiple-comparison correction.

Figure 6 and Table 6 show that F1 decreases as the boundary moves deeper over the tested range. The shallowest boundary, $b = 3$, achieves 69.21, 73.01, and 71.15 F1 on Qwen3-4B, Qwen3-8B, and Llama-3.1-8B-Instruct, respectively, improving on b^* by 1.95, 1.30, and 1.68 points. At approximately one-quarter depth, both Qwen models remain within 1.3 points of native text, whereas Llama is 6.69 points below it. At half depth, Llama also falls 7.78 points below static reuse.

Shallower boundaries require more computation at read time: $b = 3$ reruns 33 of 36 layers for Qwen and 29 of 32 for Llama, compared with 30 and 27 at the fixed boundaries. Checkpoint size is independent of b for a fixed document and backbone. Differences are calculated before rounding the displayed means. The sweep measures quality only; end-to-end latency was evaluated at b^* and was not remeasured at the additional boundaries.

Table 6: Exploratory boundary sweep on 800 questions per backbone. F1 is in percent; differences are checkpoint minus the indicated reference in percentage points, with 95% paired cluster-bootstrap intervals. Rerun counts recomputed layers. Share is the fraction of the native-text minus static-reuse F1 gap recovered by the checkpoint.

b	Rerun	F1	Δ vs text	Δ vs static	Δ vs b^*	Share
Qwen3-4B						
			$+2.57$	$+15.49$	$+1.95$	
3	33	69.21	$[+1.15, +4.05]$	$[+12.59, +18.49]$	$[+0.53, +3.42]$	1.20
			-1.29	$+11.64$	-1.91	
9	27	65.35	$[-3.26, +0.67]$	$[+8.78, +14.56]$	$[-3.55, -0.28]$	0.90
			-5.29	$+7.63$	-5.91	
12	24	61.35	$[-7.65, -2.99]$	$[+4.90, +10.43]$	$[-8.02, -3.77]$	0.59
			-6.93	$+5.99$	-7.55	
18	18	59.71	$[-9.68, -4.22]$	$[+3.84, +8.16]$	$[-10.23, -4.93]$	0.46
$b^* = 6$	30	67.26	Fixed boundary, text 66.64, static reuse 53.72			
Qwen3-8B						
			$+1.51$	$+7.83$	$+1.30$	
3	33	73.01	$[+0.32, +2.72]$	$[+5.48, +10.27]$	$[+0.13, +2.49]$	1.24
			-0.09	$+6.23$	-0.30	
9	27	71.41	$[-1.44, +1.22]$	$[+3.84, +8.68]$	$[-1.33, +0.69]$	0.99
			-1.39	$+4.93$	-1.60	
12	24	70.11	$[-3.06, +0.24]$	$[+2.57, +7.32]$	$[-3.08, -0.16]$	0.78
			-3.37	$+2.95$	-3.58	
18	18	68.13	$[-5.59, -1.20]$	$[+1.17, +4.79]$	$[-5.62, -1.59]$	0.47
$b^* = 6$	30	71.71	Fixed boundary, text 71.50, static reuse 65.18			
Llama-3.1-8B-Instruct						
			-0.09	$+6.81$	$+1.68$	
3	29	71.15	$[-1.34, +1.15]$	$[+4.46, +9.21]$	$[+0.25, +3.13]$	0.99
			-6.69	$+0.20$	-4.93	
8	24	64.55	$[-8.84, -4.60]$	$[-2.04, +2.46]$	$[-6.62, -3.28]$	0.03
			-7.65	-0.76	-5.89	
11	21	63.59	$[-10.10, -5.31]$	$[-2.57, +1.01]$	$[-7.98, -3.90]$	-0.11
			-14.68	-7.78	-12.91	
16	16	56.56	$[-17.61, -11.80]$	$[-10.33, -5.23]$	$[-15.75, -10.12]$	-1.13
$b^* = 5$	27	69.47	Fixed boundary, text 71.24, static reuse 64.35			

E SELECTION ANALYSIS

Retrieval setup and scope. BGE-M3 ranks the same 61 candidate documents using either $q \oplus d_r$ or q. The selected documents are shared across backbones and delivery methods, so the comparison isolates the retrieval-query input. This study tests receiver conditioning with a standard dense retriever; it does not evaluate the lexical scorer under an equal index budget. BGE-M3 uses embeddings computed from candidate document contents, whereas the end-to-end system uses the compact hashed addresses of Section 3.3.

Target retrieval. The two queries select the target document on 518 of 800 questions (64.75%) for query-only selection and 720 of 800 (90.00%) for receiver-aware selection. These selections are identical across the three backbones and the two delivery modes. Of the 286 questions whose selected document changes, 225 change from a miss to a hit, 23 from a hit to a miss, and 38 remain misses. A miss denotes selection of a document other than the designated target; it does not imply that the selected document contains no useful information.

Paired selection gains. Table 7 reports receiver-aware minus query-only F1 on the same 800 questions. The 98.75% percentile intervals use 100,000 paired cluster-bootstrap resamples with seed 2026091038. The questions belong to distinct groups, and comparisons share the resampling indices. The confidence level follows the selection family defined in Appendix C.

Receiver conditioning changes the selected document on 286 questions. Among these, F1 improves on 148–163 and decreases on 12–18, depending on the backbone and delivery method; the remaining questions are tied. The other 514 questions retain the same document and have unchanged F1 in these comparisons. The gains are similar across text and checkpoint delivery, but this similarity is not a statistical test of equal gains across delivery methods. Appendix F reports the separate text-versus-checkpoint and quantization comparisons.

Table 7: Receiver-aware minus query-only F1 on the same 800 questions, in percentage points, with 98.75% paired cluster-bootstrap confidence intervals.

Backbone	Delivery	F1 gain	Confidence interval
Qwen3-4B	Text	$+16.27$	$[12.88, 19.74]$
	Checkpoint	$+15.74$	$[12.44, 19.16]$
Qwen3-8B	Text	$+17.19$	$[13.73, 20.72]$
	Checkpoint	$+17.12$	$[13.72, 20.62]$
Llama-3.1-8B-Instruct	Text	$+15.52$	$[12.19, 18.98]$
	Checkpoint	$+16.47$	$[13.17, 19.90]$

F READ-PATH ANALYSIS

This appendix supplements Section 4.3. All comparisons use the same 800 questions and receiver-selected documents on each backbone.

Per-question outcomes. Table 8 reports wins, losses, and ties by token F1. Static reuse loses to both native text and reconstruction on substantially more questions than it wins. Reconstruction and native text have identical F1 on 88.5–93.3% of questions; identical F1 need not imply identical answers. Static reuse differs from native text at the first generated token on 254, 142, and 191 questions for Qwen3-4B, Qwen3-8B, and Llama-3.1-8B-Instruct, respectively. These questions account for approximately 12.1 of 12.9, 5.8 of 6.3, and 6.5 of 6.9 percentage points of the aggregate F1 loss, or 91–94%. This concentration describes where the loss occurs, rather than establishing that first-token divergence causes the loss.

Table 8: Per-question outcomes on 800 questions per backbone. Each entry gives wins/losses/ties of the first method against the second by token F1. The last row counts questions whose first generated token under static reuse differs from that under native text.

	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
Text vs. static reuse	162/49/589	112/45/643	114/45/641
Reconstruction vs. static reuse	164/46/590	107/37/656	89/43/668
Reconstruction vs. text	39/34/727	28/26/746	35/57/708
First token differs from text	254	142	191

Recovery of the static-reuse loss. Let Δ_{loss} be native-text F1 minus static-reuse F1, and Δ_{rec} be reconstruction F1 minus static-reuse F1. The ratio $R = \Delta_{\text{rec}}/\Delta_{\text{loss}}$ measures the fraction of the mean F1 gap recovered; $R > 1$ means that reconstruction exceeds native text. The statistic $\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$ measures recovery beyond half of the loss.

Table 9 reports paired cluster-bootstrap intervals. The six planned comparisons of loss and recovery beyond half the loss use 100,000 resamples with seed 2026091038 and 99.1667% intervals. Descriptive intervals for Δ_{rec} and R use 20,000 resamples with seed 20260916 and a 95% confidence level. Resampling preserves the pairing of methods within question groups; the 800 questions in this study form 800 distinct groups.

Reconstruction improves over static reuse on all three backbones. Recovery beyond half of the loss is supported on both Qwen models by the 99.1667% intervals. On Llama-3.1-8B-Instruct, the estimated recovery is 74.4%, but the corresponding interval for $\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$ includes zero. The

descriptive 95% interval for R, $[0.52, 0.98]$, uses a lower confidence level and does not establish the same stronger conclusion.

Table 9: Read-path effects on 800 questions per backbone. F1 differences are in percentage points; R is dimensionless. Paired cluster-bootstrap confidence levels are 99.1667% for Δ_{loss} and $\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$, and 95% for Δ_{rec} and R.

Backbone	Statistic	Estimate	Confidence interval
Qwen3-4B	Δ_{loss}	12.92	$[8.98, 16.95]$
	$\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$	7.08	$[4.54, 9.68]$
	Δ_{rec}	13.54	$[10.64, 16.50]$
	R	1.05	$[0.92, 1.20]$
Qwen3-8B	Δ_{loss}	6.32	$[3.01, 9.74]$
	$\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$	3.37	$[1.39, 5.41]$
	Δ_{rec}	6.53	$[4.18, 9.00]$
	R	1.03	$[0.84, 1.30]$
Llama-3.1-8B-Instruct	Δ_{loss}	6.89	$[3.61, 10.24]$
	$\Delta_{\text{rec}} - \frac{1}{2}\Delta_{\text{loss}}$	1.68	$[-0.39, 3.77]$
	Δ_{rec}	5.13	$[2.90, 7.40]$
	R	0.74	$[0.52, 0.98]$

Text delivery and quantization. Under receiver-aware selection, checkpoint-minus-text F1 differences are $+0.62$ $[-1.08, +2.31]$ on Qwen3-4B, $+0.21$ $[-1.08, +1.49]$ on Qwen3-8B, and -1.76 $[-3.44, -0.12]$ on Llama-3.1-8B-Instruct. These exploratory 95% intervals include zero on both Qwen models and favor native text on Llama.

The INT8-minus-BF16 checkpoint differences are $+0.08$ $[-0.96, +1.15]$, -0.33 $[-1.01, +0.32]$, and -0.13 $[-0.65, +0.34]$ F1 points, respectively, with exploratory 95% intervals. The point estimates differ by at most 0.33 points, and all three intervals include zero. These comparisons do not identify INT8 quantization as the main source of the Llama gap.

Implementation and payload accounting. Native-text and checkpoint outputs are reused from the selection study. Before generating static-reuse outputs, a 16-question check on each backbone reproduced the text and checkpoint output tokens exactly. The serialized payloads in Table 2 include a 29-byte checkpoint header and a 33-byte static-KV header. These are stored-object sizes; the end-to-end transaction sizes in Appendix H also include request and response framing.

Storage at matched precision. To separate representation size from precision, Table 10 gives storage estimates for a common length of 64 backbone tokens, excluding headers. A BF16 checkpoint uses $2mh$ bytes for hidden width h. For an illustrative INT8 static cache, we assume one FP16 scale per token, KV head, layer, and key or value tensor. Its size is $2mLH_{\text{KV}}(d_h + 2)$ bytes, where $d_h = 128$. This INT8 cache is a storage estimate, not an evaluated read path; no quality or latency claim is made for it. Llama's actual window lengths vary, so these fixed-length estimates differ from its measured mean payload.

Table 10: Storage estimates at 64 backbone tokens, excluding headers. INT8 formats include FP16 scales. Values are bytes.

Representation	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
BF16 checkpoint	327,680	524,288	524,288
BF16 static KV	9,437,184	9,437,184	8,388,608
INT8 checkpoint	163,968	262,272	262,272
INT8 static KV (estimate)	4,792,320	4,792,320	4,259,840

G DEVELOPMENT-STAGE READ-PATH STUDY

A development-stage experiment compared the three read paths on 500 MuSiQue questions using Qwen3-4B. The receiver holds the first-hop document and selects from a 62-entry directory containing the final-hop document. The selected document was fixed before answer generation and shared by all communicating methods.

Table 11 reports answer quality. Reconstruction achieves 35.65 F1, compared with 23.24 for static reuse and 35.77 for native text. Its paired difference from native text is -0.12 $[-2.11, +1.89]$ F1 points. Share of text gain divides each method's improvement over no communication by the corresponding native-text improvement. This denominator differs from the static-to-text gap used for R in Appendix F.

Table 11: Development-stage read-path comparison on Qwen3-4B using 500 MuSiQue questions. Communicating methods use the same selected document. F1 and EM are in percent. Share measures the F1 gain over no communication as a percentage of the corresponding native-text gain.

Read path	F1	EM	Share of text gain
No communication	10.26	2.2	—
Full static KV reuse	23.24	11.0	50.9%
Reconstruction (ours)	35.65	24.4	99.5%
Native text	35.77	23.6	100%

An earlier BF16 reconstruction variant transmitted lower-layer KV together with a boundary residual. On 200 2WikiMultihopQA questions with two receiver cases per question, it improved F1 over static reuse by 10.8 $[7.4, 14.2]$ points. Both intervals in this appendix are 95% percentile intervals from 20,000 paired bootstrap resamples over source-question groups. These development results informed the later study design and are reported separately from the 800-question evaluation.

H END-TO-END EVALUATION DETAILS

Workload and measurement. The three backbones use the same 400 questions and the lexical selector of Section 3.3. These questions were initially held out for Qwen3-4B and subsequently reused for the two 8B evaluations. The sender and receiver run in separate processes on separate RTX 4090 GPUs, connected through local TCP in a controlled virtual network. The network is configured at 1 Gbps and 5 ms round-trip latency. Measured round-trip latency and goodput are 5.13–5.14 ms and 0.547–0.561 Gbps for Qwen3-4B, and 6.09–6.16 ms and 0.395–0.403 Gbps for the two 8B evaluations. The 8B runs use GPUs shared with unrelated processes; their activity is not controlled by the benchmark. Sender and receiver remain on separate devices. TTFT includes selection, prompt construction, receiver preparation, TCP exchange, payload decoding and launch, and first-token computation. For every question and restart, each method receives three warmup repetitions followed by seven measured repetitions. Method order alternates across repetitions and its starting order is balanced across questions. CUDA synchronization precedes request timing and the first-token timestamp; CUDA events measure GPU stages.

For each question and delivery method, the latency observation is the median of 21 measurements: seven repeats in each of three independent process restarts. Quality uses one deterministic generation per question and method; the restarts are timing repetitions, not independent quality samples. Intervals use 20,000 paired cluster-bootstrap resamples over question groups. The full sample contains 400 groups. These intervals quantify variation across questions, not across independently sampled GPU-load or network conditions.

Variation across restarts. Table 12 reports each restart separately. Paired TTFT improves in every restart for every backbone, but absolute latency varies substantially in the 8B runs. For example, native-text TTFT on Llama ranges from 96.11 to 156.46 ms across restarts. These observations describe the three recorded runs, not a distribution over independently sampled machine loads or network conditions.

Table 12: TTFT by independent process restart. Each entry averages 400 question medians, with seven measurements per question and delivery mode within that restart. Reduction is the mean of paired per-question relative reductions. These descriptive results do not replace the primary estimates based on 21 measurements per question.

Backbone	Restart	Text (ms)	Checkpoint (ms)	Difference (ms)	Reduction (%)
Qwen3-4B	1	103.91	96.65	-7.25	6.98
	2	104.31	97.10	-7.21	6.91
	3	105.05	97.73	-7.33	6.97
Qwen3-8B	1	218.01	204.91	-13.10	6.01
	2	227.95	214.30	-13.65	5.97
	3	203.57	191.68	-11.89	5.84
Llama-3.1-8B-Instruct	1	149.39	143.29	-6.11	4.09
	2	156.46	149.58	-6.88	4.37
	3	96.11	91.59	-4.52	4.86

Table 13: Mean stage durations over 400 per-question medians of 21 measurements. Each cell gives native text / checkpoint. Times are in milliseconds; the final row gives mean application-layer bytes per transaction. GPU stages use CUDA timing; other stages use wall-clock timing.

Stage	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
Selection	1.43 / 1.43	3.02 / 3.02	3.00 / 3.00
Prompt construction	0.80 / 0.79	1.99 / 1.99	1.88 / 1.88
Receiver preparation (GPU)	49.8 / 50.6	97.9 / 102.9	69.9 / 74.7
TCP exchange	5.1 / 16.0	5.5 / 25.5	5.4 / 18.7
Payload decoding / launch	0.02 / 0.26	0.04 / 0.65	0.04 / 0.58
First-token forward (GPU)	52.1 / 43.8	101.9 / 82.4	71.3 / 57.9
TTFT	104.3 / 97.1	207.2 / 195.2	148.9 / 142.6
Application-layer bytes	397 / 164,123	397 / 262,427	380 / 244,744

Stage timing and communication. Table 13 adds the remaining stages and transaction sizes to the timing breakdown in Figure 5. Checkpoint delivery shortens the final first-token forward pass by 8.3, 19.5, and 13.4 ms, while receiver preparation increases by 0.8, 4.9, and 4.8 ms, respectively. TCP exchange runs concurrently with preparation. Stage differences are calculated before rounding. The recorded stage durations mix GPU and wall-clock measurements and overlap in time, so they cannot be added to recover TTFT.

Transaction sizes include request and response framing but exclude session setup, directory distribution, calibration, and shutdown. Checkpoint transactions carry approximately 413, 661, and 645 times as many application-layer bytes as native text, respectively. The TTFT improvement therefore reflects computation reuse and overlap under the measured conditions, despite the larger transferred payloads.

Paired latency and quality estimates. Table 14 reports the intervals underlying Table 3. TTFT reduction is the mean per-question relative reduction. Completion change is $100(\overline{T}_{\mathrm{ckpt}}/\overline{T}_{\mathrm{text}} - 1)$, where T is the per-question median completion latency. The P95 ratio divides the 95th percentile of checkpoint latencies by that of native-text latencies.

The evaluation criteria were fixed before generation: the upper bound of the TTFT difference below -1 ms, the lower bound of relative TTFT reduction above 5%, the upper bound of completion change below 5%, and a P95 completion ratio point estimate at most 1.05. Quality preservation requires lower bounds above -1 F1 point and -2 EM points; both content-control differences require lower bounds above zero. Both Qwen models meet these criteria. Llama reduces TTFT by 4.23%, below the specified 5% target, and its quality intervals do not establish the specified margins. Its quality intervals also include zero, so they do not establish a decrease relative to native text. The Qwen3-8B P95 ratio meets the point-estimate criterion, but its interval extends to 1.080; the criterion does not provide an interval-based guarantee of less than 5% tail-latency increase.

Table 14: Paired end-to-end estimates on all 400 questions. Each cell gives the estimate and its 95% cluster-bootstrap interval. Differences and ratios compare checkpoint delivery with native text unless otherwise indicated. F1 and EM differences are in percentage points.

Metric	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
TTFT difference (ms)	-7.21 $[-7.22, -7.20]$	-12.05 $[-12.31, -11.70]$	-6.30 $[-6.41, -6.19]$
TTFT reduction (%)	6.91 [6.90, 6.92]	5.81 [5.65, 5.91]	4.23 [4.16, 4.30]
Completion change (%)	-0.13 $[-2.16, 2.16]$	0.44 $[-1.02, 1.99]$	-1.72 $[-3.25, -0.30]$
P95 completion ratio	0.997 [0.996, 0.998]	0.995 [0.994, 1.080]	1.009 [0.997, 1.014]
Checkpoint − text F1	1.88 $[-0.38, 4.16]$	1.15 $[-0.33, 2.70]$	-0.82 $[-2.14, 0.44]$
Checkpoint − text EM	0.75 $[-1.50, 3.25]$	1.75 $[-0.50, 4.00]$	-0.75 $[-2.50, 1.00]$
F1 vs. no communication	22.56 [17.66, 27.43]	21.18 [16.29, 26.11]	19.17 [14.42, 23.91]
F1 vs. deranged checkpoint	22.34 [17.41, 27.31]	25.36 [20.47, 30.33]	25.81 [21.16, 30.54]

Completion-latency estimator. The ratio of means was specified as the primary estimator before evaluation. The secondary mean of per-question relative completion changes is $+2.16\%$ $[-0.92, +5.70]$ on Qwen3-4B, $+1.28\%$ $[-0.60, +3.40]$ on Qwen3-8B, and -0.83% $[-2.06, +0.45]$ on Llama-3.1-8B-Instruct. Unlike the ratio of means, it gives equal weight to each question's relative change. For Qwen3-4B, its interval extends above 5%. Completion latency also depends on generated answer length; its change is not a direct measure of per-token decoding throughput.

Target-document subset. The selector retrieves the target document on the same 310 questions for each backbone. Table 15 reports the corresponding paired estimates using the same estimators as the full sample. Both Qwen models retain the latency and quality conclusions above; Llama's quality margins remain unresolved and its TTFT reduction remains below 5%. This subset conditions on successful retrieval and does not replace the full-sample evaluation.

I READ-PATH IMPLEMENTATION CHECKS

Synthetic-sequence checks validate static KV reuse before the 800-question read-path evaluation. This appendix describes the checks, a numerical discrepancy found during validation, and the amendment made before any task-level static-reuse outputs were generated.

Positional correction and execution. The checks use each model's own rotary frequencies, including Llama-3.1-8B-Instruct's RoPE scaling. They verify identity and round trips of the key rephasing operator, compare re-phased layer-0 keys with keys computed directly at the target positions, and test shared-prefix reuse against a one-shot prefill. The shared-prefix test compares first-token logits and all-layer document KV after inserting a 64-token document under its original prefix. Additional checks verify serialized payload sizes, the Qwen full-depth endpoint, and 24 decoding steps against the reference implementation using the same spliced cache.

The fixed numerical tolerances are relative L2 error at most 0.06 and cosine similarity at least 0.9985 for cache tensors; for first-token logits, they are relative L2 error at most 0.15, cosine similarity at least 0.993, and Jensen–Shannon divergence at most 0.05.

Execution-shape discrepancy. The initial validation stopped before task evaluation for two reasons. One Qwen3-4B empty-prefix case had a different first-token argmax despite satisfying all five numerical tolerances. Its cache relative L2 error was 0.0098, logit relative L2 error was 0.0132, and

Table 15: End-to-end estimates on the 310 questions where the selector retrieves the target document. Each cell gives the estimate and its 95% paired cluster-bootstrap interval. Metric definitions and units match Table 14.

Metric	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
TTFT difference (ms)	-7.21 $[-7.22, -7.19]$	-12.19 $[-12.37, -12.01]$	-6.34 $[-6.46, -6.21]$
TTFT reduction (%)	6.91 $[6.90, 6.92]$	5.88 $[5.82, 5.93]$	4.26 $[4.17, 4.34]$
Completion change (%)	-0.13 $[-2.24, 2.38]$	0.36 $[-0.97, 1.83]$	-1.15 $[-2.39, -0.04]$
P95 completion ratio	0.997 $[0.996, 0.997]$	0.995 $[0.994, 1.080]$	1.011 $[1.004, 1.015]$
Checkpoint − text F1	2.05 $[-0.71, 4.91]$	1.38 $[-0.52, 3.38]$	-0.95 $[-2.37, 0.45]$
Checkpoint − text EM	0.97 $[-1.94, 3.87]$	2.26 $[-0.65, 5.16]$	-0.65 $[-2.90, 1.61]$
F1 vs. no communication	28.83 $[22.82, 34.65]$	30.30 $[24.64, 35.88]$	28.09 $[22.69, 33.57]$
F1 vs. deranged checkpoint	28.79 $[22.63, 34.94]$	32.30 $[26.31, 38.14]$	34.04 $[28.57, 39.55]$

Jensen–Shannon divergence was 1.5×10^{-4}. An additional bitwise-equality check on layer-0 keys also failed across different sequence-length execution shapes on all three backbones, although the errors remained within the numerical tolerances.

Subsequent diagnostic runs localized the differences to the layer-0 key projection: embeddings and RMSNorm outputs remained bitwise identical. In the affected empty-prefix case, ordinary native execution in 64-token and 8-token chunks was bitwise identical to static splicing, and both differed from a 72-token one-shot prefill. The one-shot reference's two leading logits were tied at BF16 precision. The same pattern occurred in two additional synthetic cases on the Qwen models. Thus, chunked execution alone was sufficient to produce these argmax changes; static splicing introduced no additional difference in those cases.

Amended validation. An amendment dated September 19, 2026 retained all five numerical tolerances, the task samples, and the statistical comparisons. First-token argmax agreement became a recorded diagnostic. The extra bitwise check was replaced by an FP32 algebraic RoPE check: starting from the same pre-rotation key tensor, it compares direct rotation to the target position with source-position rotation followed by re-phasing. A second check applies the operator to unit phase vectors, so its sensitivity does not depend on the key's energy distribution.

The tolerance 2×10^{-3} was fixed before the new synthetic cases were generated. A first-order FP32 rounding estimate gives a phase discrepancy of approximately 3.7×10^{-4} at the largest tested position; the tolerance allows additional rotation arithmetic error and is not a strict worst-case bound. Validation used the original synthetic case and four additional cases from a new seed, with prefix lengths 0, 128, and 1024 and position shifts 64, 512, and 2048. All required checks passed on every applicable combination for all three backbones before task evaluation.

Negative controls show why the algebraic check matters. An off-by-one position error on Qwen produces relative L2 error of approximately 0.03, below the inherited BF16 threshold of 0.06, but above the FP32 tolerance by more than an order of magnitude. The amendment and both sets of diagnostic results will be released with the preregistration documents upon publication.

J ADDITIONAL READ-PATH BASELINES

Scope. These comparisons were added after the preregistered analyses and are exploratory. They use the 800 questions, receiver-selected documents, prompts, greedy decoding, and scoring of Sec-

tion 4.3. Both methods start from the full static KV of the selected document, byte-identical to the static-reuse payload, and differ only in how they adapt it to the receiver's prefix. All settings were fixed before any generation. Intervals are 95% paired cluster-bootstrap intervals over the question groups of Appendix C.

KVCOMM. We follow the released implementation of Ye et al. (2025). An anchor stores the write-time K/V of a document and its offset to the K/V formed after a receiver prefix. For each token index, weights over anchors are a softmax, at the released temperature of 1, which we did not tune, of the negative mean absolute difference between write-time K/V, separately for keys and values; the weighted offsets are added to the write-time K/V, and layer 0 is unchanged. The anchor pool holds the default 20 anchors, built from development questions disjoint from the evaluation set and frozen before evaluation. In BF16, the pool occupies 378 MB of receiver memory for the Qwen backbones and 315 MB for Llama-3.1-8B-Instruct. Keys are compared and offset in the write-time position frame and then re-phased to the receiver positions, which equals the released arithmetic when positions coincide. Anchors shorter than the document are not used; documents without such an anchor keep the static K/V, as in the released code. The primary result applies the correction to every question. The released entropy gate ($\gamma = 0.3$) would instead prefill some questions densely; we report that variant separately.

CacheBlend. We follow the released implementation of Yao et al. (2025). Layer 0 and the check layer recompute every document row. At the check layer, tokens are ranked by the squared deviation between recomputed and cached values, and the top 15% of document tokens (30% as a sensitivity setting) are recomputed in all later layers, while the other rows keep the cached K/V. The ratio applies to document tokens only, because the receiver prefix is computed in every read path. Attention uses exact position-causal masks.

Results. Table 16 reports answer quality and read-time document computation, and Table 17 reports paired differences. KVCOMM shows no detectable improvement over static reuse on any backbone: F1 changes by -1.21 to $+1.06$ points, and every interval includes zero. Its anchor weights are nearly uniform, with entropy within 1% of the maximum, so the correction approximately adds the mean offset of the anchor pool. In this setting, offsets measured for other documents and prefixes did not transfer to new receivers. KVCOMM targets pipelines in which context changes recur, and its authors note that fully dynamic contexts are not yet covered (Ye et al., 2025); here every receiver prefix is private and distinct. The entropy gate prefills 3.9–22.8% of questions densely and raises F1 by 0.5–1.5 points. CacheBlend changes F1 relative to static reuse by $+0.49$ to $+2.54$ points at a 15% ratio, with an interval excluding zero only on Qwen3-4B, and by $+1.88$ to $+7.20$ points at 30%, with intervals excluding zero on both Qwen backbones. Every baseline remains below reconstruction on every backbone, by 2.93–14.75 points, and all intervals of these differences exclude zero. Reconstruction performs more read-time document computation than CacheBlend, 83–84% of document layer–token computations against 19–34%, while transferring 32–57 times fewer bytes. As a check of the shared pipeline, static reuse rerun on 16 questions per backbone on each GPU reproduced the original tokens exactly, and KVCOMM reproduced static reuse exactly on the questions without a usable anchor.

Table 16: Additional read-path baselines on 800 questions per backbone. F1 and EM are in percent. Recompute is the fraction of document layer–token computations performed at read time, given as a range over backbones.

	Qwen3-4B		Qwen3-8B		Llama-3.1-8B-Instruct		
Read path	F1	EM	F1	EM	F1	EM	Recompute
Native text	66.64	61.38	71.50	64.38	71.24	66.38	100%
Full static KV reuse	53.72	48.88	65.18	57.00	64.35	58.88	0%
KVCOMM (all questions)	52.51	47.25	66.24	58.00	65.24	58.13	0%
KVCOMM (entropy gate)	54.03	48.62	66.76	58.50	65.85	59.00	3.9–22.8%
CacheBlend (15%)	56.26	50.75	66.74	58.00	64.84	58.63	18.8–19.6%
CacheBlend (30%)	60.92	55.12	68.78	60.50	66.22	59.88	33.5–33.7%
Reconstruction (ours)	67.26	62.62	71.71	64.12	69.47	63.88	83.3–84.4%

Table 17: Paired F1 differences on 800 questions per backbone, in percentage points, with 95% paired cluster-bootstrap intervals. Static and Recon. denote full static KV reuse and reconstruction (ours). Differences use unrounded scores and can differ by 0.01 from differences of the rounded entries in Table 16.

Difference	Qwen3-4B	Qwen3-8B	Llama-3.1-8B-Instruct
KVCOMM (all) − Static	-1.21 [-3.01, $+0.55$]	$+1.06$ [-0.49, $+2.64$]	$+0.89$ [-1.13, $+2.93$]
KVCOMM (all) − Recon.	-14.75 [-17.80, -11.75]	-5.47 [-7.82, -3.15]	-4.24 [-6.50, -1.98]
KVCOMM (gate) − Static	$+0.31$ [-1.62, $+2.24$]	$+1.58$ [-0.05, $+3.26$]	$+1.51$ [-0.57, $+3.64$]
KVCOMM (gate) − Recon.	-13.23 [-16.24, -10.28]	-4.95 [-7.24, -2.72]	-3.62 [-5.77, -1.49]
CacheBlend (15%) − Static	$+2.54$ [$+0.67$, $+4.37$]	$+1.56$ [-0.05, $+3.19$]	$+0.49$ [-1.14, $+2.13$]
CacheBlend (15%) − Recon.	-11.00 [-13.86, -8.19]	-4.97 [-7.20, -2.82]	-4.64 [-6.67, -2.64]
CacheBlend (30%) − Static	$+7.20$ [$+4.68$, $+9.75$]	$+3.60$ [$+1.52$, $+5.73$]	$+1.88$ [-0.08, $+3.87$]
CacheBlend (30%) − Recon.	-6.34 [-8.66, -4.03]	-2.93 [-4.75, -1.18]	-3.25 [-5.10, -1.44]