KVTRANS: TOWARDS SEAMLESS CROSS-CONTEXT KV CACHE REUSE FOR EFFICIENT AGENTIC COMMUNICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-based multi-agent systems repeatedly re-encode received messages, even though their key–value (KV) states have already been computed during sender generation. Reusing these states can reduce redundant computation, but differences in agent instructions, interaction histories, and token positions make direct cross-context reuse unreliable. We introduce KVTRANS, a training-free framework for adapting sender-generated KV caches to receiver contexts among agents sharing the same model. KVTRANS combines rotary position alignment, receiver-referenced key and value rescaling, and dynamic modulation of receiver-instruction keys during decoding. Across four models on MMLU debate, KVTRANS improves macro-averaged accuracy from 59.61% under direct reuse to 65.44%, compared with 64.88% for text communication, while reducing receiver-side prefill tokens by 59.14%. Timing measurements on four matched role pairs show a 57.04% reduction in prefill time, including pre-decoding cache-adaptation overhead. In WebSailor experiments on BrowseComp-EN and BrowseComp-ZH, KVTRANS partially recovers the quality degradation caused by direct reuse while processing only 53.9% and 62.3% of the text baseline's receiver-side prefill tokens, respectively. These results demonstrate that lightweight receiver-side adaptation can improve the quality–efficiency trade-off of cross-context KV reuse without retraining model weights.

1 INTRODUCTION

LLM-based multi-agent systems increasingly rely on inter-agent communication to combine complementary roles, critiques, and evidence (Du et al., 2023; Liang et al., 2024). However, text-based communication introduces substantial redundant computation: when one agent generates a message, every receiving agent must re-encode that message under its own context, even though the sender has already computed its KV states during generation. In fully connected debate, the resulting message-prefill computation grows as $O(N^2)$ with the number of agents. Reusing sender-generated KV states provides a natural way to avoid this repeated encoding (Figure 1), reducing the corresponding message-prefill computation to $O(N)$ while preserving the original message generation process.

Recent work has explored KV cache-mediated communication, where agents transmit computed Key–Value states instead of raw text (Zou et al., 2026; Liu et al., 2025). However, transferred KV states are not context-free. In our four-model MMLU study, direct KV reuse reduces macro accuracy from 64.88% to 59.61%, yet the effect varies substantially across models and role pairs: relative accuracy retention falls to 77% for one model while matching or exceeding the text baseline for others. This heterogeneity suggests that reliable cross-context KV reuse depends on sender–receiver context compatibility and therefore requires explicit receiver-side alignment.

We formalize this setting as *cross-context KV reuse*, where the sender and receiver share model weights but differ in role instructions, interaction histories, and message positions. This setting introduces three coupled sources of mismatch. *Position Shift* places the same message at different rotary positions; *Magnitude Misalignment* alters the scales of transferred keys and values relative to receiver-local states; and *History Information Conflict* reflects sender-conditioned information

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.

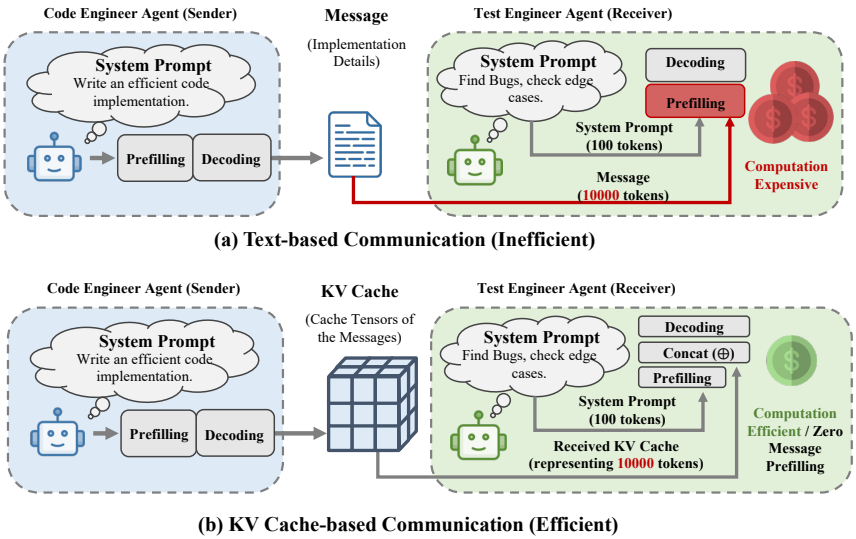


Figure 1: Text versus KV communication. Text re-encodes a received message under each receiver's context; KV transfers sender-produced states. Cache adaptation and continuation still incur work; token counts illustrate the communication-cost reduction achieved by reusing sender states.

embedded in the transferred states that may interfere with the receiver's instructions and context. These discrepancies motivate the three components of KVTRANS.

To address these discrepancies, we propose KVTRANS, a training-free framework for adapting sender-generated KV caches to receiver contexts. RoPE Alignment maps incoming keys from sender-side to receiver-side rotary positions, while receiver-referenced L2-Rescaling calibrates key and value norms to receiver-local statistics. Dynamic Attention Amplification then identifies heads with stronger receiver-instruction preference at each decoding step and selectively rescales instruction keys after foreign-cache insertion. The first two components operate on the transferred cache before insertion, whereas the third modulates receiver-conditioned attention during continuation. KVTRANS keeps all model weights frozen and requires no training.

We evaluate the quality–efficiency trade-off across both debate and collaborative agent settings. In the four-model MMLU study, KVTRANS recovers macro accuracy from 59.61% under Direct KV to 65.44%, comparable to the 64.88% Text baseline, while reducing receiver-side prefill tokens by 59.14%. Measured prefill time decreases from 4.62 s to 1.99 s, a 57.04% reduction including cache-adaptation overhead. These savings grow with both agent count and interaction depth. We further integrate KVTRANS into WebSailor for BrowseComp deep-research tasks. On BrowseComp-EN, KVTRANS raises Pass@1 from 10.00% under Direct KV to 28.33%, versus 34.67% for Text, while using 53.9% of the Text prefill tokens. On BrowseComp-ZH, it achieves 33.33% Pass@1, versus 26.67% for Direct KV and 40.00% for Text, while using 62.3% of the Text prefill tokens.

Our contributions are threefold:

- **Cross-context reuse characterization.** We characterize the heterogeneous behavior of direct cross-context KV reuse across models and role pairs, and identify three coupled mismatches—position shift, magnitude misalignment, and history information conflict—that hinder reliable reuse.
- **Training-free cache adaptation.** We propose KVTRANS, a training-free framework that adapts sender-generated KV states to receiver contexts through RoPE Alignment, L2-Rescaling, and Dynamic Attention Amplification without model retraining.
- **Quality–efficiency evaluation.** We evaluate KVTRANS in both controlled multi-agent debate and long-horizon deep-research settings, including deployment in WebSailor with explicit receiver-side prefill and runtime accounting.

2 PRELIMINARIES

Setting. We consider a multi-agent system in which agents exchange natural-language messages while solving a task, and analyze a single communication step between a sender and a receiver. Both run the same causal language model with parameters θ and share the same tokenizer, but operate under different role instructions and interaction histories. Superscripts (S), (R), and (M) denote sender-side, receiver-side, and message quantities, respectively. The sender generates a message M under its own context, and the receiver incorporates M into its context before continuing computation. We restrict our study to same-model reuse; cross-model transfer is outside our scope.

Text and KV communication. Let $C = (K, V)$ denote a KV cache. Under text communication, the sender transmits the tokens of M, and the receiver recomputes the corresponding message states under its own context, obtaining $\widehat{C}^{(M)}$. Under KV communication, the sender transmits its message cache $C^{(M)}$, from which an adapted cache $\widetilde{C}^{(M)}$ is inserted into the receiver context without re-encoding the message tokens. Direct reuse is the special case $\widetilde{C}^{(M)} = C^{(M)}$, whereas a cache-adaptation method transforms $C^{(M)}$ using receiver-side information.

Writing $C^{(R)}$ for the receiver's local-context cache, the two paths differ at the message representation inserted before subsequent computation:

$$\underbrace{C^{(R)} \oplus \widehat{C}^{(M)}}_{\text{text}} \qquad \text{versus} \qquad \underbrace{C^{(R)} \oplus \widetilde{C}^{(M)}}_{\text{KV}}, \tag{1}$$

where $\oplus$ denotes concatenation along the sequence dimension. Any subsequent continuation B is then processed from the corresponding cache, and its resulting states may therefore differ between the two paths. In general, $C^{(M)} \neq \widehat{C}^{(M)}$ because the same message is conditioned on different sender and receiver contexts; reducing this discrepancy is the central problem studied in this work. Cache substitution changes the receiver's context states, but does not replace the receiver's logits with the sender's logits.

Matched comparison. For a matched comparison, the two paths use the same message boundaries, the same continuation B, and the same decoding rule, so that cache substitution is the only experimental intervention. Avoiding message recomputation reduces receiver-side computation rather than eliminating all costs associated with communication: sender generation, cache preparation and adaptation when applicable, and continuation processing still incur computation. Formal prefill and decoding notation, together with cache-materialization details, are provided in Appendix A.

3 CORE CHALLENGES IN CROSS-CONTEXT KV CACHE REUSE

While KV cache-mediated communication avoids redundant message re-encoding, a central question remains: can sender-generated KV states be reliably reused under a different receiver context? A message is encoded under the sender's role instructions, interaction history, and token positions, whereas the receiver incorporates that same message under a different prefix and local context. Direct transfer therefore exposes a cross-context representation mismatch between sender-generated and receiver-native states. We first characterize the reliability of direct reuse across models and role pairs, and then analyze three coupled dimensions of this mismatch: Position Shift, Magnitude Misalignment, and History Information Conflict.

3.1 FEASIBILITY ANALYSIS OF CROSS-CONTEXT KV CACHE REUSE

We measure direct reuse using the four-model MMLU debate protocol described in Section 5, covering eight expert roles and all 56 ordered off-diagonal sender–receiver pairs (Du et al., 2023; Liang et al., 2024; Hendrycks et al., 2021).

Direct KV retains 91.9% of the Text baseline macro accuracy on average (Table 1), but its effect is highly heterogeneous across models. It underperforms Text on Qwen2.5-3B-Instruct (Qwen et al., 2025), Qwen3-4B-Instruct-2507 (Yang et al., 2025), and LLaMA3.2-3B-Instruct (Meta AI, 2024), while exceeding Text on Ministral-3-3B-Instruct-2512 (Liu et al., 2026).

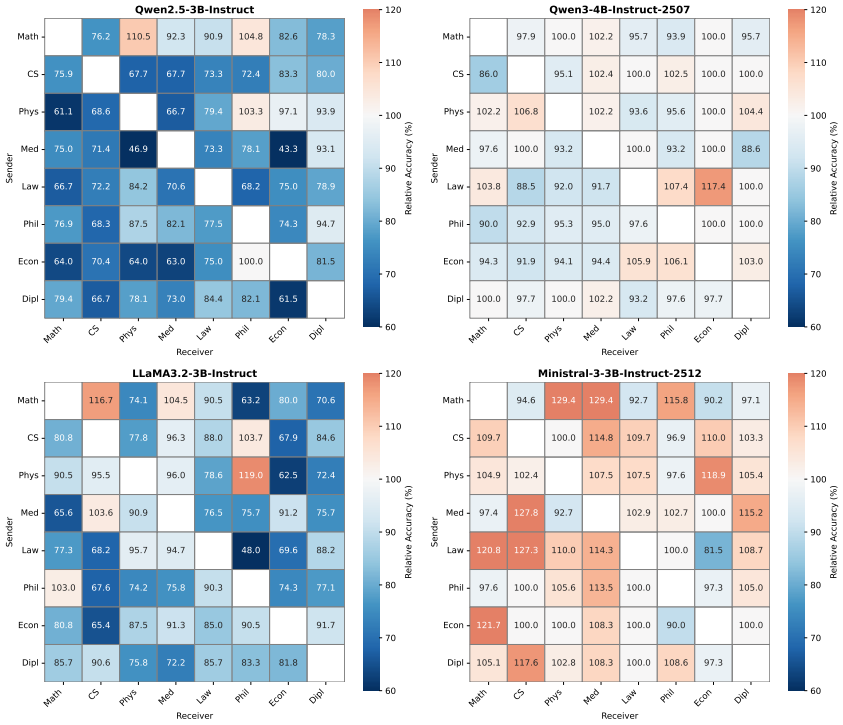


Figure 2: Relative accuracy of Direct KV communication relative to matched Text communication across ordered sender–receiver expert pairs. Each cell shows $100A^{\mathrm{KV}}_{s,r}/A^{\mathrm{text}}_{s,r}$; diagonal cells are omitted. Expert roles are Math (Mathematics), CS (Computer Science), Phys (Physics), Med (Medicine), Law, Phil (Philosophy), Econ (Economics), and Dipl (Diplomacy). Blue indicates degradation below the Text baseline, white indicates parity, and red indicates improvement. The substantial variation across role pairs suggests that direct KV reuse depends on sender–receiver context compatibility, motivating context-specific alignment.

Figure 2 further shows substantial variation across ordered sender–receiver pairs. These results suggest that the reliability of direct KV reuse depends on sender–receiver context compatibility, motivating receiver-side alignment.

3.2 Analysis of Cross-Context KV Discrepancy

The feasibility results show that direct cross-context KV reuse is not uniformly reliable. The same message can acquire different internal states when conditioned on different system prompts, interaction histories, and prefix lengths. We analyze three coupled forms of this discrepancy (Figure 3), which affect complementary aspects of attention computation and can occur simultaneously. *Position Shift* alters the positional phase of the message; *Magnitude Misalignment* changes the scale at which transferred keys compete for attention and values contribute to the attention output; and *History-Conditioned Conflict* captures how sender-conditioned information interacts with the receiver's own instructions and evidence. These three mismatches motivate the corresponding components introduced in Section 4.

Position Shift. For message token j, a cached RoPE key produced by the sender is

$$k_j^{(S)} = R(p_j^{(S)})u_j^{(S)}, \tag{2}$$

where $u_j^{(S)}$ is the sender-conditioned pre-rotation key and $p_j^{(S)}$ is its sender-side position (Su et al., 2023). When the receiver has a different prefix length, the corresponding receiver-side position becomes $p_j^{(R)}$. Directly reusing $k_j^{(S)}$ therefore retains a rotary phase associated with $p_j^{(S)}$ rather than

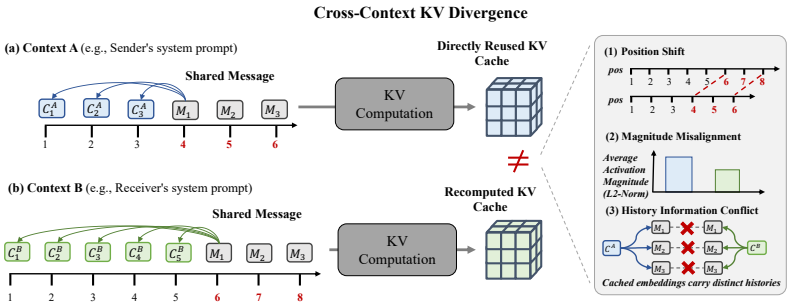


Figure 3: Three sources of cross-context KV discrepancy. The same message is encoded under sender and receiver contexts with different positions, activation scales, and history-conditioned directions and values. These mismatches motivate position transformation, scale alignment, and dynamic attention enhancement, respectively.

$p_j^{(R)}$, misaligning the message key with receiver-side queries. This positional mismatch motivates explicitly transforming transferred keys to their receiver-side rotary positions.

Magnitude Misalignment. Even after positional alignment, sender- and receiver-conditioned states can have different activation scales. An attention logit can be written as

$$\frac{q^\top k}{\sqrt{d}} = \frac{\|q\|_2 \|k\|_2 \cos(q,k)}{\sqrt{d}},$$

showing that key magnitude directly affects competition in the attention softmax, while value magnitude affects the scale of each token's contribution to the attention output. Because RoPE applies an orthogonal transformation and therefore preserves vector norms, positional alignment alone cannot correct such scale differences. This motivates calibrating transferred keys and values against receiver-local statistics.

History Information Conflict. Position and norm alignment leave context-dependent directions and value content unresolved. Sender-generated keys and values remain conditioned on the sender's role instructions and interaction history. Once inserted into the receiver context, transferred keys can alter how receiver queries allocate attention between transferred and receiver-local states, while transferred values alter the content contributed to the attention output. The receiver must therefore incorporate incoming information while remaining consistent with its own instructions and local context, motivating modulation of receiver-instruction states rather than uniform intervention.

The next section formalizes the corresponding operations.

4 Robust Cross-Context KV Cache Reuse

KVTrans is a training-free framework that adapts sender-generated message states to receiver contexts without model retraining. RoPE Alignment and L2-Rescaling modify transferred caches before insertion, while Dynamic Attention Amplification modulates receiver-side attention during subsequent decoding. Initial independent generations remain part of ordinary task computation and do not constitute an additional calibration round.

4.1 RoPE Alignment

For layer ℓ, KV head g, and message token j, let $k_{\ell,g,j}^{(M)}$ and $v_{\ell,g,j}^{(M)}$ denote the sender-generated key and value states for token j in the message cache $C^{(M)}$, and let $p_j^{(S)}$ and $p_j^{(R)}$ denote its sender-side and receiver-side positions, respectively. We transform

$$k_{\ell,g,j}^{\text{pos}} = R(p_j^{(R)}) R(p_j^{(S)})^{-1} k_{\ell,g,j}^{(M)}, \qquad v_{\ell,g,j}^{\text{pos}} = v_{\ell,g,j}^{(M)}. \tag{3}$$

This removes the sender-side rotary phase and applies the corresponding receiver-side phase. For standard RoPE with shared frequencies, the transformation depends only on $p_j^{(R)} - p_j^{(S)}$. Values

are not rotated. This corrects explicit position offsets, but not the contextual dependence of the underlying states.

4.2 L2-RESCALING

We calibrate keys and values independently against receiver-local cache states already available from the receiver's prompt and interaction history. Let $\mathcal{M}$ and $\mathcal{R}$ denote the message and receiver-reference positions, respectively, and write $x^K = k$ and $x^V = v$. For each layer ℓ, KV head g, and $X \in \{K, V\}$, we compute

$$\begin{aligned} \mu_{\ell,g}^{X,M} &= \frac{1}{|\mathcal{M}|} \sum_{j \in \mathcal{M}} \|x_{\ell,g,j}^{X,\mathrm{pos}}\|_2, \quad \mu_{\ell,g}^{X,R} = \frac{1}{|\mathcal{R}|} \sum_{i \in \mathcal{R}} \|x_{\ell,g,i}^{X,\mathrm{ref}}\|_2, \\ \alpha_{\ell,g}^{X} &= \frac{\mu_{\ell,g}^{X,R}}{\mu_{\ell,g}^{X,M} + \epsilon}, \qquad\qquad \widetilde{x}_{\ell,g,j}^{X} = \alpha_{\ell,g}^{X} x_{\ell,g,j}^{X,\mathrm{pos}}, \end{aligned} \tag{4}$$

where $\epsilon > 0$ is a small constant for numerical stability. The scaling coefficient is shared across message tokens within each layer, KV head, and state type, preserving their relative norm ratios and directions. No additive shift is applied. The resulting mean norm is calibrated toward the receiver-local reference while leaving the within-message directional structure unchanged. Because the reference states are already present in the receiver cache, this calibration requires no receiver-side re-encoding of the message. For orthogonal RoPE rotations, scalar rescaling and position correction commute (Appendix D).

4.3 DYNAMIC ATTENTION AMPLIFICATION

To mitigate residual context mismatch, we dynamically select heads with stronger relative preference for receiver instructions. Let $\mathcal{S}$ denote system-instruction positions, $\mathcal{O}_t$ the remaining positions at decoding step t, and $\bar{k}_{\ell,g}^{\mathcal{A}} = |\mathcal{A}|^{-1} \sum_{i \in \mathcal{A}} k_{\ell,g,i}$ the pre-amplification key centroid. For query head h with KV head $g(h)$, define

$$d_{\ell,h}(t) = \cos(q_{\ell,h}(t), \bar{k}_{\ell,g(h)}^{\mathcal{S}}) - \cos(q_{\ell,h}(t), \bar{k}_{\ell,g(h)}^{\mathcal{O}_t}). \tag{5}$$

This score measures relative query–centroid similarity, not attention mass. We standardize it within each layer and select

$$z_{\ell,h}(t) = \frac{d_{\ell,h}(t) - \mathrm{mean}_{h'}\, d_{\ell,h'}(t)}{\mathrm{std}_{h'}\, d_{\ell,h'}(t) + \epsilon}, \quad \mathcal{H}_\ell(t) = \{h : z_{\ell,h}(t) > \tau\}. \tag{6}$$

The selection is recomputed at every layer and decoding step. For selected heads, the associated system keys are scaled by γ; values and other keys remain unchanged. We fix τ and γ on development data. GQA handling and implementation details are given in Appendix D.

The receiver inserts $(\widetilde{K}^{(M)}, \widetilde{V}^{(M)})$ and applies this dynamic selection during decoding. Cache transformation and selection overheads are included in our cost accounting.

5 EXPERIMENTS

We evaluate quality and receiver-side message-processing cost in two settings: MMLU(Hendrycks et al., 2021) debate and WebSailor(Li et al., 2025) collaboration on BrowseComp-EN/ZH(Wei et al., 2025; Zhou et al., 2025). Table 1 reports the two settings separately rather than averaging heterogeneous tasks into a single score. BBH in debate (Suzgun et al., 2023) and HotPotQA with GAM (Yang et al., 2018; Yan et al., 2025) generalization results are provided in Appendix F. MMLU additionally serves as the controlled setting for measuring runtime and cache-adaptation overhead. For the collaborative setting, we report Pass@1 as the mean success rate over repeated rollouts.

5.1 EXPERIMENTAL SETUP

Competition-based communication. Our MMLU debate protocol (Hendrycks et al., 2021) uses eight expert roles, forming 56 ordered sender–receiver pairs after excluding same-role pairs. For

Table 1: Quality and receiver-side prefill tokens. Bold denotes Ours, not the column best. Relative rows are percentages of Text; for prefill, 40.86% means 59.14% fewer tokens, not a speedup. [a]Four-model macro; per-model results in Appendix E. [b]Pass@1 and per-question prefill tokens averaged over rollouts on a fixed subset per language; EN and ZH are reported separately. See Appendices I and K for details.

	Competition (Debate)		**Collaboration (WebSailor)**			
	MMLU[a]		BrowseComp-EN[b]		BrowseComp-ZH[b]	
Method	Acc. (%)	Prefill tok.	Pass@1 (%)	Prefill tok.	Pass@1 (%)	Prefill tok.
Text	64.88	8,331.8	34.67	583,271	40.00	285,912
Direct KV	59.61	3,404.4	10.00	193,944	26.67	145,844
Ours	65.44	3,404.4	28.33	314,580	33.33	178,047
Relative to Text (%)						
Direct KV	91.87	40.86	28.85	33.25	66.67	51.01
Ours	100.85	40.86	81.71	53.93	83.33	62.27

each pair, we sample 50 questions from the sender role's domain. The two experts first answer independently, exchange their answers, and each perform one revision; the two initial answers and two revised answers are then provided to a same-base judge. KV reuse is applied both when experts process exchanged messages during revision and when the judge processes the four expert messages. All debate runs use greedy decoding with matched generation limits. We evaluate Qwen2.5-3B-Instruct, Qwen3-4B-Instruct-2507, LLaMA3.2-3B-Instruct, and Ministral-3-3B-Instruct-2512. Full protocol details are provided in Appendix A.

Collaboration-based communication. We evaluate WebSailor (Li et al., 2025) on BrowseComp-EN/ZH, reusing tool-summary message states at the research agent. For each language, we fix a question subset and run multiple independent rollouts per question with matched decoding and tool budgets across methods. We report averages over rollouts, with Ours using the full three-component method. Full protocol and implementation details are provided in Appendix I.

Baselines and metrics. Text re-encodes exchanged messages, Direct KV reuses their caches without correction, and Ours applies all components in Section 4. Debate accuracy is based on the final judge answer. Prefill tokens count receiver-side request tokens actually processed. MMLU results are macro averages, whereas WebSailor results are per-question averages over sampled rollouts. Relative metrics are normalized to Text. Detailed scoring and cost definitions are in Appendices A and K.

5.2 Main Results

Competition. Direct KV reduces MMLU macro accuracy from 64.88% under Text communication to 59.61%, while Ours recovers it to 65.44%. Ours improves over Direct KV across all four models. BBH results are in Appendix F.

Collaboration. On BrowseComp-EN, Ours raises Pass@1 from 10.00% under Direct KV to 28.33% (Text: 34.67%) while using 53.9% of Text prefill tokens. On BrowseComp-ZH, it raises Pass@1 from 26.67% to 33.33% (Text: 40.00%) while using 62.3% of Text prefill tokens. HotPotQA results under GAM are in Appendix F.

5.3 MMLU Efficiency Analysis

MMLU requires 8,331.8 receiver-side prefill tokens for Text versus 3,404.4 for both Direct KV and Ours, a 59.14% reduction. System prompts remain prefilled, while generated messages are reused; detailed cost boundaries are defined in Appendix K.

For four matched expert pairs, average prefill time decreases from 4.6209 s for Text to 1.9853 s for Ours, a 57.04% reduction. The measurement ends before the first decoding step and includes RoPE

Table 2: Prefill time on four MMLU sender–receiver pairs. Mean Ours/Text time is 42.96%, corresponding to a 57.04% reduction. Timing includes RoPE Alignment and L2-Rescaling overhead.

Pair	Text (s)	Ours (s)	Rel. (%)
CS→Math	5.0730	2.0477	40.36
Law→CS	5.9669	2.8592	47.92
Physics→Medicine	3.5577	1.8331	51.52
Economics→Philosophy	3.8859	1.2012	30.91
Mean	4.6209	1.9853	42.96

Table 3: Sequential component ablation on matched expert pairs. "Rel." is accuracy relative to the corresponding Text baseline. Results average over 10 common pairs for LLaMA3.2-3B-Instruct and 12 for Qwen2.5-3B-Instruct.

Configuration	LLaMA3.2-3B-Ins		Qwen2.5-3B-Ins	
	Acc. (%)	Rel. (%)	Acc. (%)	Rel. (%)
Text	52.60	100.00	55.50	100.00
Direct KV	44.20	84.03	42.33	76.27
+ RoPE Alignment	46.00	87.45	42.67	76.88
+ L2-Rescaling	46.60	88.59	51.00	91.89
Full KVTRANS	53.00	100.76	55.00	99.10

Alignment and L2-Rescaling overhead. End-to-end timing, including decoding-time overhead, is reported in Appendix E.2.

6 ANALYSIS

Component effects. Table 3 reveals model-dependent contributions from the three components. RoPE Alignment alone improves accuracy by 1.80 percentage points on LLaMA3.2-3B and 0.34 points on Qwen2.5-3B, leaving substantial gaps to Text. Adding L2-Rescaling yields a larger gain on Qwen2.5 (+8.33 points) than on LLaMA (+0.60), whereas Dynamic Attention Amplification further improves LLaMA by 6.40 points and Qwen2.5 by 4.00. This pattern suggests that positional correction alone is insufficient and that the relative benefits of magnitude calibration and receiver-instruction modulation differ across models. These gains are conditional on the preceding components because the ablation is sequential.

Why dynamic head selection? We compare dynamic selection with fixing the heads selected at the first decoding step and amplifying all heads uniformly. Across four matched expert pairs, dynamic selection achieves 76.0% mean accuracy, versus 71.5% for fixed selection and 72.0% for all-head amplification. The gains are concentrated in Law–CS and Physics–Medicine, where dynamic selection exceeds both alternatives by 10 and 6 percentage points, respectively. These results support adapting the selected heads during decoding, while also showing that the benefit varies across role pairs. Full pairwise results are provided in Appendix D.1.

Scaling with communication. Figure 4 shows that absolute prefill savings increase with interaction depth and expert count. From two to five rounds, the Text-minus-Ours token gap grows from 3.18k to 15.38k; from two to four experts, it grows from 3.75k to 7.40k. Quality remains competitive as interaction deepens: Ours matches Direct KV at two rounds and exceeds it by 2.5–5.0 percentage points at three to five rounds (Appendix G.1). At five rounds, Ours reaches 77.5% accuracy, compared with 72.5% for Direct KV and 74.5% for Text. Expert-count accuracy is nonmonotonic, however, so larger communication savings should not be interpreted as guaranteed accuracy gains from adding agents.

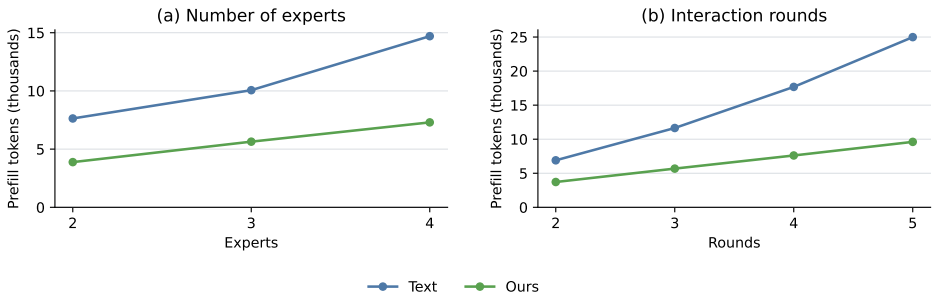


Figure 4: Receiver-side prefill tokens under expert-count and round-count scaling. The expert-count panel averages science and humanities equally; the round-count panel averages four role pairs. Absolute token savings grow with both expert count and interaction depth. Full results are in Appendix G.1.

7 Related Work

Our work studies training-free cross-context KV reuse among same-model agents, distinct from identical-prefix reuse, learned inter-model communication, and training-based cache adaptation. Further comparisons are provided in Appendix L.

KV caching and prefix reuse. RadixAttention and Prompt Cache reuse identical prefixes across requests (Zheng et al., 2024; Gim et al., 2024), whereas CacheBlend and Cache-Craft selectively recompute cached states for RAG (Yao et al., 2025; Agarwal et al., 2025). KVCOMM (Ye et al., 2025) addresses training-free cross-context reuse with positional correction and anchor-based contextual offsets; our method instead uses receiver-local calibration and decode-time receiver-instruction modulation. KVComm (Shi et al., 2026) focuses on selective layer-wise KV sharing.

Multi-agent and latent communication. Multi-agent systems commonly coordinate through natural language or structured messages (Wu et al., 2023; Hong et al., 2024). DroidSpeak shares KV caches across fine-tuned variants without cross-context alignment (Liu et al., 2025). Cache-to-Cache and ThoughtComm learn communication interfaces through training (Fu et al., 2026; Zheng et al., 2025), while LatentMAS uses layer-wise KV states as latent working memory and recomputes before reasoning (Zou et al., 2026). In contrast, we adapt existing KV states at inference time among same-model agents while retaining standard autoregressive decoding.

8 Conclusion

KVTrans enables training-free cross-context KV reuse through RoPE Alignment, L2-Rescaling, and Dynamic Attention Amplification. On MMLU, it recovers macro accuracy from 59.61% under Direct KV to 65.44% while reducing receiver-side prefill tokens by 59.14%; across four matched expert pairs, measured prefill time decreases by 57.04%. On BrowseComp-EN/ZH, KVTrans substantially recovers the degradation from direct reuse while using 53.9%/62.3% of the Text prefill tokens. Additional BBH and HotPotQA results are provided in the appendix.

AI use statement

In this work, we used generative AI tools to assist with manuscript preparation, including language polishing, improving clarity, and structuring technical descriptions. We also used GPT-5.5 as an LLM-based evaluator for automated scoring in both the BrowseComp-EN and BrowseComp-ZH evaluations. Its role was limited to applying a predefined judging protocol; it was not used to formulate research hypotheses, design the proposed method, or determine the scientific conclusions.

The evaluation setup is documented in Appendix I. All AI-assisted outputs were reviewed by the authors, who take full responsibility for the final content, claims, and conclusions of this work.

ETHICS STATEMENT

This work presents a training-free framework for efficient multi-agent communication through KV cache reuse. We have considered potential ethical implications and address them as follows:

Human subjects and data privacy. This research does not involve human subjects or collect personal data. The experiments use publicly available benchmark datasets, including MMLU, BBH, HotpotQA, BrowseComp-EN, and BrowseComp-ZH. Generation experiments use publicly available open-weight language models; GPT-5.5 is used only as an automated evaluator for part of the BrowseComp evaluation, as disclosed in the AI Use Statement. No privacy-sensitive information is processed or disclosed.

Environmental impact. Our method aims to reduce computational costs in multi-agent systems by eliminating redundant prefill computation. The measured 57% reduction in receiver-side prefill time may reduce the computational and energy costs associated with prefilling, although we do not directly measure energy consumption or carbon emissions.

Potential applications and dual use. The proposed KV cache reuse mechanism is a general computational optimization technique for multi-agent communication. It may improve the efficiency of applications such as collaborative reasoning, research assistance, and educational tools, but the same efficiency gains could also benefit harmful applications. The method does not introduce new model capabilities or alter model safety mechanisms directly; nevertheless, its deployment should follow the safety and access-control practices appropriate to the underlying models and applications.

Broader impacts. By reducing redundant computation in multi-agent communication, this work may lower the computational cost of deploying collaborative AI systems, including in resource-constrained settings. At the same time, lower inference costs may increase the accessibility and scale of such systems, making responsible deployment and evaluation of downstream applications important.

REPRODUCIBILITY STATEMENT

We have made substantial efforts to ensure the reproducibility of this work. The method section (Section 4) provides complete algorithmic descriptions of RoPE Alignment, L2-Rescaling, and Dynamic Attention Amplification, with mathematical formulations for all transformations.

Detailed experimental configurations are documented in Appendix A (debate protocol and model configurations), Appendix B (debate prompt construction, representative system instructions, few-shot examples, and output formats), Appendix D (implementation details including norm statistics, head scoring, and GQA handling), and Appendix I (WebSailor integration and sampling specifications). Detailed experimental results are reported across the appendices: Appendix E provides per-model MMLU results and end-to-end timing measurements; Appendix F reports generalization results on BBH and HotpotQA under the GAM protocol; and Appendix G.1 presents communication-scale analyses over expert count and interaction depth. Cost-accounting definitions and measurement boundaries are detailed in Appendix K, distinguishing processed prefill tokens from cache-hit tokens and other protocol-level costs.

All MMLU and BBH experiments use greedy decoding with matched generation limits. WebSailor experiments use temperature sampling with three fixed-seed rollouts per question. Models are specified by their HuggingFace-compatible identifiers: Qwen2.5-3B-Instruct, Qwen3-4B-Instruct-2507, LLaMA3.2-3B-Instruct, Ministral-3-3B-Instruct-2512, and Tongyi-DeepResearch-30B-A3B. The appendices report the corresponding inference backends, numerical precision, serving configurations, and method hyperparameters, including the γ and τ values used in evaluation.

We commit to releasing the complete implementation code upon paper acceptance, including our modified SGLang fork for KV cache grafting with RoPE shift correction and L2-rescaling operations. The release will include prompt-construction scripts, evaluation pipelines, environment specifications, and the benchmark question splits used in our experiments. MMLU, BBH, HotpotQA, BrowseComp-EN, and BrowseComp-ZH are used subject to their respective licensing terms.

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, 2025. URL https://arxiv.org/abs/2502.15734.

Jiaming Cheng, Subhransu Das, and Rajiv Ramnath. When does latent communication pay? a causal audit of relayed KV caches in multi-agent LLMs, 2026. URL https://arxiv.org/abs/2608.04893.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate, 2023. URL https://arxiv.org/abs/2305.14325.

Zhuoyun Du, Runze Wang, Huiyu Bai, Zouying Cao, Xiaoyong Zhu, Yu Cheng, Bo Zheng, Wei Chen, and Haochao Ying. Enabling agents to communicate entirely in latent space. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026. URL https://aclanthology.org/2026.acl-long.1248/.

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*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/474ada926b331d78f06d95e8913111cc-Abstract-Conference.html.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, 2024. URL https://arxiv.org/abs/2311.04934.

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.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. Metagpt: Meta programming for a multi-agent collaborative framework, 2024. URL https://arxiv.org/abs/2308.00352.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. WebSailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025. URL https://arxiv.org/abs/2507.02592.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of EMNLP*, pp. 17889–17904, 2024. URL https://aclanthology.org/2024.emnlp-main.992/.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, Baptiste Rozière, Baudouin De Monicault, Clémence Lanfranchi, Corentin Barreau, Cyprien Courtot, Daniele Grattarola, Darius Dabert, Diego de las Casas, Elliot Chane-Sane, Faruk Ahmed, Gabrielle Berrada, Gaëtan Ecrepont, Gauthier Guinet, Georgii Novikov, Guillaume Kunsch, Guillaume Lample, Guillaume Martin, Gunshi Gupta, Jan Ludziejewski, Jason Rute, Joachim Studnia, Jonas Amar, Joséphine Delas, Josselin Somerville Roberts, Karmesh Yadav, Khyathi Chandu, Kush Jain, Laurence Aitchison, Laurent Fainsin, Léonard Blier, Lingxiao Zhao, Louis Martin, Lucile Saulnier, Luyu Gao, Maarten Buyl, Margaret Jennings, Marie Pellat, Mark Prins, Mathieu Poirée, Mathilde Guillaumin, Matthieu Dinot, Matthieu Futeral, Maxime Darrin, Maximilian Augustin, Mia Chiquier, Michel Schimpf, Nathan Grinsztajn, Neha Gupta, Nikhil Raghuraman, Olivier Bousquet, Olivier

Duchenne, Patricia Wang, Patrick von Platen, Paul Jacob, Paul Wambergue, Paula Kurylowicz, Pavankumar Reddy Muddireddy, Philomène Chagniot, Pierre Stock, Pravesh Agrawal, Quentin Torroba, Romain Sauvestre, Roman Soletskyi, Rupert Menneer, Sagar Vaze, Samuel Barry, Sanchit Gandhi, Siddhant Waghjale, Siddharth Gandhi, Soham Ghosh, Srijan Mishra, Sumukh Aithal, Szymon Antoniak, Teven Le Scao, Théo Cachet, Theo Simon Sorg, Thibaut Lavril, Thiziri Nait Saada, Thomas Chabal, Thomas Foubert, Thomas Robert, Thomas Wang, Tim Lawson, Tom Bewley, Tom Bewley, Tom Edwards, Umar Jamil, Umberto Tomasini, Valeriia Nemychnikova, Van Phung, Vincent Maladière, Virgile Richard, Wassim Bouaziz, Wen-Ding Li, William Marshall, Xinghui Li, Xinyu Yang, Yassine El Ouahidi, Yihan Wang, Yunhao Tang, and Zaccharie Ramzi. Ministral 3, 2026. URL https://arxiv.org/abs/2601.08584.

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 for cross-llm communication and multi-llm serving, 2025. URL https://arxiv.org/abs/2411.02820.

Meta AI. Llama 3.2 model card, 2024. URL https://github.com/meta-llama/llama-models/blob/main/models/llama3_2/MODEL_CARD.md.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Xiangyu Shi, Marco Chiesa, Gerald Q. Maguire Jr., and Dejan Kostic. KVComm: Enabling efficient LLM communication through selective KV sharing. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=F7rUng23nw.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL https://arxiv.org/abs/2504.12516.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation, 2023. URL https://arxiv.org/abs/2308.08155.

B. Y. Yan, Chaofan Li, Hongjin Qian, Shuqi Lu, and Zheng Liu. General agentic memory via deep research, 2025. URL https://arxiv.org/abs/2511.18423.

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, 2025. URL https://arxiv.org/abs/2505.09388.

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.

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*, 2025. URL https://openreview.net/forum?id=yGOytgjurF.

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 *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://arxiv.org/abs/2312.07104.

Yujia Zheng, Zhuokai Zhao, Zijian Li, Yaqi Xie, Mingze Gao, Lizhu Zhang, and Kun Zhang. Thought communication in multiagent collaboration. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/b2b502c3629beadda06311386d2c6f73-Abstract-Conference.html.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, Yuxin Gu, Sixin Hong, Jing Ren, Jian Chen, Chao Liu, and Yining Hua. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese, 2025. URL https://arxiv.org/abs/2504.19314.

Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, 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, 2026. URL https://arxiv.org/abs/2511.20639v4. Version 4; accepted at ICML 2026.

A Protocols and Reproducibility

Prefill and decoding notation. For a token sequence x and an optional preceding cache C_{pre}, define

$$\big(\mathbf{z}_x, C_x\big) = \mathrm{Prefill}_\theta(x; C_{\mathrm{pre}}), \qquad C_{\mathrm{full}} = C_{\mathrm{pre}} \oplus C_x, \tag{7}$$

where $\mathbf{z}_x$ is the next-token logit vector and C_x contains the states computed for x. For sequence length T, the key and value tensors have shape $L \times H_{\mathrm{KV}} \times T \times d$, omitting the batch dimension, where H_{KV} is the number of KV heads. We write $(y, C_y) = \mathrm{Decode}_\theta(\mathbf{z}_0; C_0)$ for a decoded sequence and its incremental states.

Let $P^{(S)}$ and $P^{(R)}$ denote the sender and receiver contexts from Section 2. Their local prefills give $(\mathbf{z}^{(S)}, C^{(S)}) = \mathrm{Prefill}_\theta(P^{(S)}; \varnothing)$ and $(\mathbf{z}^{(R)}, C^{(R)}) = \mathrm{Prefill}_\theta(P^{(R)}; \varnothing)$. The sender generates $(M, C^{(M)}) = \mathrm{Decode}_\theta(\mathbf{z}^{(S)}; C^{(S)})$, where $C^{(M)} = (K^{(M)}, V^{(M)})$ remains conditioned on

$P^{(S)}$. Under Text communication, the receiver recomputes the message states as $(\mathbf{z}^{(R,M)}, \widehat{C}^{(M)}) = \mathrm{Prefill}_\theta(M; C^{(R)})$. Both paths then process the same continuation B:

$$\begin{aligned}
\left(\mathbf{z}_B^{\text{text}}, C_B^{\text{text}}\right) &= \mathrm{Prefill}_\theta\left(B; C^{(R)} \oplus \widehat{C}^{(M)}\right), \\
\left(y_{\text{text}}, C_{y_{\text{text}}}\right) &= \mathrm{Decode}_\theta\left(\mathbf{z}_B^{\text{text}}; C^{(R)} \oplus \widehat{C}^{(M)} \oplus C_B^{\text{text}}\right), \\
\left(\mathbf{z}_B^{\text{KV}}, C_B^{\text{KV}}\right) &= \mathrm{Prefill}_\theta\left(B; C^{(R)} \oplus \widetilde{C}^{(M)}\right), \\
\left(y_{\text{KV}}, C_{y_{\text{KV}}}\right) &= \mathrm{Decode}_\theta\left(\mathbf{z}_B^{\text{KV}}; C^{(R)} \oplus \widetilde{C}^{(M)} \oplus C_B^{\text{KV}}\right).
\end{aligned} \tag{8}$$

Cache substitution avoids receiver-side recomputation of M, but does not substitute sender logits or eliminate the forward computation for B.

Cache materialization and continuation. Token selection and KV materialization are distinct. Given logits $\mathbf{z}_{t-1}$ and cache C_{t-1},

$$y_t \sim \pi(\cdot \mid \mathbf{z}_{t-1}), \qquad (\mathbf{z}_t, C_{y_t}) = \mathrm{Forward}_\theta(y_t; C_{t-1}), \qquad C_t = C_{t-1} \oplus C_{y_t}. \tag{9}$$

Here π includes greedy decoding as a deterministic case, and $\mathrm{Forward}_\theta$ denotes a one-token cached forward pass. Because y_t has no KV entry until this pass, transmitting a complete message cache may require materializing the final generated token. This cost, together with processing B after cache insertion, is included in our accounting.

B Debate Prompt

The debate protocol uses role-specific system prompts, fixed message templates, and runtime substitutions. Braces below denote runtime fields rather than literal prompt text.

Stage 1 Prompt Construction (MMLU Mathematics Expert)
(1) System Prompt

```
You are the Mathematics Expert.  Your responsibilities:
- Identify the mathematical domain
(algebra/calculus/number theory/geometry/linear algebra/
probability & statistics/logic).
- Use formal definitions, theorems, and stepwise
derivations; check edge cases and units when applicable.
- Keep reasoning concise, precise, and strictly relevant
to the question and options.
ReAct protocol (strict):
1) Thought:  step-by-step reasoning (equations,
transformations, short proofs, sanity checks).
2) Action:  your conclusion and brief justification.  On
the final line, output only Final Answer:  [X], where X
is one of {A,B,C,D}.
```

(2) Few-shot Examples (5 examples from `mmlu_college_mathematics_4`)

```
Question:  Let V be the set of all real polynomials p(x).  Let
transformations T and S be defined on V by T:p(x) → xp(x) and S:p(x)
→ p'(x) = d/dx p(x), and interpret (ST)(p(x)) as S(T(p(x))).  Which of
the following is true?
Options:
A) ST = 0
B) ST = T
C) ST = TS
D) ST - TS is the identity map of V onto itself.
Answer:  D


Question:  A tank initially contains a salt solution of 3 grams of salt
dissolved in 100 liters of water.  A salt solution containing 0.02 grams
of salt per liter of water is sprayed into the tank at a rate of 4 liters
per minute.  The sprayed solution is continually mixed with the salt
solution in the tank, and the mixture flows out of the tank at a rate of
4 liters per minute.  If the mixing is instantaneous, how many grams of
salt are in the tank after 100 minutes have elapsed?
```

```
Options:
A) 2
B) 2 − e^{−2}
C) 2 + e^{−2}
D) 2 + e^{−4}
Answer: D

[... 3 more examples with the same format ...]
```

(3) Target Question

```
Question: {question_text}
Options:
A) {option_a}
B) {option_b}
C) {option_c}
D) {option_d}
```

(4) Role Instruction

```
Strictly respond in the role assigned for this round. Maintain the
role's tone and responsibilities.
End with exactly one final-answer line: Final Answer: [A|B|C|D].
Immediately afterward, output <END_OF_TURN> on the final line.
```

Stage 2 and Stage 3 Message Construction
Stage 2 (Reviser): `[Own Stage-1 History] + [Counterpart's Answer] + [Revision Instruction]`
Stage 3 (Judge): `[Judge System Prompt] + [A_1, B_1, A_2, B_2]`

Cache-Insertion Points Across Methods
Text: Answer spans are transmitted as text and re-encoded by the receiver.
Direct KV: Answer spans are transmitted as KV cache segments and inserted without adaptation.
Ours: Answer spans are transformed using RoPE Alignment and L2-Rescaling before insertion, with Dynamic Attention Amplification applied during subsequent decoding.

The debate uses two domain experts and a same-base final judge. In Stage 1, each expert receives a domain-specific system prompt, five same-domain MMLU few-shot examples, and the target question, and answers independently. In Stage 2, each expert retains its Stage 1 history, receives the counterpart's answer, and revises its response. In Stage 3, the judge receives the two initial and two revised answers in the fixed order A_1, B_1, A_2, B_2 and produces the final decision. Text supplies these messages as text, whereas Direct KV and Ours supply the corresponding answer spans as inserted caches. All methods therefore preserve the same logical message order and judge input.

The expert system prompts cover eight roles: mathematics, computer science, physics, medicine, law, philosophy, economics, and diplomacy. Each requests domain-appropriate step-by-step analysis, a brief justification, and the final line `Final Answer: [X]`, where $X \in \{A, B, C, D\}$. The pair-specific reviser prompt identifies both domains and asks the expert to compare its Stage 1 conclusion with the counterpart's response, correct errors, and retain the same output format. The judge system prompt is: "You are the final Judge. Make the most fair and accurate decision based on all provided analyses." It requests a brief adjudication followed by the same final-answer format. Outputs without a valid A–D final answer are counted as incorrect and remain in the 50-question denominator.

Each role uses five examples from the corresponding MMLU development set. For evaluation, 50 questions are sampled once from each domain's test split and held fixed across models, role pairs, and methods. The same samples are used for Qwen2.5-3B-Instruct, Qwen3-4B-Instruct-2507, LLaMA3.2-3B-Instruct, and Ministral-3-3B-Instruct. The evaluation artifact retains all eight role prompts, pair-specific reviser prompts, rendered few-shot examples, chat templates, and stop strings. BBH follows the same three-stage topology with task-specific analyst/reviser prompts and a judge output of `Final Answer: <Y>` for free-form answers.

KV reuse applies to both expert exchange and judge input. All MMLU runs use greedy decoding with matched output limits across Text, Direct KV, and Ours: 2048 tokens for each initial expert response, 2048 for each revision, and 4096 for the judge. For each ordered role pair, the same fixed 50 questions from the sender domain are used across models and methods.

MMLU experiments use the Hugging Face inference backend with each checkpoint's default numerical precision on one NVIDIA A100 80GB GPU. The full method uses attention-amplification magnitude $\gamma = 1.2$ and selection threshold $\tau = 0.1$, with no cap on the number of selected heads. BBH uses the same backend, precision, hardware, and head-selection policy, with $\gamma = 2.0$ and $\tau = 0.1$. For both datasets, we search $\gamma \in \{1.0, 1.1, \ldots, 2.0\}$ and $\tau \in \{0.0, 0.1, \ldots, 0.5\}$ on the corresponding development set and fix the selected configuration for evaluation.

C Diagnostic Formulation

The following identities characterize possible sources of discrepancy rather than attributing measured fractions of the end-to-end error. For a fixed query and sender pre-rotation key, the explicit positional contribution to the attention-logit difference is

$$\delta s_j^{\mathrm{pos}} = \frac{q^\top [R(p_j^{(S)}) - R(p_j^{(R)})] u_j^{(S)}}{\sqrt{d}}. \tag{10}$$

For matched visible positions $\mathcal{V}$, reused states (superscript U) and receiver-native states (superscript (R)) satisfy

$$o^U - o^{(R)} = \underbrace{\sum_{i \in \mathcal{V}} (a_i^U - a_i^{(R)}) v_i^{(R)}}_{\text{attention redistribution}} + \underbrace{\sum_{i \in \mathcal{V}} a_i^U (v_i^U - v_i^{(R)})}_{\text{value-content discrepancy}}. \tag{11}$$

Queries may diverge in subsequent layers or decoding steps. Fixed-message diagnostics should therefore hold message content and continuation constant and first verify that receiver-native caches inserted through the same path reproduce native computation within numerical tolerance. Attention distributions can further diagnose changes in attention allocation, complementing task-level accuracy.

For role pairs (s, r), we report absolute accuracies $A_{s,r}^{\mathrm{text}}$ and $A_{s,r}^{\mathrm{KV}}$ together with $\rho_{s,r} = 100 A_{s,r}^{\mathrm{KV}} / A_{s,r}^{\mathrm{text}}$ when the denominator is nonzero. The mean pairwise ratio is generally not equal to the ratio of macro accuracies. Diagonal entries are omitted rather than imputed.

D Method Details

Dynamic Attention Amplification details. Let $\mathcal{S}$ denote system-instruction positions and $\mathcal{O}_t$ the remaining comparison positions at decoding step t. For $\mathcal{A} \in \{\mathcal{S}, \mathcal{O}_t\}$, define the pre-amplification centroid

$$\bar{k}_{\ell,g}^{\mathcal{A}} = |\mathcal{A}|^{-1} \sum_{i \in \mathcal{A}} k_{\ell,g,i}.$$

For query head h with associated KV head $g(h)$, define

$$d_{\ell,h}(t) = \cos(q_{\ell,h}(t), \bar{k}_{\ell,g(h)}^{\mathcal{S}}) - \cos(q_{\ell,h}(t), \bar{k}_{\ell,g(h)}^{\mathcal{O}_t}). \tag{12}$$

After layer-wise standardization, we select heads with $z_{\ell,h}(t) > \tau$:

$$z_{\ell,h}(t) = \frac{d_{\ell,h}(t) - \mathrm{mean}_{h'}\, d_{\ell,h'}(t)}{\mathrm{std}_{h'}\, d_{\ell,h'}(t) + \epsilon}, \quad \mathcal{H}_\ell(t) = \{h : z_{\ell,h}(t) > \tau\}. \tag{13}$$

We define query-head gains $a_{\ell,h}(t) = 1 + (\gamma - 1)\mathbf{1}\{h \in \mathcal{H}_\ell(t)\}$. For each KV head g, $b_{\ell,g}(t)$ is the mean gain over query heads sharing that KV head; under standard MHA, this reduces to the corresponding query-head gain. Only system keys are rescaled in the current attention operation:

$$k_{\ell,g,i}^{\mathrm{attn}}(t) = [1 + (b_{\ell,g}(t) - 1)\mathbf{1}\{i \in \mathcal{S}\}]\, k_{\ell,g,i}(t), \qquad v_{\ell,g,i}^{\mathrm{attn}}(t) = v_{\ell,g,i}(t). \tag{14}$$

This scaling is transient and does not modify the stored KV cache.

Norm statistics and operation order. For $\mu_{\ell,g}^{X,M} > 0$, Equation 4 gives

$$\frac{1}{|\mathcal{M}|}\sum_{j\in\mathcal{M}}\|\widetilde{x}_{\ell,g,j}^{X}\|_2 = \mu_{\ell,g}^{X,R}\frac{\mu_{\ell,g}^{X,M}}{\mu_{\ell,g}^{X,M}+\epsilon} \approx \mu_{\ell,g}^{X,R}, \tag{15}$$

where the approximation holds when $\mu_{\ell,g}^{X,M} \gg \epsilon$. Because RoPE rotations are orthogonal, they preserve norms and commute with scalar rescaling. Our implementation computes the statistics in float32 with $\epsilon = 10^{-6}$, then casts the states back to their original dtype. Keys and values are rescaled separately without a centroid shift. The receiver reference set is protocol-specific and need not be restricted to system-prompt positions.

Head scoring and GQA. The system segment $\mathcal{S}$ is determined by the receiver prompt boundary, and the comparison set is $\mathcal{O}_t = \mathcal{V}_t \backslash \mathcal{S}$. Key centroids are computed after positional encoding. Cosine similarities are numerically stabilized, and standardization uses the sample standard deviation across query heads. A high standardized score does not imply a positive raw contrast.

For query heads $\mathcal{G}_g = \{h : g(h) = g\}$ sharing KV head g, we average their gains:

$$b_{\ell,g}(t) = \frac{1}{|\mathcal{G}_g|}\sum_{h\in\mathcal{G}_g} a_{\ell,h}(t) = 1 + (\gamma - 1)\frac{|\mathcal{G}_g \cap \mathcal{H}_\ell(t)|}{|\mathcal{G}_g|}. \tag{16}$$

Thus, query heads sharing a KV head use the same effective key scaling. If the backend expands KV heads to query-head count, the corresponding $b_{\ell,g}(t)$ is applied to each expanded copy.

For visible positions $\mathcal{V}_t$, the modified attention is

$$\begin{aligned} s_{\ell,h,i}^{\mathrm{attn}}(t) &= \left[1 + (b_{\ell,g(h)}(t) - 1)\mathbf{1}\{i \in \mathcal{S}\}\right]\frac{q_{\ell,h}(t)^\top k_{\ell,g(h),i}(t)}{\sqrt{d}}, \\ o_{\ell,h}(t) &= \sum_{i\in\mathcal{V}_t}\frac{\exp(s_{\ell,h,i}^{\mathrm{attn}}(t))}{\sum_{j\in\mathcal{V}_t}\exp(s_{\ell,h,j}^{\mathrm{attn}}(t))} v_{\ell,g(h),i}(t). \end{aligned} \tag{17}$$

The intervention rescales signed attention logits rather than probabilities, so a gain above one does not necessarily increase attention mass. Scaling is noncumulative and is applied only during decoding after foreign-cache insertion, not during initial independent generation.

D.1 Attention-Selection Ablations

Dynamic selection recomputes selected heads at every layer and decoding step. Fixed selection reuses the heads selected at the first decoding step, whereas all-head amplification applies the gain uniformly to all heads. Table 4 compares these variants on the four matched expert pairs used in Table 2.

Table 4: Attention-selection ablation accuracy (%) on four matched expert pairs. Fixed reuses first-step selections throughout decoding; All-Head amplifies all heads uniformly.

Selection	CS–Math	Law–CS	Phys.–Med.	Econ.–Phil.	Mean
Text	84.00	44.00	88.00	68.00	71.00
Direct KV	82.00	46.00	90.00	62.00	70.00
Dynamic (Ours)	82.00	56.00	96.00	70.00	76.00
Fixed	80.00	46.00	90.00	70.00	71.50
All-Head	82.00	46.00	90.00	70.00	72.00

Dynamic selection achieves the highest mean accuracy (76.00%), compared with 71.50% for Fixed and 72.00% for All-Head. The largest gaps occur on Law–CS and Physics–Medicine, where Dynamic exceeds both simplified variants by 10 and 6 points, respectively.

E DETAILED MMLU RESULTS

E.1 PER-MODEL MMLU RESULTS

Table 5 reports per-model accuracy and receiver-side prefill-token counts underlying the macro results in Table 1.

Table 5: Per-model MMLU results over 56 ordered role pairs. The macro row is the equal-model average. Direct KV and Ours process identical prefill-token counts under this protocol; this does not imply identical runtime.

Model	Method	Accuracy (%)	Prefill tok.
Qwen2.5-3B-Instruct	Text	58.39	6437.7
	Direct KV	44.93	3395.3
	Ours	57.46	3395.3
Qwen3-4B-Instruct-2507	Text	79.57	8230.6
	Direct KV	77.89	3395.3
	Ours	80.07	3395.3
LLaMA3.2-3B-Instruct	Text	54.36	8031.5
	Direct KV	44.93	3424.2
	Ours	51.68	3424.2
Ministral-3-3B-Instruct-2512	Text	67.21	10627.5
	Direct KV	70.68	3402.7
	Ours	72.54	3402.7
Macro	Text	64.88	8331.8
	Direct KV	59.61	3404.4
	Ours	65.44	3404.4

E.2 END-TO-END TIMING

Table 6 reports per-question end-to-end wall-clock time on the four expert pairs used for the prefill-time analysis, including prefill, decoding, and KVTRANS adaptation.

Table 6: End-to-end wall-clock time per question on four matched MMLU expert pairs. "Rel." is Ours divided by Text.

Expert pair	Text (s)	Ours (s)	Rel. (%)
CS–Math	138.89	103.42	74.46
Law–CS	97.14	68.81	70.84
Physics–Medicine	83.56	63.29	75.74
Economics–Philosophy	74.19	75.92	102.33
Mean	98.44	77.86	79.09

Average end-to-end time decreases from 98.44 s to 77.86 s, a 20.91% reduction, although the Economics–Philosophy pair shows a small increase.

F METHOD GENERALIZATION: BBH AND GAM

We evaluate additional settings beyond the main MMLU and WebSailor experiments. BBH examines receiver-side prefill-token reduction under another debate-style task, while HotPotQA evaluates quality recovery under the GAM collaboration protocol.

F.1 BBH: QWEN3-4B-INSTRUCT-2507

We evaluate eight directed collaborations among four BBH expert roles. Direct KV and Ours reuse the same message spans and therefore process identical receiver-side prefill-token counts.

Table 7: Receiver-side prefill tokens for eight BBH expert pairs using Qwen3-4B-Instruct-2507. Direct KV and Ours reduce the mean count by 34.10% relative to Text.

Expert pair	Text	Direct KV	Ours
Arithmetic → Boolean	1123.5	754.8	754.8
Arithmetic → Translation	1289.6	925.2	925.2
Arithmetic → Word sorting	1165.0	811.6	811.6
Boolean → Translation	921.2	720.8	720.8
Boolean → Word sorting	811.0	607.2	607.2
Translation → Arithmetic	1672.7	1219.5	1219.5
Translation → Word sorting	1598.0	1184.1	1184.1
Word sorting → Boolean	1908.6	689.5	689.5
Mean	1311.2	864.1	864.1

Across all eight pairs, cache reuse reduces receiver-side prefill tokens relative to Text, with a mean reduction of 34.10%.

F.2 GAM: HOTPOTQA COLLABORATION

We further evaluate Qwen3-4B-Instruct-2507 on HotPotQA under the GAM collaboration protocol. This setting differs from the main WebSailor evaluation and is therefore reported separately.

Table 8: Qwen3-4B-Instruct-2507 results on HotPotQA under the GAM collaboration protocol.

Method	HotPotQA F1 (%)	Tokens
Text	40.76	52,822.9
Direct KV	15.57	25,141.4
Ours	42.93	28,337.3

Ours recovers HotPotQA F1 from 15.57% under Direct KV to 42.93%, while using 28,337.3 tokens compared with 52,822.9 for Text. Relative to Direct KV, this recovery incurs additional token processing, but Ours still uses only 53.6% of the Text token count. This result provides additional evidence that cache adaptation can recover degradation from direct reuse under a distinct collaboration protocol while retaining substantial token savings.

G COMMUNICATION-SCALE ANALYSIS

G.1 EXPERT AND ROUND COUNTS

Expert-count experiments use Qwen3-4B-Instruct-2507 with one to four debating experts, excluding the final same-base judge. Each configuration uses two rounds: independent responses followed by one revision round. Communication is fully connected: each expert receives all other experts' initial responses before revision, and the judge receives all initial and revised responses. The science setting covers mathematics, computer science, physics, and medicine, while the humanities setting covers law, philosophy, economics, and diplomacy, with 50 questions per domain.

Round-count experiments use four ordered role pairs: Law–Mathematics, Philosophy–Diplomacy, Computer Science–Physics, and Medicine–Economics. Each pair uses the same 50 sender-domain questions across round counts.

Accuracy does not vary monotonically with expert count, indicating that adding more participants does not by itself guarantee better debate outcomes. Across interaction depth, Ours matches Direct

Table 9: Expert-count accuracy (%) under two-round fully connected debate. The one-expert setting has no cross-expert message to reuse; all other settings use the same communication protocol across methods.

	Science (200 questions)			Humanities (200 questions)		
Experts	Text	Direct KV	Ours	Text	Direct KV	Ours
1	83.5	–	–	66.0	–	–
2	92.0	89.0	89.0	71.5	73.0	71.5
3	86.0	90.0	90.0	72.0	68.5	73.0
4	86.0	91.0	91.0	72.0	71.0	72.5

Table 10: Accuracy (%) averaged over four role pairs as interaction depth increases. The first round contains independent responses; subsequent rounds add exchange and revision.

Rounds	Text	Direct KV	Ours
2	74.0	73.5	73.5
3	75.0	75.5	78.0
4	74.0	75.0	78.5
5	74.5	72.5	77.5

KV at two rounds and achieves higher accuracy at three to five rounds, with the largest gap appearing at five rounds (77.5% versus 72.5%).

Direct KV and Ours reuse the same message spans under this scaling protocol and therefore have identical receiver-side prefill-token counts. The multi-round experiments use the same receiver-side cost boundary as Table 1.

H Per-Model Diagnostics

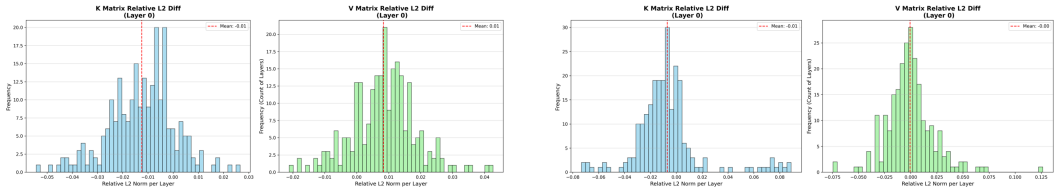
We provide complementary diagnostics for Qwen3-4B-Instruct-2507 and LLaMA3.2-3B-Instruct to examine how cross-context reuse affects attention selection and KV statistics across model architectures. Rather than attributing performance differences to a single architectural factor, these analyses illustrate model-dependent behavior under the same reuse framework.

Dynamic head selection. Figure 5 compares the head-selection scores used by Dynamic Attention Amplification. Both models exhibit substantial variation across layers and heads, supporting selective rather than uniform modulation. The detailed patterns differ across models, indicating that a fixed global head set would not capture the same receiver-instruction preferences across architectures.

KV magnitude mismatch. Figure 6 shows layer-wise KV norm statistics before and after L2-Rescaling. Directly reused states exhibit scale differences relative to receiver-local states in both models. Rescaling reduces these discrepancies by matching foreign-cache norms to receiver-local statistics, although the magnitude and layer-wise structure of the mismatch differ between models.

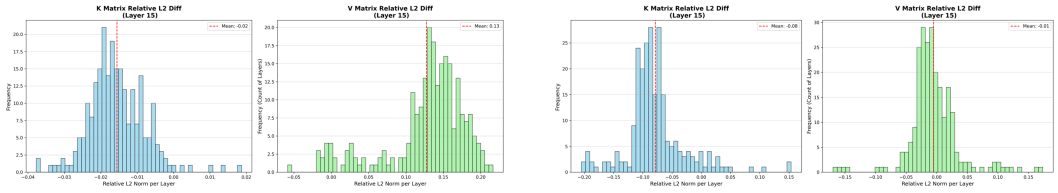
Attention behavior after reuse. Figure 7 compares layer-wise attention patterns under Text, Direct KV, and Ours. Direct reuse changes the receiver's attention allocation relative to Text, while the full method reduces part of this deviation. The remaining differences vary across layers and models, consistent with the heterogeneous task-level sensitivity observed in Table 5.

Cross-model observations. The two models show different sensitivity to direct cross-context reuse. For Qwen3-4B-Instruct-2507, Direct KV reduces MMLU accuracy from 79.57% under Text to 77.89%, while Ours recovers it to 80.07%. For LLaMA3.2-3B-Instruct, Direct KV reduces accuracy from 54.36% to 44.93%, and Ours recovers it to 51.68%. The diagnostics above likewise show model-dependent differences in head-selection patterns, KV magnitude mismatch, and attention redistribution. These results suggest that the severity and structure of cross-context mismatch



(a) Qwen3-4B-Instruct-2507. (b) LLaMA3.2-3B-Instruct.

Figure 5: Layer- and head-wise receiver-instruction preference scores used for dynamic head selection. Selection patterns vary across both decoding context and model architecture.



(a) Qwen3-4B-Instruct-2507. (b) LLaMA3.2-3B-Instruct.

Figure 6: Layer-wise KV norm statistics under cross-context reuse. L2-Rescaling calibrates foreign-cache magnitudes toward receiver-local statistics.

vary across models, while the same receiver-side alignment framework can provide recovery under both architectures.

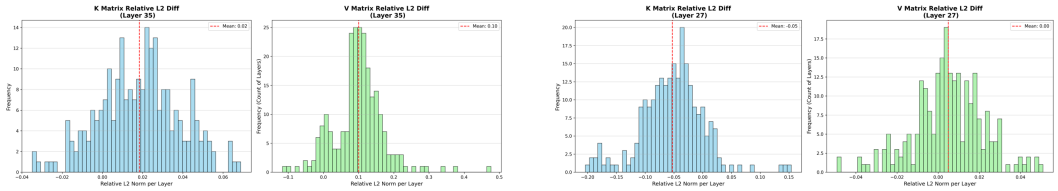
I Collaboration Experimental Details

Sampling protocol. For each BrowseComp language, we fix a subset of 100 questions and run three independent rollouts per question, yielding 300 rollouts per method and language. Text, Direct KV, and Ours use the same question subset and the same decoding and tool budgets. Pass@1 is computed over the 300 rollouts, while receiver-side prefill tokens are first averaged across rollouts for each question and then across questions. EN and ZH are reported separately because they use different question subsets.

Integration and serving. The collaboration pipeline treats the tool-summary generator as the sender and the research agent as the receiver. Reused summary states are therefore conditioned on the sender context rather than the receiver's full history. We use Tongyi-DeepResearch-30B-A3B served with SGLang at TP=8, temperature 0.85, top-p 0.95, a maximum of 40 LLM calls, and an 8192-token limit per request. The agent is forced to produce a final answer when the interaction budget is exhausted. Ours applies RoPE Alignment, L2-Rescaling, and Dynamic Attention Amplification. Both EN and ZH use GPT-5.5-based automated scoring. For ZH, we additionally apply manual equivalence correction to the scoring outputs.

Cache reuse protocol. Transferred states correspond only to the reusable summary span; receiver-side system instructions, message boundaries, and generation-boundary tokens are computed locally. After insertion, decoding continues under the receiver context. No trained adapter is used in any reported KVTrans result.

Evaluation protocol. Fixed-message diagnostics compare receiver processing under matched visible content, whereas free-running WebSailor rollouts may diverge in search actions, retrieved evidence, and trajectory length. Three rollouts per question reduce sensitivity to individual stochastic trajectories but do not eliminate sampling variability. We therefore report EN and ZH results separately and interpret failure modes only when they are supported by recorded trajectories.



(a) Qwen3-4B-Instruct-2507. (b) LLaMA3.2-3B-Instruct.

Figure 7: Layer-wise attention patterns under Text, Direct KV, and Ours. The full method reduces part of the deviation introduced by direct cross-context reuse.

J WebSailor Failure Analysis

Cross-context reuse inserts sender-generated KV states, conditioned on the sender's instructions, history, and token positions, into a receiver with a different context. Unlike prefix caching within a single receiver context, these states need not match those obtained by receiver-side re-encoding. We use controlled next-token probes and trajectory traces to examine how this mismatch affects long-horizon agentic search.

Controlled next-token divergence. To separate cache mismatch from free-running trajectory divergence, we perform a teacher-forced probe on 10 real WebSailor interaction nodes. For each node, we fix the same reference continuation and independently predict the next token at each of the first 16 positions. We compare full-text re-encoding, receiver-local cache reconstruction, and reuse of the sender-generated summary states. All paths therefore share the same generated prefix at every measured position.

Table 11: Teacher-forced next-token probe over 160 aligned positions. Top-1 agreement and JS divergence are measured relative to Text.

Cache source	Top-1 agree. (%)	Top-10 overlap	Mean JS
Receiver-local reconstruction	98.1	0.941	0.00258
Foreign summary KV	76.9	0.669	0.10548

Receiver-local reconstruction closely reproduces Text, whereas foreign summary states change the highest-probability token at 37 of 160 positions and produce substantially larger distributional divergence. The reference-token log-probability also decreases by 0.278 on average under foreign KV. Because the generated prefix is held fixed at every position, this discrepancy is present before free-running search trajectories diverge rather than being solely a consequence of an earlier sampled-token difference.

Trajectory amplification. This local discrepancy is particularly consequential in WebSailor because the agent operates in a closed loop. A changed token can alter whether the agent searches, the query it issues, the page it visits, or whether it terminates. The resulting observation then becomes part of the next history:

KV mismatch $\rightarrow$ action change $\rightarrow$ different evidence $\rightarrow$ new history $\rightarrow$ trajectory divergence.

Thus, even a local next-token discrepancy can affect subsequent observations and actions. Once different evidence has been retrieved, later cache correction cannot recover evidence that was never observed.

Partial recomputation. We further test whether recomputing only the message suffix near the generation boundary can restore the receiver-native distribution while preserving most KV reuse. Using the same 160 teacher-forced positions, we recompute the last 0, 32, or 128 reusable message tokens, or re-encode the entire reusable message body at the receiver.

Recomputing 32 or 128 tail tokens reduces the discrepancy but provides only partial recovery. Near-text behavior appears only after re-encoding the full message body, which largely removes the cor-

Table 12: Effect of receiver-side tail recomputation in the fixed-trajectory probe. Metrics are relative to Text.

Recomputed body	Top-1 agree. (%)	Mean JS
0 tokens	76.9	0.10548
32 tokens	80.6	0.08434
128 tokens	81.9	0.07475
Full body	97.5	0.00312

responding prefill reuse. This indicates that the sender-conditioned effect is not confined to a short region near the generation boundary.

Observed trajectory failures. Representative WebSailor traces include premature finalization, repeated searches, malformed or incomplete tool calls, repeated-tool loops, and budget exhaustion. Periodic or signal-triggered recomputation can suppress some repetition and excessively long trajectories, but does not consistently restore task quality. In particular, recomputation after an incorrect query or tool action only re-encodes the resulting history; it does not roll the trajectory back to the receiver-native branch. These observations motivate treating protocol failures and serving failures separately from semantic answer errors.

Implications. The controlled probes show that cross-context KV reuse can alter the receiver's conditional next-token distribution even when the visible continuation is held fixed. WebSailor can then amplify this local discrepancy through search and tool feedback. This helps explain why KVTRANS substantially recovers the degradation caused by Direct KV on BrowseComp but does not fully recover the Text baseline. The results suggest that reliable long-horizon reuse requires not only local cache alignment, but also mechanisms that preserve receiver-conditioned predictions or detect and recover from trajectory-level divergence.

K COST ACCOUNTING

MMLU prefill-token counts report receiver-side tokens actually processed during prefill, distinguishing them from total input length, cache-hit tokens, and protocol-level message counts. Prefill time is measured up to the first decoding step and includes cache adaptation and insertion before decoding, including RoPE Alignment and L2-Rescaling. End-to-end timing additionally includes decoding and Dynamic Attention Amplification. Detailed timing results are reported in Appendix E.2.

WebSailor token accounting. BrowseComp-EN/ZH results report mean receiver-side prefill tokens per question, averaged over three rollouts per method. Text counts are reconstructed using the deployment tokenizer and chat template, while KV counts use recorded request and transcript events. The accounting includes agent-request prefill and recorded transcript re-encoding, but excludes generated tokens, browser-summary generation input, scoring input, and KV-export sidecars. Accordingly, these values represent logical prefill cost within the defined receiver-side accounting boundary rather than end-to-end runtime.

L EXTENDED RELATED WORK

Multi-agent debate studies the exchange of answers and critiques (Du et al., 2023; Liang et al., 2024); our focus is the computation required to consume these messages. Reusing sender-generated message states does not eliminate sender decoding or reduce the number of communication edges.

Our distinction from anchor-based adaptation (Section 7) lies in the source of alignment information. DroidSpeak shares KV states across fine-tuned model variants (Liu et al., 2025), whereas we study same-model cross-context reuse.

Learned communication interfaces include Cache-to-Cache, ThoughtComm, and Interlat (Fu et al., 2026; Zheng et al., 2025; Du et al., 2026). LatentMAS uses layer-wise KV states as shared latent memory (Zou et al., 2026); in contrast, we adapt existing sender-generated KV states while retaining

standard message generation. Causal auditing further highlights the distinction between receiving cached states and preserving the intended message information (Cheng et al., 2026). Comparisons across these systems require matched protocols and cost definitions.