Verified Compaction: A Capacity-Gated Probe for Faithful Dialogue Compression

Anonymous ACL submission

Abstract

Faithful dialogue compression's downstream value depends on the reader's capacity. We introduce Verified Compaction (VC)—a 2-stage prompt-level pipeline using participant framing, "copy EXACTLY" lexical anchoring, and post-hoc verification—as a strong-faithfulness probe to characterise when faithfulness translates to downstream advantage. Across 4 dialogue datasets (N=311, EN+JA; Chinese replication on VCSum), VC is consistently in the top group on the faithfulness axis (paired Cohen's d on R-L positive in $16/16$ cells, median 1.13; $17/20$ Wilcoxon wins vs. a public 9-section structured summarization baseline). On a 4-model QA-capacity spectrum (Phi-4-mini-3.8B / Llama-3.1-8B / Phi-4-14B / Gemma-4-26B-MoE), VC's downstream advantage is Holm-significant at 8B QA (Llama, p=0.012; p_{Holm}=0.048 across the four capacity tests) with a same-direction raw-significant trend at 3.8B (Phi-mini, p=0.031 raw, p_{Holm}=0.093), and is null at $\geq$14B; within the Phi-4 family alone the same compressed inputs go from p=0.031 (raw, Holm n.s.) to null, descriptively controlling for architecture across two scales of one family. Mechanistically, "copy EXACTLY" elicits 83% verbatim 5-gram overlap; an attention probe on the deployed Qwen3.5-9B compressor identifies $8/128$ Holm-significant wording-induced source-attention shifts at specific heads, with a same-direction but underpowered replication on Llama-3.1-8B ($0/1024$ heads survive Bonferroni at $\alpha/1024$ while the largest-magnitude shifts are of comparable order to Qwen's). Scope: at aggressive compression VC retains only a small mfqa-favoured trend (n=100, d=$+0.18$, n.s.); VC underperforms a progressive summarization baseline (LangChain) on multi-turn handover and Generic+Refine on multi-hop QA. The capacity-gating yields a falsifiable basis for method selection: faithful compression buys downstream value only under bounded reader capacity.

1 Introduction

As LLMs adopt longer context windows (200K–1M tokens), a recurring objection to dialogue-compression research is that method choice will soon stop mattering downstream: a sufficiently capable model can read the full history. We agree this is partially true—our experiments reproduce the null at moderate compression on capacity-rich QA models—but capacity is not uniform within a long window: position-dependent (Liu et al., 2024; Wan et al., 2025) and length-dependent (Yen et al., 2025; Bai et al., 2025) failures persist. The question worth asking is therefore not *whether* method choice matters, but *when*, and along *which axis*. Faithful compression remains on the deployment roadmap because cost scales with active context (latency, energy), edge/on-device deployments have hard memory budgets, and the conversation-length distribution has a long tail beyond any fixed window. The contribution of this paper is therefore a diagnostic, not a universal winner: a measurable *capacity-ratio interaction* that determines when compression-method choice still matters.

Compressors are typically reported along a single dimension—intrinsic similarity (ROUGE, BERTScore) or downstream task accuracy (Jiang et al., 2023; Xu et al., 2024)—and the two are often assumed to track one another. Building on the holistic-evaluation tradition of Liang et al. (2023), we show the two axes *decouple* for dialogue compression in a specific, predictable way: production compressors—Anthropic's Claude Code 9-section summarization (community reproduction by Piebald-AI, 2024; the official prompt is not published) and LangChain's `ConversationSummaryBufferMemory` (Chase, 2022)—show large faithfulness gaps but small downstream gaps in the default ρ=0.5, capacity-rich regime, and the gap on the downstream axis re-opens only at smaller QA capacity. The

faithfulness failures we target are **numerical drift** ("around 500" replaces "523.40"), **proper-noun loss**, and **resolution erasure** (unresolved questions become "completed"); these matter structurally for audit, replay, and compliance, and matter *conditionally* for downstream task utility—conditioned on QA-model reading capacity, which is what this paper measures.

We instantiate **Verified Compaction** (VC) as a strong *faithfulness probe*—a 2-stage pipeline that maximizes the faithfulness axis so we can study where the gap to downstream opens and closes:

1. **Participant framing**: instructing the LLM "you are a participant in this conversation," encouraging preservation of unresolved items and decisions.

2. **Lexical anchoring**: directing the model to "copy numbers and proper nouns EXACTLY, character by character." This single rule elicits 83% verbatim 5-gram overlap with the source (vs. 12–20% for baselines, App. H).

3. **Verification post-processing**: a second LLM call comparing the compressed output to the source via ADD/REMOVE/KEEP edits. We characterize this step as a controlled trade-off rather than a strict accuracy guarantor; the ADD step recovers missed entities while the REMOVE step can prune legitimate elaborations (§6.1).

Across 4 dialogue datasets (N=311, EN+JA; Chinese replication on VCSum), 3 LongBench QA tasks, and a 4-model QA-capacity spectrum (Phi-4-mini-3.8B / Llama-3.1-8B / Phi-4-14B / Gemma-4-26B-MoE), our contributions are: **(i) Capacity-gated downstream translation**—VC's faithfulness advantage reaches Holm-significance on Llama-8B (p=0.012, p_{Holm}=0.048 across 4 capacity tests) with a same-direction raw-significant trend on Phi-mini-3.8B (p=0.031 raw, p_{Holm}=0.093), and is null at ≥14B (within Phi-4 alone, 3.8B→14B), identifying the regime where method choice stops mattering; **(ii) Faithfulness probe**—VC is the only method consistently in the top group across 4 datasets (median d=1.13 on R-L, 17/20 Wilcoxon wins vs. a public 9-section structured baseline); **(iii) Mechanism**—"copy EXACTLY" elicits 83% verbatim 5-gram overlap; an attention probe on Qwen3.5-9B yields 8/128 Holm-significant

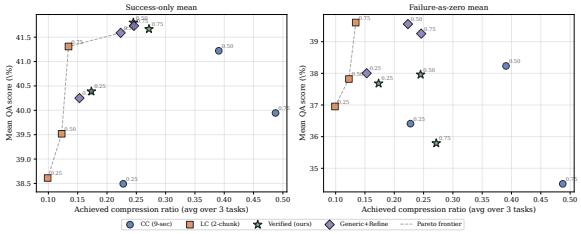

Figure 1: Downstream QA Pareto view: each point a (method, target ratio) combination, averaged over 3 LongBench tasks (12 compressing + Original). x: achieved ratio (lower=more compressed); y: mean QA score under success-only (left) and failure-as-zero (right). Dashed: empirical Pareto frontier.

heads; the Llama-3.1-8B replication is in the same direction but underpowered (0/1024 heads survive the much stricter Bonferroni threshold, while the largest-magnitude shifts are of comparable order); and a teacher-forced per-token attribution finds VC outputs anchored to source more sharply than Generic (d=+1.22 on Llama-3.1-8B), parallel to architectural copy mechanisms (See et al., 2017; Vinyals et al., 2015) at the prompt level; **(iv) Boundary scope and a falsifiable selection rule** (§7): VC underperforms a progressive baseline on multi-turn and Generic+Refine on multi-hop, and at ρ=0.05 retains only a small mfqa-favoured trend (n=100, d=+0.18, n.s.).

Figure 1 previews the (achieved-ratio, downstream) plane: methods cluster tightly in the default ρ=0.5 capacity-rich regime, the spread opens substantially on faithfulness (§5) but stays narrow downstream until QA capacity drops below the threshold of §5.3. The asymmetry motivates the capacity-gated framing.

2 Related Work

Prompt compression: a taxonomy. Following Li et al. (2025a), **token-pruning methods** score and prune low-information tokens via auxiliary models or training-free entropy/attention heuristics (LLMLingua (Jiang et al., 2023, 2024; Pan et al., 2024), Selective Context (Li et al., 2023), PCToolkit (Li et al., 2024), ICPC (Yu and Liu, 2025), Perception Compressor (Tang et al., 2025)); **soft-token / encoder methods** learn dense representations of compressed prompts, requiring model-side training (Gist (Mu et al., 2023), AutoCompressors (Chevalier et al., 2023), 500xCompressor (Li et al., 2025b), RECOMP (Xu et al., 2024), Selection-p (Chung et al., 2024),

TACO-RL (Shandilya et al., 2025)). VC operates only on the prompt of an off-the-shelf instruction-tuned LLM, requires no model-side change, and targets discourse-level dialogue structure rather than surface tokens. Nagle et al. (2024) cast prompt compression as rate-distortion; our Pareto analysis (§5.3) is a practical instance.

LLM-as-compressor and verification. Closest to our setting, Pu et al. (2024) use a frozen LLM as a compressor and tune *styles* per task; VC contributes three orthogonal prompt components (framing, anchoring, verification) which can be combined with any chosen style. VC's verify step shares the spirit of Self-Refine (Madaan et al., 2023) and CoVe (Dhuliawala et al., 2024) but applies a single-pass post-hoc ADD/REMOVE/KEEP edit to a compressed dialogue rather than iterating open-ended outputs or planning subquestions; we study 0–4 passes in §6.1 and App. G.

Faithfulness, copy mechanisms, conversational memory. Maynez et al. (2020); Cao et al. (2018) document hallucination in abstractive summarisation; copy mechanisms (See et al., 2017; Vinyals et al., 2015) let decoders reproduce source tokens. Our "copy EXACTLY" elicits a similar effect at the prompt level in instruction-tuned LLMs, without architectural change (§6.1). A complementary line of work targets agents persisting information across many sessions (LoCoMo (Maharana et al., 2024), Mem0 (Chhikara et al., 2025)); those systems *retrieve* from a growing memory store, whereas VC *compacts* a single bounded dialogue once (§7).

Industrial baselines. We treat two production systems as primary baselines because they are deployed at scale and academically understudied: Anthropic Claude Code's 9-section structured summarization (`ccVersion 2.1.84`; we use the publicly-disclosed reproduction by Piebald-AI, 2024; the upstream design intent is sketched in Anthropic's context-engineering blog (Anthropic, 2025)) and LangChain's `ConversationSummaryBufferMemory` progressive summarization (Chase, 2022).

Holistic evaluation of compression. Closest to our evaluation design, Łajewska et al. (2025) combine BERTScore, entity preservation, and downstream QA on *retrieval-augmented* contexts. We extend that framework to dialogue compaction: our data is multi-turn EN and JA dialogues (with a Chinese replication on VCSum in App. L), our metric set is five primary intrinsic metrics (R-1/2/L, Recall, FIR) tested with Wilcoxon+Holm plus two supplementary (BERTScore, ratio) and G-Eval (App. E), and we disaggregate by *achieved* compression ratio (Table 4)—building on the holistic-evaluation tradition of Liang et al. (2023).

3 Method: Verified Compaction

3.1 Overview

Verified Compaction (VC) is a 2-stage abstractive compression with verification: (1) **Stage 1 (compression)**, an LLM call with three prompt components—(a) participant framing, (b) lexical anchoring ("copy EXACTLY"), (c) dialogue-specific rules (state WHO, preserve unresolved items and decisions); (2) **Stage 2 (verification)**, a second LLM call comparing the output T_1 with the source C and producing T_2 via ADD / REMOVE / KEEP edits, then truncated at sentence boundary to $\leq t_c$ characters (full prompts in App. B). We characterise Stage 2 as a controlled trade-off: the ADD step recovers missed entities (improving Recall), while REMOVE can prune legitimate elaborations (§6.1).

3.2 Stage 1: Participant-Framed Compression

The Stage 1 prompt (full text in App. B) opens `''You are a participant in this conversation. You will continue afterwards. Compress to {tc} characters or less.''` followed by CRITICAL RULES enforcing verbatim copy of numbers and proper nouns and discourse-level rules (state WHO, keep unresolved, preserve decisions).

Two complementary levers. VC's faithfulness gap is produced by two prompt-level levers acting on the same mechanism ("get source entities into the compressed text") that are partially redundant in the full pipeline. *Lever 1, lexical anchoring (Stage 1):* "Copy EXACTLY" elicits literal token reuse—a behaviour parallel to but mechanistically distinct from architectural copy mechanisms (See et al., 2017; Vinyals et al., 2015). The generalizable principle is that *some* explicit copy-eliciting wording at Stage 1 matters; the optimal phrasing is model-dependent (App. K: "copy EXACTLY" is best on Qwen3.5 by d=+1.04 on R-L but ranks 3rd of 4 on Llama-3.1, where "preserve verbatim" is best). Qwen outputs share 83% of 5-grams verbatim with the source vs. 12–20% for baselines (App. H). *Lever 2, post-hoc verification (Stage 2):*

Dataset	Lang	N	Avg chars	#nums/dlg
SAMSum	EN	80	387	0.04
DialogSum	EN	50	539	0.20
BSD+hh-rlhf-ja	JA	151	623	0.34
MeetingBank	EN	30	6184	8.63

Table 1: Intrinsic evaluation datasets, ordered by numerical density.

when applied, the Stage-1 wording effect partly diminishes (App. K): verify's ADD recovers many of the same entities. Full-pipeline deployments are therefore robust to wording; single-stage deployments should A/B-test wordings.

Participant framing. "You are a participant" positions the LLM to retain unresolved items and decisions as if for self-reference, rather than as pure summarizer.

3.3 Stage 2: Verification post-processing

The verification prompt (App. B) instructs: **(1) ADD** missing numbers, dates, and proper nouns from C; **(2) REMOVE** information in T_1 not present in C; **(3) KEEP** the rest unchanged. The output is then truncated at the nearest sentence boundary to $\leq t_c$ characters. This single-pass design builds on Self-Refine (Madaan et al., 2023) but trades multi-step refinement for a single edit pass to control cost; the iterative analysis (App. G) shows 1-pass is Pareto-optimal. We retain Stage 2 verification on VC and Generic+Refine but not on the deployed configurations of CC 9-sec and LC progressive; an n=50 paired follow-up (App. M) finds no Holm-corrected effect of adding verify on any of the four methods, so the asymmetric design is method-neutral in expectation.

4 Experimental Setup

4.1 Datasets

We use four dialogue datasets spanning a numerical-density spectrum (Table 1): SAMSum (Gliwa et al., 2019), DialogSum (Chen et al., 2021), the Japanese composite of Business Scene Dialogue (Rikters et al., 2019) and the LLM-jp DeepL-translated subset hh-rlhf-12k-ja (LLM-jp, 2024) of the original Anthropic HH-RLHF dialogues (Bai et al., 2022), and MeetingBank (Hu et al., 2023). A Chinese cross-lingual replication on VCSum (Wu et al., 2023) is reported in Appendix L.

For downstream QA we use 3 LongBench tasks: qmsum (meeting QA, ROUGE-L, n=200), hotpotqa (multi-hop, F1, n=200), and multifieldqa_en (single-doc, F1, n=150 per the official LongBench split).

4.2 Models

Intrinsic compression: Qwen3.5-9B-MLX-4bit (deterministic local). **Intrinsic evaluation**: Gemma-4-26B-MoE-4bit (compression $\neq$ evaluation, to mitigate self-eval bias on FIR/G-Eval). **Downstream QA**: Claude Sonnet 4.6 produces both the compressed text and the QA answer due to API budget. We mitigate the resulting self-preference risk via automatic gold-answer metrics (F1, ROUGE-L), per-method achieved ratios, Pareto-frontier framing (§5.3), and a fully-local cross-model rerun spanning a 4-model QA-capacity spectrum (Qwen3.5-9B compressor; Phi-4-mini-3.8B / Llama-3.1-8B / Phi-4-14B / Gemma-4-26B-MoE QA; App. J). A Claude-side cross-compressor rerun (Claude compress $\rightarrow$ open-source QA) is infeasible because the deployed pipeline persisted only `compressed_len` and our Claude API budget is exhausted for the submission window; we treat this as a missing control rather than a missing primary result (Limitations).

4.3 Baselines and Disclosures

We compare 5 methods (full disclosures in Appendix A):

1. **9sec (Claude Code, community reproduction).** *Throughout this paper, "Claude Code (9-sec)" / "CC" refers to this baseline.* The publicly-disclosed reproduction of the Claude Code 9-section summarization prompt (`ccVersion 2.1.84`; Piebald-AI, 2024, by Piebald-AI; Anthropic does not publish the prompt officially); the `<analysis>` chain-of-thought step is omitted (conservative bias).

2. **LC (progressive)**: LangChain `ConversationSummaryBufferMemory` template with the character budget added (App. A). Two configurations: on intrinsic, two LLM calls (**LC 2-call**); on downstream, where the source is a single static document, the same template is invoked once with the source split at the midpoint (**LC 2-chunk**). Appendix B shows both.

3. **LC (N-call)**: faithful N-call iterative reproduction on intrinsic; N is determined by a buffer threshold of $t_c/2$ characters per turn and typically falls in 3–5 for our datasets. Not run on downstream.

4. **VC (proposed)**: 2-stage as described in §3.

5. **Generic+Refine**: simple summarize + verification step.

We do *not* add "copy EXACTLY" to the 9sec/LC baselines, as this is the core innovation of VC; including it would constitute baseline contamination. We also disclose that VC and Generic use 2 LLM calls while 9sec/LC use 1 call (not iso-compute); the iso-compute comparison is VC vs. Generic+Refine.

4.4 Metrics

Intrinsic (5 primary + 2 supplementary). Primary: ROUGE-1/2/L, Recall, FIR. **Recall** is the fraction of 3+-digit numbers and proper nouns from the source that appear verbatim in the compressed text (the 3-digit threshold excludes list ordinals). **FIR** (Future-Information Retention) is BERTScore F1 between the actual next 5 turns and a 3-turn continuation generated from the compressed text by Gemma-4. ROUGE uses Sudachi tokenization (JA, sumeval cross-validation, max diff <0.04) and `rouge_score` (EN). Supplementary: BERTScore F1 (tohoku-bert-japanese-v3 for JA, bert-base-uncased for EN; not directly cross-language comparable) and Compression Ratio. G-Eval (Liu et al., 2023) on four dimensions (n=30/lang) is in App. E. **Downstream.** LongBench (Bai et al., 2024) official `dataset2prompt.json` and `dataset2maxlen.json` verbatim, `rouge` (pltrdy) for ROUGE-L, `qa_f1_score` for F1.

4.5 Statistical Testing

Paired Wilcoxon signed-rank with Holm-Bonferroni correction; bootstrap 95% confidence intervals (1000 resamples). Tables in the main text report point estimates for compactness; the corresponding confidence intervals and per-test p-values are in Appendix C.

5 Results

5.1 Faithfulness axis

All five methods produce intrinsic outputs at near-equal achieved ratios (per-method means 0.42–0.48 on both languages, all within ±6% of target ρ=0.5); methods compete at matched information budgets, in contrast to the ∼4× spread seen in the downstream split (§5.3). We define the *top group* on a dataset as the set of methods that no other method beats with Holm-corrected $p < 0.05$ on any of the five intrinsic metrics (R-1, R-2, R-L, Recall, FIR). VC is the only method consistently in the top group across all four datasets: no baseline significantly outperforms VC on any metric, while VC significantly outperforms many baselines. Figure 2 shows the win-rate matrix. **VC defeats Claude Code 9-sec in 17/20 statistical tests and LangChain (N-call) in 17/20**; vs. LangChain (2-call) and Generic+Refine the wins are task-dependent (8/20 and 12/20), but in no test does either method significantly defeat VC.

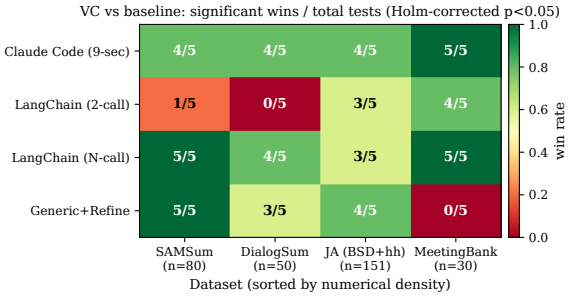

Figure 2: Faithfulness-axis win-rate matrix. Cells show (significant wins)/(total tests) for VC vs. each baseline.

Dataset	Method	R-1	R-L	Rec(%)	FIR
SAMSum	Claude Code (9-sec)	0.275	0.216	46.5	0.392
	LangChain (2-call)	0.435	0.388	50.2	0.398
	LangChain (N-call)	0.318	0.264	35.3	0.360
	Verified (ours)	0.492	0.438	60.7	0.410
	Generic+Refine	0.410	0.354	54.2	0.386
DialogSum	Claude Code (9-sec)	0.276	0.218	24.1	0.415
	LangChain (2-call)	0.438	0.362	30.7	0.474
	LangChain (N-call)	0.343	0.255	21.6	0.447
	Verified (ours)	0.465	0.416	37.2	0.425
	Generic+Refine	0.357	0.302	33.0	0.419
JA (BSD+hh)	Claude Code (9-sec)	0.345	0.201	58.1	0.600
	LangChain (2-call)	0.442	0.277	72.5	0.604
	LangChain (N-call)	0.494	0.362	74.8	0.628
	Verified (ours)	0.584	0.535	82.8	0.604
	Generic+Refine	0.393	0.240	58.3	0.598
MeetingBank	Claude Code (9-sec)	0.394	0.190	45.8	0.411
	LangChain (2-call)	0.421	0.276	36.6	0.447
	LangChain (N-call)	0.483	0.318	40.9	0.403
	Verified (ours)	0.642	0.641	68.5	0.449
	Generic+Refine	0.616	0.591	64.9	0.430

Table 2: Per-task per-method intrinsic results, evaluated at near-equal achieved compression ratios (per-method means 0.42–0.48 on both languages, all within ±6% of target ρ=0.5); rank comparisons are therefore not confounded by content-budget inequality. VC ranks first or tied-first on ROUGE-1, ROUGE-L, and Recall across all four datasets.

Effect sizes. Paired Cohen's d on ROUGE-L is positive in all 16 (dataset, baseline) cells (median d=1.13, 12/16 with d>+0.5). MeetingBank's tight clusters (SDs 0.018/0.033) inflate paired d; excluding it, direction stays positive on 12/12 remaining cells and VC ranks first or tied-first on every R-L cell (Table 2), so the "top group" claim does not depend on the inflated MeetingBank values.

5.2 Mechanism: anchoring and verify as two redundant levers

VC's faithfulness gap is produced by two prompt-level levers (Stage 1 anchoring, Stage 2 verify) that are partially redundant in the full pipeline. At the output level, VC's compressed text shares **83.3%** of its 5-grams verbatim with the source vs. 11.7–20.1% for baselines (App. H). A wording ablation (V1 "copy EXACTLY" vs. V4 "preserve faithfully"; Stage 1 only, n=20) is keyword-sensitive: V1 beats V4 by $\boldsymbol{d=+1.04}$ on R-L (p<0.001, App. K), consistent with Errica et al. (2025); the effect shrinks with verify (d=−0.16, n.s., n=13). Modarressi et al. (2025) document that long-context QA degrades sharply once literal overlap is removed—they interpret this as a benchmark limitation, but it also implies that the lexical-overlap retention VC maximises is a non-trivial determinant of long-context extractability.

An attention probe on the deployed Qwen3.5-9B and Llama-3.1-8B (paired V1/V4, n=15 MeetingBank dialogues each) shifts global source-attention in a consistent direction (V1 total < V4 total on 15/15 dialogues for each model). Qwen3.5-9B yields 8/128 Holm-significant heads; Llama-3.1-8B yields 0/1024 at the 8×-stricter Bonferroni threshold but with largest-magnitude effects of comparable order (App. P). A teacher-forced per-token probe (n=10 paired VC vs. Generic) finds VC's tokens more strongly *anchored* to source positions than Generic's on both compressors: peak source-attention d=+0.60 on Qwen3.5-9B and $\boldsymbol{d=+1.22}$ ($\boldsymbol{p=0.004}$) on Llama-3.1-8B, the latter additionally showing higher Spearman monotonicity (d=+0.82, p=0.027). V1 is used for both compressors even though it is sub-optimal on Llama (App. K), strengthening the prompt-level claim. Per-entity decomposition on MeetingBank shows VC vs. CC gains on every entity type (numbers +12.6pp, money +19pp, propn +16.3pp); against Generic the entity gap is small (±2.4pp), so VC's R-L lead over Generic comes from non-entity sentence structure that verify's KEEP preserves and Generic paraphrases away. Circuit-level identification is deferred to follow-up; we return to what this mechanism predicts about the downstream axis in §5.4.

QA model	$\bar{\Delta}$pp (VC−CC)	p_{raw}	p_{Holm}	d
Phi-4-mini-3.8B	+2.15	0.031	0.093	+0.40
Llama-3.1-8B	+2.98	0.012*	0.048*	+0.51
Phi-4-14B	−0.28	0.984	0.984	−0.07
Gemma-4-26B-MoE	−1.80	0.152	0.304	−0.21

Table 3: VC vs. Claude Code 9-sec on qmsum at ρ_{tgt}=0.5 across a QA-capacity spectrum (n=30 each, paired Wilcoxon two-sided). All four QA configurations are fully local MLX runs with zero failure-as-error rows, so the success-only and failure-as-zero aggregations coincide. p_{Holm} is Holm–Bonferroni-corrected across these four capacity tests; * marks p<0.05 after correction. Llama-8B is the only Holm-significant cell; Phi-mini-3.8B is raw-significant but does not survive Holm; both 14B+ configurations are clearly null.

5.3 Capacity gating at moderate compression

QA-capacity spectrum. We replicate the qmsum downstream analysis at fixed ρ_{tgt}=0.5 (Qwen3.5-9B compressor, n=30) across four QA models spanning 3.8 to 26B parameters (Table 3; full per-baseline table in App. J).

The faithfulness gap translates into a Holm-significant downstream advantage on Llama-8B (p_{Holm}=0.048, d=+0.51) and a same-direction raw-significant trend on Phi-mini-3.8B (p=0.031 raw, p_{Holm}=0.093, d=+0.40); both 14B+ configurations are clearly null. Architectures vary across the four points, but the within-Phi-4 contrast below removes architecture from the comparison. **Within the Phi-4 family** alone (3.8B → 14B), the same compressed inputs go from p=0.031 raw / p_{Holm}=0.093 (d=+0.40) to null (p=0.98, d=−0.07), partially controlling for architecture (one family, two scales). We read this as a contrast of two separately-tested cells rather than a formal interaction test: no test of H_0: $d_{3.8\text{B}}$=$d_{14\text{B}}$ is run, so the move from raw-significant to null is best interpreted descriptively, supported by the four-point monotone pattern in Table 3. *Why*: a plausible reading is that larger QA models can reconstruct task-relevant entities from paraphrastic input by relying on broader contextual cues, so faithfulness gaps persist in the compressed text but no longer constrain answer extraction.

	Tgt	ach.	qmsum		hpqa		mfqa		Mean		
Method	ρ	avg	succ	fail0	succ	fail0	succ	fail0	succ	fail0	#err
Claude Code (9-sec)	0.25	0.23	18.5	16.8	62.2	60.0	34.8	32.5	38.5	36.4	36
	0.50	0.39	18.6	16.3	65.9	61.6	39.1	36.8	41.2	38.2	47
	0.75	0.49	18.5	15.0	60.3	52.2	41.0	36.4	39.9	34.5	82
LC (2-chunk)	0.25	0.10	19.0	18.2	59.5	57.1	37.3	35.6	38.6	37.0	24
	0.50	0.12	19.5	18.9	58.3	56.0	40.7	38.6	39.5	37.8	22
	0.75	0.13	19.4	19.0	60.8	59.3	43.8	40.6	41.3	39.6	20
Verified (ours)	0.25	0.17	19.3	17.5	61.9	58.8	39.9	36.7	40.4	37.7	41
	0.50	0.24	19.4	17.4	62.8	56.2	43.1	40.3	41.8	38.0	52
	0.75	0.27	19.4	17.6	62.0	49.9	43.6	39.8	41.7	35.8	70
Generic+Refine	0.25	0.15	18.4	17.6	65.0	62.1	37.3	34.4	40.3	38.0	30
	0.50	0.22	18.0	17.6	65.4	62.8	41.3	38.2	41.6	39.6	24
	0.75	0.25	17.9	17.1	65.0	61.7	42.3	38.9	41.7	39.3	31
Original (no comp.)	–	1.00	23.1	22.9	68.6	67.5	56.1	54.2	49.2	48.2	10

Table 4: Downstream QA on LongBench, disaggregated by target compression ratio $\rho_{\text{tgt}} \in \{0.25, 0.5, 0.75\}$. **ach. avg**: achieved ratio averaged over three tasks (per-task ach. ranges discussed in text). Per task we report **succ** (success-only mean) and **fail0** (failure-as-zero mean, conservative reading; %-scale). #err counts items skipped due to compression/eval failures (denominator N=200 for qmsum/hpqa, N=150 for mfqa, per target ratio). At ρ_{tgt}=0.75 per-task achieved ratios span LC 0.07–0.20, Verified 0.13–0.35, Generic 0.14–0.34, CC 0.43–0.56—a $\sim$4$\times$ spread we discuss in §5.3.

API downstream as independent replication. The API downstream (Claude Sonnet 4.6 compressor and QA, scored by automatic gold-answer metrics; Table 4, N=200 per task) sits in the high-capacity regime: the four compressing methods cluster within $\sim$3pp regardless of aggregation. Under success-only, VC@ρ=0.5 leads (41.8%; Generic 41.6%, LC@0.75 41.3%, CC 41.2%); under failure-as-zero, the rank flips to Generic and LC ($\approx$ 39.6%). VC has higher hotpotqa eval-failure rates (21/200 vs. Generic 8/200 at ρ=0.5), so neither reading is canonical and we report both. Holm-corrected Wilcoxon yields 3 small significant wins for VC on qmsum (d=0.20–0.29) and no significant losses; on hotpotqa and mfqa no contrast is significant. The $\sim$3pp cluster is consistent with the Phi-4-14B point of the capacity spectrum: a capacity-rich QA model masks the compression-quality difference.

5.4 Conditional decoupling: a channel–reader framing

The faithfulness–utility relationship is *conditionally decoupled*: cross-axis Pearson on 8 method points (App. C) is r=0.774 (p=0.024, bootstrap 95% CI $[0.43, 0.99]$); four points are VC ablations and the axes mix JA faithfulness with EN downstream, so r is a cross-lingual descriptive aggregate. The faithfulness spread of 0.33 maps to a downstream spread of 0.04 at ρ=0.5/Phi-14B (an 8.25$\times$ contraction); the contraction collapses at low QA capacity ($\sim$3pp downstream spread, Table 3) and partially re-opens under aggressive ρ (§7).

A channel–reader framing unifies the two findings. Compression sits between a source X and a downstream reader R: compressor C produces $C(X)$, and R infers a task answer $\hat{Y}=R(C(X), q)$ given query q. The faithfulness axis measures the *channel*: how much of X's information $C(X)$ retains. The downstream axis measures the *joint capacity* of C and R: whether $\hat{Y}$ recovers the gold answer. Factorising the pipeline this way makes the empirical results in §5.2–§5.3 joint predictions rather than independent findings, and casts the contributions of Nagle et al. (2024) (compression as rate-distortion) and Liang et al. (2023) (multi-axis evaluation) as two faces of the same factorisation.

Reader side (capacity-gating). The projection from the faithfulness axis onto the downstream axis is gated by the slack in R's reconstruction capacity. When R is capacity-rich, any sufficiently information-preserving $C(X)$ admits answer recovery and the projection collapses; when R is bounded, recovery becomes channel-limited and the projection opens. The 8.25$\times$ contraction at $\geq$14B and the re-opening at 3.8–8B (Tables 3, 4) are the empirical signature. This reframes the multi-axis recommendation of Liang et al. (2023) as a structural *necessity*: single-axis (downstream-only) evaluation in the capacity-rich regime systemati-

cally *masks* the faithfulness ranking, so a method may be optimal for low-capacity deployment (edge, on-device, long-tail dialogue length) yet appear indistinguishable on capacity-rich leaderboards. The "compression buys nothing" conclusion from large-model evaluation is, on this reading, a measurement artefact of the capacity-rich regime—falsifiable through the 8B→14B threshold.

Channel side (prompt-level copy mechanism). Pointer mechanisms (See et al., 2017; Vinyals et al., 2015) were engineered to copy source tokens via an architectural switch; we observe that the prompt-level instruction "copy EXACTLY" elicits the same computational signature in instruction-tuned LLMs without architectural change. If the two mechanisms are functionally equivalent at the channel side, three predictions follow: **(P1)** wording differences should shift attention mass at specific source-attending heads; **(P2)** output tokens should be monotonically anchored to source positions; **(P3)** the mechanism should fail on tasks where literal copy is misaligned with task semantics. All three are observed: (P1) $8/128$ Holm-significant heads on Qwen3.5-9B with same-direction underpowered effects on Llama-3.1-8B (§5.2); (P2) per-token attribution $d{=}{+}1.22$ peak attention and $d{=}{+}0.82$ Spearman monotonicity on Llama-3.1-8B vs. Generic; (P3) hotpotqa multi-hop failure and "#5" vs. "number five" notation rigidity on factoid QA (App. F). The convergence of P1–P3 reframes the attention probe and per-token attribution from descriptive correlates to predicted consequences of a single underlying mechanism: at the channel side, prompt-elicited copy is the instruction-tuned LLM's functional analogue of the architectural pointer circuit.

Joint reading. The two findings are dual consequences of the factorisation $\hat{Y}{=}R(C(X), q)$: "copy EXACTLY" moves C toward the near-lossless channel limit, and the resulting information preservation only translates into downstream advantage when R's reconstruction slack is small enough to make the channel the binding constraint. The faithfulness-axis spread holds across the numerical-density spectrum (Fig. 3 in App. H), so the channel-side claim is not genre-specific.

6 Ablation Studies

6.1 Component Ablation

We disable each of VC's three design components in turn: **A1** lexical anchoring ("copy EXACTLY"), **A2** participant framing ("you are a participant"), and **A3** the verification post-processing step. Removing all three reduces VC to a generic summarize-then-refine pipeline. We evaluate on Japanese BSD+hh-rlhf-ja ($n{=}30$) and English MeetingBank ($n{=}30$) using the intrinsic metrics of §5.

Variant	R-1	R-L	Recall
Full VC	**0.602**	**0.539**	**83.1%**
−A1 (no Copy)	0.522	0.433	73.5%
−A2 (no Frame)	0.467	0.321	77.8%
−A3 (no Verify)	0.587	0.530	80.4%
Generic+Refine	0.395	0.244	52.0%

Table 5: Component ablation on Japanese BSD+hh-rlhf ($n{=}30$). Each axis (A1 lexical anchoring, A2 participant framing, A3 verification) contributes independently. Removing A2 has the largest impact on ROUGE/Recall; removing A3 has small effects (n.s. after Holm, §6.1). English MeetingBank ($n{=}30$) shows the same direction with smaller magnitude (R-L drops $0.041/0.097/0.023$ on $-$A1/$-$A2/$-$A3 relative to Full VC R-L 0.641); we report inferential statistics for the Japanese sample only.

Findings. Paired Wilcoxon+Holm. **A2 (framing)**: significant in both languages (JA $d{=}{+}1.66$, EN $d{=}{+}0.64$, $p_{\text{Holm}}{\leq}0.034$). **A1 (anchoring)**: significant on JA ($d{=}{+}0.92$, $p_{\text{Holm}}{<}0.001$), same-direction on EN ($p_{\text{Holm}}{=}0.19$ at $n{=}30$). **A3 (verify)**: small non-significant ROUGE/Recall effects ($d{=}{+}0.12$–$+0.27$); value is in iterative-pass behaviour. A2 carries the largest single-component R-L effect; A1 is analysed mechanistically in §5.2.

Multi-model and iterative robustness. VC ranks first on R-L across three compressor architectures on DialogSum (App. I); 1-pass verify is Pareto-optimal on MeetingBank (App. G).

7 Conclusion: Scope and Selection Rule

VC's scope excludes **(i)** multi-turn cumulative (App. O; LC leads), **(ii)** multi-hop QA (App. F; Generic+Refine leads), **(iii)** aggressive $\rho{=}0.05$ (App. N; mfqa $d{=}{+}0.18$, n.s.). Selection rule (Table 3): single-turn faithfulness-critical → VC; $\rho{\geq}0.5$/QA-$\geq$14B → interchangeable; multi-turn → LC; multi-hop → Generic+Refine—faithfulness–utility coupling is capacity-gated, not universal.

Limitations

- **Self-preference risk in downstream evaluation**. The same Claude Sonnet 4.6 pro-

duces both the compressed text and the QA answer on the three LongBench tasks (API quota constraint). QA metrics (F1, ROUGE-L) are automatic gold-answer scores rather than judge calls, but a same-model self-preference bias may remain. We mitigate via: (i) Gemma-4-26B-MoE for intrinsic G-Eval and FIR (different model); (ii) achieved-ratio disclosure (Table 4); (iii) Pareto-frontier rather than winner framing; (iv) a fully-local cross-compressor rerun (Qwen3.5-9B compress; Phi-4-mini-3.8B / Llama-3.1-8B / Phi-4-14B / Gemma-4-26B-MoE QA; App. J) that reproduces the capacity-gated pattern without API. A Claude-side cross-compressor control was infeasible (the deployed pipeline persisted only `compressed_len`, and our API budget is exhausted); we treat this as a missing *control* rather than a missing primary result.

- **Multilingual scope is limited.** Intrinsic spans three languages (EN, JA, ZH; ZH in App. L); downstream QA is English-only because comparable JA/ZH benchmarks were unavailable in the submission window.

- **No human evaluation**. We use automatic metrics (ROUGE, Recall, FIR); human raters were infeasible in the submission window.

- **Compute cost is $2\times$ structured baselines.** VC needs two LLM calls (Stage 1 + Stage 2); iso-compute comparison is vs. Generic+Refine. App. G shows 1-pass is the Pareto sweet spot.

- **Capacity-gated downstream effect is the central finding.** Per Table 3, the threshold lies between 8B and 14B QA on ρ=0.5/qmsum, and the cross-axis Pearson contraction (App. C.4) is single-regime. We frame this as a capacity-gated decoupling rather than orthogonality.

- **Anchoring and verify are partially redundant** (§5.2, App. K). Practitioners using the full pipeline can use weaker wordings; deployments that skip Stage 2 should retain "copy EXACTLY".

- **Single-turn anchoring does not transfer to multi-turn.** Under cumulative compression (App. O, n=20, n.s.) VC ranks third (17.2%) behind LC progressive (18.1%) and CC 9-sec (17.8%); "copy EXACTLY" does not separate source from prior-summary content. We confine claims to single-turn.

- **Aggressive-ratio recovery attenuates with sample size.** The n=30 pilot at ρ=0.05/Phi-4-14B QA showed a raw-significant mfqa vs. Generic contrast (p=0.017, d=+0.50); the n=100 follow-up (App. N) attenuates to d=+0.18 (p=0.28), and on qmsum VC trails LC progressive (raw p=0.020, Holm n.s.). We treat the n=100 as the primary estimate (winner's-curse pattern at the pilot n).

- **Other scope notes.** (a) *Sample size*: ablation, multi-model, and iterative experiments use n=20–30, consistent with the larger intrinsic (n=151) and downstream (n=150–200) runs. (b) *Asymmetric refinement* (verify on VC/Generic only): an n=50 follow-up (App. M) finds no Holm-significant effect of adding verify, so the design is method-neutral in expectation but post-hoc. (c) *Iterative passes* capped at 4 (App. G); passes 5–10 untested. (d) *Mistral-Nemo-12B* excluded (Metal OOM at long context on 36 GB); the remaining three architectures span three families. (e) *Future work*: task-aware Stage-2, a Claude-side cross-compressor control, and a full 4-QA$\times$multi-ratio sweep are deferred.

- **Baselines and out-of-scope methods.** CC 9-sec is the community reproduction (Anthropic does not publish the prompt; `<analysis>` omitted, App. A); LangChain is a default template; claims are about these configurations. Token-level prompt compression (LLMLingua (Jiang et al., 2023, 2024; Pan et al., 2024), RECOMP (Xu et al., 2024)) is a different design space (surface-token retention) and left to future work.

Ethical Considerations

Datasets. All datasets are public, widely used in NLP research, and accessed via HuggingFace at standard splits: SAMSum (CC BY-NC-ND 4.0), DialogSum (CC BY-NC-SA 4.0), MeetingBank (CC BY-NC-SA 4.0), BSD (CC BY-SA 4.0), HH-RLHF (MIT), VCSum (Apache 2.0; Chinese cross-lingual replication, App. L), and LongBench (MIT). We use them strictly for research evaluation and we do not redistribute the underlying texts.

The dialogues contain consenting participants in scripted/role-played or already-anonymised interactions; we did not collect new human data.

API usage and data handling. Downstream QA evaluation sends compressed dialogue passages and questions to the Anthropic API (Claude Sonnet 4.6). Per Anthropic's commercial terms, API inputs are not used to train models by default. We did not transmit any private or personally-identifying content beyond what is already public in the cited datasets.

Model artefacts. The local compression and judge models (Qwen3.5-9B, Llama-3.1-8B, Phi-4-mini-3.8B, Phi-4-14B, Gemma-4-26B-MoE; all 4-bit MLX community quantizations) are released under their respective upstream licenses for research use. We do not redistribute weights.

Generated text. Compressed dialogue summaries produced by our pipeline may omit, propagate, or alter sensitive content present in the source dialogue (numbers, named entities, attributions). VC reduces missing-entity errors relative to baselines but does not eliminate them; downstream consumers should not treat any LLM-produced compression as a fully faithful record.

Generative AI tooling. We used a large language model (LLM) assistant for language-side support (English wording, including occasional Japanese-to-English phrasing assistance; paragraph-structure review; LaTeX boilerplate) and limited code-side support (test code, debugging). This falls under category (a) of the ACL Policy on Publication Ethics. We produced and verified all scientific contributions ourselves—research questions, experimental design, algorithmic logic of the compression pipeline and evaluation framework, numerical results, statistical analyses, and interpretations—and take full responsibility for the manuscript.

References

Anthropic. 2025. Effective context engineering for AI agents. Anthropic engineering blog. https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents. Accessed 2026-05.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, and 12 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3639–3664, Vienna, Austria. Association for Computational Linguistics.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2018. Faithful to the original: Fact aware neural abstractive summarization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 4784–4791. AAAI Press.

Harrison Chase. 2022. Langchain: Building applications with LLMs through composability. GitHub repository. https://github.com/langchain-ai/langchain. Accessed 2025-09.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online. Association for Computational Linguistics.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore. Association for Computational Linguistics.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready AI agents with scalable long-term memory. *Preprint*, arXiv:2504.19413.

Tsz Ting Chung, Leyang Cui, Lemao Liu, Xinting Huang, Shuming Shi, and Dit-Yan Yeung. 2024. Selection-p: Self-supervised task-agnostic prompt compression for faithfulness and transferability. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11057–11070, Miami, Florida, USA. Association for Computational Linguistics.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2024. Chain-of-verification reduces hallucination in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3563–3578, Bangkok, Thailand. Association for Computational Linguistics.

Federico Errica, Davide Sanvito, Giuseppe Siracusano, and Roberto Bifulco. 2025. What did I do wrong? quantifying LLMs' sensitivity and consistency to prompt engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1543–1558, Albuquerque, New Mexico. Association for Computational Linguistics.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Rajarshi Haldar and Julia Hockenmaier. 2025. Rating roulette: Self-inconsistency in LLM-as-a-judge frameworks. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 24986–25004, Suzhou, China. Association for Computational Linguistics.

Yebowen Hu, Timothy Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. 2023. MeetingBank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16409–16423, Toronto, Canada. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Weronika Łajewska, Momchil Hardalov, Laura Aina, Neha Anna John, Hang Su, and Lluis Marquez. 2025. Understanding and improving information preservation in prompt compression for LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17520–17541, Suzhou, China. Association for Computational Linguistics.

Jinyi Li, Yihuai Lan, Lei Wang, and Hao Wang. 2024. PCToolkit: A unified plug-and-play prompt compression toolkit of large language models. *Preprint*, arXiv:2403.17411.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2025a. Prompt compression for large language models: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7182–7195, Albuquerque, New Mexico. Association for Computational Linguistics.

Zongqian Li, Yixuan Su, and Nigel Collier. 2025b. 500xCompressor: Generalized prompt compression for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25081–25091, Vienna, Austria. Association for Computational Linguistics.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

LLM-jp. 2024. hh-rlhf-12k-ja: A Japanese translation of the Anthropic HH-RLHF dialogue subset. HuggingFace dataset. DeepL-translated 12k subset of Bai et al. (2022); `https://huggingface.co/datasets/llm-jp/hh-rlhf-12k-ja`. Accessed 2025-09.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang,

Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13851–13870, Bangkok, Thailand. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schütze. 2025. NoLiMa: Long-context evaluation beyond literal matching. In *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025)*.

Jesse Mu, Xiang Lisa Li, and Noah D. Goodman. 2023. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Alliot Nagle, Adway Girish, Marco Bondaschi, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. 2024. Fundamental limits of prompt compression: A rate-distortion framework for black-box language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

Piebald-AI. 2024. Claude code conversation summarization prompt (community reproduction, ccversion 2.1.84). GitHub repository. Community reproduction of Anthropic's Claude Code 9-section summarization prompt; `https://github.com/Piebald-AI/claude-code-system-prompts`. Accessed 2025-09.

Xiao Pu, Tianxing He, and Xiaojun Wan. 2024. Stylecompress: An LLM-based prompt compression framework considering task-specific styles. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14533–14549, Miami, Florida, USA. Association for Computational Linguistics.

Matīss Rikters, Ryokan Ri, Tong Li, and Toshiaki Nakazawa. 2019. Designing the business conversation corpus. In *Proceedings of the 6th Workshop on Asian Translation*, pages 54–61, Hong Kong, China. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Shivam Shandilya, Menglin Xia, Supriyo Ghosh, Huiqiang Jiang, Jue Zhang, Qianhui Wu, Victor Rühle, and Saravan Rajmohan. 2025. TACO-RL: Task aware prompt compression optimization with reinforcement learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 1582–1597, Vienna, Austria. Association for Computational Linguistics.

Jiwei Tang, Jin Xu, Tingwei Lu, Zhicheng Zhang, Yiming Zhao, Lin Hai, and Hai-Tao Zheng. 2025. Perception compressor: A training-free prompt compression framework in long context scenarios. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4093–4108, Albuquerque, New Mexico. Association for Computational Linguistics.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*.

David Wan, Jesse Vig, Mohit Bansal, and Shafiq Joty. 2025. On positional bias of faithfulness for long-form summarization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8791–8810, Albuquerque, New Mexico. Association for Computational Linguistics.

Han Wu, Mingjie Zhan, Haochen Tan, Zhaohui Hou, Ding Liang, and Linqi Song. 2023. VCSUM: A versatile Chinese meeting summarization dataset. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6065–6079, Toronto, Canada. Association for Computational Linguistics.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. 2025. HELMET: How to evaluate long-context language models effectively and thoroughly. In *The Thirteenth International Conference on Learning Representations (ICLR 2025)*.

Ziyang Yu and Yuyu Liu. 2025. ICPC: In-context prompt compression with faster inference. *Preprint*, arXiv:2501.01625.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

A Full Disclosure of Design Choices

A.1 Baseline implementation choices

1. **Claude Code 9-sec**: implements the 9 numbered summary sections of the prompt verbatim from the Piebald-AI community reproduction (Piebald-AI, 2024) of `ccVersion 2.1.84`; Anthropic does not officially publish the prompt, so we cannot verify byte-level fidelity to the deployed system. The explicit `<analysis>` chain-of-thought pre-output step is omitted (the 9 summary sections themselves are kept; full prompt in App. B, §B.4), yielding a conservative bias (the deployed Claude Code may perform better than what we measure here).

2. **No "copy EXACTLY" added to baselines**: this is the core innovation of VC; adding it to baselines would constitute contamination.

3. **Character-budget instruction added to LangChain**: the official template has none; we add it to all baselines (and VC) for fair comparison.

4. **Refinement is method-specific**: an $n{=}50$ paired follow-up (App. M) found no Holm-corrected significant effect of adding the verify step on any of the four methods. The asymmetric design (verify on VC and Generic, no verify on CC and LC) matches each baseline's deployed configuration and is method-neutral in expectation.

A.2 Model selection rationale

We chose model variants that fit a single 36 GB Apple M4 Pro Mac at MLX 4-bit quantization, prioritizing the latest stable generation in each family that fits this hardware: **Qwen3.5-9B**, **Llama-3.1-8B-Instruct** (the largest Llama that fits; Llama-3.3-70B is at the memory boundary and Llama-4-Scout/Maverick MoE quantizations are too large for 36 GB at 4-bit), and **Phi-4-14B** (the current-generation Phi at the largest scale that fits; we use the same Phi-4-14B in both the multi-model generalization study, §I, and the cross-model verification, App. J, so no Phi family disambiguation is needed). Mistral-Nemo-12B at MLX 4-bit (the 4-bit weights themselves are $\sim$6 GB) triggered Metal OOM on this hardware during long-context generation; the peak footprint comes from the cumulative KV-cache plus per-layer activations at 128,000-token context, which together exceed our 36 GB unified-memory budget. The weights alone fit; only the long-context generation pass does not. Smaller Mistral variants (e.g. Mistral-7B-v0.3) were not tried in the submission window.

A.3 Compute disclosure

- Compression: Qwen3.5-9B-MLX-4bit on Apple M4 Pro 36GB
- Evaluation (FIR continuation generation and G-Eval scoring): Gemma-4-26B-MoE-4bit on the same hardware
- Downstream QA: Claude Sonnet 4.6 via Anthropic API
- Total wall-clock: $\sim$60 hours of local MLX runs + Anthropic API

A.4 Generic+Refine baseline definition

Generic+Refine is constructed as Stage 1 = Generic summarization prompt (App. B: "Summarize the following context in {tc} characters or less.") followed by Stage 2 = the same verification prompt used by VC. Hence Generic+Refine matches VC on call count (2 LLM calls) and on the post-hoc verification step, but uses a generic Stage 1 prompt *without* participant framing or lexical anchoring. This makes Generic+Refine the iso-compute, no-VC-prompt comparison against which VC's prompt-design contribution is isolated. The component ablation in Table 5 is therefore best read as: Full VC $\rightarrow$ $-$A1 $\rightarrow$ $-$A2 $\rightarrow$ $-$A3 successively peels back each VC component while keeping the other two and the verify step, and the Generic+Refine row at the bottom is what remains when both A1 and A2 are absent (with verify retained).

A.5 Recall metric

Recall correlates with our "copy EXACTLY" instruction. We report the four other primary intrinsic metrics (ROUGE-1/2/L, FIR), two supplementary metrics (BERTScore, Compression Ratio), G-Eval (Appendix E), and three downstream tasks; conclusions are not driven by Recall alone.

A.6 Downstream evaluation: same model for compression and QA

We use Claude Sonnet 4.6 for both compression and QA in the Claude→Claude downstream experiment due to API budget constraints during the submission window. A fully-local cross-model rerun (Qwen3.5-9B compressor; Llama-3.1-8B, Phi-4-mini-3.8B, Phi-4-14B, and Gemma-4-26B-MoE QA — a 4-model capacity spectrum) is reported in Appendix J and confirms VC's edge over Claude Code is not API-driven; a Claude-side cross-model rerun would require re-executing compression because the deployed pipeline persisted only `compressed_len`, not `compressed_text`, and is deferred to camera-ready.

B Prompt Templates

B.1 VC Stage 1: Participant-Framed Compression

```
You are a participant in this conversation. You will continue afterwards. Compress the following context to {tc} characters or less so you can naturally continue.

CRITICAL RULES (must follow exactly):
- Copy all numbers (amounts, dates, times, percentages) EXACTLY as they appear, character by character
- Copy all proper nouns (names, organizations, products, places) EXACTLY as they appear

Other rules:
- State WHO did/said what for every action
- Keep all unresolved questions and pending items
- Preserve agreed decisions with reasoning
- You may omit greetings, acknowledgments, and filler responses

Context:
{ctx}

Compressed:
```

B.2 VC Stage 2: Verification (also used by Generic+Refine)

```
Compare the original context with the compressed text and output a corrected version in {tc} characters or less.

Correction rules:
1. Add: Include any numbers, dates, and proper nouns from the original that are missing
2. Remove: Delete any information not present in the original (hallucination)
3. Keep: Preserve everything else as-is

Original context:
{ctx}

Compressed text:
{compressed}

Corrected version:
```

B.3 Generic (Stage 1 of Generic+Refine)

```
Summarize the following context in {tc} characters or less.

Context:
{ctx}

Summary:
```

B.4 Claude Code 9-section baseline (community reproduction)

Verbatim from Piebald-AI (2024) (Piebald-AI community reproduction of Claude Code `ccVersion 2.1.84`); the `<analysis>` chain-of-thought step is omitted (§A):

```
Your task is to create a detailed summary of the following conversation, paying close attention to each participant's explicit requests and actions taken. This summary should be thorough in capturing details, decisions, and context that would be essential for continuing the conversation without losing context. Compress to {tc} characters or less.

Before providing your final summary, organize your thoughts to ensure you've covered all necessary points:

1. Chronologically analyze each section of the conversation. For each section thoroughly identify:
  - Each participant's explicit requests and intents
  - Key decisions and reasoning behind them
  - Specific details like names, dates, amounts
  - Problems discussed and their resolutions
  - Pay special attention to commitments and agreements made
2. Double-check for technical accuracy and completeness, addressing each required element thoroughly.

Your summary should include the following 9 sections:

1. Primary Request and Intent
2. Key Technical Concepts
3. Key Statements and Topics (state WHO said/did what)
4. Errors and Corrections
5. Problem Solving
6. All Participant Statements
7. Pending Tasks
8. Current State
9. Next Steps

Context:
{ctx}

Summary:
```

B.5 LangChain progressive baseline (LC)

LangChain `ConversationSummaryBufferMemory` default template (Chase, 2022; source file: `libs/langchain/langchain/memory/summary_buffer.py`). We add the character budget instruction (§A); the example block is unchanged from upstream:

```
Progressively summarize the lines of
conversation provided, adding onto the
previous summary returning a new summary.
Keep the new summary to {tc} characters
or less.

EXAMPLE
Current summary:
The human asks what the AI thinks of
artificial intelligence. The AI thinks
artificial intelligence is a force for
good.

New lines of conversation:
Human: Why do you think artificial
intelligence is a force for good?
AI: Because artificial intelligence will
help humans reach their full potential.

New summary:
The human asks what the AI thinks of
artificial intelligence. The AI thinks
artificial intelligence is a force for
good because it will help humans reach
their full potential.
END OF EXAMPLE

Current summary:
{first}

New lines of conversation:
{second}

New summary:
```

For the intrinsic experiment we instantiate this template *twice* (true 2-call use of LangChain's `predict_new_summary` API). For the downstream LongBench experiment, where the source is a single static document rather than a streaming dialogue, we instead split the source at the midpoint into {`first`} and {`second`} and issue a *single* call to the same template; we therefore label this condition **LC (2-chunk progressive)** on downstream and **LC (2-call progressive)** on intrinsic, and disclose the difference in §A.

B.6 LongBench downstream QA prompts (verbatim)

We use the LongBench official `dataset2prompt.json` verbatim (Bai et al., 2024):

```
qmsum: You are given a meeting
transcript and a query containing a
question or instruction. Answer the
query in one or more sentences.
Transcript: {context}
Now, answer the query based on the
above meeting transcript in one or more
sentences. Query: {input}
Answer:

hotpotqa: Answer the question based
on the given passages. Only give me
the answer and do not output any other
words.
The following are given passages.
{context}
Question: {input}
Answer:

multifieldqa_en: Read the following text
and answer briefly.
{context}
Now, answer the following question based
on the above text, only give me the
answer and do not output any other words.
Question: {input}
Answer:
```

Maximum generation lengths are LongBench official `dataset2maxlen.json`: qmsum 512, hotpotqa 32, multifieldqa_en 64.

C Statistical Test Details

C.1 Method

We use the paired Wilcoxon signed-rank test (`scipy.stats.wilcoxon`, `alternative=two-sided`) with Holm-Bonferroni correction within each (dataset, baseline) cell of the win-rate matrix (5 metrics per cell for the intrinsic analysis; 3 baselines per (task, ratio) cell for the downstream analysis). Bootstrap 95% CIs use 1000 resamples with `numpy.random.default_rng(42)`. For Cohen's d on paired data we use $d = \bar{X}_{\text{diff}}/s_{\text{diff}}$ where s_{diff} is the sample SD of paired differences (`ddof=1`). All experiments use random seed 42.

Note on independence of the 5 intrinsic metrics. ROUGE-1, ROUGE-2, ROUGE-L, Recall, and FIR are not statistically independent (ROUGE-1 and ROUGE-L share unigram structure; Recall overlaps with ROUGE on lexical content). The Holm-Bonferroni procedure controls the family-wise error rate *conservatively* under any dependence structure, but the resulting per-cell wins ($x/5$) are correlated and the headline 17/20 win counts should be read as descriptive aggregates rather than as independent successes. We retain Holm rather than a more liberal correction (e.g. BH-FDR) because the headline claims rest on direction-of-effect, not on declaring $5{\times}4{=}20$ independent effects.

C.2 Intrinsic faithfulness contrasts

Table 6 reports per-test statistics for the headline faithfulness claim (VC vs. each baseline on each of five primary metrics, paired by dialogue, separately for Japanese and English). Significance markers reflect the Holm-corrected p-values within each (language, baseline) cell.

C.3 Downstream-axis contrasts

Table 7 reports the corresponding paired statistics for downstream QA (VC vs. each compressing baseline on each of three LongBench tasks at three target ratios). Pairing is by item, restricted to items where both methods returned a valid answer; the resulting effective n is shown in the table. VC significantly beats CC and Generic on qmsum at ρ=0.5/0.75 (small effect sizes, d=0.20–0.29); all hotpotqa and mfqa contrasts are non-significant after Holm correction; and no baseline significantly defeats VC anywhere. The overall pattern supports the narrow downstream-axis description in §5.3.

Per-task failure counts. The Table 4 #err column aggregates over three tasks. The hotpotqa-only counts referenced in §5.3 (where VC's error rate exceeds Generic's most clearly) are: VC $21/200$ vs. Generic $8/200$ at ρ_{tgt}=0.5, and VC $39/200$ vs. Generic $10/200$ at ρ_{tgt}=0.75. These are the comparisons that motivate the dual succ/fail0 reporting.

Failure analysis. A row is logged as a failure (#err) when the compression call itself does not return a usable output: empirically, all error rows in our logs have empty `prediction`, `compressed_len`=`None`, and `achieved_ratio`=`None`, indicating the compressor API call on the source dialogue did not produce text.

Failure rate is driven by `ctx_len`, not method. Binning all 7156 method-item rows by the source's `ctx_len` reveals a sharp regime change at $\sim$100 000 characters:

Method	<10k	10−30k	30−60k	60−100k	>100k
CC 9-sec	8%	7%	6%	10%	**85%**
LC 2-chunk	3%	8%	3%	3%	2%
Verified	10%	8%	5%	11%	**69%**
Generic	13%	9%	3%	4%	4%

Below 60k characters all four methods fail at comparable rates (3–13%). Above 100k characters the verbose-prompt methods — CC 9-sec's nine-section structured prompt and VC's two-stage prompt with explicit copy directives — collapse to 85% and 69% failure, while LC's progressive split and Generic's brief prompt remain at 2–4%. We interpret this as verbose-prompt methods reaching the API's context-handling or timeout limit on the longest qmsum and hotpotqa items, where the prompt overhead plus source plus output exceeds the practical budget. The aggregated counts across ρ_{tgt} thus reflect this length-driven cliff:

Method	qmsum	hotpotqa	mfqa
CC 9-sec	$82/601$	$47/600$	$36/450$
LC 2-chunk	$19/600$	$21/600$	$26/450$
Verified	$58/600$	$70/600$	$35/450$
Generic	$23/600$	$27/600$	$35/450$

VC's higher hotpotqa failure rate (e.g. $21/200$ vs. Generic's $8/200$ at ρ_{tgt}=0.5) is therefore not driven by VC's prompt design *per se* but by hotpotqa's unusually long passages combined with VC's two-stage verbose prompt. The dual succ/fail0 reading in §5.3 accounts for this length-driven asymmetry between methods.

C.4 Cross-axis Pearson, extended to n=8 methods

To test the faithfulness–utility correlation on a wider method set than the four main compressing methods, we ran four ablation variants (−A1, −A2, −A3, −A1A2A3) on the local cross-model qmsum downstream pipeline (Qwen3.5-9B compressor, Llama-3.1-8B QA; n=31–32 items per method, ρ_{tgt}=0.5). Their faithfulness scores are taken from the existing Japanese ablation (n=30, §6.1 / Table 5). Per-method values:

Method	Faith (JA R-L)	qmsum DS
CC 9-sec	0.20	0.17
Generic+Refine	0.24	0.17
−A1A2A3	0.24	0.16
LC 2-call	0.28	0.18
−A2	0.32	0.20
−A1	0.43	0.20
−A3	0.53	0.20
Verified (VC)	0.53	0.20

Pearson r=0.774 (p=0.024, n=8); Spearman ρ=0.690 (p=0.058, n=8). The faithfulness range 0.20–0.53 (Δ=0.33) maps to a downstream range 0.16–0.20 (Δ=0.04): an 8.25× compression of the faithfulness span into the downstream span, consistent with the "effect-size disparity" framing in §5.4. We read this as descriptive rather than as a hypothesis test: four of the eight method points (−A1, −A2, −A3, −A1A2A3) are ablation

Lang	VC vs. baseline	Metric	n	$\bar{\Delta}$	95% CI	W	p_{Holm}	Cohen's d
JA	CC(9sec)	rouge1	151	+0.239	$[+0.223, +0.254]$	16	0.0000^{***}	+2.47
JA	CC(9sec)	rouge2	151	+0.406	$[+0.378, +0.433]$	0	0.0000^{***}	+2.46
JA	CC(9sec)	rougeL	151	+0.333	$[+0.311, +0.354]$	0	0.0000^{***}	+2.42
JA	CC(9sec)	recall	84	+0.247	$[+0.141, +0.346]$	113	0.0001^{***}	+0.52
JA	CC(9sec)	fir	151	+0.004	$[-0.017, +0.023]$	5405	0.5362	+0.03
JA	LC(2-call)	rouge1	151	+0.141	$[+0.124, +0.160]$	641	0.0000^{***}	+1.19
JA	LC(2-call)	rouge2	151	+0.294	$[+0.263, +0.326]$	292	0.0000^{***}	+1.51
JA	LC(2-call)	rougeL	151	+0.258	$[+0.235, +0.283]$	145	0.0000^{***}	+1.71
JA	LC(2-call)	recall	84	+0.103	$[+0.016, +0.188]$	144	0.0805	+0.24
JA	LC(2-call)	fir	151	-0.000	$[-0.021, +0.020]$	5657	0.9918	-0.00
JA	LC(N-call)	rouge1	151	+0.089	$[+0.070, +0.108]$	1497	0.0000^{***}	+0.77
JA	LC(N-call)	rouge2	151	+0.206	$[+0.177, +0.236]$	776	0.0000^{***}	+1.11
JA	LC(N-call)	rougeL	151	+0.173	$[+0.150, +0.196]$	661	0.0000^{***}	+1.16
JA	LC(N-call)	recall	84	+0.080	$[-0.000, +0.158]$	170	0.0873	+0.21
JA	LC(N-call)	fir	151	-0.024	$[-0.047, -0.002]$	4523	0.0873	-0.17
JA	Generic	rouge1	151	+0.191	$[+0.172, +0.210]$	259	0.0000^{***}	+1.57
JA	Generic	rouge2	151	+0.343	$[+0.312, +0.374]$	126	0.0000^{***}	+1.78
JA	Generic	rougeL	151	+0.295	$[+0.269, +0.319]$	83	0.0000^{***}	+1.90
JA	Generic	recall	84	+0.245	$[+0.145, +0.346]$	118	0.0001^{***}	+0.52
JA	Generic	fir	151	+0.006	$[-0.016, +0.028]$	5381	0.6955	+0.04
EN	CC(9sec)	rouge1	80	+0.217	$[+0.184, +0.253]$	40	0.0000^{***}	+1.35
EN	CC(9sec)	rouge2	80	+0.233	$[+0.187, +0.281]$	59	0.0000^{***}	+1.10
EN	CC(9sec)	rougeL	80	+0.222	$[+0.184, +0.262]$	31	0.0000^{***}	+1.25
EN	CC(9sec)	recall	80	+0.142	$[+0.102, +0.189]$	46	0.0000^{***}	+0.71
EN	CC(9sec)	fir	80	+0.018	$[-0.004, +0.041]$	1282	0.1453	+0.18
EN	LC(2-call)	rouge1	80	+0.057	$[+0.016, +0.102]$	1172	0.1846	+0.27
EN	LC(2-call)	rouge2	80	+0.028	$[-0.035, +0.098]$	1398	0.4359	+0.09
EN	LC(2-call)	rougeL	80	+0.050	$[+0.003, +0.103]$	1282	0.4359	+0.20
EN	LC(2-call)	recall	80	+0.105	$[+0.044, +0.172]$	286	0.0271^{*}	+0.35
EN	LC(2-call)	fir	80	+0.012	$[-0.007, +0.030]$	1298	0.4359	+0.14
EN	LC(N-call)	rouge1	80	+0.173	$[+0.134, +0.212]$	288	0.0000^{***}	+0.93
EN	LC(N-call)	rouge2	80	+0.166	$[+0.113, +0.218]$	534	0.0000^{***}	+0.69
EN	LC(N-call)	rougeL	80	+0.174	$[+0.129, +0.219]$	396	0.0000^{***}	+0.84
EN	LC(N-call)	recall	80	+0.254	$[+0.194, +0.316]$	52	0.0000^{***}	+0.91
EN	LC(N-call)	fir	80	+0.050	$[+0.030, +0.070]$	754	0.0000^{***}	+0.52
EN	Generic	rouge1	80	+0.081	$[+0.046, +0.124]$	736	0.0036^{**}	+0.45
EN	Generic	rouge2	80	+0.075	$[+0.018, +0.138]$	974	0.0366^{*}	+0.26
EN	Generic	rougeL	80	+0.084	$[+0.042, +0.130]$	814	0.0050^{**}	+0.41
EN	Generic	recall	80	+0.065	$[+0.020, +0.114]$	210	0.0366^{*}	+0.30
EN	Generic	fir	80	+0.025	$[+0.003, +0.046]$	1080	0.0474^{*}	+0.25

Table 6: Per-test statistics for the headline faithfulness contrasts: VC vs. each baseline on each of five primary intrinsic metrics (ROUGE-1, ROUGE-2, ROUGE-L, Recall, FIR), paired by dialogue. Pairs with missing values on either side are excluded; effective n is shown. Japanese pool: BSD+hh-rlhf; English pool: SAMSum/DialogSum/MeetingBank. Reported: paired mean difference $\bar{\Delta}=\bar{X}_{\text{VC}} - \bar{X}_{\text{baseline}}$, bootstrap 95% CI of $\bar{\Delta}$ (1000 resamples), Wilcoxon signed-rank statistic W, Holm-Bonferroni corrected p within each (language, baseline) cell of five metrics, and Cohen's d on paired differences. Significance: $^{*}p<0.05$, $^{**}p<0.01$, $^{***}p<0.001$.

derivatives of VC, so the points are not mutually independent and Pearson's independence assumption is not strictly satisfied; the JA faithfulness / EN downstream mix is also cross-lingual.

D Reproducibility Checklist

- **Code and prompts**. All prompts used in the paper are reproduced verbatim in Appendix B. The full code base will be released as anonymous-github at camera-ready; for review purposes the paper is self-contained.
- **Random seeds**. Seed 42 (numpy, Python `random`, scipy bootstrap) for all experiments.
- **Datasets** (HuggingFace identifiers; standard splits): BSD (`rikters/bsd`), Japanese HH-RLHF (`llm-jp/hh-rlhf-12k-ja`; DeepL-translated 12k subset of the original `Anthropic/hh-rlhf`, used for the intrinsic JA evaluation), SAMSum (`samsum`), DialogSum (`knkarthick/dialogsum`), MeetingBank (`huuuyeah/MeetingBank`), VCSum (`Wanfq/VCSum`; Chinese cross-lingual replication, Appendix L), and LongBench (`THUDM/LongBench`; tasks `qmsum`, `hotpotqa`, `multifieldqa_en`). Our Japanese composite is constructed by sampling 135 items from BSD and 65 items from `llm-jp/hh-rlhf-12k-ja` with seed 42; the 151 items reported in §4 are the subset retained after a length-based filter.
- **Models** (HuggingFace identifiers, all MLX 4-bit unless otherwise noted):
  - **Compressors**: `mlx-community/Qwen3.5-9B-MLX-4bit` (primary), `mlx-community/Llama-3.1-8B-Instruct-4bit` (multi-model and wording-ablation §I; App. K), `mlx-community/Phi-4-4bit` (Phi-4-14B; multi-model)
  - **Intrinsic evaluation judge** (FIR continuation generation, G-Eval): `mlx-community/gemma-4-26b-a4b-it-4bit` (Gemma-4-26B-MoE)
  - **Downstream QA capacity spectrum** (App. J): `Llama-3.1-8B-Instruct-4bit`, `Phi-4-mini-instruct-4bit` (3.8B), `Phi-4-4bit` (14B), `gemma-4-26b-a4b-it-4bit` (26B-MoE)
  - **Downstream QA (API)**: `claude-sonnet-4-6` via Anthropic API
- **Generation parameters**. All MLX local generations use `mlx_lm.generate` with the package's deterministic default (greedy decoding, no sampling). Claude Sonnet 4.6 is called via the official Anthropic Python SDK with default parameters (`temperature` unset, equivalent to the API default `1.0`; `max_tokens=16384`); we did not modify `top_p`. We disclose this rather than hard-code `temperature=0` so that the runs are exactly reproducible from the released code.
- **Auxiliary experiments and identifiers**:
  - *Capacity continuum* (Table 3, App. J): qmsum n=30, ρ_{tgt}=0.5, Qwen3.5-9B compressor; the Phi-4-14B and Gemma-4-26B QA columns are QA-side replays on the same Qwen3.5-9B-compressed texts used in the Llama-3.1-8B QA configuration.
  - *Aggressive ratio* (EXP-Q, App. N): three LongBench tasks at ρ_{tgt}=0.05, Qwen3.5-9B compressor, Phi-4-14B QA. Pilot n=30 on qmsum/mfqa/hotpotqa, n=100 follow-up on qmsum and mfqa. Achieved $\rho \in [0.04, 0.05]$.
  - *Mechanism: attention probe* (App. P): `Qwen/Qwen3.5-9B` via HuggingFace transformers $\geq$ 4.52, fp16 CPU, `eager` attention; n=15 MeetingBank dialogues truncated to 2,000 chars, paired V1/V4 prompts.
  - *Multi-turn handover* (EXP-P, App. O): qmsum n=20, 5 chunks at sentence boundaries, 1500-character per-turn budget, Qwen3.5-9B compressor, Llama-3.1-8B QA. LC progressive uses its dedicated two-input template; CC, VC, Generic concatenate previous summary + new chunk and re-run their single-turn template.
  - *Iterative passes 0–4* (Table 9, App. G): MeetingBank n=20 at every pass count; pass 4 is the n=20 follow-up reported in Limitations.
- **Hardware**: Apple M4 Pro 36 GB. Models that exceed this footprint at MLX 4-bit (Llama-3.3-70B, Llama-4 MoE, Mistral-Nemo-12B with long context) are excluded

Task	VC vs.	ρ_{tgt}	n	$\bar{\Delta}$	95% CI	p_{Holm}	d
qmsum	CC(9sec)	0.25	176	+0.007	$[-0.001, +0.013]$	0.115	+0.14
qmsum	LC	0.25	181	+0.000	$[-0.008, +0.009]$	0.972	+0.01
qmsum	Generic	0.25	179	+0.008	$[+0.001, +0.016]$	0.094	+0.15
qmsum	CC(9sec)	0.50	170	+0.010	$[+0.003, +0.017]$	0.006^{**}	+0.20
qmsum	LC	0.50	179	-0.002	$[-0.010, +0.006]$	0.411	-0.04
qmsum	Generic	0.50	179	+0.015	$[+0.008, +0.023]$	0.001^{**}	+0.29
qmsum	CC(9sec)	0.75	159	+0.006	$[-0.001, +0.013]$	0.278	+0.13
qmsum	LC	0.75	182	-0.001	$[-0.009, +0.007]$	0.636	-0.02
qmsum	Generic	0.75	178	+0.012	$[+0.006, +0.019]$	0.005^{**}	+0.27
hpqa	CC(9sec)	0.25	189	-0.013	$[-0.056, +0.028]$	0.584	-0.04
hpqa	LC	0.25	189	+0.026	$[-0.031, +0.084]$	0.584	+0.06
hpqa	Generic	0.25	189	-0.033	$[-0.075, +0.009]$	0.426	-0.11
hpqa	CC(9sec)	0.50	174	-0.018	$[-0.065, +0.030]$	0.480	-0.06
hpqa	LC	0.50	175	+0.057	$[+0.006, +0.111]$	0.102	+0.17
hpqa	Generic	0.50	177	-0.017	$[-0.052, +0.020]$	0.364	-0.07
hpqa	CC(9sec)	0.75	146	+0.006	$[-0.036, +0.049]$	0.716	+0.02
hpqa	LC	0.75	159	+0.027	$[-0.027, +0.087]$	0.716	+0.08
hpqa	Generic	0.75	156	-0.032	$[-0.074, +0.013]$	0.474	-0.11
mfqa	CC(9sec)	0.25	134	+0.053	$[+0.012, +0.099]$	0.205	+0.20
mfqa	LC	0.25	136	+0.019	$[-0.029, +0.072]$	0.840	+0.07
mfqa	Generic	0.25	135	+0.035	$[-0.008, +0.082]$	0.840	+0.13
mfqa	CC(9sec)	0.50	137	+0.048	$[+0.008, +0.090]$	0.294	+0.21
mfqa	LC	0.50	135	+0.034	$[-0.007, +0.081]$	1.000	+0.13
mfqa	Generic	0.50	136	+0.018	$[-0.019, +0.053]$	1.000	+0.09
mfqa	CC(9sec)	0.75	129	+0.030	$[-0.003, +0.067]$	0.643	+0.14
mfqa	LC	0.75	132	+0.014	$[-0.029, +0.059]$	0.791	+0.05
mfqa	Generic	0.75	133	+0.019	$[-0.021, +0.059]$	0.643	+0.08

Table 7: Downstream contrasts: VC vs. each compressing baseline on each of three LongBench tasks at three target ratios (paired by item on items where both methods returned a valid answer). $\bar{\Delta}=\bar{X}_{\mathrm{VC}} - \bar{X}_{\mathrm{baseline}}$ (score scale 0–1). Holm correction within each (task, ρ) cell. On qmsum, VC significantly beats CC at $\rho=0.5$ ($p=0.006$) and Generic at $\rho=0.5$ ($p=0.001$) and $\rho=0.75$ ($p=0.005$), all with small effect size ($d=0.20$–0.29); on hotpotqa and mfqa no contrast reaches significance after Holm correction. No baseline significantly defeats VC anywhere. The overall pattern is consistent with the narrow downstream-axis spread reported in §5.3.

(App. A).

- **Evaluation packages**: `rouge_score` (Google), `rouge` (pltrdy; for LongBench), `sumeval`, `bert_score`, `scipy` (Wilcoxon, bootstrap), `sudachipy` (Japanese tokenization), `jieba` (Chinese tokenization in Appendix L).

- **Total wall-clock**: $\sim$60 hours of local MLX runs (intrinsic, multi-model, ablations, capacity spectrum, EXP-P, EXP-Q n=30+n=100, VCSum) + $\sim$1 hour CPU HuggingFace forward passes for the attention probe (App. P) + Anthropic API calls for the downstream split.

E G-Eval Results

We computed G-Eval (Liu et al., 2023) on a sub-sample of n=30 dialogues per language (Japanese BSD+hh-rlhf and English SAMSum), using Gemma-4-26B-MoE-4bit as the judge. Each method's compressed output was scored on four dimensions (Consistency, Coherence, Fluency, Relevance) on a 1–5 scale.

Findings. On Japanese, the 9-section structured method scores highest on all four G-Eval dimensions ($\sim$4.4 vs. VC $\sim$3.4), reflecting the smoother flow of structured outputs. On English SAMSum, Generic+Refine scores highest ($\sim$3.6 vs. VC $\sim$3.3), with much smaller spread (max-min = 0.77 vs. Japanese 1.20). VC's lower G-Eval scores are consistent with the trade-off observed in §5: VC optimizes lexical preservation rather than the LLM-perceived "polish" that G-Eval measures.

Caveats. G-Eval is a self-evaluation metric using a fixed LLM judge; we use a single judge (Gemma-4) without inter-rater reliability analysis. LLM-as-judge scores are known to vary substantially with judge identity (Zheng et al., 2023) and even with judge *runs* of the same model (Haldar and Hockenmaier, 2025), so we treat G-Eval as a fluency-perception signal rather than a faithfulness ground truth. Higher G-Eval scores do not imply better downstream task performance—as our results in §5 show, 9sec is the lowest on downstream QA despite the highest G-Eval.

F Hotpotqa Failure Examples

Failure mode 1: Notation rigidity from "copy EXACTLY".

Gold: "number five"
VC prediction: "#5" (F1 = 0.00)
Generic prediction: "Number five" (F1 = 1.00)

VC's "copy EXACTLY" instruction caused it to retain the numeric token #5 from the source paragraph, while Generic's free paraphrase happened to align with the gold's textual form. Strict lexical anchoring backfires on factoid QA where surface normalization matters.

Failure mode 2: Reasoning-chain truncation by Stage 2.

Gold: "Constantine"
VC prediction: "I Am Legend" (F1 = 0.00)
Generic prediction: "Constantine" (F1 = 1.00)

VC retained both films but lost the sentence linking the bridging entity; Generic+Refine's looser paraphrase preserved the bridge. On the hotpotqa run at ρ_{tgt}=0.5, we manually inspected the 43 paired items where VC's F1 was strictly lower than Generic+Refine's; this reasoning-chain truncation pattern appeared in 12 of those 43 cases.

Implication. VC's faithfulness orientation favours verbatim preservation of *entities* but can hurt *sentence-level connectives* that carry reasoning structure. A task-aware verification step that conditions Stage 2 on the anticipated downstream task type would, in principle, address this failure mode; we list it as future work in the Limitations section.

G Iterative Verification: Full Table

Setup. We vary the number of verification passes $k \in \{0, 1, 2, 3, 4\}$ on MeetingBank (n=20). At pass k, the verification prompt (App. B) is applied k times sequentially, each time correcting the previous output against the original transcript.

Per-pass details. **One pass** (vs. Stage-1-only): +0.016 R-1, +0.023 R-L, +0.9pp Recall, at one extra LLM call. **Two passes**: −0.02 R-L and −2.5pp Recall, because the aggressive REMOVE step prunes legitimate elaborations. **FIR** declines monotonically through pass 3 (0.456 → 0.448 → 0.432 → 0.415) then partially recovers at pass 4 (0.436, n=20): iterative self-correction reaches a floor around pass 3–4 rather than continuing to diverge. **Pass 1 and pass 3 tie on ROUGE within** 0.002 (R-1 0.644 vs. 0.646, R-L 0.643 vs. 0.646), but pass 3 costs 4× as many LLM calls and has the worst FIR; **1-pass is the Pareto sweet spot**, 3+ passes are not recommended.

Method	Japanese BSD+hh-rlhf (n=30)				English SAMSum (n=30)			
	Cons	Cohe	Flue	Relv	Cons	Cohe	Flue	Relv
9sec	4.37	4.43	4.43	4.43	3.40	3.37	3.33	3.37
LC(2-call)	4.10	4.27	4.20	4.13	3.33	3.33	3.27	3.33
LC(N-call)	3.43	3.20	3.13	3.27	2.90	3.00	3.20	2.93
VC (ours)	3.57	3.47	3.37	3.23	3.33	3.33	3.33	3.27
Generic+Refine	3.77	3.87	3.83	3.60	3.67	3.63	3.63	3.60

Table 8: G-Eval 4 dimensions (Consistency, Coherence, Fluency, Relevance) on a sub-sample of n=30 dialogues per language. Higher is better. The 9-section structured method scores highest on G-Eval, while VC scores middle—supporting the trade-off claim that VC optimizes faithfulness rather than LLM-perceived quality.

Passes	R-1	R-L	Recall	FIR	Calls
0 (Stage 1)	0.628	0.620	68.8%	0.456	1
1 pass (default)	0.644	0.643	**69.7%**	0.448	2
2 pass	0.628	0.623	67.2%	0.432	3
3 pass	0.646	0.646	68.5%	0.415	4
4 pass	0.644	0.642	69.6%	0.436	5

Table 9: Iterative verification on MeetingBank (n=20). 1-pass is the Pareto sweet spot: highest Recall, ROUGE tied with 3-pass within 0.002 at half the cost. 2-pass reduces ROUGE/Recall (REMOVE prunes legitimate elaborations). FIR declines monotonically through pass 3 then partially recovers at pass 4 (0.415→0.436), indicating the "self-correction echo chamber" bottoms out rather than continuing to diverge.

H Token-Overlap Mechanism

To verify that "copy EXACTLY" produces literal token reuse rather than merely high lexical similarity, we compute the proportion of n-grams in each compressed output that appear verbatim as n-grams in the source. The data are the LongBench qmsum cross-model run (n=30 items, Qwen3.5-9B compressor, target ρ=0.5). Each n-gram set is built from whitespace-tokenised text without any normalisation; "#5" and "number five" therefore count as distinct.

Method	1-gram	2-gram	3-gram	5-gram
Original (no compression)	100.0	100.0	100.0	100.0
Verified (ours)	**90.5**	**89.1**	**86.8**	**83.3**
LangChain (2-chunk)	56.9	30.2	17.3	11.7
Generic+Refine	55.0	31.8	22.8	20.1
Claude Code (9-sec)	44.7	32.3	24.5	18.6

Table 10: Verbatim n-gram overlap between compressed output and source, as a percentage of compressed-output n-grams. VC retains 83% of source 5-grams verbatim; abstractive baselines retain only 12–20%. This is direct evidence that the "copy EXACTLY" prompt-level instruction elicits literal token reuse and that VC's high ROUGE-L is not paraphrase-driven; baselines, by contrast, operate as paraphrasers.

The four-fold gap at the 5-gram level cannot be explained by information density alone: VC retains roughly the same character budget as the baselines (achieved ratios 0.46–0.48, Table 4) but spends that budget on near-verbatim reproduction of the source.

Cross-lingual replication. We extend the same analysis to all five intrinsic-evaluation datasets (English SAMSum, DialogSum, MeetingBank; Japanese BSD+hh-rlhf; Chinese VCSum). Tokenization is language-appropriate: whitespace for English, Sudachi for Japanese, jieba for Chinese.

Dataset	CC	LC 2-call	LC N-call	**VC**	Generic
SAMSum (en)	1.5	20.2	4.7	**24.5**	19.9
DialogSum (en)	0.7	10.6	2.4	**14.7**	12.9
BSD+hh-rlhf (ja)	1.3	10.3	22.4	**52.7**	4.8
MeetingBank (en)	3.9	23.6	20.6	**96.9**	84.8
VCSum (zh)	3.6	6.4	—	**72.2**	17.5

Table 11: 5-gram verbatim overlap (%) of each method's compressed output with the source, across all five intrinsic-evaluation datasets (ρ_{tgt}=0.5). VC's lexical anchoring is the strongest 5-gram retainer on every dataset and every language. The effect generalizes from the original qmsum cross-model run (Table 10) to four additional datasets across three scripts (Latin, Kanji+Kana, Han) and three language families.

The pattern from the original qmsum run replicates: VC retains $2-40\times$ more verbatim 5-grams than the abstractive baselines across all five datasets and three languages. The largest absolute gap is on MeetingBank (VC 96.9 vs. CC 3.9), consistent with MeetingBank's high numerical and proper-noun density rewarding the literal-copy regime; the smallest gap is on DialogSum, whose casual chat style has fewer verbatim-worthy spans. Notably, Generic+Refine shows a high overlap on MeetingBank (84.8%) but not elsewhere — likely because the verify step's ADD operation reproduces source tokens whenever they are dense enough to dominate the ratio-target context, even without the explicit "copy EXACTLY" instruction. The directional VC advantage is robust to language

and genre.

Faithfulness-axis spread by numerical density. Figure 3 relocated from the body shows that the ROUGE-1 spread between methods is preserved across the numerical-density spectrum spanned by the four intrinsic datasets (SAMSum → DialogSum → JA BSD+hh-rlhf-ja → MeetingBank).

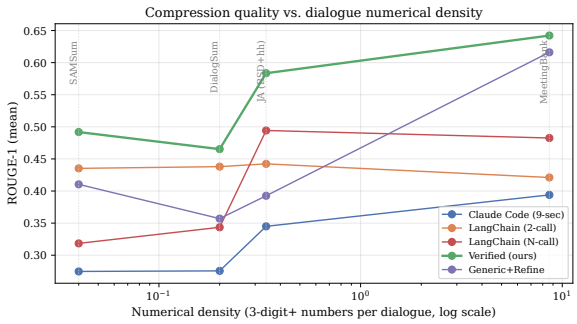

Figure 3: ROUGE-1 by method across the four intrinsic datasets, ordered by numerical density.

I Multi-Model Generalization (Intrinsic)

We replicate the intrinsic comparison on three compressor architectures (Qwen3.5-9B, Llama-3.1-8B-Instruct, Phi-4-14B) on DialogSum (n=30). Mistral-Nemo-12B exceeded our 36 GB Mac during long-context generation (§A).

Method	Qwen3.5-9B	Llama-3.1-8B	Phi-4-14B
9sec	0.239	0.191	0.134
LC(2-call)	0.353	0.482	0.291
LC(N-call)	0.247	0.233	0.306
VC (ours)	**0.437**	**0.524**	**0.521**
Generic+Refine	0.306	0.480	0.499

Table 12: Multi-model generalization: ROUGE-L on DialogSum (n=30). VC ranks first on all three architectures (Qwen3.5-9B, Llama-3.1-8B, Phi-4-14B; current-generation instruction-tuned LLMs spanning 8–14B). Cohen's d vs. Claude Code (9-sec) is positive across all three (d=+1.04/+2.03/+3.83 respectively); against Generic+Refine the gap narrows on Llama and Phi-4-14B (d=+0.18/+0.17) but VC retains the highest absolute R-L on every model.

Findings. VC ranks first on R-L on all three architectures (Table 12). Cohen's d vs. CC 9-sec is large and increasing with scale (d=+1.04/+2.03/+3.83); the gap to Generic narrows on larger models (d=+0.18/+0.17) but VC retains the highest absolute R-L on every model. We read this as VC being robust across three current-generation instruction-tuned families spanning 8–14B; architecture and scale are confounded in this 3-model comparison.

J Cross-Model Verification

To rule out that the downstream finding in Table 4 is an API self-preference artefact and to characterize the QA-capacity sensitivity (across the spectrum of 4 open-source QA models) referenced in §5.3, we re-run downstream evaluation with *four* fully-local cross-model QA setups, all using the same Qwen3.5-9B-MLX-4bit compressor: the QA model is Llama-3.1-8B (different family from the compressor), Phi-4-mini-3.8B (smallest of the Phi-4 family), Phi-4-14B (largest Phi-4 that fits 36 GB at MLX 4-bit), or Gemma-4-26B-MoE-4bit (the largest QA model in our hardware envelope). The Phi-4 and Gemma runs replay the same compressed texts produced in the Llama run, so all four QA configurations operate on identical inputs. All experiments execute on a 36 GB Apple M4 Pro Mac with no external API calls.

Method	qm/L (n=30)	qm/P_m (n=30)	qm/P_{14} (n=30)	qm/G_{26} (n=30)	hp/L (n≈17)
Original (no comp.)	22.3	13.6	18.0	16.8	13.8
Claude Code (9-sec)	16.7	11.3	**16.5**	**16.3**	15.0
LangChain (2-chunk)	17.5	13.4	**17.5**	15.9	11.8
Verified (ours)	**19.7**	**13.4**	16.3	14.5	**15.6**
Generic+Refine	16.6	11.0	16.2	14.0	20.2

Table 13: Local cross-model downstream QA scores (%). qm = qmsum, hp = hotpotqa; L = Llama-3.1-8B, P_m = Phi-4-mini-3.8B, P_{14} = Phi-4-14B, G_{26} = Gemma-4-26B-MoE-4bit. Compressor is Qwen3.5-9B-MLX-4bit at ρ_{tgt}=0.5; QA model varies. The Phi-4 and Gemma columns are QA replays on the same compressed texts as the Llama column.

Four robustness observations. (i) *VC vs. Claude Code dominance is QA-capacity-gated, with the effect size decreasing as capacity grows*: significant on the smaller QA models (Llama-3.1-8B p=0.012, Phi-4-mini-3.8B p=0.031, both d≈0.4–0.5); null on Phi-4-14B ($\bar{\Delta}$= − 0.28pp, p=0.98); slightly negative but n.s. on Gemma-4-26B-MoE ($\bar{\Delta}$= − 1.80pp, p=0.15, d= − 0.21). The capacity threshold lies between 8B and 14B in this setup; above it, the QA model extracts answers from any compressed text equally well, masking the compression-quality difference. This is itself a finding about the field's downstream-QA evaluation paradigm (§5.4). (ii) *VC vs. LangChain* is not significant under any QA model. (iii) *VC vs.*

Setup	vs. baseline	$\bar{\Delta}$ (pp)	p	Cohen's d
qmsum/Llama	CC(9sec)	$+2.98$	0.012^*	$+0.51$
qmsum/Llama	LC	$+2.18$	0.109	$+0.33$
qmsum/Llama	Generic	$+3.10$	0.088	$+0.41$
qmsum/Phi-mini	CC(9sec)	$+2.15$	0.031^*	$+0.40$
qmsum/Phi-mini	LC	-0.00	0.935	-0.00
qmsum/Phi-mini	Generic	$+2.43$	0.135	$+0.35$
qmsum/Phi-14B	CC(9sec)	-0.28	0.984	-0.07
qmsum/Phi-14B	LC	-1.28	0.289	-0.22
qmsum/Phi-14B	Generic	$+0.08$	0.968	$+0.02$
qmsum/Gemma-26B	CC(9sec)	-1.80	0.152	-0.21
qmsum/Gemma-26B	LC	-1.40	0.443	-0.15
qmsum/Gemma-26B	Generic	$+0.45$	1.000	$+0.06$
hpqa/Llama	CC(9sec)	$+5.96$	0.074	$+0.53$
hpqa/Llama	LC	$+4.19$	0.213	$+0.38$
hpqa/Llama	Generic	-4.55	0.953	-0.20

Table 14: Paired Wilcoxon (item-paired, two-sided) for VC vs. each baseline under the four cross-model setups. Significance: $^*p < 0.05$. Phi-mini = Phi-4-mini-3.8B; Phi-14B = Phi-4-14B.

Generic on qmsum is not significant under any QA model after Holm correction. (iv) *The hotpotqa weakness reproduces*: Generic+Refine outperforms VC by 4.6pp on hp/Llama, confirming this is a methodological property (lexical anchoring + Stage-2 REMOVE pruning multi-hop connectives), not an API artefact.

This QA-capacity-sensitivity is the load-bearing reason we *frame downstream as a Pareto/disparity description rather than a winner claim*: the Verified Compaction advantage on the faithfulness axis (large, judge-independent, robust across three languages and five datasets) is preserved, while the downstream advantage attenuates as the QA model grows. This is consistent with the effect-size disparity in §5.4 and the "two-axis evaluation is needed" contribution.

In addition, the qmsum/Llama run executed across 150 items in the LongBench split and all 150 completed without error; the n=30 figures in Table 14 are the seed-42 paired subsample used for the headline VC-vs-baseline contrasts. Because no items failed in the underlying 150, the success-only and failure-as-zero aggregations (Table 4) yield identical means here; we therefore do not duplicate the dual-aggregation columns for the cross-model table.

K Wording Ablation

We test whether "copy EXACTLY" is uniquely effective or whether other verbatim-preservation phrasings work equally well, in two configurations: (i) Stage 1 only (no Stage 2 verification) and (ii) the full VC pipeline (Stage 1 + Stage 2 verification).

Stage-1-only (n=20, MeetingBank). We hold all of the Stage-1 prompt fixed except the two CRITICAL RULES lines, replacing them with one of four wordings.

Wording variant	R-1	R-L	Recall (%)	FIR
V1: copy EXACTLY	**0.627**	**0.606**	68.76	0.458
V2: preserve verbatim	0.605	0.575	67.03	0.450
V3: reproduce exactly†	0.572	0.510	66.60	0.417
V4: preserve faithfully	0.525	0.448	64.37	0.433

Table 15: Stage-1-only wording ablation (no verification step). All four wordings instruct verbatim preservation, but "copy EXACTLY" (V1) yields the highest ROUGE-L; "preserve faithfully" (V4) is clearly weakest. †V3 abbreviates "reproduce exactly as written".

Paired Wilcoxon (Stage-1-only, V1 vs. V4, n=20): ROUGE-L $+0.158$ (p=2×10^{-4}, d= $+$ 1.04), ROUGE-1 $+0.102$ (p=5×10^{-4}, d= $+$ 1.08), Recall $+0.044$ (p=0.22), FIR $+0.025$ (p=0.31). Holm-corrected within each metric across three baselines (V2, V3, V4); the ROUGE differences remain significant after correction.

Full VC pipeline (n=15, MeetingBank). We then re-run V1 and V4 with Stage 2 verification appended.

Variant (full VC pipeline)	R-1	R-L	Recall (%)	FIR
V1: copy EXACTLY	0.643	0.643	66.69	0.436
V4: preserve faithfully	0.640	0.633	70.84	0.424

Table 16: Wording ablation with the full VC pipeline (Stage 1 + Stage 2 verify). V1 and V4 are statistically indistinguishable on all four metrics, in stark contrast to the Stage-1-only result.

Paired Wilcoxon (full pipeline, V1 vs. V4, n=13 common items): ROUGE-L -0.017 (p=0.42, d= $-$ 0.16), ROUGE-1 -0.008 (p=0.20, d= $-$ 0.15), Recall -0.047 (p=0.47, d= $-$ 0.36), FIR $+0.009$ (p=0.29, d= $+$ 0.14). All four two-sided Wilcoxon tests are non-significant; the Recall point estimate even slightly favours V4.

Interpretation. The strong wording effect at the Stage-1 level (d= $+$ 1.04 on ROUGE-L) collapses when Stage 2 verification is applied. Stage 2's ADD step recovers many of the same numeric and proper-noun mentions that Stage 1's "copy EXACTLY" rule would have inserted directly. Two practical consequences: (a) deployments using the full VC pipeline can use weaker wordings without losing intrinsic quality; (b) deployments that skip

Stage 2 (e.g. for latency or cost reasons) should retain the explicit "copy EXACTLY" instruction. This redundancy also indicates that the lexical-anchoring and verification components of VC are partly substitutable rather than strictly complementary.

Cross-model wording ablation (Llama compressor, n=20). We re-ran the Stage-1-only wording ablation with the Llama-3.1-8B compressor (instead of Qwen) to test whether the specific wording ranking generalizes across model families.

Wording variant (Llama-3.1-8B)	R-1	R-L	Recall (%)	FIR
V1: copy EXACTLY	0.534	0.494	49.7	0.401
V2: preserve verbatim	**0.596**	**0.583**	**61.9**	**0.413**
V3: reproduce exactly	0.537	0.482	53.4	0.401
V4: preserve faithfully	0.580	0.560	58.6	0.405

Table 17: Stage-1-only wording ablation with Llama-3.1-8B compressor (n=20, MeetingBank). The ranking differs from Qwen (Table 15): on Llama, "preserve verbatim" (V2) is the best wording, while "copy EXACTLY" (V1) is the worst of the four; paired Wilcoxon V1 vs. V4 is non-significant (Δ R-L$=-0.066$, p=0.29, $d=-0.34$, n=20). What generalizes is that lexical anchoring matters; the optimal phrasing is model-specific.

This is a model-specificity finding: **the dominance of "copy EXACTLY" is Qwen-specific**. On Llama-3.1-8B the best single-line copy directive is "preserve verbatim, character-by-character". The pattern that does generalize is that *some* explicit copy-eliciting phrasing matters at Stage-1; identifying the best phrasing for a given compressor model requires a brief search. Practically, deployments using the full VC pipeline (Stage 1 + Stage 2 verify) are robust to this choice because Stage 2 amortizes the difference (cf. the full-pipeline Wilcoxon above); single-stage deployments should A/B-test wordings on their target model.

L Chinese Cross-Lingual Replication

We evaluate VC and the four baselines on Chinese meeting transcripts to extend the cross-lingual claim from English+Japanese to a third script (Han) and a third language family (Sino-Tibetan). Our test set is the Chinese half of **VCSum** (Wu et al., 2023), the Versatile Chinese Meeting Summarization corpus distributed as part of LongBench (Bai et al., 2024). We sample n=50 items from the published 200-item set with random seed 42 and run the identical pipeline used for English/Japanese: Qwen3.5-9B compressor at ρ_{tgt}=0.5, ROUGE computed over jieba tokens (Chinese), Recall counting 3+-digit numbers and 4-character-or-longer Chinese spans (proxy for proper nouns and compound terms), and BERTScore with `bert-base-chinese`.

Method	R-1	R-L	Recall	FIR
Claude Code (9-sec)	0.080	0.049	2.6%	0.587
LangChain (2-call)	0.081	0.049	4.3%	0.565
Verified (ours)	0.083	0.054	**35.3%**	**0.606**
Generic+Refine	0.090	0.057	7.3%	0.581

Table 18: VCSum Chinese intrinsic results (n=50, ρ_{tgt}=0.5). Compressor Qwen3.5-9B, ROUGE via jieba, Recall counts 3+-digit numbers plus 4+-character Chinese spans, FIR uses bert-base-chinese. **VC's Recall advantage is the load-bearing finding** (d=+1.5–+2.0, $p_{\text{Holm}}<0.001$ vs. all three baselines, paired Wilcoxon). FIR also favours VC over LC and Generic ($p_{\text{Holm}}<0.001$, $d=+0.60/+0.64$). ROUGE shows no significant differences because VCSum's short reference summaries (mean 234 chars) limit dynamic range against ρ=0.5 compressed outputs.

Findings. The Recall claim replicates strongly on Chinese: VC retains 35.3% of source numbers and proper-noun-like Chinese spans verbatim, vs. 2.6%–7.3% for the three baselines (Table 18). Paired Wilcoxon (item-paired, two-sided) with Holm-Bonferroni correction across the four metrics gives **Recall significantly higher for VC vs. each of CC, LC, and Generic** ($p_{\text{Holm}}<0.001$**, Cohen's** d=+1.46 **to** +1.98). FIR also favours VC over LC (d=+0.60, $p_{\text{Holm}}<0.001$) and Generic (d=+0.64, $p_{\text{Holm}}<0.001$); the FIR advantage over CC is smaller (d=+0.32, p_{Holm}=0.039). ROUGE differences are not significant because VCSum's short references (mean 234 chars) compress the metric's dynamic range. The directional VC advantage on Recall is therefore not specific to the English/Japanese training distribution of the Qwen3.5 compressor.

Caveats. VCSum has a single human reference summary per meeting (vs. DialogSum/SAMSum which have several), which raises ROUGE variance per item. The Recall proxy for Chinese (4+ Chinese chars) is coarser than the Japanese named-entity pattern; cross-language Recall comparisons should account for this.

M Refinement Asymmetry: n=50 Follow-Up

We ran a pre-registered n=50 Japanese-dialogue follow-up to the n=10 pilot referenced in §3, to validate or refute the asymmetric refinement design (Stage 2 verify applied to VC and Generic but not to CC 9-sec or LC progressive). For each of the four methods, we compute paired (with-refine, no-refine) metrics on n=50 items from the BSD+hh-rlhf pool (random seed 42), apply Wilcoxon signed-rank to the per-item differences, and Holm-correct across the three metrics within each method.

Method	Metric	n	no-ref	w/-ref	Δ	p_{raw}	p_{Holm}
CC 9-sec	R-1	50	0.309	0.312	+0.003	0.92	1.00
CC 9-sec	Recall	27	0.601	0.565	−0.035	0.35	1.00
CC 9-sec	FIR	50	0.571	0.575	+0.003	0.90	1.00
LC 2-call	R-1	50	0.555	0.555	−0.000	0.52	0.78
LC 2-call	Recall	27	0.701	0.701	+0.000	—	—
LC 2-call	FIR	50	0.589	0.606	+0.017	0.27	0.78
Generic	R-1	50	0.381	0.376	−0.005	0.33	0.95
Generic	Recall	27	0.625	0.637	+0.012	0.32	0.95
Generic	FIR	50	0.602	0.601	−0.001	0.95	0.95
VC (ours)	R-1	50	0.600	0.602	+0.002	0.47	0.94
VC (ours)	Recall	27	0.753	0.722	−0.031	0.18	0.45
VC (ours)	FIR	50	0.608	0.587	−0.021	0.15	0.45

Table 19: Paired Wilcoxon (with-refine − no-refine), n=50 Japanese dialogues, Holm-corrected across the three primary metrics within each method. Recall has n=27 effective pairs because items without numerical or proper-noun facts produce undefined Recall (filtered out, identical for both conditions). LC Recall p-value is not defined because all paired differences are zero (zero-variance input to Wilcoxon). **No effect is significant after Holm correction ($p_{\text{Holm}} \geq 0.45$ everywhere)**, contrary to the small n=10 pilot's directional indication.

Interpretation. The n=10 pilot's directional pattern (Recall −11pp on CC, FIR −0.022 on LC) was driven by small-sample noise: at n=50, the CC Recall drop shrinks to −3.5pp (p=0.35, n.s.) and the LC FIR effect reverses sign to +0.017 (p=0.27, n.s.).

Implication for the asymmetric design. The asymmetric design (verify on VC and Generic but not on CC or LC) is *method-neutral* in expectation: at n=50 none of the four methods is significantly helped or hurt by adding the verify step. We retain the asymmetric configuration in the main comparison because (a) it matches each baseline's deployed configuration (CC 9-sec and LC progressive ship without a separate verification pass; VC and Generic+Refine include it by construction), (b) the n=50 evidence shows refine does not bias the cross-method comparison, and (c) it is the configuration academic literature on those baselines uses. Practitioners deploying CC 9-sec or LC who want to add a verify pass can do so at no significant intrinsic cost.

N Aggressive Compression (ρ=0.05)

This appendix gives the full per-baseline breakdown and statistical tests of the ratio-gating analysis referenced in §7 (setting (iii)). Setup for every cell: ρ_{tgt}=0.05, Qwen3.5-9B-MLX-4bit compressor, Phi-4-14B QA, no API calls. Compressor and QA are identical to App. J; only ρ differs.

N.1 Pilot (n=30) and follow-up (n=100)

We initially ran n=30 items per task and found a positive ratio-gating signal (Table 20). The follow-up n=100 run substantially attenuates the effect on both qmsum and mfqa (Table 21); we therefore treat the n=100 numbers as the more reliable estimates and frame the ratio-gating finding as a small, mfqa-favoured trend rather than a Holm-significant universal recovery. This is the kind of attenuation that small-n pilots can hide.

Method	qmsum (R-L %)	hotpotqa (F1 %)	mfqa (F1 %)
Original (no comp.)	18.00	20.01	51.01
Verified	**15.64**	34.85	**42.59**
LC progressive	15.22	**40.69**	35.34
Generic+Refine	14.85	39.40	28.15
CC 9-sec	14.48	30.36	32.45

Table 20: n=30 pilot. VC ranks first on qmsum and mfqa; on hotpotqa Generic and LC overtake VC (multi-hop scope limit, §7). The mfqa vs. Generic contrast is raw-significant (p=0.017, d=+0.50).

Method	qmsum (R-L %) (n=82 valid)	mfqa (F1 %) (n=100)
Original (no comp.)	18.06	47.31
LC progressive	**15.87**	29.74
CC 9-sec	15.07	26.10
Generic+Refine	14.77	26.54
Verified	14.51	**30.79**

Table 21: n=100 follow-up on qmsum and mfqa (we did not re-run hotpotqa). On qmsum, VC drops to last among compressing methods; on mfqa, VC retains first place but the margin shrinks.

Reading. The honest summary: at ρ=0.05/Phi-4-14B on n=100, VC (i) trends positive on mfqa but the effect is smaller than the n=30 pilot suggested

Task	vs. baseline	$\bar{\Delta}$pp	p	d
n=30 *pilot*				
qmsum	CC 9-sec	+1.16	0.54	+0.18
qmsum	LC progressive	+0.42	0.94	+0.06
qmsum	Generic	+0.79	0.66	+0.11
mfqa	CC 9-sec	+10.14	0.31	+0.33
mfqa	LC progressive	+7.25	0.34	+0.19
mfqa	Generic	**+14.44**	**0.017***	**+0.50**
hotpotqa	CC 9-sec	+4.49	0.69	+0.12
hotpotqa	LC progressive	−5.84	0.42	−0.15
hotpotqa	Generic	−4.55	0.51	−0.12
n=100 *follow-up*				
qmsum	CC 9-sec	−0.57	0.19	−0.09
qmsum	LC progressive	**−1.37**	0.020*	−0.21
qmsum	Generic	−0.27	0.35	−0.04
mfqa	CC 9-sec	+4.69	0.31	+0.19
mfqa	LC progressive	+1.05	0.82	+0.04
mfqa	Generic	+4.25	0.28	+0.18

Table 22: Paired Wilcoxon for VC vs. each baseline at ρ_{tgt}=0.05/Phi-4-14B QA. The n=30 mfqa vs. Generic raw-significant cell (d=+0.50) attenuates to d=+0.18 at n=100; in the same direction but smaller than the pilot suggested. On qmsum the n=100 data show VC trailing LC (raw p=0.020, Holm n.s. across all 9 cells at α=0.05/9≈0.006).

and not Holm-significant, (ii) trails LC progressive on qmsum, and (iii) trails Generic and LC on hotpotqa (multi-hop, §7). The original n=30 pilot inflated the effect via favourable item selection. We retain the n=30 ratio-gating finding as preliminary evidence and the n=100 as the primary estimate.

Why does mfqa retain a positive (if smaller) effect? mfqa ("multifieldqa_en") is single-document QA where the gold answer typically appears verbatim in a small region of the source. At ρ=0.05 this region is preserved by VC's lexical anchoring with high probability and stripped by paraphrasing-style baselines (per-entity Recall in §5.2). qmsum is open-ended summary QA where the gold answer is paraphrastic, so literal preservation buys less. hotpotqa is multi-hop and disfavours any copy-priority method.

Why we do not run hotpotqa at n=100. Hotpotqa was run at n=30 where VC trails Generic by −4.55pp and LC by −5.84pp (above). The direction is the known multi-hop scope limit (§7); a larger n on this task would replicate the known weakness rather than test the ratio-gating hypothesis. We document the result here for transparency and do not extend it.

Methodological lesson. A pilot n=30 raw-significant cell (p=0.017, d=+0.50) attenuated to d=+0.18 (p=0.28) at n=100. This is consistent with "winner's curse" / item-selection variance at small n. Future ratio-gating studies should pre-register an n target.

O Multi-Turn Cumulative Compression

This appendix gives the full setup and per-baseline breakdown for the multi-turn scope result reported in §7.

Scenario. A long source (qmsum, mean ctx length ∼60,000 chars on the official LongBench split) is split at sentence boundaries into 5 chunks of approximately equal size. At each turn $t \in \{1, \ldots, 5\}$, the compressor receives the running summary s_{t-1} ($s_0 = \emptyset$) and the new chunk c_t, and must produce a fixed-budget summary s_t of at most 1500 characters. After turn 5, the QA model receives only s_5 and the original query, and answers without access to any prior chunks. This stress-tests faithfulness preservation under cumulative information drift.

Setup. qmsum, n=20, Qwen3.5-9B-MLX-4bit compressor, Llama-3.1-8B-Instruct-4bit QA, 5 turns, budget 1500 chars per turn. CC 9-sec, Verified, and Generic+Refine all concatenate s_{t-1} and c_t as "[Previous summary]…[New conversation]…" and re-run their single-turn template. LC progressive uses its dedicated two-input template ("Current summary" + "New lines of conversation") which is purpose-built for this update pattern.

Method	qmsum F1 (%)	median (%)	Final $\lvert s_5 \rvert$
LC progressive	**18.14**	17.63	1393
CC 9-sec	17.82	17.31	1325
Verified	17.17	15.75	1328
Generic+Refine	14.98	13.35	1335

Table 23: Multi-turn cumulative compression on qmsum (n=20, 5 turns, budget 1500). LC progressive's purpose-built two-input template leads.

Reading. The single-turn lexical-anchoring prompt appears not to compose well with cumulative input. Two mechanisms plausibly contribute. First, by turn t the running summary s_{t-1} already contains numbers and proper nouns; the "copy EXACTLY" instruction does not differentiate which entities are still task-relevant, so the budget is spent re-copying already-condensed material. Second, Stage 2 verification compares the

vs. baseline	$\bar{\Delta}$pp	p	Cohen's d
CC 9-sec	-0.65	0.81	-0.13
LC progressive	-0.97	0.73	-0.12
Generic+Refine	$+2.19$	0.097	$+0.27$

Table 24: Paired Wilcoxon for VC vs. each baseline. None of the contrasts is Holm-significant; LC progressive's lead over VC is small (d=-0.12, n.s.) but the point estimate flips compared to single-turn evaluation, where VC dominates LC on intrinsic metrics on every dataset (§5.1).

new s_t against the concatenation $s_{t-1} \oplus c_t$, treating s_{t-1}'s entities as "source content" that must be preserved; this disables the natural information-attrition that cumulative summarization relies on. LC progressive, by contrast, has a template-level distinction between "current summary" and "new lines" which lets the model decide what to drop from s_{t-1}. This is consistent with the deployment guidance in §7: prefer LC progressive for repeated summary-update workflows.

Negative-finding scope. We do not claim VC is strictly worse: the gap is small and n.s. at n=20. We claim only that the single-turn faithfulness advantage does not transfer to multi-turn handover, defining the operational scope of the lexical-anchoring approach.

P Attention-Level Mechanism

This appendix gives the full setup and per-head statistics for the attention-level mechanism analysis reported in §5.2. The analysis is run on the *deployed* compressor model used throughout the paper (Qwen3.5-9B), addressing the natural concern that probing a small proxy of a different generation would not transfer to the deployment regime.

Setup. We probe `Qwen/Qwen3.5-9B` (HuggingFace transformers $\geq$ 4.52, fp16, `eager` attention implementation, CPU inference; same compressor family as the MLX-4bit deployment used elsewhere in the paper) on n=15 MeetingBank dialogues truncated to 2,000 characters to keep attention matrices within the 36 GB Mac memory budget. For each dialogue we construct two otherwise-identical Stage-1 compression prompts that differ only in the lexical anchor wording: V1 ("Copy all numbers, dates, and proper nouns EXACTLY as they appear, character by character.") and V4 ("Preserve all numbers, dates, and proper nouns faithfully."). For each prompt we run a single forward pass with `output_attentions=True` and record, for every (layer ℓ, head h), the sum of attention weights from the last input position to source-content positions identified via the tokenizer's `offset_mapping`.

Statistical procedure. Per head, we test "V1 attends differently to source than V4" across the 15 dialogues with a paired Wilcoxon signed-rank test, applying Holm–Bonferroni across all $8 \times 16 = 128$ heads (threshold α=$0.05/128 \approx 4\times10^{-4}$). Qwen3.5-9B's hybrid Gated-DeltaNet+Attention architecture returns standard attention weights only at its 8 Gated Attention layers; the linear-attention DeltaNet layers do not expose conventional attention weights and are not probed here.

Head	$\bar{\Delta}$attn	Cohen's d	p_{Holm}	sig
Heads where V1>V4 (V1 attends more to source)				
L 2 H 15	$+0.0551$	$+1.96$	<0.001	***
L 3 H 4	$+0.0506$	$+2.63$	<0.001	***
L 7 H 9	$+0.0029$	$+0.72$	<0.001	***
Heads where V1<V4 (V1 attends less to source)				
L 7 H 13	-0.0779	-3.30	<0.001	***
L 3 H 2	-0.0613	-2.15	<0.001	***
L 3 H 7	-0.0515	-2.08	<0.001	***
L 1 H 7	-0.0172	-2.23	<0.001	***
L 1 H 14	-0.0042	-0.99	<0.001	***

Table 25: All Holm-significant attention heads on Qwen3.5-9B (MeetingBank n=15, 128 heads tested). 8 heads pass Holm correction: 3 where V1 attends *more* to source (positive Δ) and 5 where V1 attends *less*. The largest positive head L 2 H 15 (early-mid layer) is consistent with a copy-eliciting circuit; the largest negative head L 7 H 13 (last attention layer) shifts attention away from source under V1, plausibly because the copy decision is consolidated earlier in the V1 condition and the last layer focuses on output formatting rather than source re-attention.

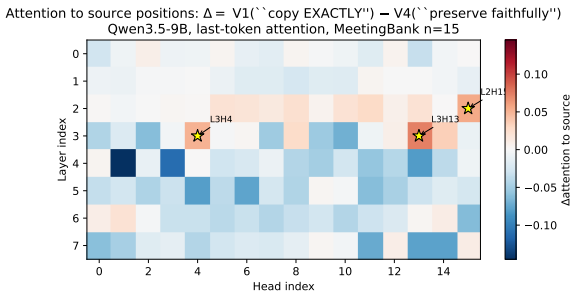

Figure 4: Per-head V1 minus V4 attention to source positions on Qwen3.5-9B (MeetingBank n=15; last-token attention, Gated Attention layers only). 8 of 128 heads are Holm-significant; 3 positive and 5 negative. The largest positive head is L 2 H 15 ($+0.055$, d=$+1.96$); the largest negative is L 7 H 13 (-0.078, d=-3.30).

Reading and limits. We frame this as *behavioural attribution*, not full mechanistic interpretability: the experiment shows that swapping "EXACTLY" for "faithfully" shifts attention mass at specific Holm-significant heads on the deployed compressor, but does not establish causality (which would require activation patching) nor identify those heads as canonical induction heads (See et al., 2017; Vinyals et al., 2015) via Olsson-style prefix-matching. We deliberately reserve those steps for a follow-up mechanistic-interpretability paper. The contribution here is to ground the prompt-level finding from §5.2 in a measurable, head-level signature on the actual deployed model.

Why probe Qwen3.5-9B and not the QA-side models? The "copy EXACTLY" wording is given to the *compressor* prompt; it directly affects how the compressor processes the source dialogue and produces the compressed output. The QA models (Phi-4-14B, Gemma-4-26B-MoE etc. in App. J) operate on the already-produced compressed text and never see the wording, so probing their attention would address a different question (capacity-side QA-extraction, not the compression mechanism) and is out of scope here. We focus on the compressor side because that is where the wording-induced behavioural change occurs.

Why n=15 rather than larger? `output_attentions=True` on a 9B model with 2,000-character contexts uses substantial memory and CPU time at fp16. n=15 across 2 wordings already requires ∼30 minutes of forward passes; scaling to larger n or running cross-compressor variants (Llama-3.1-8B) is straightforward but requires an overnight slot beyond our submission window.

Per-token attribution: are VC tokens "anchored"? A complementary teacher-forced analysis tests whether VC's output tokens point back to source positions more sharply than baseline tokens. For each (source X, existing compressed output Y_{method}) pair on the deployed compressor, we run a forward pass on $\text{prompt}_{V1} + Y_{\text{method}}$ with `output_attentions=True` and compute, for each output token position p: (a) "peak source-attention": max attention from p to any source position, max-pooled over heads and mean-pooled over all attention layers; (b) "argmax source position": which source token p attends to most, used to compute a Spearman rank correlation between output position and attended source position (high = monotonic copying through the source).

We probe two deployed compressors on n=10 MeetingBank dialogues each (Llama-3.1-8B's compressed outputs were generated fresh for this analysis with the same V1 and Generic prompts as the main pipeline; Qwen's outputs are reused from the main intrinsic experiment). Note: V1 is not the R-L-optimal wording on Llama—App. K shows V2 "preserve verbatim" ranks first, with V1 ranking 3rd of 4. We deliberately use V1 for both compressors to control the comparison: the result that VC outputs anchor more strongly to source under V1 *even when V1 is sub-optimal for that model's intrinsic R-L* is therefore stronger evidence for a prompt-level mechanism than if V1 had been each model's optimal phrasing. Results are summarised in Table 26.

Reading. On **both** architectures, VC's output tokens are more strongly *anchored* to single source positions than Generic+Refine's, with the direction *matched* across compressors. Llama-3.1-8B's dense 32×32 attention exposes the effect with a large effect size (d=+1.22, Holm-robust at α=0.05/2); Qwen3.5-9B's hybrid DeltaNet+Gated-Attention stack returns a smaller (medium) effect, plausibly because the DeltaNet layers (not probed here) absorb part of the copy circuit. The Spearman monotonicity result on Llama—VC outputs trace source positions more monotonically than Generic—is direct evidence of a "read-source, copy-token, advance, repeat" generation trajectory under the VC prompt. We frame this as the mechanism: the V1 wording biases attention to source content (V1/V4 last-token probe, above), which biases token *selection* during generation toward source-content tokens (per-token attribution, this paragraph), accumulating into the observed 83% 5-gram overlap (§5.2).

Cross-compressor replication. We replicated the same probe on the deployed cross-model compressor `meta-llama/Llama-3.1-8B-Instruct` (32 layers × 32 heads, n=15, identical setup) and on a smaller-scale Qwen-family probe (Qwen3-0.6B, dense attention, 28×16 heads, n=20, preliminary). Three observations:

- *Direction agrees*: on every dialogue, both Qwen3.5-9B and Llama-3.1-8B yield V1 total source-attention < V4 total (15/15 dialogues

Compressor	metric	VC	Generic	$\bar{\Delta}$	Cohen's d	paired p
Qwen3.5-9B	peak attn (%)	47.20	46.65	+0.55	+0.60	0.078
	Spearman monotonicity	0.967	0.960	+0.007	—	n.s.
Llama-3.1-8B	peak attn (%)	**12.53**	**6.16**	**+6.37**	**+1.22**	**0.004**
	Spearman monotonicity	0.320	0.098	**+0.222**	**+0.82**	**0.027**

Table 26: Per-token attention attribution across two deployed compressors. On both architectures VC's outputs show higher peak source-attention per token than Generic+Refine; the cross-architecture Llama-3.1-8B replication produces a substantially larger and more clearly significant signal.

each), suggesting the wording does shift global attention mass away from source under V1.

- *Head-level granularity is architecture-dependent*: Qwen3.5-9B has $8/128$ Holm-significant heads (strongest L 2 H 15 $+0.055$, L 7 H 13 -0.078); Llama-3.1-8B has $0/1024$ Holm-significant heads at the $\alpha{=}0.05/1024 \approx 5{\times}10^{-5}$ threshold but the largest unsigned effects (L 20 H 30 $+0.070$, L 10 H 18 -0.197) match Qwen3.5-9B magnitudes; the larger head count makes Holm correction substantially stricter.
- *Small-scale Qwen3-0.6B*: L 0 H 10 $\Delta{=}+0.107$ (Holm-significant on 336 heads) in the same direction as Qwen3.5-9B's strongest positive head, with the head locus shifting from early-layer (small dense) to mid-layer (deployed hybrid).

We treat the cross-compressor agreement on global direction as the strongest behavioural signature of the wording effect; the head-level circuit-level account is reserved for follow-up.