BEYOND TOKENS: MODEL COMMUNICATION THROUGH COMMON STATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models can communicate through internal representations, but transfer depends on how the receiver uses them. We introduce common-state conditioning (CS-QKV): a feed-forward interface maps sharer key–value features to a continuous message added before a frozen receiver's attention-input normalization. Native projections construct the conditioned queries, keys, and values, while the direct residual carries the current layer input. Communication occurs during prefix processing; the receiver then generates independently. Across 15 larger-to-smaller Qwen3 pairs, Full CS exceeds default Cache-to-Cache (C2C) in 13. At the largest size gap, 32B→0.6B, it raises four-task mean accuracy from 32.40% to 75.84%, gaining 43.44 points over the native receiver and 19.55 over C2C. Under the primary T0 protocol, single-run matched controls with sharer-KV-only inputs (No-H) gain 3.80 and 2.14 points over directly predicting Q/K/V increments. Receiving controls examine how native computations construct these updates; a separately trained first-order variant reaches 75.23% on 8B→0.6B. Selected cross-family applications and a locked SciQ test extend the evidence, while prompt and readout comparisons delimit relative advantages. The results support using native normalization and projections to parameterize cross-model attention updates.

1 INTRODUCTION

Language models increasingly collaborate by exchanging answers, explanations, and plans. Multiagent debate uses generated responses to refine an answer (Du et al., 2024), while AutoGen organizes such exchanges into conversational workflows (Wu et al., 2023). Text provides an interface between independently trained models, but producing an intermediate message requires autoregressive generation. Cache-to-Cache (C2C) instead projects and fuses key–value (KV) caches between frozen models (Fu et al., 2026); KV-cache alignment and Latent Cache Flow extend this direction through shared representations and compressed exchange (Dery et al., 2026; Rossi et al., 2026). Once internal features are available, a distinct question remains: *how should they enter the receiver's computation?*

An attention layer derives its queries, keys, and values from a common input through learned projections (Vaswani et al., 2017). In the pre-normalized receivers studied here, that input is a normalized hidden state (Zhang and Sennrich, 2019; Yang et al., 2025). Directly predicting attention increments requires an interface to learn updates compatible with the frozen attention and downstream layers. Learning inputs to an existing computation offers another route: continuous prompts condition frozen language models (Li and Liang, 2021; Lester et al., 2021), and model stitching learns connections between frozen representations (Bansal et al., 2021). These precedents motivate studying the receiving location, rather than treating external features as interchangeable attention operands.

We propose **common-state conditioning (CS)**. A feed-forward interface maps sharer KV features to a continuous message m at each eligible receiver-prefix position. The receiver constructs Q/K/V from $N(h + m)$ and retains h on the direct residual branch (Figure 1). The *common state* is this receiver-local projection input, not a universal coordinate system shared by the two models. Unlike fixed learned prompts, the message depends on the current request and a second model's features. Unlike an autoregressive latent reasoning sequence (Hao et al., 2024), communication here uses a feed-forward interface during prefix processing. Both backbones remain frozen; after the prefix, the receiver generates without further sharer or interface calls.

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.

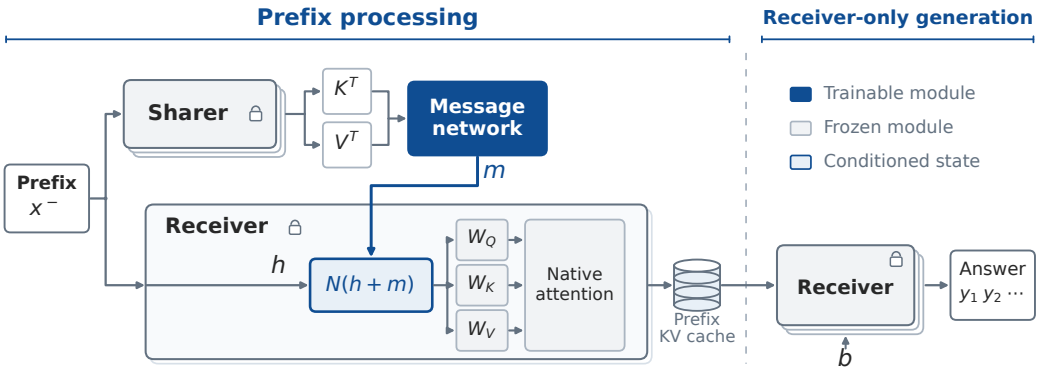


Figure 1: **Prefix-only common-state conditioning (No-H).** A frozen sharer processes prefix x^- once. Independent trainable message networks at receiver layers map cached K^T, V^T to m; frozen native projections construct Q/K/V from $N(h+m)$. The native-attention block includes Q/K head normalization, RoPE, attention, and output projection. The receiver's layerwise prefix KV cache supports boundary-token b processing and receiver-only generation. Dark blue marks trainable modules, gray frozen modules, and pale blue conditioned states; normalization parameters remain frozen. Direct residuals carrying h and feed-forward sublayers are omitted, as are layer/position indices and receiver superscripts. Full additionally supplies $N(h)$ to each message network.

Our contributions are a prefix-only common-state interface, a complete scale study of all 15 larger-to-smaller Qwen3 pairs, and matched CS–Direct comparisons on two anchor pairs. Additive, remapped, and first-order controls examine receiving choices; cross-family applications, a locked SciQ test, and execution measurements characterize scope and cost.

2 Motivating Study: From Sharer Strength to Receiver Gain

A stronger source of information does not, by itself, determine the quality of a model that receives it. Figure 2 examines the complete four-sharer series for a fixed Qwen3-1.7B receiver. Native sharer accuracy rises from 76.53% to 88.58%, whereas default C2C yields 58.68–62.47%, below the native receiver's 63.03%. No-H CS reaches 68.11–79.15% on the same pairs. Its advantage over C2C is 9.09 points with the smallest sharer and 16.68 with the largest.

This comparison motivates a receiving problem: useful sharer features must also be presented in a form that works with the receiver's frozen computation. The plot is a descriptive slice of Table 1, not a separate causal experiment. Section 6 subsequently fixes feature access and online insertion to compare receiving rules directly.

3 Latent Communication through a Common State

3.1 From sharer features to a receiving message

Following C2C terminology, let T be the *sharer* and S the *receiver*, both frozen. These are the representation-sharing and answer-generating roles, respectively. For receiver layer j and prefix position i, fixed correspondences $g(j)$ and $a(i)$ identify a sharer layer and position. The communicated feature is

$$z^T_{j,i} = \left[\text{flat}\, K^T_{g(j),a(i)}; \text{flat}\, V^T_{g(j),a(i)}\right]. \tag{1}$$

Here K^T, V^T are stored cache tensors; in Qwen3, cached keys have already undergone sharer key-head normalization and RoPE. Flattening joins head and channel dimensions. A layerwise message network $A_{\phi,j}$ maps these features to receiver hidden width d_S. With eligibility mask $M_{j,i}$,

$$m_{j,i} = M_{j,i} A_{\phi,j}(z_{j,i}), \qquad x^c_{j,i} = N_j(h_{j,i} + m_{j,i}). \tag{2}$$

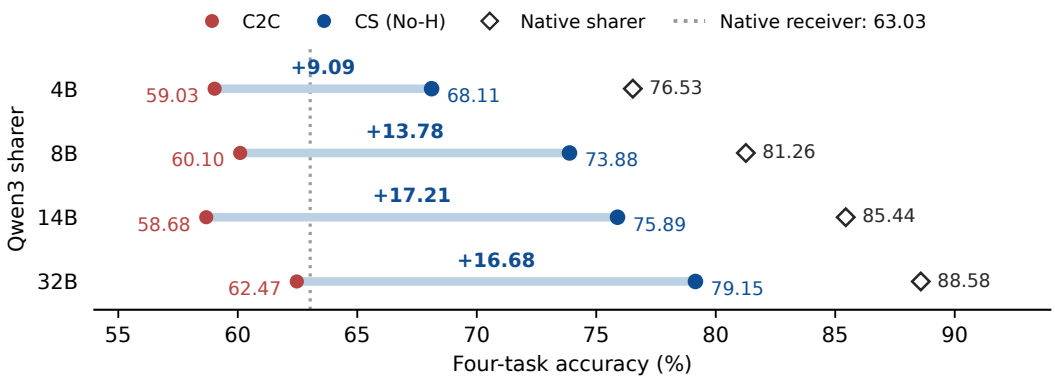


Figure 2: **Same receiver, different use of sharer information.** Each row fixes one sharer and the Qwen3-1.7B receiver. Colored points show C2C and No-H CS; labels above the connecting segments give CS–C2C differences in percentage points. Diamonds show native sharer scores, and the dashed line marks the native receiver. All values come from the complete scale study in Table 1.

$h_{j,i}$ is the current conditioned run's layer input, and N_j is its frozen attention-input normalization. In **No-H**, $z = z^T$; in **Full**, $z = [z^T; N_j(h)]$. Both occurrences of N_j use the same native module. No-H excludes receiver state from the message network, but its receiving update still depends on h.

Each message network contains an input projection, three residual two-linear MLP blocks with non-affine RMS normalization, GELU, and dropout 0.1, and a zero-initialized output projection. Layers have independent parameters. We train one interface per model pair and reuse it across tasks. The core No-H interfaces contain approximately 0.51B trainable parameters in total, executed only during prefix conditioning.

3.2 Letting native projections interpret the message

CS-QKV computes the raw attention operands from the common state:

$$q^c_{j,i} = W^S_{Q,j} x^c_{j,i}, \qquad k^c_{j,i} = W^S_{K,j} x^c_{j,i}, \qquad v^c_{j,i} = W^S_{V,j} x^c_{j,i}. \tag{3}$$

We omit projection biases; where present, the native modules apply them once. Qwen3 applies head normalization and RoPE to raw Q/K, followed by native attention and output projection (Yang et al., 2025). **CS-KV** instead uses $q = W_Q N(h)$ and conditions only K/V. This channel choice is independent of Full/No-H; its Q is native for the current h, which earlier communication may already have changed.

For CS-QKV, writing $\mathcal{A}_j$ for native attention including its projections, the block first computes

$$u_j = h_j + \mathcal{A}_j(N_j(h_j + m_j)), \tag{4}$$

then applies its native feed-forward sublayer, including its normalization and residual. The bypass carries the current h_j without adding m_j. Altered prefix attention propagates into later representations and caches; no extra message tokens are appended. CS-KV uses separate Q and K/V inputs and is not represented by this single-input shorthand.

One message, coordinated attention updates. Hold h fixed at the current conditioned-layer input and stack $W = [W_Q; W_K; W_V]$. The local raw-operand increment is

$$\Delta o = W \delta x, \qquad \delta x = N(h + m) - N(h). \tag{5}$$

Biases cancel in this difference. It is a local comparison at the same h, not the total difference from a separately executed native model. The interface selects a change in normalized input, and frozen projections determine its effect on the attention channels. This raw-space constraint does not imply semantic alignment between models.

For RMS normalization (Zhang and Sennrich, 2019), the message changes the normalized direction and effective scale of the raw operands. Thus, even with fixed sharer features, the No-H operand increment Δo depends on h. Subsequent Q/K head normalization further transforms these operands. Appendix A gives the decomposition; Section 6 compares alternative receiving rules.

3.3 CONDITION THE PREFIX, THEN LEAVE GENERATION TO THE RECEIVER

In the core Qwen3 protocol, we divide the known input into a communicated prefix x^- and a final boundary token b. The sharer prefills x^- once, without the reference answer. The receiver conditions the prefix, forming state $\mathcal{H}_{S,\phi}(x^-)$, then processes b and generates

$$p_\phi(y \mid x) = \prod_t p_S\big(y_t \mid y_{<t}, b, \mathcal{H}_{S,\phi}(x^-)\big)\,. \tag{6}$$

This state is the layerwise receiver KV cache. Prefix Q affects answers indirectly through later-layer prefix states and caches. Boundary and answer computation use native receiver modules without additional sharer or interface calls; their hidden states can still differ from an unconditioned run. Appendix D describes the historical cross-family execution variant.

We optimize the interface with cross-entropy on the final assistant response. Let $\mathcal{G}$ denote the prescribed logical training groups, and $\mathcal{I}_g$ their supervised answer-token positions. The implemented reduction is

$$\mathcal{L}(\phi) = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \frac{-1}{|\mathcal{I}_g|} \sum_{t \in \mathcal{I}_g} \log p_\phi(y_t \mid x, y_{<t}). \tag{7}$$

The group definition and weighting are retained across matched comparisons; this is not replaced with a global token mean. Gradients traverse the frozen receiver and its prefix cache to reach ϕ. Sharer features are read-only, but receiver caches are not detached. Zero-message checks and gradient-path checks are reported in Appendix A.

4 EXPERIMENTAL SETUP

Models and training. The principal study uses the post-trained Qwen3 models of sizes 0.6B, 1.7B, 4B, 8B, 14B, and 32B (Yang et al., 2025). We evaluate all 15 pairs with a larger sharer than receiver. Dense layers are aligned from the end; for 8B→0.6B and 8B→1.7B, sharer layers 8–35 map to receiver layers 0–27. The cross-family cases use their model-specific tokenizers and layer correspondences (Appendix D).

Interfaces are trained on 493,818 processed OpenHermes examples, with 4,989 validation examples (Teknium, 2023). The core budget is one epoch, 1,286 updates, global batch 384, and maximum length 2,048; the 0.6B anchor uses 132,635,762 answer-supervision tokens. AdamW uses learning rate 10^{-4}, weight decay 0.01, 128 warmup updates followed by linear decay, and gradient clipping at 1 (Loshchilov and Hutter, 2019). Unless otherwise noted, runs use seed 42 and final checkpoints. Appendix A records the recipe and execution variants.

Tasks and comparisons. The primary T0 protocol uses non-thinking, greedy generation capped at 64 tokens. We evaluate OpenBookQA (500), ARC-Challenge (1,150), MMLU-Redux (5,632), and C-Eval (12,342) (Mihaylov et al., 2018; Clark et al., 2018; Gema et al., 2025; Huang et al., 2023). ACC is the equal-weight mean of four task accuracies. One fixed, deterministic extractor scores every method using the response and options, without access to the reference answer. It prioritizes explicit final-answer declarations, then leading labels, followed by fixed option-matching fallbacks. Recognized placeholder or ambiguous-answer patterns and unresolved outputs count as incorrect; all questions remain in their task denominators. Appendix A specifies the rule and prompts. The English benchmarks and C-Eval contain different questions.

The full matrix compares default C2C, Full CS, and No-H CS with matched native references across separately trained configurations; physical batches and hardware vary across runs. The receiving controls match sharer inputs, online prefix insertion, and training budgets, with approximately matched effective parameter counts. Each controlled configuration uses one training run unless otherwise noted. The four primary tasks were used during development. SciQ is a separate confirmation whose protocol was locked before inference, without task-specific retraining.

5 MAIN RESULTS: QWEN3 MODEL COMMUNICATION

Small receivers benefit across the complete study. Table 1 reports all 15 larger-to-smaller pairs. Full CS exceeds C2C in 13 pairs and the native receiver in 12. No-H exceeds each reference in 12

Table 1: **Complete Qwen3 communication study.** All models are Qwen3. The first four scores are No-H task accuracies; the remaining scores are macro averages. Parentheses give percentage-point differences from C2C. Receiver and Sharer are native references; bold marks the best of C2C, Full, and No-H for each pair.

	No-H task accuracy (%)				Four-task macro accuracy (%)				
Receiver	OBQA	ARC-C	MMLU-R	C-Eval	Receiver	C2C	Full (Δ)	No-H (Δ)	Sharer
Sharer: Qwen3-1.7B									
0.6B	49.40	56.78	39.60	37.34	32.40	47.86	**59.51** (+11.65)	45.78 (-2.08)	63.03
Sharer: Qwen3-4B									
0.6B	72.80	80.78	63.76	58.45	32.40	52.38	60.29 (+7.91)	**68.95** (+16.57)	76.53
1.7B	70.80	80.00	63.83	57.82	63.03	59.03	66.70 (+7.67)	**68.11** (+9.09)	76.53
Sharer: Qwen3-8B									
0.6B	80.80	85.04	68.11	60.63	32.40	51.31	69.21 (+17.90)	**73.65** (+22.33)	81.26
1.7B	78.00	85.13	68.25	64.15	63.03	60.10	73.13 (+13.03)	**73.88** (+13.78)	81.26
4B	76.00	85.13	69.62	67.75	76.53	**76.64**	75.75 (-0.90)	74.63 (-2.02)	81.26
Sharer: Qwen3-14B									
0.6B	84.20	88.26	70.92	65.14	32.40	53.31	75.94 (+22.62)	**77.13** (+23.81)	85.44
1.7B	80.20	87.39	70.19	65.78	63.03	58.68	74.87 (+16.19)	**75.89** (+17.21)	85.44
4B	84.40	88.35	73.15	69.37	76.53	75.80	78.48 (+2.68)	**78.82** (+3.02)	85.44
8B	87.20	89.48	73.28	67.42	81.26	79.29	**79.40** (+0.11)	79.34 (+0.05)	85.44
Sharer: Qwen3-32B									
0.6B	73.00	82.26	66.80	64.44	32.40	56.29	**75.84** (+19.55)	71.62 (+15.34)	88.58
1.7B	85.20	88.87	73.76	68.77	63.03	62.47	**79.19** (+16.72)	79.15 (+16.68)	88.58
4B	91.00	92.61	77.93	75.34	76.53	75.40	83.87 (+8.46)	**84.22** (+8.82)	88.58
8B	88.00	92.17	77.63	73.26	81.26	79.56	82.23 (+2.66)	**82.77** (+3.20)	88.58
14B	90.40	91.83	77.75	72.87	85.44	**84.66**	82.81 (-1.85)	83.21 (-1.45)	88.58

pairs and both simultaneously in 11. Both variants exceed the native receiver in all nine pairs with a 0.6B or 1.7B receiver. Full also exceeds C2C in all nine; No-H does so in eight, except 1.7B→0.6B. At 8B→0.6B, No-H CS-QKV reaches 73.65%, compared with 32.40% for the receiver and 51.31% for C2C, with gains across the English benchmarks and C-Eval.

More capable sharers can help, but their value depends on receiving. For the fixed 1.7B receiver, No-H rises from 68.11% to 79.15% as the sharer grows from 4B to 32B (Figure 2). The 0.6B series is not monotonic: No-H reaches 77.13% with 14B and falls to 71.62% with 32B, while Full remains near 76% at both sizes. Sharer capability alone therefore does not predict receiver gain.

Benefits extend beyond the smallest receiver, but not to every pair. With a 32B sharer, No-H reaches 84.22% on a 4B receiver, above its native 76.53% and C2C's 75.40%. Both CS variants fall below the native receiver at 8B→4B, 14B→8B, and 32B→14B. This matrix compares complete communication systems; the following matched controls examine receiving rules without attributing the entire CS–C2C gap to a single design choice.

6 Which Receiving Choices Matter?

6.1 Matched sharer access and online computation

Unless otherwise specified, these comparisons use No-H message inputs. Direct-QKV predicts raw increments from the same sharer KV features,

$$o^{\mathrm{Direct}} = o^0(h) + D_\phi(z_T), \qquad o^0(h) = WN(h), \tag{8}$$

and applies the same subsequent head normalization and position operations. Like CS, it recomputes current-layer prefix attention and propagates the altered state forward. This avoids conflating the receiving rule with whether communication occurs during prefill or after a completed cache. Internal widths are adjusted for parameter matching rather than held equal regardless of output dimension.

Table 2: **Matched receiving rules with sharer-KV-only message inputs (No-H).** CS and Direct use online prefix injection, the same training budget, and approximately matched interface parameters. Bold compares the two interfaces; Δ is their four-task ACC difference (CS minus Direct).

Method	Params (M)	OBQA	ARC-C	MMLU-R	C-Eval	ACC	Δ
Qwen3-8B → Qwen3-0.6B							
Native receiver	—	38.00	38.52	32.95	20.13	32.40	—
Direct-QKV (No-H)	507.33	76.80	80.96	64.26	57.39	69.85	—
CS-QKV (No-H)	509.39	**80.80**	**85.04**	**68.11**	**60.63**	**73.65**	+3.80
Native sharer	—	85.00	90.96	75.62	73.46	81.26	—
Qwen3-8B → Qwen3-1.7B							
Native receiver	—	64.80	75.30	58.06	53.95	63.03	—
Direct-QKV (No-H)	512.20	75.00	82.52	67.31	62.15	71.74	—
CS-QKV (No-H)	513.52	**78.00**	**85.13**	**68.25**	**64.15**	**73.88**	+2.14
Native sharer	—	85.00	90.96	75.62	73.46	81.26	—

Table 3: **Message input, receiving rule, and channel choice.** Complete $2 \times 2 \times 2$ comparison for an 8B sharer and 0.6B receiver. Values are four-task macro accuracy. Full (KV + receiver hidden state) is retrained under the matched execution settings (Appendix A).

Rule	KV	QKV
KV + receiver hidden state		
Direct	62.70	67.83
CS	**65.67**	**68.92**

Rule	KV	QKV
KV only (No-H)		
Direct	69.23	69.85
CS	**74.33**	**73.65**

CS improves T0 accuracy in the primary single-run comparisons. Gains are 3.80 and 2.14 points (Table 2), with all four tasks agreeing. The 95% paired intervals, [2.92, 4.70] and [1.20, 3.07], quantify question uncertainty at fixed checkpoints (Appendix C.3). Jointly unique-answer questions contribute 3.12 of the 3.80 points under full task denominators; this partition does not isolate an extraction effect. Across three independent training seeds on 8B→0.6B, the paired CS–Direct gain is $+5.16 \pm 1.43$ percentage points (mean $\pm$ sample standard deviation; Appendix C.4). Relative advantages also depend on the prompt and readout protocol: alternative instructions reverse the 0.6B candidate-readout ordering on the diagnostic panel (Appendix A.5).

Input and channel choices are separate from the receiving rule. Table 3 shows that No-H CS exceeds No-H Direct for both KV and QKV, by 5.10 and 3.80 points. No-H CS-KV reaches 74.33%, 0.68 points above No-H CS-QKV. With H-conditioned messages on a 1.7B receiver, however, CS reaches 71.03% and Direct 72.34%. Receiving rule, message input, and channel selection interact: neither adding Q conditioning nor reading receiver state is uniformly beneficial.

Sharer features outperform the tested receiver-only controls. A 512.35M-parameter H-only interface reaches 36.54% on 0.6B, versus 73.65% for No-H CS-QKV. Historical same-data SFT and self-sharer CS reach 34.19% and 38.35%. Following the C2C-C architecture described by Fu et al. (2026), we implement Enhanced cache and evaluate it under our protocol. With 705.79M parameters, it reaches 49.27% and 62.22% on the two receivers (Appendix C).

6.2 Native maps, first-order reception, and the full update

An additive native map does not recover the result. Native-additive receiving uses $WN(h) + Wm$, retaining the frozen projections and direct residual but adding the message after normalization. Its message network shares the CS architecture, initial weights, 509,386,752 trainable parameters, data order, and update budget. Native-additive receiving reaches 66.62%, compared with 73.65% for CS; fixed candidate readout gives 69.14% and 73.54%. Native projection alone is therefore insufficient in this recipe. This comparison does not isolate normalization from effective message scale or optimization.

Table 4: **Which receiving constructions support transfer?** No-H, 8B→0.6B. ACC averages four tasks (19,624 questions); bold marks the highest point estimate. $m = A_\phi(z_T)$; $\delta x = N(h+m) - N(h)$; $r(h)$ is the RMS denominator (Appendix C.5). Candidate results: Tables 12 and 16.

Rule	Raw operands	Params (M)	ACC
Direct-QKV (No-H)	$WN(h) + D_\phi(z_T)$	507.33	69.85
Native-additive	$WN(h) + Wm$	509.39	66.62
Remapped CS	$WN(h) + RW\delta x$	509.39	63.57
CS-QKV (No-H)	$WN(h+m)$	509.39	73.65
Lin-CS	$WN(h) + WJ_N(h)m$	509.39	75.23
Scale-only	$WN(h) + WD_\gamma m/r(h)$	509.39	**75.75**

Table 5: **Cross-family transfer from Mistral to Gemma.** Task columns show CS-Full. Receiver, CS-Full, and Sharer at right are matched macro averages; Δ_R compares CS-Full with the native receiver. These are applicability cases using Full interfaces, not matched No-H receiving ablations.

Receiver	OBQA	ARC-C	MMLU-R	C-Eval	Receiver	CS-Full	Sharer	Δ_R
Sharer: Mistral-Small-24B								
Gemma-3-270M	60.20	61.91	43.47	34.39	18.82	**49.99**	79.90	+31.17
Gemma-3-1B	44.00	52.87	40.48	27.53	29.01	**41.22**	79.90	+12.21
Gemma-3-4B	63.20	70.70	55.49	40.92	53.72	**57.57**	79.90	+3.85

First-order receiving retains strong transfer. Lin-CS is trained from the same initialization with

$$o^{\mathrm{Lin}} = WN(h) + WJ_N(h)m, \qquad m = A_\phi(z_T), \tag{9}$$

where $J_N(h)m$ is the normalization's Jacobian–vector product. The update agrees with CS to first order at zero message; the separately trained message network remains nonlinear. Lin-CS reaches 75.23% versus 73.65% for No-H CS-QKV, improving all four tasks; candidate accuracy is 75.10% versus 73.54% on the 1,024-question panel. A separately trained Scale-only rule omits the Jacobian's radial correction and reaches 75.75%. For fixed h, m, its receiving input is a positive scalar multiple of $N(h+m)$ (Appendix C.5). Strong transfer thus does not require the exact finite nonlinear update or the full first-order correction in this configuration. These single-pair controls do not measure approximation error along a trained CS trajectory; scale and timing studies retain their CS checkpoints.

Native receiving coordinates matter in the tested control. Remapped CS retains $\delta x = N(h+m) - N(h)$ but uses $o^0(h) + RW\delta x$, where R is a fixed signed permutation per raw head, layer, and channel. Only the message increment changes coordinates; native residual and downstream weights remain fixed. For a fixed δx, the map retains rank, singular spectrum, and per-head increment norms, but not attention interactions or loss geometry. Training from the anchor initialization yields 63.57%, below CS on all four tasks and candidate readout (Appendix C).

These rules all retain native projections, but construct different message increments. With fixed sharer features, CS, Lin-CS, and Scale-only modulate the increment through the current h; No-H Direct and native-additive do not. The results support these native receiving constructions in the tested setting without isolating state dependence, scale, or optimization as a unique cause.

7 Transfer and Execution Scope

Selected other backbones also use the messages. Table 5 reports a fixed Mistral sharer with three Gemma receivers. Every displayed pair improves its matched native macro score. Each pair has its own trained Full interface; these selected applications use historical execution paths rather than matched No-H controls. The 270M Gemma interface exceeds its backbone size. Appendix D specifies model versions, token/layer alignment, and execution differences.

Table 6: **Transfer to locked SciQ.** Accuracy on 1,000 test questions without support passages. Mean is the predeclared average over all three templates. The common native sharer is repeated for readability.

Method	T0	T1	T2	Mean
Qwen3-8B → Qwen3-0.6B				
Receiver	70.10	74.60	78.40	74.37
CS	94.80	94.90	94.20	**94.63**
Direct	93.90	94.80	94.50	94.40
Sharer	95.40	95.00	95.10	95.17

Method	T0	T1	T2	Mean
Qwen3-8B → Qwen3-1.7B				
Receiver	88.00	88.10	87.30	87.80
CS	94.20	95.00	93.90	**94.37**
Direct	93.70	94.30	94.30	94.10
Sharer	95.40	95.00	95.10	95.17

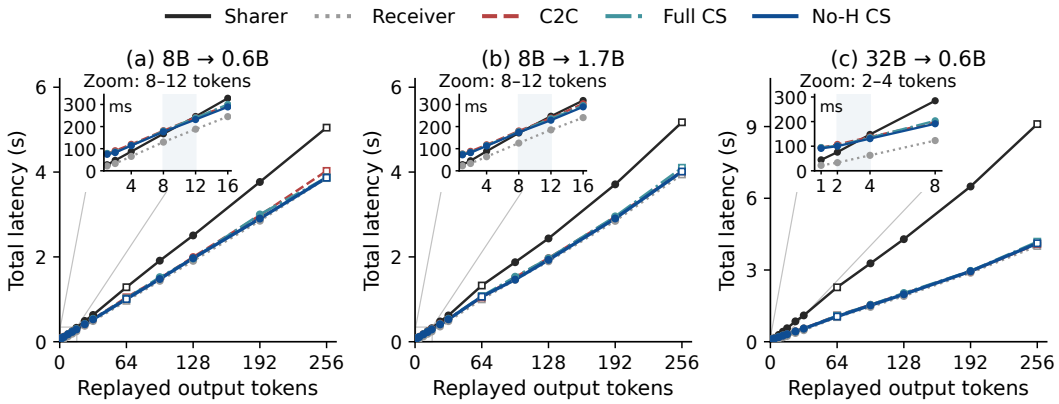


Figure 3: **Prefix preparation is amortized over receiver decoding.** Thirteen measured lengths on H200/HF; insets magnify the early crossover with token and millisecond coordinates. Shading brackets adjacent measured means, not a fitted exact crossing. All methods use the same within-pair token trajectory; replay does not measure its answer quality.

A locked task confirms transfer without separating the interfaces. SciQ test (Welbl et al., 2017) is evaluated without support passages or task-specific training, after locking options, checkpoints, and the three-template mean. No-H CS improves native 0.6B and 1.7B by 20.27 and 6.57 points, finishing within 0.80 points of the sharer (Table 6). CS–Direct differences are only 0.23 and 0.27 points, with paired intervals spanning zero. These results support transfer without establishing superiority over Direct. Appendix D.3 reports the common one-question overlap sensitivity check.

Transfer persists across the tested prompt templates. On the fixed 1,024-question panel, CS exceeds native 0.6B and 1.7B under all three templates, by 28.91–39.06 and 10.25–12.79 points, respectively (Table 19). CS also exceeds Direct in each of these ACC comparisons, with T2 gains of 2.93 and 0.68 points. Both model prefixes are recomputed when the instruction changes, so these comparisons measure system-level prompt sensitivity. The separate likelihood diagnostic in Appendix A.5 examines how method ordering changes under fixed-prefix scoring.

7.1 Prefix cost and receiver-only continuation

We measure 13 output lengths from 1 to 256 tokens using HF/SDPA, BF16, batch one, and one H200 per request. The same 32 short prompts and within-pair token trajectories are replayed for three rounds. Loading, tokenization, and scoring are excluded. Appendix E gives the measurement protocol and the complete plotted values.

No-H becomes faster than the native sharer between the measured 8- and 12-token points for both 8B pairs, and between 2 and 4 tokens for 32B→0.6B. C2C and Full share these brackets. At 256 tokens, No-H takes 3.867, 4.012, and 4.118 seconds: speedups of 1.305×, 1.289×, and 2.211× over

the native sharer, computed as ratios of mean latencies. C2C obtains similar decoding benefits; No-H is slightly slower than C2C at 256 tokens for 32B→0.6B.

Natural short answers need not accelerate: a separate 8B→0.6B sample records 179.78 ms for No-H versus 152.97 ms for native 8B, with unequal output lengths. Short-text communication reaches 77.44% on 0.6B and 76.74% on 1.7B; the sharer's actual messages average only 6.60–6.66 tokens (Appendix D.5). Task accuracy and replay latency remain separate measurements, not a measured joint quality–latency advantage.

8 Related Work

Communication through text and internal representations. Multiagent debate refines answers through text (Du et al., 2024), and AutoGen organizes agent conversations (Wu et al., 2023). Coconut uses hidden states for within-model reasoning (Hao et al., 2024); model stitching connects frozen representations (Bansal et al., 2021). Activation communication combines intermediate activations before continuing the receiver's forward pass (Ramesh and Li, 2025). C2C learns KV fusion and gating (Fu et al., 2026); KV-cache alignment uses shared representations (Dery et al., 2026); Latent Cache Flow translates and compresses KV across contexts (Rossi et al., 2026). CS injects sharer-derived messages before attention-input normalization, preserving the direct residual.

Continuous conditioning and context compression. Adapters insert trainable modules (Houlsby et al., 2019), and LoRA parameterizes low-rank weight updates (Hu et al., 2022). Prefix and prompt tuning condition frozen models with learned continuous vectors (Li and Liang, 2021; Lester et al., 2021). Gist tokens summarize reusable prompts (Mu et al., 2023), AutoCompressors create soft context summaries (Chevalier et al., 2023), and the In-context Autoencoder learns memory slots through reconstruction and language modeling (Ge et al., 2024). CS derives messages from another model's KV features and injects them at existing receiver-prefix positions before normalization, keeping receiver projections frozen and appending no message tokens. Its objective is cross-model receiving.

Knowledge transfer and cooperative decoding. Knowledge distillation transfers behavior through teacher distributions (Hinton et al., 2015) or generated sequences (Kim and Rush, 2016); rationale supervision trains smaller models to reproduce explanations (Hsieh et al., 2023). CS instead uses fresh sharer features per request while keeping both backbones frozen. Speculative decoding combines fast drafts with target-model verification (Leviathan et al., 2023; Chen et al., 2023); EAGLE predicts features to construct drafts that the target still checks (Li et al., 2024). CS ends communication after prefix processing and generates with the conditioned receiver, without target verification.

9 Conclusion

CS-QKV provides a prefix-only interface for communication between frozen language models. It maps sharer KV features to messages added before the receiver's attention-input normalization, allowing native projections to construct coordinated Q/K/V updates while the direct residual retains the current layer input. The receiver then generates without further sharer or interface calls. Across the complete Qwen3 scale study, this design substantially improves small receivers; the primary T0 matched comparisons also show gains over direct increment prediction. Selected cross-family applications and the locked SciQ test extend the evidence for transfer to other backbones and tasks.

The receiving controls show that strong transfer need not require every component of the original construction: query conditioning and the exact finite nonlinear update are not necessary in the tested configurations. These findings motivate studying receiving rules that use native computations to interpret external messages. Relative advantages depend on the input and readout protocol, while fixed-workload timing shows how prefix preparation is amortized over decoding. The central implication is that model communication should consider both the information supplied by the sharer and the computation through which the receiver uses it.

AI USE STATEMENT

AI tools were used to aid and polish writing, including improving language, wording, and organization, and to support research ideation and execution, including experiment-planning discussions, code development, and analysis and interpretation of experimental results. The authors are responsible for the final content and its verification.

ETHICS STATEMENT

This work studies communication between pretrained language models using existing datasets and benchmarks. Such systems may inherit biases and generate inaccurate or harmful outputs. Improved benchmark performance should not be interpreted as evidence of safety or fairness. Downstream use should respect applicable model and dataset usage terms and include safeguards and evaluation appropriate to the intended application.

REPRODUCIBILITY STATEMENT

Sections 3–4 specify the receiving computation and core experimental setting. Appendix A provides the architecture, optimization settings, execution algorithm, exact prompt instructions, answer-extraction rule, and candidate-readout definition. Primary comparisons use fixed task instructions and one deterministic answer extractor. Appendices B and C report complete task-level results; Appendix C.3 specifies paired intervals at fixed checkpoints. Appendix D records cross-family alignment and execution differences. Appendix D.3 separates the locked transfer test from prompt diagnostics, and Appendix E specifies timing boundaries and reports the plotted measurements. Core training comparisons use final checkpoints and seed 42 unless otherwise noted.

REFERENCES

Yamini Bansal, Preetum Nakkiran, and Boaz Barak. Revisiting model stitching to compare neural representations. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/01ded4259d101feb739b06c399e9cd9c-Abstract.html.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023. URL https://arxiv.org/abs/2302.01318.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, 2023. doi: 10.18653/v1/2023.emnlp-main.232. URL https://aclanthology.org/2023.emnlp-main.232/.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Lucio M. Dery, Zohar Yahav, Henry Prior, Qixuan Feng, Jiajun Shen, and Arthur Szlam. Latent space communication via K-V cache alignment. *arXiv preprint arXiv:2601.06123*, 2026. URL https://arxiv.org/abs/2601.06123.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 11733–11763, 2024. URL https://proceedings.mlr.press/v235/du24e.html.

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://openreview.net/forum?id=LeatkxrBCi.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/0b276510ec2d3f6613a8b60c41ff0438-Abstract-Conference.html.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with MMLU? In *Proceedings of NAACL-HLT (Volume 1: Long Papers)*, pages 5069–5096, 2025. doi: 10.18653/v1/2025.naacl-long.262. URL https://aclanthology.org/2025.naacl-long.262/.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024. URL https://arxiv.org/abs/2412.06769.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799, 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, 2023. doi: 10.18653/v1/2023.findings-acl.507. URL https://aclanthology.org/2023.findings-acl.507/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-Eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/c6ec1844bec96d6d32ae95ae694e23d8-Abstract-Datasets_and_Benchmarks.html.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, 2016. doi: 10.18653/v1/D16-1139. URL https://aclanthology.org/D16-1139/.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, 2021. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL-IJCNLP*, pages 4582–4597, 2021. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 28935–28948, 2024. URL https://proceedings.mlr.press/v235/li24bt.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of EMNLP*, pages 2381–2391, 2018. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems*, volume 36, pages 19327–19352, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/3d77c6dcc7f143aa2154e7f4d5e22d68-Abstract.html.

Vignav Ramesh and Kenneth Li. Communicating activations between language model agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 51094–51116. PMLR, 2025. URL https://proceedings.mlr.press/v267/ramesh25a.html.

Maximillian Rossi, Prajwal Raghunath, and Eugene Wu. Latent cache flow: Model-to-model communication without text, 2026. URL https://arxiv.org/abs/2605.22863.

Teknium. OpenHermes 2.5. Dataset, 2023. URL https://huggingface.co/datasets/teknium/OpenHermes-2.5.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, 2017. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413/.

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 conversation. *arXiv preprint arXiv:2308.08155*, 2023. URL https://arxiv.org/abs/2308.08155.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://arxiv.org/abs/1910.07467.

A Implementation and Experimental Setting

A.1 Model roles and message networks

The *sharer* provides prefix representations and the *receiver* generates the answer. Both are released post-trained language models whose backbone parameters remain frozen. Full messages additionally use the normalized receiver hidden state; No-H messages use only the sharer's K/V. This input distinction is independent of conditioning KV or QKV.

Each receiving layer has its own feed-forward message network: a bias-free input projection, three residual two-linear MLP blocks, and a bias-free output projection to the receiver hidden width. The blocks use non-affine RMS normalization, bias-free linear maps, GELU, and dropout 0.1. The output projection is initialized to zero. The No-H hidden widths are 1,504 and 1,440 for the 0.6B and 1.7B receivers; matched Direct widths are 1,304 and 1,312. The resulting effective trainable parameter counts appear in Table 2.

Let L be the number of receiving layers, w the message-network width, d_T^{KV} the flattened width of one sharer K or V tensor, and d_S the receiver hidden width. The total CS interface sizes are

$$P_{\text{No-H}} = L\big[(2d_T^{\mathrm{KV}} + d_S)w + 6w^2\big], \tag{10}$$

$$P_{\text{Full}} = L\big[(2d_T^{\mathrm{KV}} + 2d_S)w + 6w^2\big]. \tag{11}$$

For the two core No-H interfaces these give 509,386,752 and 513,515,520 parameters. These totals include the separate networks at all receiving layers; the receiver's native normalization and projection weights remain frozen.

For the two 8B-sharer comparisons, zero-based sharer layers 8–35 map to receiver layers 0–27. Sharer features concatenate the selected cached K/V head-channel dimensions. In Qwen3, cached sharer K has already undergone its key-head normalization and RoPE. The interface maps these features to a receiver-width message. The 0.6B receiver's raw Q/K/V widths are 2,048/1,024/1,024, giving a stacked projection of shape $4,096 \times 1,024$. Receiver updates precede its query/key head normalization, RoPE, and grouped-query KV repetition. Projection biases, where present, are applied once in the native path; incremental projections do not add them again.

The last receiver layer's prefix query has no path to a later layer's cache or the answer objective under this handoff protocol. The matched Direct control therefore omits its unused final-prefix Q output head. This avoids counting disconnected output parameters in the budget comparison.

Table 7: Core interface training and evaluation settings.

Frozen-backbone interface training	
Training / validation	493,818 / 4,989 processed OpenHermes examples
Supervision	Final assistant response and EOS; 132,635,762 supervised tokens for the 0.6B core anchor
Budget	One epoch; 1,286 updates; global batch 384; maximum length 2,048
Optimizer	AdamW; learning rate 10^{-4}; $(\beta_1, \beta_2) = (0.9, 0.999)$; $\epsilon = 10^{-8}$; weight decay 0.01
Schedule	128 warmup updates, then linear decay; gradient clipping at 1.0
Model selection	Final checkpoint; training seed 42 unless otherwise noted
Primary generated-answer evaluation	
Execution	Hugging Face Transformers; BF16; SDPA; non-thinking
Decoding	Greedy; maximum 64 new tokens
Tasks	OpenBookQA 500; ARC-C 1,150; MMLU-Redux 5,632; C-Eval 12,342
Aggregation	Task accuracy, followed by an equal-weight four-task mean
Scoring	One fixed, deterministic answer extractor shared by all methods (Appendix A.4)

A.2 Prefix execution and training objective

For the core Qwen3 path, render the prompt with its native chat template and generation prompt. The final token is the generation boundary b; all preceding tokens form x^-. The common tokenizer gives $a(i) = i$. At each receiving layer, $M_{j,i} = 1$ at every nonpadding position in x^-, including

its chat-template tokens, and zero elsewhere. Training preflight checks enforce this same prefix selection. The receiver builds its conditioned prefix cache, then processes b and all answer tokens through its native continuation path. Neither the boundary nor answer tokens are sent to the sharer or receive messages. Padding and prompt tokens do not contribute to the answer objective. Backbone parameters are frozen, but autograd remains connected through receiver states and prefix caches. The historical cross-family execution is specified separately in Appendix D.2.

The loss in Equation 7 averages answer-token losses within logical groups and then averages groups. Matched runs retain that reduction and data order. At both the 0.6B and 1.7B receiving scales, the CS–Direct controls use four-example logical groups and physical batch caps of 8/12/16 for long/medium/short sequences; changing physical packing does not redefine the logical groups. Full interfaces are retrained under these shared execution settings, yielding 68.92/71.03% for the two receiving scales. The scale matrix pools separate training runs.

Algorithm 1 Core Qwen3 prefix-only common-state communication

Require: Frozen sharer T, receiver S; message networks A_ϕ; prefix x^-, boundary b; layer/position correspondences; masks M_j; Full or No-H input; KV or QKV channels

$Z_T \leftarrow \mathrm{Prefill}_T(x^-)$ ▷ No answer tokens
Initialize the receiver prefix from its native embeddings
for receiver layers $j = 0, \ldots, L_S - 1$ **do**
  $x_j^0 \leftarrow N_j(h_j)$; select and align cached sharer K/V features z_j^T
  if Full input **then**
    $z_j \leftarrow [z_j^T; x_j^0]$
  else
    $z_j \leftarrow z_j^T$
  end if
  $m_j \leftarrow M_j \odot A_{\phi,j}(z_j)$; $x_j^c \leftarrow N_j(h_j + m_j)$
  $k_j, v_j \leftarrow$ native K/V projections of x_j^c
  if QKV channels **then**
    $q_j \leftarrow$ native Q projection of x_j^c
  else
    $q_j \leftarrow$ native Q projection of x_j^0
  end if
  Apply native Q/K head normalization and RoPE
  Compute causal prefix attention; retain h_j on the direct residual branch
  Save receiver K/V and apply the native feed-forward sublayer
end for
Process boundary b using the receiver and conditioned prefix state
Generate the answer with the receiver only
During training, backpropagate answer loss through receiver caches to ϕ.

A.3 RMS DECOMPOSITION AND FIRST-ORDER RECEIVING

Here h is held fixed at the current conditioned-layer input. With stacked raw projection bias b_o, $o^0 = WN(h) + b_o$ and $o^{\mathrm{CS}} = WN(h+m) + b_o$, so $\Delta o = W[N(h+m) - N(h)]$. Bias cancels in this local difference. It is not the end-to-end difference between independently executed conditioned and native models: earlier communication can already have changed h.

Writing $N(h) = D_\gamma h / r(h)$ and $r(h) = \sqrt{\|h\|^2/d + \epsilon}$ gives the exact bias-free identity in real arithmetic

$$WN(h+m) = \frac{r(h)}{r(h+m)} WN(h) + \frac{WD_\gamma m}{r(h+m)}. \tag{12}$$

The message influences both the raw native-operand scale and the injected term. Qwen3 subsequently applies per-head normalization and RoPE to Q/K, while V has neither operation. Thus this identity does not imply the same scaling of the final attention logits, and the raw increment's column-space constraint need not survive nonlinear head normalization. The empirical controls compare learned interfaces, not separate summands of this identity.

For the linearized receiver,

$$J_N(h)m = D_\gamma \left[\frac{m}{r(h)} - \frac{h(h^\top m)}{d\, r(h)^3} \right]. \tag{13}$$

For fixed h and $\epsilon > 0$, $N(h+m) = N(h) + J_N(h)m + O(\|m\|^2)$ near zero message. Lin-CS preserves gradients through both h and m. Squared norms, inner products, and reciprocal factors are computed in FP32, then cast to the native normalization output type before projection. All 28 receiving layers use normalization epsilon 10^{-6}. The message network remains nonlinear and is trained separately; only the local receiving update is linearized. Its performance is not a measurement of Taylor approximation error along a trained CS trajectory.

Native-additive receiving instead uses $N(h) + m$. Remapped CS uses $o^0(h) + RW\delta x$, where $\delta x = N(h+m) - N(h)$ and R is a fixed independent signed permutation within each raw head, layer, and channel (mapping seed 1701). The final-prefix Q map is the identity. Only the message increment is transformed: the original residual, feed-forward layers, and attention output projection remain unchanged. Because $R^\top R = I$, RW and W share a singular spectrum, but they do not preserve the message's interaction with the native operands. All three controls start from the same initial message weights as No-H CS and use the same training budget.

A.4 Generated-answer evaluation and exact instructions

All generated-answer ACCs use the same fixed deterministic extractor. It reads only the response and labeled answer options; the reference label is used afterwards to score the extracted prediction. Empty or unresolved predictions remain incorrect in the full task denominator. The four primary tasks participated in development; the SciQ protocol was locked before its inference.

The extractor first applies Unicode NFKC normalization, removes Markdown bold/backtick markers, and trims surrounding whitespace. It rejects a recognized `A/B/C/D` placeholder anywhere in the response. It then considers the last explicit English or Chinese answer declaration and attempts to read its option label, rejecting its fixed patterns for alternative or negated answers. If this does not produce a prediction, it rejects multiple listed options, tries a leading option label, and applies fixed fallbacks for a bare letter, an explicit declaration, a concluding or opening answer statement, or a unique option-label/text match. Option-text matching normalizes case and punctuation, requires at least four normalized characters, and accepts a unique matching option. Extraction uses fixed regular-expression and text-matching rules with a predefined precedence, without a model-based judge or access to reference answers.

For illustration, with options A=`water vapor`, B=`liquid water`, C=`solid ice`, and D=`carbon dioxide`, the following outcomes follow from that rule:

Response	Extracted label
`The correct answer is B.`	B
`B`	B
`liquid water`	B
`The answer is A. Final answer:  C.`	C
`Final answer:  A or B`	unresolved
`The correct answer is A/B/C/D`	unresolved

Exact input instructions. The question and labeled options retain their original language. T0 appends three newline characters followed by the block below to that question/options text. The resulting content is a user message rendered with the native Qwen3 chat template, `add_generation_prompt=True`, and `enable_thinking=False`. The English instruction is also used for C-Eval. The final line is part of the user instruction; it is distinct from the receiver-only forced answer prefix used for candidate scoring. Long lines below wrap typographically without adding newline characters to the strings.

```
Instructions:
- Carefully read the question and all options.
- Select the single most correct answer.
- Respond ONLY in the following format: "The correct answer is A/B/C/D".
- Do not include any explanations, additional text, or punctuation besides the answer.

The correct answer is
```

T1 and T2 replace that instruction block with the following exact strings, respectively. The question, option order, and diagnostic question panel are held fixed; both models' rendered prefixes are recomputed.

```
Instructions:
- Select the single most correct option.
- Respond with only its uppercase option letter.
- Do not include explanations or any other text.
```

```
Instructions:
- Select the single most correct option.
- Respond in the format "Final answer: <letter>", replacing <letter> with the chosen uppercase
↪  option letter.
- Do not include explanations or any other text.
```

We use the native chat templates of the core 0.6B, 1.7B, and 8B models and fixed candidate-tokenization settings. Generation uses greedy decoding with a maximum of 64 new tokens.

Answer-form diagnostic. The "unique final answer" analysis classifies the generated response separately from ACC extraction. It selects the final declaration when present, distinguishes a placeholder or multiple answer from a unique final option, and otherwise applies fixed option-matching fallbacks. On the 0.6B CS–Direct QKV comparison, 18,860 questions where both responses have a unique final answer contribute 3.12 points to the 3.80-point macro gap; 745 where only CS has a unique final answer contribute 0.68 points. The remaining 16 Direct-only and three neither-unique questions contribute approximately zero. Each contribution retains the full denominator of its task before averaging tasks. This partitions the observed score difference by answer form; it does not isolate the causal contribution of extraction.

A.5 Candidate readout and execution checks

The four-task candidate diagnostic uses 256 fixed questions per task, for 1,024 in total. For both receiver sizes and all T0/T1/T2 input templates, the receiver-only continuation prefix remains `The correct answer is`, with token IDs $[785, 4396, 4226, 374]$. The candidate strings include a leading space; labels A/B/C/D have token IDs $362/425/356/422$. Full-context tokenization verifies that appending this prefix and each candidate preserves the preceding token boundaries. Neither the continuation prefix nor candidate labels are included in sharer prefill.

Let $\mathcal{C}$ contain the evaluated candidate labels and let $s(c)$ be the sum of the candidate sequence's vocabulary log probabilities, conditional on the fixed answer prefix. Under the legal-option protocol, $\mathcal{C}$ includes only labels with nonempty option text. The prefix and EOS are not scored, and there is no length normalization. Prediction is $\arg\max_{c\in\mathcal{C}} s(c)$, with ties resolved by A/B/C/D order. The reported diagnostic loss is the candidate-normalized classification NLL,

$$\mathrm{NLL}_{\mathrm{candidate}} = \log \sum_{c\in\mathcal{C}} \exp s(c) - s(y). \tag{14}$$

Table 12 reports a task-equal mean of per-question NLLs. Multi-token candidate branches use separate cache copies; single-token candidates can share the same next-token logits without modifying the cache. Candidate ACC is a separate likelihood readout, not a re-parsing of generated answers. The historical T0 construction panel uses $\mathcal{C} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}\}$; the prompt-sensitivity panel uses the

legal-option protocol, excluding an empty option on one ARC-C item. This distinction leaves the displayed T0 accuracies unchanged.

Sensitivity of the likelihood diagnostic. Under T0/T1/T2, candidate ACC on 0.6B is 73.54/73.05/72.95% for CS and 70.02/74.12/75.29% for Direct, with differences of $+3.52/-1.07/-2.34$ points. On 1.7B, the corresponding scores are 73.05/74.51/74.32% and 73.14/74.22/74.12%, with differences of $-0.10/+0.29/+0.20$ points. Differences are computed before rounding. The forced answer prefix remains fixed even when it differs from the requested response format. This diagnostic therefore measures a distinct input/readout protocol; its relative ordering need not agree with the answer ACCs in Table 19.

Zero-message and gradient checks cover CS and Direct on both receiving scales. Each path records nonzero gradients on all 56 prefix K/V cache tensors; at the second update, all 224 CS parameter tensors and all 279 Direct tensors receive gradients. The first update can have zero upstream gradients because the output projection starts at zero. Four zero-message test cases reproduce the native segmented path with measured maximum error zero; the temporary check weights are not used for evaluation. Boundary and answer processing make no additional sharer or interface calls.

An independent evaluation rerun of the 0.6B final checkpoints on all 19,624 questions yields 73.6434% for CS and 69.8650% for Direct. The corresponding primary scores are 73.6462% and 69.8509%. This reproduces the aggregate difference, not every generated token; it does not constitute an independent retraining experiment.

B Complete Qwen3 Results

Table 8 lists native references once per model. Table 9 gives the full interface task vectors behind the scale study under the common answer-extraction rule. Full and No-H are separate trained interfaces. All 15 model pairs are retained, including those below a native reference. Macro accuracy is the equal-weight average over OpenBookQA, ARC-C, MMLU-Redux, and C-Eval. The English column averages the first three tasks and is not a translated counterpart of C-Eval.

Table 8: Native task scores, listed once per model. The reference scores are shared by all matching model pairs in the scale study.

Model	OBQA	ARC-C	MMLU-R	C-Eval	English	Macro
Native Qwen3 models						
0.6B	38.00	38.52	32.95	20.13	36.49	32.40
1.7B	64.80	75.30	58.06	53.95	66.06	63.03
4B	79.20	87.30	71.43	68.19	79.31	76.53
8B	85.00	90.96	75.62	73.46	83.86	81.26
14B	90.20	93.13	79.83	78.60	87.72	85.44
32B	93.80	95.04	83.04	82.43	90.63	88.58

Table 9: Complete Qwen3 interface results under the common evaluation protocol. English averages the first three tasks; Macro averages all four.

Pair	Method	OBQA	ARC-C	MMLU-R	C-Eval	English	Macro
Sharer: Qwen3-1.7B							
1.7B→0.6B	C2C	50.40	54.35	45.24	41.45	50.00	47.86
1.7B→0.6B	Full CS	63.60	69.57	53.71	51.15	62.29	59.51
1.7B→0.6B	No-H CS	49.40	56.78	39.60	37.34	48.59	45.78
Sharer: Qwen3-4B							
4B→0.6B	C2C	57.20	60.96	48.17	43.19	55.44	52.38
4B→0.6B	Full CS	66.40	72.70	55.50	46.57	64.87	60.29
4B→0.6B	No-H CS	72.80	80.78	63.76	58.45	72.45	68.95

Table 9 (continued)

Pair	Method	OBQA	ARC-C	MMLU-R	C-Eval	English	Macro
4B→1.7B	C2C	60.00	71.13	54.14	50.84	61.76	59.03
4B→1.7B	Full CS	71.40	78.78	61.74	54.89	70.64	66.70
4B→1.7B	No-H CS	70.80	80.00	63.83	57.82	71.54	68.11
Sharer: Qwen3-8B							
8B→0.6B	C2C	53.20	60.26	47.43	44.36	53.63	51.31
8B→0.6B	Full CS	77.00	80.61	63.37	55.86	73.66	69.21
8B→0.6B	No-H CS	80.80	85.04	68.11	60.63	77.98	73.65
8B→1.7B	C2C	63.40	71.74	54.31	50.96	63.15	60.10
8B→1.7B	Full CS	79.80	83.91	67.28	61.55	77.00	73.13
8B→1.7B	No-H CS	78.00	85.13	68.25	64.15	77.13	73.88
8B→4B	C2C	79.60	87.65	71.41	67.91	79.56	76.64
8B→4B	Full CS	83.00	86.52	69.71	63.76	79.74	75.75
8B→4B	No-H CS	76.00	85.13	69.62	67.75	76.92	74.63
Sharer: Qwen3-14B							
14B→0.6B	C2C	55.80	61.74	49.01	46.71	55.51	53.31
14B→0.6B	Full CS	82.20	87.48	70.03	64.04	79.90	75.94
14B→0.6B	No-H CS	84.20	88.26	70.92	65.14	81.13	77.13
14B→1.7B	C2C	60.20	70.96	53.41	50.14	61.52	58.68
14B→1.7B	Full CS	81.00	86.00	69.23	63.24	78.74	74.87
14B→1.7B	No-H CS	80.20	87.39	70.19	65.78	79.26	75.89
14B→4B	C2C	78.00	86.43	71.27	67.49	78.57	75.80
14B→4B	Full CS	84.80	87.74	73.05	68.32	81.86	78.48
14B→4B	No-H CS	84.40	88.35	73.15	69.37	81.97	78.82
14B→8B	C2C	81.20	89.65	74.11	72.19	81.65	79.29
14B→8B	Full CS	86.60	89.83	74.08	67.10	83.50	79.40
14B→8B	No-H CS	87.20	89.48	73.28	67.42	83.32	79.34
Sharer: Qwen3-32B							
32B→0.6B	C2C	58.60	65.22	52.47	48.86	58.76	56.29
32B→0.6B	Full CS	81.60	86.26	70.83	64.67	79.56	75.84
32B→0.6B	No-H CS	73.00	82.26	66.80	64.44	74.02	71.62
32B→1.7B	C2C	67.40	74.61	56.36	51.53	66.12	62.47
32B→1.7B	Full CS	84.80	89.39	73.74	68.84	82.64	79.19
32B→1.7B	No-H CS	85.20	88.87	73.76	68.77	82.61	79.15
32B→4B	C2C	78.60	86.00	70.38	66.63	78.33	75.40
32B→4B	Full CS	89.20	92.70	78.12	75.44	86.67	83.87
32B→4B	No-H CS	91.00	92.61	77.93	75.34	87.18	84.22
32B→8B	C2C	80.80	90.17	74.45	72.82	81.81	79.56
32B→8B	Full CS	88.20	91.91	76.69	72.10	85.60	82.23
32B→8B	No-H CS	88.00	92.17	77.63	73.26	85.93	82.77
32B→14B	C2C	89.20	92.61	79.15	77.69	86.99	84.66
32B→14B	Full CS	90.20	92.17	77.70	71.17	86.69	82.81
32B→14B	No-H CS	90.40	91.83	77.75	72.87	86.66	83.21

C Task-Level Receiving Comparisons

The following tables expand the receiving comparisons. Table 10 contains the full input/channel ablation and the second receiver. Tables 11 and 12 give task vectors for the five original receiving constructions; Table 16 adds the Scale-only comparison. The initial message weights, data order, and training budget are shared among CS, native-additive, remapped, and linearized receiving; their trained messages subsequently differ. Appendices C.4 and C.5 report supplementary training repeats and the Scale-only control.

Table 10: Task-level receiving ablations under the common evaluation protocol. H denotes Full message inputs, including the receiver state. Full interfaces are retrained under the matched execution settings (Appendix A).

Input / receiving rule	OBQA	ARC-C	MMLU-R	C-Eval	ACC
Qwen3-8B → Qwen3-0.6B					
H / CS-KV	70.20	78.17	61.88	52.44	65.67
H / CS-QKV	76.20	80.26	63.03	56.18	68.92
H / Direct-KV	65.00	75.83	58.61	51.35	62.70
H / Direct-QKV	71.40	80.00	63.32	56.59	67.83
No-H / CS-KV	80.60	85.13	69.48	62.11	74.33
No-H / CS-QKV	80.80	85.04	68.11	60.63	73.65
No-H / Direct-KV	73.60	79.74	64.26	59.34	69.23
No-H / Direct-QKV	76.80	80.96	64.26	57.39	69.85
Qwen3-8B → Qwen3-1.7B					
H / CS-QKV	76.80	82.35	65.84	59.13	71.03
H / Direct-QKV	76.40	83.22	67.63	62.13	72.34
No-H / CS-QKV	78.00	85.13	68.25	64.15	73.88
No-H / Direct-QKV	75.00	82.52	67.31	62.15	71.74

Table 11: Task-level receiving-construction accuracy under the common evaluation protocol. All methods use No-H message inputs.

Method	OBQA	ARC-C	MMLU-R	C-Eval	ACC
Qwen3-8B → Qwen3-0.6B					
CS-QKV (No-H)	80.80	85.04	68.11	60.63	73.65
Lin-CS	81.60	85.74	69.85	63.73	75.23
Direct-QKV (No-H)	76.80	80.96	64.26	57.39	69.85
Native-additive	73.80	79.04	60.12	53.52	66.62
Remapped CS	71.40	75.30	56.73	50.83	63.57

Table 12: Fixed candidate readout: 256 questions per task, 1,024 in total. Candidate ACC and NLL are task-equal means; NLL renormalizes over evaluated candidates.

Rule	OBQA	ARC-C	MMLU-R	C-Eval	Candidate ACC	Candidate NLL
CS-QKV (No-H)	80.08	86.72	68.75	58.59	73.54	0.688
Lin-CS	81.64	86.72	69.92	62.11	75.10	0.673
Direct-QKV (No-H)	75.39	82.81	67.19	54.69	70.02	0.774
Native-additive	73.83	82.42	64.84	55.47	69.14	0.821
Remapped CS	70.31	76.95	61.72	49.61	64.65	0.966

C.1 SOURCE AND ADAPTATION CONTROLS

Table 13: Source and adaptation controls on the primary four tasks. Receiver SFT uses learning rate 2×10^{-5}; interface methods use 10^{-4}.

Method	0.6B receiver	1.7B receiver
Native receiver	32.40	63.03
H-only (controlled study)	36.54	—
Receiver SFT	34.19	60.28
Self-sharer CS	38.35	46.17
C2C (8B sharer)	51.31	60.10
Full CS-KV (scale study)	68.60	67.39
Full CS-QKV (scale study)	69.21	73.13
Native 8B sharer	81.26	81.26
Enhanced cache (reimplementation)	49.27	62.22

H-only removes sharer features from training and uses the receiver hidden state to construct the message. Self-sharer CS uses the same model as both information source and receiver in two execution roles. Receiver SFT updates the backbone rather than the interface. Following the C2C-C architecture in Appendix A.1.3 of Fu et al. (2026), our Enhanced cache reimplementation applies separate three-layer MLPs to sharer K and V before the default C2C cache concatenation, feature fusion, dynamic weighting, and gating. Both tested pairs have 1,024-dimensional sharer and receiver KV features. The added MLPs use width 1,024, GELU, LayerNorm, residual connections, and dropout 0.1; the interface totals 705.79M trainable parameters. Fusion operates on completed native prefix caches, after key-head normalization and RoPE. We evaluate this reconstruction under our training and evaluation protocol, rather than reproducing the original paper's Table 9 configuration. The scale-study Full comparisons and the matched receiving comparisons remain distinct.

C.2 INTERPRETING THE FIRST-ORDER RESULT

Lin-CS scores 75.23% ACC, compared with 73.65% for No-H CS-QKV, and improves each of the four task scores. Candidate ACC is 75.10% versus 73.54% on the separate 1,024-question panel. Thus, a first-order rule can retain strong communication without the exact finite nonlinear receiving update in this configuration. This is one model pair and one formal training run; the scale study and latency measurements retain their independently trained CS checkpoints. Table 14 reports the corresponding paired uncertainty, including the candidate interval that contains zero.

C.3 PAIRED UNCERTAINTY FOR THE REPORTED COMPARISONS

Table 14 reports existing pointwise 95% percentile bootstrap intervals, with 10,000 paired question resamples and statistical seed 42. For the four-task generation and candidate comparisons, questions are resampled with replacement within each task, using the same selected questions for both methods; the resampled task accuracies are then averaged equally. For SciQ, the three template outputs travel together with each sampled question, and the statistic is the mean over templates. These intervals describe question uncertainty conditional on the fixed trained checkpoints. They do not estimate variation across training seeds or provide simultaneous coverage for all comparisons.

Table 14: Paired differences in percentage points. Generation uses the full 19,624-question four-task evaluation; Candidate ACC uses the 1,024-question panel; SciQ uses the locked 1,000-question three-template mean. All CS/Direct entries use No-H inputs.

Comparison and model pair	Metric	Δ	95% paired interval
CS − Direct, 8B→0.6B	Generation ACC	+3.80	$[+2.92, +4.70]$
CS − Direct, 8B→1.7B	Generation ACC	+2.14	$[+1.20, +3.07]$
Lin-CS − CS, 8B→0.6B	Generation ACC	+1.58	$[+0.87, +2.31]$
Lin-CS − Native-additive, 8B→0.6B	Generation ACC	+8.61	$[+7.64, +9.62]$
Lin-CS − CS, 8B→0.6B	Candidate ACC	+1.56	$[-0.10, +3.22]$
CS − Direct, 8B→0.6B	SciQ ACC	+0.23	$[-0.60, +1.10]$
CS − Direct, 8B→1.7B	SciQ ACC	+0.27	$[-0.47, +1.03]$

The Lin-CS generation comparison and its separate candidate diagnostic have different uncertainty: the candidate interval includes zero. The SciQ intervals likewise do not separate CS and Direct, while both methods improve their native receiver. An interval containing zero is not an equivalence test.

C.4 Independent training repeats on the 0.6B receiver

Table 15 reports three independent training runs per method for Qwen3-8B→Qwen3-0.6B under T0. Try 1 denotes the designated primary final checkpoints; Try 2 and Try 3 are additional runs with fresh initializations, training orders, and dropout randomness. Each reported run uses the prescribed 1,286-update training budget and final-checkpoint evaluation.

The architecture and training recipe follow Appendix A. CS and Direct retain widths 1,504 and 1,304 and effective parameter counts 509.39M and 507.33M. The fixed training protocol pairs the data order, logical groups, supervision masks, physical batch schedule, and evaluation batches within each try. Its seed-specific batch plans specify 132,635,762, 132,633,036, and 132,634,672 supervised tokens for Try 1–3, respectively, with the same plan for both methods within each try.

Table 15: Independent training repeats for No-H receiving rules. Task accuracies and ACC are percentages; ACC is the equal-weight four-task mean over 19,624 questions. Δ is CS minus Direct within each try, in percentage points, computed before rounding.

Method	OBQA	ARC-C	MMLU-R	C-Eval	ACC	Δ
Try 1: primary checkpoints						
CS-QKV	80.80	85.04	68.11	60.63	73.65	+3.80
Direct-QKV	76.80	80.96	64.26	57.39	69.85	—
Try 2						
CS-QKV	82.00	86.78	69.23	64.40	75.60	+5.06
Direct-QKV	76.20	82.70	65.71	57.58	70.55	—
Try 3						
CS-QKV	81.20	86.96	68.59	72.98	77.43	+6.64
Direct-QKV	75.80	83.30	64.75	59.30	70.79	—

Across the three runs, mean ACC $\pm$ sample standard deviation is 75.56 ± 1.89 for CS and 70.40 ± 0.49 for Direct. The paired CS–Direct difference is $+5.16 \pm 1.43$ points. These are sample standard deviations across training runs (denominator $n - 1$, $n = 3$), not confidence intervals or question-bootstrap uncertainty. CS leads in all three reported runs and on all four tasks within each run. Its observed variability remains larger than Direct's; three seeds provide limited evidence about broader training variability. These repeats supplement the primary checkpoints; the scale matrix and the other model pair are not three-run averages.

C.5 SEPARATING THE SCALE TERM FROM FIRST-ORDER RECEIVING

For the same 8B→0.6B No-H setting, we train a Scale-only control whose receiving input and raw operands are

$$x^{\text{scale}} = N(h) + \frac{D_\gamma m}{r(h)}, \qquad o^{\text{scale}} = W x^{\text{scale}}, \qquad r(h) = \sqrt{\|h\|^2/d + \epsilon}. \tag{15}$$

Here $m = A_\phi(z_T)$. Relative to Lin-CS in Equation 13, this removes the rank-one correction $-D_\gamma h(h^\top m)/(d\, r(h)^3)$ from the message increment. The native normalization gain and current-state scale remain; this is not a constant rescaling or a new trainable gate. Gradients pass through both h and m, including the scale denominator. Projection-bias conventions, subsequent head normalization and RoPE, and the direct residual are unchanged.

Scale-only is trained independently from the same original initialization as the primary No-H CS, Native-additive, and Lin-CS runs, with the same message network, 509,386,752 parameters, data order, masks, logical loss reduction, physical schedule, and update budget. It is not obtained by deleting a term from a trained Lin-CS checkpoint. The existing comparator checkpoints are reused; only Scale-only adds a training run, using the primary training seed and final-checkpoint selection.

Table 16: Scale-only and existing No-H controls on 8B→0.6B. Generation ACC uses the full 19,624-question T0 evaluation; Candidate ACC uses the separate fixed 1,024-question panel. Values are percentages; the two readouts are not averaged together.

Receiving rule	Generation ACC	Candidate ACC
Native-additive	66.62	69.14
Scale-only	75.75	74.22
Lin-CS	75.23	75.10
CS-QKV	73.65	73.54
Direct-QKV	69.85	70.02

Scale-only exceeds Native-additive by 9.13 generation points (95% paired interval $[8.11, 10.14]$) and primary CS by 2.10 points ($[1.43, 2.77]$). For Lin-CS minus Scale-only, the generation difference is -0.52 points ($[-1.20, +0.17]$), whereas the candidate difference is $+0.88$ points ($[-0.59, +2.35]$). Scale-only's candidate differences from Native-additive and CS are $+5.08$ ($[+2.64, +7.52]$) and $+0.68$ ($[-1.17, +2.54]$) points, respectively. All intervals use 10,000 paired question resamples within tasks and equal task weighting, with statistical seed 42; as in Appendix C.3, they are conditional on fixed checkpoints and do not estimate training variability.

This one-pair, one-training-seed comparison shows that the retained scale term can support strong transfer without Lin-CS's rank-one correction. It does not establish reliable superiority over Lin-CS: the point ordering reverses between readouts, and both corresponding intervals include zero. The separately trained interfaces can adapt differently, and the native gain and state-dependent scale remain coupled. The results therefore do not identify a unique optimization or semantic cause, nor replace the checkpoints used in the scale or timing studies.

A local amplitude identity. For the same current state h and message m, RMS normalization gives the exact identity in real arithmetic

$$\begin{aligned} x^{\text{scale}} &= \frac{D_\gamma(h+m)}{r(h)} = \alpha(h,m) N(h+m), \\ \alpha(h,m) &= \frac{r(h+m)}{r(h)} > 0. \end{aligned} \tag{16}$$

With native projection bias b_o, this implies $o^{\text{scale}} - b_o = \alpha(o^{\text{CS}} - b_o)$; without bias, the raw operands are proportional. In the bias-free Qwen3 setting, Q/K head RMS normalization cancels positive scaling in the zero-epsilon idealization, but cancellation need not be exact with the native nonzero epsilon. V has no head normalization and retains the position-dependent scaling. Independently

trained interfaces generally produce different messages and subsequent layer inputs, while finite-precision computations introduce rounding. This local identity therefore establishes neither end-to-end model equivalence nor a causal explanation of the measured accuracy differences.

D Selected Cross-Family Cases and Evaluation Protocols

D.1 The three Mistral-to-Gemma pairs in the main text

The released checkpoints are `Mistral-Small-24B-Instruct-2501`, `gemma-3-270m-it`, `gemma-3-1b-it`, and `gemma-3-4b-it`. The Mistral-to-Gemma series holds the sharer fixed across three receiver sizes. These retrospectively selected cases use Full interfaces and their own native-reference evaluations. They are not replications of the No-H CS–Direct controls.

Table 17: Complete task scores for the three Mistral-to-Gemma pairs shown in the main text.

Path	OBQA	ARC-C	MMLU-R	C-Eval	English	Macro
Mistral-Small-24B → Gemma-3-270M						
Native receiver	26.00	17.83	14.51	16.94	19.44	18.82
Full CS	60.20	61.91	43.47	34.39	55.19	49.99
Native sharer	89.20	92.61	77.15	60.64	86.32	79.90
Mistral-Small-24B → Gemma-3-1B						
Native receiver	34.20	29.04	28.69	24.12	30.65	29.01
Full CS	44.00	52.87	40.48	27.53	45.78	41.22
Native sharer	89.20	92.61	77.15	60.64	86.32	79.90
Mistral-Small-24B → Gemma-3-4B						
Native receiver	62.40	64.87	52.20	35.42	59.82	53.72
Full CS	63.20	70.70	55.49	40.92	63.13	57.57
Native sharer	89.20	92.61	77.15	60.64	86.32	79.90

Table 18: Layer and interface sizes for the displayed cross-family cases. Layer indices are zero-based; mappings align terminal layers. Repeated counts reuse of a selected sharer layer.

Pair	Depths T/S	Receiving layers	Repeated	Sharer → receiver layers	Params (M)
Mistral-Small-24B→Gemma-3-4B	40/34	34	0	6–39→0–33	528.86
Mistral-Small-24B→Gemma-3-1B	40/26	26	0	14–39→0–25	523.06
Mistral-Small-24B→Gemma-3-270M	40/18	18	0	22–39→0–17	528.45

Each model uses its own tokenizer and chat template; equal text does not imply equal token indices. Layer counts and receiver-layer coverage are shown in Table 18. Each pair trains its own interface. These cross-family results establish applicability to other backbones; SciQ instead evaluates already trained interfaces on a new task.

D.2 Cross-family position, layer, and execution correspondences

After rendering the same canonical messages with each model's native chat template, valid content tokens are assigned UTF-8 byte spans within their original message. Let $e_T(t)$ and $e_S(i)$ denote their ending byte offsets. Position alignment selects

$$a(i) = \operatorname*{argmax}_{t:\,\mathrm{msg}(t)=\mathrm{msg}(i),\; e_T(t)\le e_S(i)} (e_T(t), t), \tag{17}$$

where the maximum is lexicographic in endpoint and token index. An empty candidate set gives no message ($M_{j,i} = 0$). This selects one sharer token without averaging; several receiver tokens may

reuse it. Template-only tokens, special tokens, and tokens crossing message boundaries receive no content mapping. For byte-fallback tokens with overlapping offsets, only the final complete piece receives a valid span. The alignment is causal in message-content endpoints.

All three displayed pairs use zero-based terminal layer alignment $g(j) = L_T - L_S + j$, for $j = 0, \ldots, L_S - 1$, with every physical receiver layer covered. For Mistral-24B's 40 layers, the three Gemma receivers use source layers 22–39 (270M), 14–39 (1B), and 6–39 (4B). There are no repeated source-layer selections in these three pairs.

These historical Full implementations use the same receiving input $N(h + m)$ and retain the direct residual. During inference, they prefill the complete rendered receiver prompt once, inject only at positions with valid content mappings, and remove the conditioning hooks before native continuation. Template positions without content mappings receive no message; there is no separately executed final-boundary-token forward. During training, the receiver uses full-sequence masked teacher forcing: the sharer sees only the known prompt, answer positions have mapping -1, and the answer loss backpropagates through the complete causal receiver computation. This differs from the core Qwen3 segmented prefix-cache path in Algorithm 1. The core checks on a separate boundary call and an unused final-prefix Q head apply to that segmented path.

D.3 Locked SciQ confirmation

SciQ uses its 1,000-question test split without support passages. Checkpoints, option order, and the three-template mean are fixed before inference, with no SciQ training or score-based tuning. Table 6 gives every generated-answer condition under the common extractor. Both interfaces raise the native receiver's mean accuracy substantially; their mutual differences are 0.23 and 0.27 points on the two receiving scales, with paired intervals reported in Table 14. A lexical overlap check against the 493,818 training examples and 19,388 development-question texts flagged one item. Excluding it yields mean ACCs of 94.63/94.39% for CS/Direct on 0.6B and 94.36/94.09% on 1.7B. The shared 999-question exclusion is a sensitivity check; the locked 1,000-question results remain primary.

D.4 Answer-template sensitivity

The fixed diagnostic panel contains 256 questions from each primary task. T0 is the primary instruction; T1 requests only an option letter; T2 requests a final-answer marker. Both the sharer and receiver prefixes are recomputed when the instruction changes. Table 19 reports answer ACC under the common deterministic extraction rule.

Table 19: Answer accuracy under three templates on the fixed 1,024-question panel (%). Native denotes the unassisted receiver; CS and Direct use No-H interfaces. Δ is CS minus Direct.

Template	Native	CS	Direct	Δ
Qwen3-8B → Qwen3-0.6B				
T0	34.77	73.83	70.51	+3.32
T1	41.02	74.41	72.95	+1.46
T2	45.21	74.12	71.19	+2.93
Qwen3-8B → Qwen3-1.7B				
T0	62.30	72.56	71.97	+0.59
T1	61.72	74.51	73.73	+0.78
T2	64.26	74.80	74.12	+0.68

On 0.6B, CS's ACC remains within 0.58 points across these templates and exceeds the native receiver by 28.91–39.06 points. Native unresolved-answer rates are 17.97%, 0.49%, and 0.59% under T0, T1, and T2; the corresponding rates on 1.7B are 1.76%, 0.29%, and 0.10%. The native T0 outputs use historical batch-one inference, whereas T1/T2 use length-bucketed batches. These observations describe protocol sensitivity without attributing the entire change to answer extraction. CS's point estimates exceed Direct under all three templates on both receivers, with T2 differences of 2.93 and 0.68 points. The primary comparison remains the fixed T0 result.

D.5 TEXT COMMUNICATION AS AN ALTERNATIVE CHANNEL

The same 8B sharer is prompted for solution-style or structured advice with caps of 256 or 512 tokens, after which each receiver answers. These are request-condition names: the observed messages are short answers, commonly of the form `The correct answer is C.` Across 19,624 sharer outputs per condition, mean output lengths are 6.6005/6.6009 tokens for solution 256/512 and 6.6634/6.6636 for structured 256/512; every median is 7. The maxima are 10 tokens for solution and 17 for structured, and no advice output reaches its cap. Thus these conditions evaluate short-text answer communication.

Table 20 retains all eight receiver conditions. Its serial latency is the sum of separately measured warm sharer and receiver times; serial TTFT adds the complete sharer time to the receiver's time to its first answer token. These natural-generation measurements use a separate protocol from fixed-token replay.

Table 20: Short-text answer communication under the requested advice conditions. Both models run in sequence. Caps are upper limits; sharer tokens are realized means over 19,624 generated messages. Receiver lengths and full serial times are also reported.

Request	Cap	ACC	Sharer tokens	Receiver tokens	Serial time (ms)	Serial TTFT (ms)
Qwen3-8B sharer / Qwen3-0.6B receiver						
Solution	256	77.44	6.6005	7.34	238.41	148.10
Solution	512	77.44	6.6009	7.34	239.87	147.38
Structured	256	77.19	6.6634	7.32	238.12	148.36
Structured	512	77.21	6.6636	7.32	241.17	149.28
Qwen3-8B sharer / Qwen3-1.7B receiver						
Solution	256	76.55	6.6005	6.62	230.61	148.42
Solution	512	76.56	6.6009	6.62	229.85	147.42
Structured	256	76.74	6.6634	6.67	230.48	148.67
Structured	512	76.74	6.6636	6.67	231.78	149.19

Text guidance exceeds CS on these task scores. Avoiding intermediate text is a property of the CS execution path, not a claim that text is always less accurate or more expensive. The comparison does not assign the latency of one protocol to the accuracy of another.

E TIMING PROTOCOL AND COMPLETE PLOTTED MEASUREMENTS

The timing study uses 32 questions (eight per primary task), with 113–325 input tokens and median 129. Each request occupies one H200 at batch one; eight independent replicas distribute the workload. The software is PyTorch 2.11.0+cu129, CUDA build 12.9, Transformers 5.6.0, BF16/SDPA, and two CPU threads per worker. Native conditions load only their own model; interface conditions load the sharer, receiver, and interface.

Each method replays the same within-pair native-receiver token trajectory at lengths 1, 2, 4, 8, 12, 16, 24, 32, 64, 96, 128, 192, and 256. EOS does not terminate replay. Each of three rounds excludes two warm-up questions per GPU, method, and length. Method order alternates by rank and round, and each question remains assigned to the same GPU. Total latency includes prefix processing, the first output calculation, $N - 1$ continuation steps, synchronization, and common instrumentation; model loading, tokenization, and scoring are excluded. TTFT uses CUDA events.

The 64- and 256-token measurements were collected in an earlier run of the same protocol; all remaining lengths, including 16, use the dense-grid run. Comparisons at each fixed length are paired within their measurement run. In Figure 3, open squares identify those two retained lengths and circles identify the dense-grid measurements. No cross-run rescaling is applied.

Table 21: Fixed-workload timing for Qwen3-8B→0.6B. Each cell averages 32 prompts over three rounds.

Qwen3-8B→0.6B										
	Total latency (ms)					Time to first token (ms)				
Tokens	Sharer	Recv.	C2C	Full	No-H	Sharer	Recv.	C2C	Full	No-H
1	27.355	22.079	77.997	78.433	74.521	27.320	22.041	77.939	78.378	74.466
2	47.213	32.935	90.679	84.707	82.416	20.726	15.664	75.081	66.978	67.157
4	87.014	65.117	119.679	115.092	113.899	20.774	15.919	73.757	65.982	67.671
8	167.464	129.777	181.243	179.330	176.077	20.520	16.229	73.754	66.706	68.221
12	245.580	188.291	241.288	239.579	231.871	20.855	16.159	73.479	66.783	66.676
16	328.102	245.035	295.670	300.207	288.985	20.733	15.880	72.527	67.206	65.996
24	490.173	375.431	415.106	419.233	412.358	21.119	16.376	72.800	66.379	66.658
32	630.994	487.017	532.976	539.946	527.296	20.562	15.953	72.612	66.195	66.080
64	1284.979	964.286	1046.640	1020.798	1006.093	20.301	16.117	75.405	67.479	66.888
96	1910.927	1434.653	1497.464	1522.758	1478.946	20.900	15.585	73.469	66.513	66.437
128	2505.541	1906.225	1996.266	1961.958	1969.264	20.756	15.836	74.152	66.163	67.211
192	3766.386	2851.447	2979.672	3001.059	2903.877	20.734	15.787	74.354	67.683	66.595
256	5047.428	3858.144	4027.143	3874.806	3866.848	20.599	16.433	76.597	66.537	66.450

Table 22: Fixed-workload timing for Qwen3-8B→1.7B. Each cell averages 32 prompts over three rounds.

Qwen3-8B→1.7B										
	Total latency (ms)					Time to first token (ms)				
Tokens	Sharer	Recv.	C2C	Full	No-H	Sharer	Recv.	C2C	Full	No-H
1	27.714	22.636	75.958	75.151	72.204	27.677	22.601	75.902	75.105	72.157
2	45.459	33.031	87.931	83.063	82.059	20.877	15.855	72.783	65.828	66.885
4	88.506	63.657	117.600	114.230	112.316	20.873	15.780	72.454	65.342	66.800
8	171.078	125.359	180.042	176.712	173.674	21.043	15.821	73.586	66.351	67.238
12	246.706	185.162	238.587	235.784	228.877	20.710	15.799	72.554	66.022	66.129
16	317.908	240.607	303.134	294.859	289.225	20.372	15.592	73.631	65.345	65.776
24	478.372	358.637	426.548	416.537	408.429	20.484	15.580	74.349	65.725	66.159
32	619.359	484.790	540.552	540.511	528.211	20.130	15.791	73.930	67.288	65.850
64	1325.073	999.160	1045.208	1075.731	1062.086	21.125	16.262	74.982	69.087	69.592
96	1874.448	1458.028	1518.575	1532.900	1464.278	20.106	16.046	74.528	67.424	66.128
128	2434.832	1895.575	1971.282	1967.277	1929.536	20.079	15.899	73.120	66.176	65.726
192	3709.913	2848.069	2942.566	2954.930	2906.372	20.378	15.621	73.270	67.389	67.298
256	5173.295	3943.998	4017.367	4101.687	4012.283	21.020	16.445	76.082	69.102	69.050

Table 23: Fixed-workload timing for Qwen3-32B→0.6B. Each cell averages 32 prompts over three rounds.

Qwen3-32B→0.6B										
	Total latency (ms)					Time to first token (ms)				
Tokens	Sharer	Recv.	C2C	Full	No-H	Sharer	Recv.	C2C	Full	No-H
1	44.400	22.028	94.073	94.568	91.008	44.364	21.988	94.013	94.511	90.952
2	75.798	33.076	107.045	101.649	100.160	37.646	15.846	91.658	84.208	84.910
4	147.720	62.840	137.472	134.945	130.642	38.257	15.562	91.547	84.261	84.894
8	284.713	122.906	197.194	203.202	191.167	38.087	15.391	91.175	86.314	84.805
12	418.659	181.260	258.742	262.539	255.009	37.913	15.528	91.912	85.834	86.060
16	563.113	237.197	315.331	311.631	313.374	38.435	15.399	91.186	83.713	84.923
24	851.818	363.266	439.367	432.991	430.802	38.707	15.714	92.032	83.342	84.519
32	1100.504	481.510	557.410	553.440	547.843	38.130	15.792	91.819	83.486	84.122
64	2276.588	1029.484	1098.406	1090.931	1054.922	38.650	16.948	96.869	87.362	86.201
96	3277.765	1454.144	1543.878	1544.602	1521.584	38.221	15.801	93.903	84.270	85.618
128	4289.880	1910.801	2014.749	2038.517	1983.535	38.130	15.879	93.054	85.155	84.293
192	6493.056	2883.196	2942.113	2964.033	2947.297	37.861	15.870	90.905	84.032	84.663
256	9104.388	4004.557	4089.756	4184.321	4118.490	39.104	16.740	96.357	87.520	88.602

For both 8B-sharer pairs, all three interface paths are slower than the native sharer at eight tokens and faster at twelve. For 32B→0.6B, the corresponding measured change occurs between two and four. These are adjacent measurement brackets, not fitted continuous thresholds. The curves describe a fixed output workload, not the quality of a natural answer of that length.

At 256 tokens, No-H CS takes 3.867/4.012/4.118 seconds in the three plotted pairs. C2C takes 4.027/4.017/4.090 seconds, showing that both interfaces benefit from receiver-only continuation. In a separate natural short-answer comparison at 8B→0.6B, No-H CS takes 179.78 ms and the native sharer 152.97 ms, with different generated lengths. Thus a favorable fixed-length crossover does not imply lower latency for every request.