CSGR: Compressed Source Gradient Replay for Scalable Self-Supervised Retriever Training

Anonymous ACL submission

Abstract

Retriever training with language-modeling (LM) supervision has emerged as a promising annotation-free paradigm. However, scaling this paradigm exposes a severe memory bottleneck. Unlike contrastive learning, LM-supervised training requires maintaining heavy multi-layer activations for every candidate. As the retriever backbone grows, the activation footprint of the query–candidate scoring path inflates rapidly, so candidate-pool expansion and backbone expansion compete for the same limited memory budget. We propose **C**ompressed **S**ource **G**radient **R**eplay (CSGR) to address this joint activation bottleneck. CSGR consists of two coupled components. **S**ource **G**radient **R**eplay (SGR) enables cross-device document sharing by caching candidate activation boundaries and locally replaying gradients, preserving cache-boundary gradients under deterministic replay assumptions without globally storing massive computation graphs. **C**ompressed **S**ource **M**emory (CSM) replaces costly full-token source exposure with compact trainable compression slots, reducing the language model's activation overhead and leaving memory headroom for larger retriever backbones. This framework achieves system-level scalability on a single 8×80 GB node: CSGR supports a 256-document shared candidate pool, a setting where full-token baselines consistently exhaust memory during backpropagation. Beyond memory efficiency, the compressed CSM interface yields observed matched deltas of 1.59 and 1.93 nDCG@10 over our full-token reproductions at the 1B and 3B scales. Enabled by this scalable training path, a Qwen3-8B retriever reaches 65.61 CoIR nDCG@10 as a larger-backbone endpoint.

1 Introduction

Dense retrievers (Karpukhin et al., 2020; Izacard et al., 2022; Wang et al., 2022b) are central to retrieval-augmented applications such as open-domain question answering, retrieval-augmented generation, and code search. However, training them traditionally requires large-scale query–document annotations, which are scarce in specialized domains such as code, law, medicine, and enterprise text (Li et al., 2025). Recent work circumvents this bottleneck via language-modeling supervision, where conditioning a language model on a relevant document lowers its prediction loss (Guu et al., 2020; Shi et al., 2024; Cai et al., 2026). We focus on this paradigm in its next-token prediction (NTP) form, using loss differentials as annotation-free training signals. While prior work such as Revela scales along model size and local batch size (Cai et al., 2026), a critical complementary question remains: can the candidate pool itself scale beyond a device-local set while keeping source memories and retriever scores trainable without exceeding the strict memory constraints of standard GPU hardware?

Scaling this candidate-pool axis exposes two coupled bottlenecks. First, the pool becomes a massive gradient-carrying computation graph rather than a simple set of vectors. Unlike contrastive learning, where expanding a pool to M candidates mainly adds M embeddings and a similarity matrix (Karpukhin et al., 2020; He et al., 2020; Gao et al., 2021), NTP training requires backpropagating through both a retriever score path and a language-model source-memory path for each candidate. A naive large pool therefore keeps heavy activations for scoring, source encoding, source memories, and cross-attention live until backward, expanding with candidate size. Second, the standard full-token source interface is costly: each candidate exposes L source states, yet the NTP loss collapses all token-level interactions into a single document-level scalar gradient. Consequently, superficial lexical shortcuts reach the retriever through this same narrow channel. Scalable training must therefore address both the graph-level activation footprint and the source interface.

Existing techniques address pieces of these bottlenecks but not their combination. Kernel-level optimizations such as FlashAttention and activation checkpointing reduce local footprints, but they do not change the fact that a larger NTP pool spans both retriever and language-model paths. Contrastive scaling recipes such as GradCache (Gao et al., 2021) also fail to transfer directly: they assume a single-vector candidate boundary, whereas NTP training must recover gradients through multi-layer key–value memories and scoring paths after cutting this complex graph. The core obstacle is thus combining a gradient-preserving graph cut with a memory-efficient source interface that remains feasible after candidates are shared across devices.

We therefore view scalable training through two coupled design questions: where to cut and replay the cross-device computation graph, and what source interface the language model should read. The first determines whether a global pool remains trainable; the second determines memory feasibility and shapes the cue channel.

We propose **C**ompressed **S**ource **G**radient **R**eplay (CSGR) to address these questions. First, **S**ource **G**radient **R**eplay (SGR) operationalizes the boundary cut: it caches candidate source memories and retriever embeddings as detached leaves, shares them across devices, streams targets in chunks, and replays local computations with accumulated boundary gradients. Under deterministic replay assumptions, these gradients match the monolithic full-batch gradient (Proposition 1). Second, **C**ompressed **S**ource **M**emory (CSM) replaces full exposure over all L tokens with $K \ll L$ trainable compression slots appended to each sequence. Only slot states pass to cross-attention, reducing readable states by L/K. This fits the SGR cache boundary within single-node memory while providing a controlled interface change whose quality effects we evaluate empirically.

In summary, our contributions are:

- **System-level scalability.** On a single 8×80 GB node, CSGR enables training with a 256-document shared candidate pool at 55.20 GB max-rank reserved memory, whereas full-token baselines consistently trigger backward OOM under identical hyperparameters due to the massive activation footprint of the uncompressed path.
- **The CSGR framework.** We integrate SGR, which provides exact cache-boundary replay under deterministic assumptions, and CSM, a trainable sequence-to-slots source interface. In matched local controls, CSM yields observed deltas over full-token reproductions of 1.59 and 1.93 nDCG@10 at the 1B and 3B scales.
- **Larger-backbone endpoint.** Enabled by CSGR, a Qwen3-8B retriever scales successfully and reaches 65.61 CoIR nDCG@10.

2 Related Work

Retriever training via next-token prediction. Dense retrievers are trained with labels, mined negatives, synthetic queries, or teacher distillation (Karpukhin et al., 2020; Izacard et al., 2022; Wang et al., 2022b; Xiao et al., 2023), and domain adaptation extends this to low-resource domains via generated queries or pseudo labels (Wang et al., 2022a; Bonifacio et al., 2022; Dai et al., 2023). A parallel line connects retrieved content to language-model likelihood: REALM trains a retriever end-to-end with masked LM (Guu et al., 2020), RAG and Atlas integrate retrieval into generation (Lewis et al., 2020; Izacard et al., 2023), RETRO builds retrieval into the decoder architecture (Borgeaud et al., 2022), and reader-to-retriever distillation transfers reader evidence into retriever scores (Izacard and Grave, 2021). REPLUG scores candidates by perplexity reduction and ensembles their likelihoods; each candidate forward pass is independent, so its pool is bounded by sequential compute rather than the gradient-coupling obstacle we address (Shi et al., 2024). Revela is the closest prior retriever training system based on NTP loss: it trains a dense retriever with cross-document attention weighted by retriever scores and shows that larger local batches and model size improve retrieval quality, while candidate-pool scaling across devices is a complementary axis (Cai et al., 2026).

Scaling and compression: technical building blocks. For expanding the candidate pool in contrastive retrieval, in-batch negatives (Karpukhin et al., 2020), momentum queues (He et al., 2020), and GradCache (Gao et al., 2021) are the main tools. GradCache decouples the contrastive loss backward from the encoder backward, caching per-sample gradients and replaying a single encoder pass per document to recover the corresponding full-batch gradients; the key enabler is that a contrastive candidate is a single vector. SGR generalises this detach-and-replay decomposition to

training via NTP loss, where the cached boundary is a multi-layer KV memory and the replay must recover gradients through both source memories and retriever scores under the cache-boundary assumptions of Proposition 1. For source compression, learnable tokens prepended to aggregate context appear in prefix tuning (Li and Liang, 2021), memory transformers (Burtsev et al., 2020), compressive transformers (Rae et al., 2020), Set Transformer (Lee et al., 2019), and Perceiver (Jaegle et al., 2021); these apply slot compression for inference efficiency or parameter reduction. CSM applies the same mechanism as a memory-efficient source interface: the compression point determines how much source state the language model must hold and process, and our measurements use this compact interface to test the quality–memory tradeoff empirically.

3 Preliminaries

Retriever training via next-token prediction. A dense retriever E_θ maps each sequence D_i to $r_i = E_\theta(D_i)$ and assigns target–source scores $s_{ij} = \mathrm{sim}(r_i, r_j)$. For a top-$K_s$ candidate set $\mathcal{S}_i$ obtained from a frozen retriever snapshot or a teacher index, scores are normalised into routing weights

$$\alpha_{ij} = \frac{\exp(s_{ij}/\tau)}{\sum_{k\in\mathcal{S}_i}\exp(s_{ik}/\tau)}, \qquad j \in \mathcal{S}_i, \quad (1)$$

with temperature τ. A separate language model predicts target tokens while conditioning on the target prefix and routed sources; the retriever and this language model are two independent models. The language model uses one shared parameter set in two passes: a source encoding pass that encodes each source into its memory, and a mixed prediction pass that consumes the routed source memories to predict target tokens. For source exposure pattern $\mathcal{A}$, the loss is

$$\mathcal{L}_{i,\mathcal{A}} = -\sum_t \log p_\phi\big(x_{i,t} \mid x_{i,<t}, \{D_j, \alpha_{ij}\}_{j\in\mathcal{S}_i}; \mathcal{A}\big). \quad (2)$$

The candidate selection itself is detached, so gradients reach the retriever only through $\partial\mathcal{L}_{i,\mathcal{A}}/\partial s_{ij}$ over the routed candidates; there is no qrel-style direct signal. Under this supervision scheme, the candidate's representation at each routed layer ℓ is the key-value cache $(K_j^\ell, V_j^\ell) \in \mathbb{R}^{L\times 2Hd_k}$, which the language model must attend to in the mixed prediction pass while gradients propagate back through cross-attention to the source-side parameters.

The score channel. Equations 1–2 have an immediate consequence that motivates the rest of the paper. For any source exposure pattern $\mathcal{A}$, the chain rule converts every language-model interaction with candidate D_j—across all target tokens, all routed layers, all attention heads, and all token-level cues the language model exploits—into a single centred scalar that updates the retriever score s_{ij}:

$$-\frac{\partial\mathcal{L}_{i,\mathcal{A}}}{\partial s_{ij}} = \frac{\alpha_{ij}}{\tau}\Big(U_{ij}^{\mathcal{A}} - \textstyle\sum_{k\in\mathcal{S}_i}\alpha_{ik}U_{ik}^{\mathcal{A}}\Big), \qquad U_{ij}^{\mathcal{A}} = -\frac{\partial\mathcal{L}_{i,\mathcal{A}}}{\partial\alpha_{ij}}. \quad (3)$$

We give the derivation in Appendix E.2; for our purposes Eq. 3 is an accounting identity, not an optimisation result. It identifies the scalar route through which any source exposure pattern $\mathcal{A}$ affects the retriever. The same identity also exposes a credit-assignment bottleneck: token-level cues that help the language model predict the next token, whether decisive evidence or incidental lexical overlap, enter the retriever update through the same document-level score gradient. Source exposure is therefore a supervision-interface choice, not only a memory implementation detail.

4 Method

Model Overview. CSGR comprises two separable components, motivated by what inflates when the candidate pool and retriever backbone grow together. Enlarging the pool drives up three resident quantities at once—the retriever's multi-layer forward activations, the language model's multi-layer source memories, and the cross-attention workspace between targets and candidates—while increasing the retriever backbone further raises the cost of the score path; detaching and all-gathering cached states severs the gradient to the source encoder and retriever. CSM (Section 4.1) attacks both the footprint and the supervision interface: it inserts a learnable bottleneck between source tokens and the mixed prediction pass, shrinking readable source states from L tokens to K slots and changing which token-level evidence can reach the score channel. SGR (Section 4.2) attacks the connectivity: it gathers detached source memories, streams mixed prediction in chunks, and replays accumulated boundary gradients through the source and retriever computations, matching the full-batch gradient at the cache boundary under deterministic

replay assumptions. Figure 1 shows how the two modules interact in the training pipeline.

4.1 Compressed Source Memory

Challenges Analysis. In retriever training via NTP loss, each candidate's token-level KV states must remain resident throughout the forward pass so the language model can cross-attend to every source position, and its intermediate activations must be retained until the backward completes so NTP gradients can propagate back to the source-side parameters. The number of source computation graphs live at once therefore equals the candidate pool size, bounded by each rank's device memory. Growing the pool requires a proportionally larger activation budget, imposing a hard memory ceiling that gradient accumulation alone cannot relieve. There is also a supervision-interface concern. Full-token exposure lets the language model use exact identifiers, formatting templates, boilerplate, or other local continuation cues, but Eq. 3 collapses all such utility into one document-level score update. More reader access can lower NTP loss without necessarily producing a cleaner retriever update.

Method Rationale. To address these two issues, we replace the interface that exposes all source tokens with a compact, learned representation formed inside the source self-attention pass. The goal is to reduce the number of source states that each target must keep resident and attend over, while introducing a trainable bottleneck through which source evidence must pass before it can affect the document-level score channel. Towards this end, CSM, denoted APPENDED LATENTS in experiments, appends K learnable compression slots to the source sequence, processes them jointly with source tokens in the source encoding pass, and exposes only the resulting slot key-value states to the mixed prediction pass. Because the slots attend over the full document before any interaction with the target, the bottleneck is not a hard truncation: it is a contextualized, trainable source summary. In the reported controls, this lowers the memory footprint and keeps the CoIR endpoints competitive with full-token exposure, consistent with the possibility that the compact interface reduces some incidental local cues while retaining document-level evidence useful for retriever supervision.

Slot Compression. Let $D_j = (x_{j,1}, \ldots, x_{j,L})$ be a source sequence of L tokens. CSM appends K learnable compression slots $c = (c_1, \ldots, c_K)$, with $c_k \in \mathbb{R}^d$ shared across sources and trained end-to-end, to the source input and runs a single forward pass of the source encoder. After the pass, only the slot key-value states at each routed layer ℓ are retained:

$$M_j^\ell = \mathrm{KV}^\ell_{L+1:L+K}\big([D_j;\, c]\big) \in \mathbb{R}^{K \times 2Hd_k}. \quad (4)$$

where $\mathrm{KV}^\ell_{L+1:L+K}$ selects the appended slot positions, H is the number of attention heads, and d_k is the per-head key dimension. In the mixed prediction pass the language model at routed layer ℓ attends to M_j^ℓ (size K) instead of the raw source KV (size L). Thus the exposure-dependent routed cost scales with the number of readable source states—K rather than L, an L/K reduction relative to FULL ATTENTION; Appendix E.1 gives the full accounting. The results below report a $K{=}16$ local CSM row above the matched full-token local row while exposing far fewer states to the language model.

4.2 Source Gradient Replay

Challenges Analysis. Even with compact slot memories, sharing them across devices while preserving gradient correctness runs into a wall that standard data parallelism cannot resolve. Any cross-device sharing attempt faces the same three dead ends: all-gathering the full KV states is communication-prohibitive and requires large activation buffers for the backward; distributing cross-attention requires non-standard kernels; and caching states as detached leaf tensors severs the computation graph from the source-side path, leaving it without any NTP gradient. Standard data parallelism assumes each sample's forward-backward is self-contained—a condition violated here because the NTP loss of a target depends on the live computation graph of a source that may sit on another device.

Method Rationale. To address this, SGR treats the chosen source exposure as the cache boundary: compact CSM slot states $M_j = \{M_j^\ell\}_{\ell \in \mathcal{R}}$ in the default system, and raw source-token KV states in the full-token control. Towards this end, SGR first caches source memories and retriever embeddings as detached leaf tensors, all-gathers those leaves into a global pool across devices, streams targets through the pool one micro-batch at a time, returns accumulated boundary gradients to the owner ranks, and replays local source and

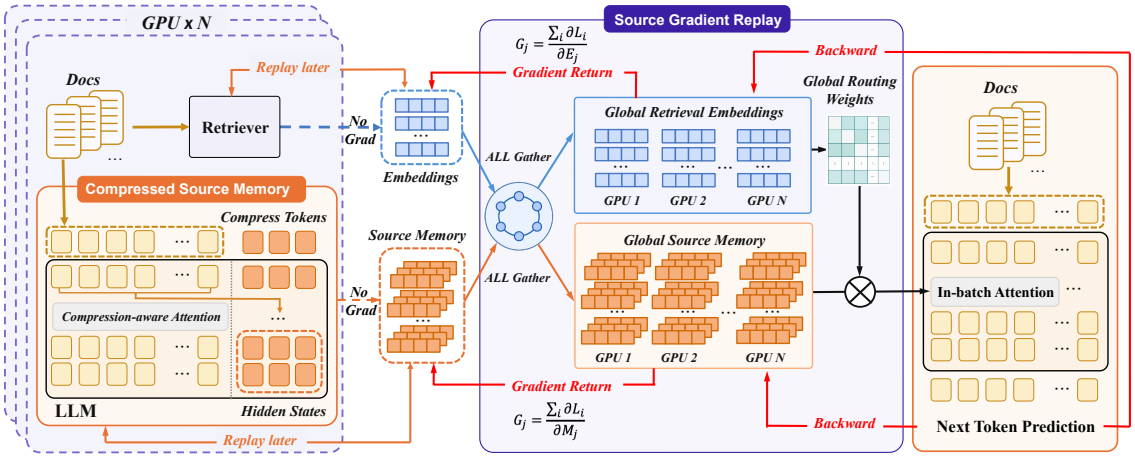


Figure 1: Overview of CSGR's two modules. Compressed Source Memory (CSM) locally encodes each source document into retrieval embeddings and compact memory states using compression tokens. Source Gradient Replay (SGR) all-gathers these detached cache boundaries into global pools for routed next-token prediction, then returns accumulated embedding and memory gradients to the owning GPU and replays the local retriever and CSM computations.

retriever encoders. Because the source representation is decoupled at the chosen exposure boundary before remote consumption, each streamed mixed-prediction chunk becomes self-contained; accumulated leaf gradients are then replayed through the source-side path and retriever encoders, matching the full-batch chain-rule gradient at the cache boundary when the replay assumptions hold.

Encode–cache–stream–return–replay. The protocol runs in six stages.
(1) Local encode–cache. Each rank encodes its local sources into source-exposure memories M_j (CSM memories in the default system) and retriever passage embeddings, and encodes local targets into retriever query embeddings. The memory and embedding outputs are stored as detached leaves, decoupled from their encoder computation graphs.
(2) All-gather. Detached source memory leaves and passage embedding leaves from all G devices are all-gathered into global pools. For CSM source memories, forward communication volume is $O(GB_{\text{loc}}|\mathcal{R}|Kd)$—proportional to K, not L; the full-token replay control uses the same protocol with an L-sized boundary.
(3) Stream targets. Each target D_i computes its NTP loss $\mathcal{L}_i$ over streamed chunks of $\{\hat{M}_j\}_{j\in\mathcal{S}_i}$, so the peak mixed-prediction workspace scales with the streamed chunk, e.g., $O(|\mathcal{R}|TK)$ for one CSM candidate. Routing weights α_{ij} are recomputed live from the retriever scores, so gradients along the routing path (Eq. 3) flow without interruption. Each accumulated source memory gradient $G_j \mathrel{+}= \partial\mathcal{L}_i/\partial\hat{M}_j$ is added to the corresponding leaf buffer, while query and passage embedding leaves collect gradients through the retriever scores.
(4) Gradient return. When candidates are shared across ranks, each consumer rank has only its local target contribution to a shared source or passage leaf. SGR therefore sums source memory and passage embedding gradients across the share group, implemented as all-reduce over the padded gathered buffers followed by slicing the owner rows on each rank.
(5) Replay. After gradient return, each local owner row holds $G_j = \sum_{i:\, j\in\mathcal{S}_i} \partial\mathcal{L}_i/\partial\hat{M}_j$ and the corresponding passage embedding gradient. SGR reruns the local query encoder, passage encoder, and source-exposure path with these boundary gradients as upstream gradients, propagating to retriever, source encoder, and slot parameters where applicable.
(6) Optimizer step. The optimizer step is taken after the backward pass for mixed prediction and all replay backwards have contributed to the same parameter-gradient buffers. The mixed prediction pass is not re-run during replay; only the local owner-side encoders—the language model's clean source pass and the retriever—are re-run to convert cached boundary gradients into parameter gradients.

Replay identity at the cache boundary. Let θ_{src} denote the source-side parameters that produced the cached memories, including CSM slots and source-pass language-model adapters where applicable, and let θ_{ret} denote the retriever parameters used by the query and passage encoders. Assume

that replay uses the same parameters and forward values as the cached source and retriever computations, either because the computations are deterministic or because the same stochastic masks are replayed. Also assume that boundary gradients for gathered source memories and passage embeddings are summed over all consumer ranks and returned to the owner ranks.

Proposition 1 (Gradient identity of SGR at the cache boundary)**.** *Let $\mathcal{L} = \sum_i \mathcal{L}_i$ be the total NTP loss over a batch. Under the SGR protocol,*

$$\begin{aligned} \nabla_{\theta_{\text{src}}} \mathcal{L}\big|_{\text{SGR}} &= \nabla_{\theta_{\text{src}}} \mathcal{L}\big|_{\text{full-batch}}, \\ \nabla_{\theta_{\text{ret}}} \mathcal{L}\big|_{\text{SGR}} &= \nabla_{\theta_{\text{ret}}} \mathcal{L}\big|_{\text{full-batch}}, \\ \frac{\partial \mathcal{L}}{\partial s_{ij}}\bigg|_{\text{SGR}} &= \frac{\partial \mathcal{L}}{\partial s_{ij}}\bigg|_{\text{full-batch}}. \end{aligned} \tag{5}$$

The proof follows from the chain rule applied to gradients through both source memories and retriever scores; see Appendix C. Real-model one-block and 8-rank communication checks for full-token and CSM boundaries are also reported there. The equality is not an end-to-end claim about our stochastic training stack: the reported runs use bf16, ZeRO-2, and LoRA dropout 0.1, and we have not audited exact stochastic-mask replay. Unless masks are replayed exactly, the implementation should be read as a stochastic numerical approximation whose empirical role is to restore a trainable source path.

Single-node pool feasibility. After SGR, the forward and backward passes for each target's mixed prediction step are self-contained: they read detached cached memories and accumulate boundary gradients. Batches are therefore re-partitionable under standard data parallelism, and the effective source batch grows to $GB_{\text{loc}}K_s$ on a single node without custom cross-attention kernels. In cost terms, FULL ATTENTION exposes L source states to each reader, CSM exposes $K \ll L$ states, and CSM+SGR keeps the K-sized boundary while adding all-gathered states and reduced boundary gradients; Appendix E.1 gives the full accounting.

5 Experiments

We evaluate CSGR along three axes: quality–efficiency trade-off, CSM as a source interface, and SGR as a replay mechanism for larger candidate pools.

Evaluation protocol. We use CoIR as the main in-domain benchmark and report nDCG@10; BEIR serves as a cross-domain direction check. Table 1 includes published baselines to orient absolute performance, but our causal comparisons are internal: FULL ATTENTION is our matched full-token reproduction, and CSM, stop-gradient sharing, and SGR are run under the same training and evaluation pipeline. The batch docs rows vary the materialized candidate batch by design: after self-masking, the 16-doc and 256-doc rows route over 15 and 255 candidates, respectively, so the softmax normalization set changes as part of the batch docs intervention rather than as a separate confound. The K=16 attribution chain, batch-docs rows, and BEIR row should be read as matched observed deltas rather than statistical-significance claims; query-level paired tests and multi-seed batch-docs estimates remain future validation. A bounded lexical contamination audit flags only 250 of 2.61M checked evaluation items, and the CSGR ordering remains after removing StackOverflowQA, where overlap is most concentrated. The appendix gives exact hyperparameters, candidate-pool definitions, run provenance, and contamination-audit details.

5.1 Results and Analysis

We next separate endpoint quality, component attribution, and replay feasibility.

Main CoIR results. Table 1 compares CSGR with internal references and published CoIR/REVELA context. The causal comparison is internal: FULL ATTENTION is our matched full-token REVELA-style reproduction, while CSM and CSGR change the source interface and pool execution under the same protocol. Published REVELA columns are copied only to situate the numbers in prior reporting; CSGR is numerically higher by +2.45 with a 1B retriever (58.60 to 61.10) and +2.57 with a 3B retriever (60.10 to 62.67), but these deltas are not used for attribution because they mix reporting protocols. The matched 3B FULL ATTENTION column (59.95) is included to make that denominator explicit for the 3B source-interface check in Appendix D. With the same Llama-3.2-1B language model and a Qwen3-8B retriever, the CSGR scale endpoint reaches 65.61 mean CoIR, +2.94 over the 3B CSGR endpoint. The largest 8B-over-3B gains are Apps Retrieval (+11.81), COIR CodeSearchNet (+6.29), CodeFeedback MT (+5.46), and Code-

Table 1: Main CoIR results (nDCG@10, %). Published context copies CoIR (Li et al., 2025) and REVELA (Cai et al., 2026); gray/API columns are external supervised, weakly supervised, or zero-shot references. FULL columns are our internal full-token REVELA-style reproductions under the matched protocol, with the 3B FULL column included to separate matched attribution baselines from published 3B context. Our rows use seed 42 and Llama-3.2-1B as the language model; CSGR uses 256 batch docs before self-mask. Bold/underline are descriptive.

	Published external context					matched	Ours	Pub.	matched	Ours scale checks		Published external context		
Task	BM25	E5-PT	BGE	REPLUG	REVELA	FULL	CSGR	REVELA	FULL	CSGR	CSGR	E5-Mis.	Ada-2	Voyage-2
Size/K	–	0.3B	0.6B	1B	1B	1B	1B/K16	3B	3B	3B/K16	8B/K16	7B	API	API
Apps Retrieval	4.80	10.60	14.70	13.90	19.40	17.93	20.98	26.60	24.23	31.81	**43.62**	23.50	8.70	26.50
CosQA	15.60	27.10	26.40	20.10	30.20	32.59	33.16	29.00	29.87	**33.47**	32.11	33.20	28.90	29.80
Synthetic Text2SQL	25.00	48.90	46.90	53.90	55.00	57.68	58.90	55.90	59.49	61.90	61.18	68.00	58.30	**69.30**
COIR CodeSearchNet	40.90	35.20	58.30	50.80	64.00	79.32	**84.76**	62.60	82.38	77.16	83.45	67.40	74.20	81.80
CodeSearchNet CC	54.00	50.50	53.70	62.80	70.00	74.73	76.69	69.10	74.23	77.13	**78.16**	64.80	69.13	73.45
CodeTransOcean Contest	47.80	56.30	62.60	61.50	81.10	74.87	77.90	83.20	76.57	81.77	**85.30**	80.60	53.34	72.77
CodeTransOcean DL	34.40	32.20	30.20	33.30	34.20	33.73	33.41	**34.50**	34.21	33.62	32.84	31.70	26.04	27.28
StackOverflow QA	70.20	86.90	80.60	76.30	85.70	84.85	90.76	88.30	86.73	91.25	**93.00**	91.00	72.40	87.68
CodeFeedback ST	68.10	70.40	69.30	66.00	76.20	76.20	77.54	78.80	76.09	79.04	**81.40**	76.40	47.12	65.35
CodeFeedback MT	59.20	46.20	47.90	42.90	70.40	56.20	56.93	**73.00**	55.70	59.60	65.06	36.40	17.74	28.74
Mean	42.00	46.40	49.10	48.20	58.60	58.81	61.10	60.10	59.95	62.67	**65.61**	57.30	45.60	56.30

TransOcean Contest (+3.53); CosQA, Synthetic Text2SQL, and CodeTransOcean DL drop by 1.36, 0.72, and 0.78. The support table below keeps the 1B attribution chain at K=16 before assigning deltas.

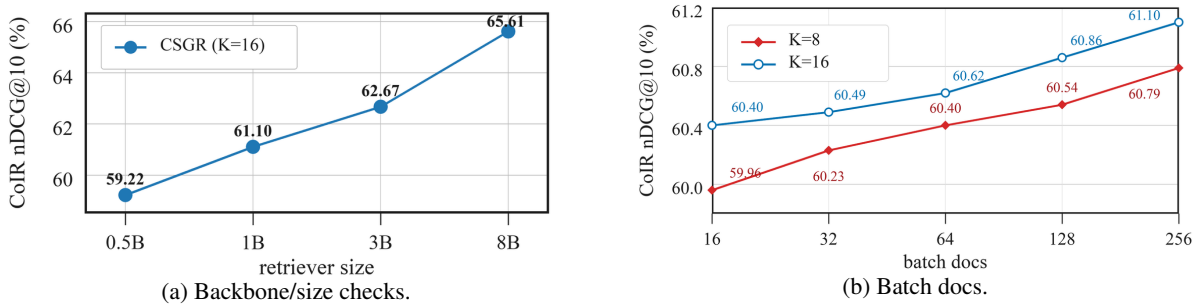
System support. Table 2 uses K=16 throughout the 1B decomposition. The local source-interface step is +1.59 nDCG@10 (58.81 to 60.40). CSGR enables the 256-doc batch setting; in the intended larger routed-candidate / softmax-normalization regime, the seed-42 run improves from 60.40 to 61.10. This is a fixed-K observed local-to-global delta: the routed candidate set and softmax normalization set grow from 15 to 255 candidates by design. The table also reports the same-hardware memory/status diagnostics behind the feasibility claim, while Appendix Table 12 adds step time and Table 3 reports the completed batch-docs and K-support sweeps. The remaining controls isolate feasibility and connectivity. Stop-gradient CSM uses the same gathered CSM boundary for the forward pass, but routing scores are computed from detached query/passage embeddings and no source-memory or passage-embedding boundary gradients are returned or replayed. This drops to 46.04, while same-hardware full-token baselines under the 256-document ZeRO-2/SGR hyperparameters consistently exhaust memory during backpropagation. We therefore report full-token SGR as an uncompressed quality/memory reference rather than as a same-hardware 80GB competitor.

Figure 2 reports CSGR endpoints above the matched local FULL ATTENTION reproduction and published REVELA context across fixed-K=16 backbone endpoints: the Qwen2.5-0.5B support point (59.22), Llama-3.2 1B/3B retrievers, and the Qwen3-8B endpoint. At 3B, the local CSM source-exposure check reaches 61.88, while the fixed-K=16 CSGR endpoint reaches 62.67; together these rows support the larger-backbone and batch-docs behavior around the main 1B attribution chain.

Table 2: 1B attribution, batch docs, and memory/status rows (CoIR nDCG@10, %; peak reserved GB for completed rows). The local CSM, stop-gradient, and CSGR rows use K=16, forming the headline 1B attribution and connectivity chain. Stop-grad CSM keeps the same gathered CSM boundary for the forward pass, but detaches query/passage routing embeddings and omits source-memory and passage-embedding gradient return/replay. Step-time diagnostics are in Appendix Table 12.

Setting	K	Docs	CoIR	Peak	Status
K=16 attribution chain					
FULL ATTENTION local	Full	16	58.81	72.30	done
CSM local	16	16	60.40	58.53	done
CSGR	16	256	**61.10**	55.20	done
Connectivity and feasibility checks					
Stop-grad CSM	16	256	46.04	42.86	done
Full-token SGR	Full	256	–	OOM	fail
Full-token SGR (H200 ref.)	Full	256	60.60	86.36	done

CSM as a memory-efficient source interface. APPENDED LATENTS is the compact source-interface family used for the reported 1B K=16 attribution chain. At 3B, the matched local source-interface check similarly improves from 59.95 with FULL ATTENTION to 61.88 with APPENDED LATENTS K=8, while EXTRACTED LATENTS reaches 60.64 (Appendix Tables 6 and 18); these compressed rows expose far fewer source states than full-token attention. Table 3 shows that the 256-document setting is not a brittle K choice: completed CSGR rows stay in a narrow 60.13–61.10 CoIR band for $K \in \{2, 4, 8, 16, 32\}$, with



(a) Backbone/size checks. (b) Batch docs.

Figure 2: Scale checks with 256 batch docs. Left: fixed-K=16 retriever/backbone endpoint checks with Llama-3.2-1B as the language model. Right: batch-docs rows for K=8 and K=16.

Table 3: Operating-point support sweeps. Left: materialized document batch at fixed K. Right: CSGR compression-token sweep with 256 docs. Scores are CoIR nDCG@10.

Docs	K=8	K=16	K	CoIR
16	59.96	60.40	2	60.13
32	60.23	60.49	4	60.48
64	60.40	60.62	8	60.79
128	60.54	60.86	16	**61.10**
256	60.79	**61.10**	32	60.80

K=16 highest in this sweep. The batch-docs sweep likewise moves monotonically from 16 to 256 materialized docs for both K=8 and K=16, though this remains an operating-point sweep rather than a multi-seed scaling law.

BEIR is a cross-domain direction check: CSM is the only compressed row above FULL ATTENTION (+0.65; Appendix Table 22), but we do not use it as headline evidence or mechanism proof. Appendix E.4 gives a post-hoc lexical-overlap probe; it is hypothesis-generating evidence about possible shortcut sensitivity, not a causal explanation of the CSM gain. Overall, the reported rows indicate that CSM lowers the memory footprint and remains competitive on endpoint quality; the small CoIR and BEIR deltas are descriptive rather than significance-tested.

SGR correctness and pool feasibility. A Llama-3.2-1B one-block check matches full-graph layer gradients exactly for full-token and CSM boundaries when dropout is disabled, and a companion 8-rank check adds the all-gather/all-reduce gradient-return path. These checks verify the cache-boundary identity under deterministic or exact-mask replay assumptions; Appendix C gives the full setup and Appendix C.1 records the remaining gap to a dropout-on full-stack parity test.

Table 2 and Figure 2 show the component split: SGR restores a trainable 256-document single-node pool, while CSM fits it within the 80GB budget at 55.20 GB max-rank reserved memory on the same 8×A800-80GB setup. The full-token SGR quality row is an H200-profiled reference that reserves 86.36 GB; on the same 8×A800-80GB ZeRO-2 path, full-token baselines exhaust memory during backpropagation.

Table 4: Deterministic SGR correctness checks.

Check	Boundary	fp32	bf16
One-block	Full, CSM	0 rel. err.	0 rel. err.
8-rank	Full, CSM	$< 8\times10^{-7}$ rel.	> 0.99996 cos.

6 Discussion and Conclusion

CSGR combines compact source exposure with cache-boundary replay for NTP-supervised retriever training. On the Revela code corpus with CoIR as the primary evaluation, CSM reduces language-model memory while remaining competitive with full-token exposure, and SGR restores trainable sharing for 256 batch docs. The Qwen3-8B endpoint reaches 65.61 CoIR nDCG@10, showing compatibility with a larger retriever backbone. The 1B K=16 attribution chain gives a +1.59 source-interface delta (58.81 to 60.40); with the intended larger routed-candidate / softmax-normalization regime, the 256-doc CSGR run improves from 60.40 to 61.10. K=16 stop-gradient sharing drops to 46.04, CSGR fits at 55.20 GB max-rank reserved memory on the same 8×A800-80GB setup, and the corresponding full-token ZeRO-2/SGR path consistently exhausts memory during backpropagation. Ultimately, by decoupling memory constraints from gradient connectivity, CSGR establishes a highly practical foundation for scaling annotation-free retrieval paradigms. Query-level paired tests and contamination-filtered retraining or re-evaluation remain future work.

Limitations

Our study is scoped to self-supervised retriever training via NTP loss on the Revela code corpus, with CoIR as the primary in-domain evaluation and BEIR as a direction check. The language model is Llama-3.2-1B in all of our internal rows; only the separate LoRA-tuned retriever varies in scale (Qwen2.5-0.5B, Llama-3.2-1B/3B, Qwen3-8B). Whether the ordering among source interfaces transfers to larger language models, encoder-only retrievers, or non-code corpora is left open. Compact latent capacity inside the source sequence is examined over $K \in \{2, 4, 8, 16, 32\}$ in support sweeps and may interact with sequence length, candidate set size, or routing temperature outside the swept range. The 256-document rows also change per-GPU target batch and the softmax candidate set, so they are system-support evidence rather than isolated pool-size estimates. We do not yet report multi-seed estimates or query-level paired bootstrap/randomization tests for the $K{=}16$ attribution chain or BEIR direction check. Memory profiling combines earlier rank-0 local profiles with a same-hardware max-over-ranks audit for the 256-document rows; the full-token SGR quality row remains an H200 reference, while the corresponding 8×A800-80GB ZeRO-2 path exhausts memory during backpropagation. Appendix F.3 reports a bounded lexical overlap audit between the training corpus and CoIR; however, that audit is not a semantic memorization test or a contamination-filtered retraining run, so external benchmark comparisons remain reporting context. All claims are restricted to offline public-benchmark experiments; we make no operational claims about online retrieval quality, safety, or robustness on unsafe, copyrighted, or privacy-sensitive corpora.

References

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Unsupervised dataset generation for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2387–2392. Association for Computing Machinery.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Mikhail S. Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V. Sapunov. 2020. Memory transformer. *Preprint*, arXiv:2006.11527.

Fengyu Cai, Tong Chen, Xinran Zhao, Sihao Chen, Hongming Zhang, Tongshuang Wu, Iryna Gurevych, and Heinz Koeppl. 2026. Revela: Dense retriever learning via language modeling. In *The Fourteenth International Conference on Learning Representations*.

Zhuyun Dai, Vincent Y. Zhao, Ji Ma, Yi Luan, Jianmo Ni, Jing Lu, Anton Bakalov, Kelvin Guu, Keith B. Hall, and Ming-Wei Chang. 2023. Promptagator: Few-shot dense retrieval from 8 examples. In *The Eleventh International Conference on Learning Representations*. OpenReview.net.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. 2021. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 316–321, Online. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Gautier Izacard and Edouard Grave. 2021. Distilling knowledge from reader to retriever for question answering. In *9th International Conference on Learning Representations*. OpenReview.net.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. 2021. Perceiver: General perception with iterative attention.

In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4651–4664. PMLR.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753. PMLR.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xiangyang Li, Kuicai Dong, Yi Quan Lee, Wei Xia, Hao Zhang, Xinyi Dai, Yasheng Wang, and Ruiming Tang. 2025. CoIR: A comprehensive benchmark for code information retrieval models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22074–22091, Vienna, Austria. Association for Computational Linguistics.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, Mexico City, Mexico. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Kexin Wang, Nandan Thakur, Nils Reimers, and Iryna Gurevych. 2022a. GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2345–2360. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022b. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

A Main Matched Experiments

The appendix is organized by the evidence chain rather than by the order in which experiments were produced. We first collect the main matched experiments, then separate system feasibility, gradient correctness, backbone/operating-point diagnostics, mechanism probes, and reproducibility details. This keeps headline claims, system diagnostics, exploratory interpretation, and audit information distinct.

Table 5: Appendix result inventory.

Block	Included evidence
Main matched experiments	Internal endpoints, task rows, and external context.
System feasibility	Cost accounting, memory profiles, and communication diagnostics.
Gradient correctness	Cache-boundary proof, real-block checks, and replay scope.
Backbone/operating-point diagnostics	3B check, 256-doc sweeps, BEIR, and support sweeps.
Mechanism probes	Exposure bookkeeping, score channel, loss diagnostics, and case study.
Reproducibility details	Protocol, environment, audit, commands, provenance, and asset terms.

A.1 Matched Source-Exposure Controls

This section collects the matched endpoint evidence used for causal attribution in the main text. FULL ATTENTION is the internal full-token REVELA-style reproduction under our codebase, training corpus, optimizer, seed protocol, and evaluation pipeline. Table 6 reports the seed-42 1B rows used for the K=16 attribution chain; all rows use `top_k=4`, 16 materialized docs, R = all routed layers, final checkpoint 2803, and the same ten

CoIR tasks. The APPENDED LATENTS $K{=}16$ row is the local CSM row used in that chain. Table 7 gives task-level clean evaluation rows for diagnostics and contamination sensitivity. Auxiliary capacity-screening rows are omitted when their full hyperparameter ledger is not part of the paper. The matched 3B check is grouped with backbone diagnostics because it serves as larger-backbone support.

Table 6: Internal 1B source-representation endpoints (CoIR nDCG@10).

Exposure	CoIR mean
FULL ATTENTION	58.81
EOS STATE	58.55
MEAN STATE	53.09
APPENDED LATENTS $K{=}16$	**60.40**

A.2 External CoIR Baseline Context

Table 8 reports copied external baselines from CoIR (Li et al., 2025) and REVELA (Cai et al., 2026), plus API references. These rows orient absolute scale only. Causal attribution in the paper uses the internal FULL ATTENTION reproduction, CSM, stop-gradient, and SGR controls in Table 1.

B System Feasibility

B.1 Language-Model Exposure Cost Accounting

Source exposure also controls the routed language model's variable memory footprint. Let B be target sequences per device, K_s candidate sources per target, $|\mathcal{R}|$ routed layers, T target positions participating in routed reading, d_{model} hidden width, and $m_{\mathcal{A}}$ the number of readable states exposed per source under $\mathcal{A}$. Absorbing constants for precision, heads, and forward/backward storage,

$$M_{\mathcal{A}} \approx M_{\text{base}} + c_{\text{kv}}\, B\, K_s\, |\mathcal{R}|\, m_{\mathcal{A}}\, d_{\text{model}} + c_{\text{attn}}\, B\, K_s\, |\mathcal{R}|\, T\, m_{\mathcal{A}}, \tag{6}$$

with $m_{\mathcal{A}} = L$ for FULL ATTENTION, K for APPENDED LATENTS, D for EXTRACTED LATENTS, k_{tok} for QUERY-SELECTED EVIDENCE, and 1 for EOS STATE/MEAN STATE. Eq. 6 describes only the exposure-dependent term; measured peak memory in Section 5 also reflects model, optimiser, ZeRO partitioning, non-routed activations, source-side extraction overhead, and allocator reserve, which are why the empirical $\sim 28\%$ memory gap is much smaller than the nominal L/K ratio.

Table 10 expands Eq. 6 by exposure operator. It tracks the exposure-dependent routed language-model term, not the full training memory model. The source-side overhead is described in prose rather than stored as long table cells: FULL ATTENTION stores full source-token KV states; EOS STATE selects the EOS-position state; MEAN STATE pools over source states; APPENDED LATENTS runs the source pass with K learned latent positions; EXTRACTED LATENTS adds latent-query extraction attention over source states; and QUERY-SELECTED EVIDENCE scores or selects source tokens before gathering selected KV states.

B.2 System and Memory Diagnostics

This section collects the feasibility evidence: matched memory summaries, the profiled shared-pool rows behind the compact main-text table, source-memory communication diagnostics, and memory-scaling figures. These diagnostics explain training footprint and are not alternate endpoint selection rules.

Table 11 supports the main-text memory diagnostic. FULL ATTENTION is the internal full-token REVELA-style reproduction. Local rows report rank-0 reserved CUDA profiles rather than multi-seed or max-over-ranks statistics. The $K{=}16$ APPENDED LATENTS row is the local quality endpoint used for attribution; its local rank-0 memory profile was not rerun. The $K{=}8$ profile is a separate memory diagnostic and is not used as the attribution endpoint.

Table 12: Profiled memory and throughput diagnostics behind Table 2.

Setting	HW/status	Docs	Cand.	Peak GB	Step s
FULL ATTENTION local	8×80GB / done	16	15	72.30	5.84
CSM local profile ($K{=}8$)	8×80GB / done	16	15	52.30	6.20
Stop-grad CSM ($K{=}16$)	8×A800 / done	256	255	42.86	5.00
CSGR ($K{=}16$)	8×A800 / done	256	255	55.20	4.64
Full-token SGR	8×A800 / OOM	256	255	–	–
Full-token SGR	H200 / done	256	255	86.36	4.49

For Table 12, *Peak* is reserved CUDA memory. Local rows are earlier rank-0 profiles; 256-document 8×A800 rows are same-hardware max-over-ranks audits. Runtime fields are parsed from instrumented logs. The 8×A800 full-token SGR row exhausts memory during backpropagation, with an NVIDIA peak of 81199 MiB; the H200 row is an uncompressed quality reference with CoIR 60.60 rather than a same-hardware 80GB competitor.

Table 7: Task-level internal CoIR endpoints (nDCG@10, %).

Task	CSGR 0.5B	FULL ATTENTION 1B	CSGR 1B	FULL ATTENTION 3B	CSGR 3B	CSGR 8B
Apps Retrieval	26.67	17.93	20.98	24.23	31.81	43.62
CosQA	30.47	32.59	33.16	29.87	33.47	32.11
Synthetic Text2SQL	57.11	57.68	58.90	59.49	61.90	61.18
COIR CodeSearchNet	73.99	79.32	84.76	82.38	77.16	83.45
CodeSearchNet CC	75.33	74.73	76.69	74.23	77.13	78.16
CodeTransOcean Contest	73.11	74.87	77.90	76.57	81.77	85.30
CodeTransOcean DL	33.28	33.73	33.41	34.21	33.62	32.84
StackOverflow QA	87.23	84.85	90.76	86.73	91.25	93.00
CodeFeedback ST	77.01	76.20	77.54	76.09	79.04	81.40
CodeFeedback MT	57.97	56.20	56.93	55.70	59.60	65.06
Mean	59.22	58.81	61.10	59.95	62.67	65.61

Table 8: External CoIR baseline context (nDCG@10, %).

Task	BM25	E5-PT	BGE	REPLUG	REVELA 1B	REVELA 3B	E5-Mis.	Ada-2	Voyage-2
Apps Retrieval	4.80	10.60	14.70	13.90	19.40	26.60	23.50	8.70	26.50
CosQA	15.60	27.10	26.40	20.10	30.20	29.00	33.20	28.90	29.80
Synthetic Text2SQL	25.00	48.90	46.90	53.90	55.00	55.90	68.00	58.30	69.30
COIR CodeSearchNet	40.90	35.20	58.30	50.80	64.00	62.60	67.40	74.20	81.80
CodeSearchNet CC	54.00	50.50	53.70	62.80	70.00	69.10	64.80	69.13	73.45
CodeTransOcean Contest	47.80	56.30	62.60	61.50	81.10	83.20	80.60	53.34	72.77
CodeTransOcean DL	34.40	32.20	30.20	33.30	34.20	34.50	31.70	26.04	27.28
StackOverflow QA	70.20	86.90	80.60	76.30	85.70	88.30	91.00	72.40	87.68
CodeFeedback ST	68.10	70.40	69.30	66.00	76.20	78.80	76.40	47.12	65.35
CodeFeedback MT	59.20	46.20	47.90	42.90	70.40	73.00	36.40	17.74	28.74
Mean	42.00	46.40	49.10	48.20	58.60	60.10	57.30	45.60	56.30

Table 13: Source-memory communication diagnostics for shared-pool rows.

Setting	Source comm. rank/step	Aggregate per step	Max-rank avg time
Stop-grad CSM	28 MiB	224 MiB	3.43 ms
Full-token SGR (L=160)	1.64 GiB	13.13 GiB	15.80 ms
CSGR (K=16)	168 MiB	1.31 GiB	9.99 ms

Table 13 counts source-state communication only and excludes unrelated optimizer or data-parallel traffic. Time is the logged max-rank average for the source-memory gather/return path.

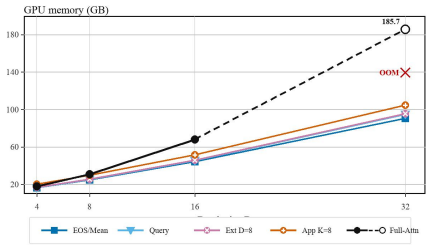


Figure 3: Materialized-doc memory scaling for the 0.5B seed 42 sweep. Completed points report rank-0 peak reserved CUDA memory; the red cross marks the FULL ATTENTION 32-doc OOM allocation-at-failure value. The dashed endpoint extrapolates the completed FULL ATTENTION–EOS STATE/MEAN STATE gap, approximately $0.093B^2$ GB.

C Gradient Correctness

Proof of Proposition 1. The proposition is a chain-rule identity at the cache boundary. It assumes that the replayed source and retriever computations use unchanged parameters and reproduce the cached forward values, either deterministically or by replaying the same stochastic masks. It also assumes that boundary gradients accumulated on gathered source memory and passage embedding leaves are summed over all consumer ranks and returned to the owner ranks before replay. *Source memory path.* By the chain rule, $\nabla_{\theta_{\text{src}}}\mathcal{L} = \sum_j \left(\sum_{i:\, j\in\mathcal{S}_i} \partial\mathcal{L}_i/\partial M_j\right) \cdot \partial M_j/\partial\theta_{\text{src}}$. SGR's accumulation and gradient-return steps compute exactly $G_j = \sum_{i:\, j\in\mathcal{S}_i} \partial\mathcal{L}_i/\partial\hat{M}_j$ at the owner rank; since $\hat{M}_j$ and M_j agree in forward value, the replay step applies $G_j \cdot \partial M_j/\partial\theta_{\text{src}}$, recovering the full-batch chain-rule expression. *Routing score path.* $\partial\mathcal{L}/\partial s_{ij} = \sum_i \partial\mathcal{L}_i/\partial s_{ij}$. In SGR the routing weights α_{ij} are live during the stream step, so the score derivatives match those of the full-batch computation. Let $q_i(\theta_{\text{ret}})$ and $p_j(\theta_{\text{ret}})$ be the query and passage embeddings used to form s_{ij}. The full-batch retriever gradient is the sum of query-embedding contributions and passage-embedding contributions over all routed target–source pairs. SGR accumulates the same query and passage leaf gradients through the live scores,

Table 9: Source exposure patterns compared in the matched CoIR line.

Operator	Source memory	Size	Role
FULL ATTENTION	full token KV	L	full KV baseline
APPENDED LATENTS	appended latent KV	K	appended slot exposure
EXTRACTED LATENTS	extracted latent memories	$D = 8$	extracted memory exposure
EOS STATE	EOS-position KV	1	single state compression control
MEAN STATE	mean-pooled state	1	pooling control
QUERY-SELECTED EVIDENCE	selected tokens	$k_{\text{tok}} = 8$	selected token control

Table 10: Language model exposure complexity. Constants for precision, heads, and forward/backward storage are omitted.

Operator	$m_{\mathcal{A}}$	Routed target-source term
FULL ATTENTION	L	$O(BK_s\|\mathcal{R}\|TL)$
EOS STATE	1	$O(BK_s\|\mathcal{R}\|T)$
MEAN STATE	1	$O(BK_s\|\mathcal{R}\|T)$
APPENDED LATENTS (K)	K	$O(BK_s\|\mathcal{R}\|TK)$
EXTRACTED LATENTS (D)	D	$O(BK_s\|\mathcal{R}\|TD)$
QUERY-SELECTED EVIDENCE token selection	k_{tok}	$O(BK_s\|\mathcal{R}\|T\,k_{\text{tok}})$

Table 11: Matched 1B local endpoint and memory summary.

Method	CoIR	Peak GB	80GB share
FULL ATTENTION	58.81	72.30	90.4%
EOS STATE	58.55	48.20	60.3%
App. lat. K=16	60.40	–	–
App. lat. K=8 profile	–	52.30	65.4%

sums shared passage gradients across consumer ranks, and replays the query and passage encoders with unchanged parameters and matching forward values. Linearity of differentiation therefore gives $\nabla_{\theta_{\text{ret}}}\mathcal{L}\big|_{\text{SGR}} = \nabla_{\theta_{\text{ret}}}\mathcal{L}\big|_{\text{full-batch}}$. □

Real-model boundary verification. We run a deterministic real-weight check on layer 0 of a Llama-3.2-1B checkpoint. The check instantiates a one-transformer-block model from the real checkpoint, uses batch size 2, source length 32, reader length 16, K=8 compression slots for CSM, eager attention, seed 1234, and disabled dropout. For each dtype and boundary, we compare two copies of the same model: a full graph that backpropagates through source memory directly, and an SGR replay graph that builds source memory under `no_grad`, detaches the boundary K/V tensors as leaves, backpropagates the reader loss to those leaves, rebuilds the live source memory, and calls `torch.autograd.backward` with the cached boundary gradients. The objective is synthetic and deterministic; its role is only to produce stable nonzero gradients through the patched in-batch attention path.

Table 14: Real-model SGR boundary-gradient check.

Dtype	Boundary	Grad cos.	Rel. err.	Max err.	Loss err.
fp32	full-token	0.9999999999997	0	0	0
fp32	CSM	0.9999999999999	0	0	0
bf16	full-token	1.0	0	0	0
bf16	CSM	1.0	0	0	0

Table 15: Real-model gradient and boundary-gradient norms.

Dtype	Boundary	Full grad	SGR grad	Full bdy.	SGR bdy.
fp32	full-token	0.448451	0.448451	0.006967	0.006967
fp32	CSM	0.242102	0.242102	0.002131	0.002131
bf16	full-token	0.448377	0.448377	0.006966	0.006966
bf16	CSM	0.242276	0.242276	0.002134	0.002134

In this deterministic real-block setting, SGR replay matches full-graph layer gradients exactly within the reported precision for both full-token and CSM boundaries, in both fp32 and bf16. This supports the local cache-boundary identity under deterministic replay.

Distributed communication spot check. We also run a companion 8-rank real-model communication check that adds the distributed all-gather/all-reduce return path. The check compares the SGR gradient-return path against a monolithic full graph under the same deterministic source-boundary setup, using both full-token and CSM boundaries. In fp32, the distributed path matches the monolithic graph up to below 8×10^{-7} relative norm error; in bf16, both boundaries remain above 0.99996 gradient cosine similarity. This check targets the communication and owner-gradient return path; it is still not a claim about ZeRO partitioning, stochastic LoRA dropout mask replay, optimizer updates, or multi-step training drift.

Table 16: Eight-rank SGR gradient-return spot check.

Dtype	Boundary	Metric	Agreement
fp32	full-token	Rel. error	$< 8\times10^{-7}$
fp32	CSM	Rel. error	$< 8\times10^{-7}$
bf16	full-token	Grad cosine	> 0.99996
bf16	CSM	Grad cosine	> 0.99996

C.1 Stochastic Replay Scope

The mathematical identity in Proposition 1 is exact only when replay reproduces the cached source and retriever forward values. This holds for deterministic computations and for stochastic computations whose masks or RNG states are exactly replayed. The completed training runs use the command template in Appendix F.2, including `-lora_dropout 0.1`, bf16, and DeepSpeed ZeRO-2. We do not claim that these completed runs replay LoRA dropout masks exactly. Therefore the theorem should be read as a cache-boundary identity for the deterministic or exact-mask case, while the reported CSGR training results are empirical evidence that the stochastic approximation remains trainable and useful under the audited training stack.

This distinction is important for attribution. The K=16 stop-gradient row in Table 2 tests connectivity by using the same gathered CSM boundary in the forward pass, while detaching query/passage routing embeddings and omitting source-memory and passage-embedding gradient return/replay. It collapses to 46.04 CoIR, whereas CSGR reaches 61.10 with the same dropout setting and training protocol. That comparison supports the practical need for replay as a connectivity control; the fixed-K observed local-to-global delta is instead the seed-42 K=16 comparison in Table 20. CSGR enables the 256-doc batch setting; in the intended larger routed-candidate / softmax-normalization regime, the seed-42 run improves from 60.40 to 61.10. It is not a proof that the dropout-on SGR update equals a monolithic dropout-on full graph step.

Table 17: Scope of SGR gradient checks and training evidence.

Check	Masks	Evidence
One-block	disabled	exact fp32/bf16
8-rank return	disabled	fp32 $< 8\times10^{-7}$; bf16 > 0.99996
Training rows	dropout 0.1	empirical evidence

D Backbone and Operating-Point Diagnostics

This section collects larger-backbone and support diagnostics: the matched 3B check, 256-document operating-point and K sweeps, the BEIR generalization check, and routed-layer sweeps with explicit local configuration. These rows are seed-42 support diagnostics unless explicitly stated otherwise.

D.1 Matched 3B Larger-Backbone Check

Table 18: Matched 3B CoIR check (nDCG@10, %).

Exposure	3B CoIR mean
FULL ATTENTION	59.95
APPENDED LATENTS K=8	**61.88**
EXTRACTED LATENTS D=8	<u>60.64</u>

Table 19: Run provenance for the matched 3B check.

Method	CoIR mean
FULL ATTENTION	59.95
APPENDED LATENTS K=8	61.88
EXTRACTED LATENTS	60.64

All rows in Tables 18 and 19 use a Llama-3.2-3B retriever with the same Llama-3.2-1B language model, `top_k=4`, 16 materialized docs, seed 42, and complete ten-task CoIR evaluation. They vary only the source exposure pattern, support the endpoint ordering, and are not used for the 80GB memory headline because the 3B FULL ATTENTION row reserves about 91 GB.

D.2 256-Document Operating-Point Sweeps

Tables 20 and 21 report seed-42 CoIR nDCG@10 percentages. *Docs* is the total materialized source pool before self-mask; 16 is the local CSM row, and 256 is the configured CSGR share-group pool. Ret. and Ref. indicate enabled retrieval/reference settings.

Table 20: Materialized candidate-batch sweep.

Docs	K=8 CoIR	K=16 CoIR
16	59.96	60.40
32	60.23	60.49
64	60.40	60.62
128	60.54	60.86
256	60.79	61.10

Table 21: CSGR g=256 compression-token support.

K	Ret.	Ref.	CoIR
2	1	1	60.13
4	1	1	60.48
8	1	1	60.79
16	1	1	61.10
32	1	1	60.80

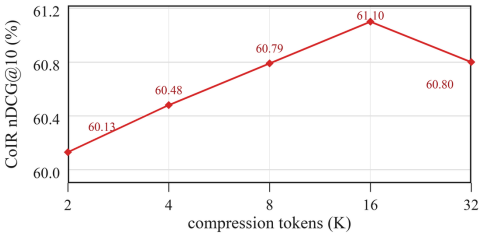


Figure 4: CSGR compression-token sensitivity with a 256-document pool. Scores are CoIR nDCG@10; exact values are in Table 21.

D.3 BEIR Generalization Check

Table 22: Cross-domain BEIR direction check (nDCG@10, %).

Exposure	BEIR	Δ
FULL ATTENTION	42.93	–
EOS STATE	39.92	−3.01
MEAN STATE	39.29	−3.64
APPENDED LATENTS K=1	37.28	−5.65
APPENDED LATENTS K=8	**43.58**	+0.65
EXTRACTED LATENTS D=8	39.82	−3.11
QUERY-SELECTED EVIDENCE k=8	42.85	−0.08

The BEIR cross-domain summary in Table 22 is a seed-42 generalization check: each source exposure pattern is trained only on the Revela code training corpus and the resulting retriever is evaluated on the natural-language 13-task BEIR suite without BEIR-specific fine-tuning, following the cross-domain protocol used by prior NTP retriever work (Cai et al., 2026). Table 23 records per-row seed 42 BEIR provenance for reproducibility. We use these rows as a direction check for interpreting the matched CoIR results. CoIR endpoint numbers remain in the CoIR endpoint tables rather than being repeated here. For QUERY-SELECTED EVIDENCE top-k = 8, HotpotQA and FEVER are included (51.2020 and 63.4750 nDCG@10 respectively), giving the 13-task mean of 42.85 reported in the main text.

D.4 Additional Support Sweeps

The following support sweeps answer narrower questions only when the local configuration is explicit in the table. These auxiliary timing diagnostics are not matched endpoint runs and should not be compared numerically with the main attribution table. Earlier auxiliary capacity-screening rows are omitted from numeric reporting because their full hyperparameter ledger is not included in the paper.

Table 23: BEIR 13-task nDCG@10 snapshot.

Method	BEIR mean
FULL ATTENTION	42.93
EOS STATE	39.92
MEAN STATE	39.29
APPENDED LATENTS (K = 1)	37.28
APPENDED LATENTS K=8	43.58
EXTRACTED LATENTS (D = 8)	39.82
QUERY-SELECTED EVIDENCE top-k = 8	42.85

Table 24: Completed 1B APPENDED LATENTS routed-layer sweeps.

Config	CoIR
$K = 4, D = 0, R = 4$, default	60.52
$K = 8, D = 0, R = 0$, all layers	60.11
$K = 8, D = 0, R = 1$, default	58.48
$K = 8, D = 0, R = 2$, default	58.93
$K = 8, D = 0, R = 4$, default	60.78
$K = 8, D = 0, R = 4$, contiguous early	51.97
$K = 8, D = 0, R = 4$, contiguous middle	57.22
$K = 8, D = 0, R = 4$, strided early	55.19
$K = 8, D = 0, R = 4$, strided middle	58.26
$K = 8, D = 0, R = 8$, default	60.64

D.4.1 Omitted auxiliary capacity rows

Auxiliary compact-exposure capacity sweeps that lack a complete run-level hyperparameter ledger are intentionally omitted from numeric tables and figures. In earlier drafts, capacity-screening rows and local endpoint rows shared the phrase "seed-42 support sweep" even when their batch profile, routed-layer setting, checkpoint, or evaluation bucket differed. To keep provenance auditable, we do not report those CoIR means here and do not use them for endpoint selection, BEIR interpretation, or the fixed-K CSGR attribution.

D.4.2 Additional routed-layer sweeps

This appendix reports additional completed CoIR ablations used to characterize routed-layer timing. Scores are nDCG@10 percentages. These runs contextualize the source exposure analysis, but they are not part of the matched endpoint table. In these sweeps, the run-id convention above applies unless the table gives explicit provenance: `d0` disables extracted-latent slots, `k0` disables appended-latent slots, and `r0` usually denotes all interaction layers.

The most useful pattern in these routed-layer sweeps is qualitative. For the K=8 APPENDED LATENTS rows, early routed-layer interaction is weak (51.97 CoIR), while default or all-layer inter-

Table 25: Completed EXTRACTED LATENTS routed-layer sweeps.

Config	CoIR
1B, $D = 2$, $K = 0$, $R = 4$, default	58.85
1B, $D = 4$, $K = 0$, $R = 4$, default	57.80
1B, $D = 8$, $K = 0$, $R = 0$, all layers	58.18
1B, $D = 8$, $K = 0$, $R = 1$, default	54.93
1B, $D = 8$, $K = 0$, $R = 2$, default	55.79
1B, $D = 8$, $K = 0$, $R = 4$, default	58.79
1B, $D = 8$, $K = 0$, $R = 8$, default	57.09
1B, $D = 8$, $K = 0$, $R = 8$, late-half	58.60
1B, $D = 16$, $K = 0$, $R = 4$, default	57.74

actions are near 60–61 CoIR. This indicates sensitivity to routed-layer timing.

E Mechanism Probes

This section gives the source-exposure and score-channel accounting, then keeps the loss diagnostics and post-hoc CoIR case study used to interpret possible behavior. These analyses are explanatory support rather than causal proof of a lexical-shortcut mechanism.

E.1 Source Exposure Bookkeeping

This section gives the longer bookkeeping that was compressed out of the main text. Let $\mathcal{A}$ denote one source exposure pattern. It specifies the readable memory exposed by each source document and the target-side computation that reads that memory. We write the exposed memory as

$$M_j^{\ell,\mathcal{A}} = \mathcal{M}_{\mathcal{A}}^{\ell}(D_j)$$

and the language-model contribution as

$$\Delta H_{ij}^{\ell,\mathcal{A}} = \mathcal{C}_{\mathcal{A}}^{\ell}(H_i^{\ell}, M_j^{\ell,\mathcal{A}}).$$

At routed layers, the target state is updated as

$$\widetilde{H}_i^{\ell} = H_i^{\ell} + \gamma_{\ell} \sum_{j \in \mathcal{S}_i} \alpha_{ij} \Delta H_{ij}^{\ell,\mathcal{A}}, \qquad \ell \in \mathcal{R}, \quad (7)$$

where $\mathcal{R}$ is the routed-layer set and γ_{ℓ} is the injection strength at layer ℓ. Changing $\mathcal{A}$ leaves the nominal NTP objective and routing rule intact, but changes the exposed memories $M_j^{\ell,\mathcal{A}}$ and the language-model contribution $\Delta H_{ij}^{\ell,\mathcal{A}}$ that can produce NTP utility.

Different source exposure patterns instantiate $M_j^{\ell,\mathcal{A}}$ differently. FULL ATTENTION exposes full source-token key-value states (K_j^{ℓ}, V_j^{ℓ}). EOS STATE exposes $(K_{j,\text{eos}}^{\ell}, V_{j,\text{eos}}^{\ell})$. MEAN STATE exposes averaged key-value states. APPENDED LATENTS exposes the appended latent slice

$$\text{slice}_{L+1:L+K}(\text{KV}^{\ell}(D_j, c_1, \ldots, c_K)),$$

where $c_1, \ldots, c_K$ are learned latent positions appended to the source pass. EXTRACTED LATENTS exposes

$$\text{KV}^{\ell}(\text{Attn}(Q_{\text{lat}}^{\ell}, H_j^{\ell}, H_j^{\ell})),$$

where learned latent queries extract D source-conditioned memories. QUERY-SELECTED EVIDENCE exposes selected source-token key-value states. These cases differ in what the language model can read from each candidate before token-level utility is collapsed into document-level feedback.

E.2 Score-Channel Derivation

For a fixed target, checkpoint, candidate set, and source exposure pattern, write the routed language-modeling loss as

$$\mathcal{L}_{i,\mathcal{A}} = -\sum_t \log p_{\phi}(x_{i,t} \mid x_{i,<t}, \{\widetilde{H}_i^{\ell}\}_{\ell \in \mathcal{R}}).$$

Define the uncentered NTP utility of source D_j as

$$U_{ij}^{\mathcal{A}} = -\partial \mathcal{L}_{i,\mathcal{A}} / \partial \alpha_{ij}.$$

If $U_{ij}^{\mathcal{A}}$ is large, locally allocating more routing mass to D_j would lower the NTP loss more. Let $G_i^{\ell} = \partial \mathcal{L}_{i,\mathcal{A}} / \partial \widetilde{H}_i^{\ell}$ be the loss gradient at routed layer ℓ. From Eq. 7, the utility can be expanded as

$$\begin{aligned} U_{ij}^{\mathcal{A}} &= -\sum_{\ell \in \mathcal{R}} \gamma_{\ell} \left\langle G_i^{\ell}, \Delta H_{ij}^{\ell,\mathcal{A}} \right\rangle \\ &= -\sum_{\ell \in \mathcal{R}} \sum_t \gamma_{\ell} \left\langle G_{i,t}^{\ell}, \Delta H_{ij,t}^{\ell,\mathcal{A}} \right\rangle. \end{aligned} \quad (8)$$

Thus token-level and layer-level language-model contributions from a candidate document are compressed into one document-level scalar $U_{ij}^{\mathcal{A}}$.

For fixed i and $\mathcal{A}$, write $\alpha_j = \alpha_{ij}$, $s_j = s_{ij}$, and $U_j = U_{ij}^{\mathcal{A}}$. The routing weights are

$$\alpha_k = \frac{\exp(s_k/\tau)}{\sum_{m \in \mathcal{S}_i} \exp(s_m/\tau)}. \quad (9)$$

For $j, k \in \mathcal{S}_i$,

$$\frac{\partial \alpha_k}{\partial s_j} = \frac{1}{\tau} \alpha_k (\mathbf{1}[j = k] - \alpha_j). \quad (10)$$

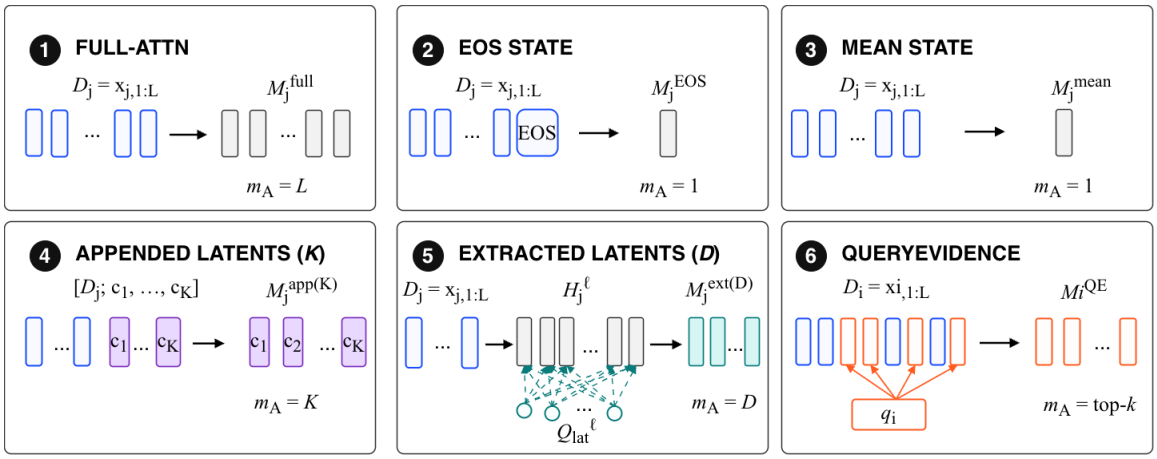


Figure 5: Source exposure patterns organized by compression locus and learnability. FULL ATTENTION exposes raw source-token states; APPENDED LATENTS forms compact states inside the source self-attention pass; EXTRACTED LATENTS forms compact states outside it; EOS STATE and MEAN STATE use non-learned summaries, and QUERY-SELECTED EVIDENCE uses non-parametric token selection. All six pass through the score channel of Eq. 3.

Using $U_k = -\partial \mathcal{L}_{i,\mathcal{A}} / \partial \alpha_k$ and the chain rule,

$$\frac{\partial \mathcal{L}_{i,\mathcal{A}}}{\partial s_j} = \sum_{k \in \mathcal{S}_i} \frac{\partial \mathcal{L}_{i,\mathcal{A}}}{\partial \alpha_k} \frac{\partial \alpha_k}{\partial s_j} \tag{11}$$

$$= -\frac{1}{\tau} \sum_{k \in \mathcal{S}_i} U_k \alpha_k (\mathbf{1}[j = k] - \alpha_j) \tag{12}$$

$$= -\frac{1}{\tau} \alpha_j \left(U_j - \sum_{k \in \mathcal{S}_i} \alpha_k U_k \right). \tag{13}$$

Defining the centered document credit $C_{ij}^{\mathcal{A}} = U_{ij}^{\mathcal{A}} - \mathbb{E}_{k \sim \alpha_i}[U_{ik}^{\mathcal{A}}]$, we obtain

$$\begin{aligned} g_{ij}^{\mathcal{A}} &\triangleq -\frac{\partial \mathcal{L}_{i,\mathcal{A}}}{\partial s_{ij}} \\ &= \frac{1}{\tau} \alpha_{ij} \left(U_{ij}^{\mathcal{A}} - \mathbb{E}_{k \sim \alpha_i}[U_{ik}^{\mathcal{A}}] \right) \\ &= \frac{1}{\tau} \alpha_{ij} C_{ij}^{\mathcal{A}}. \end{aligned} \tag{14}$$

This is a local accounting identity for any softmax-routed differentiable loss; it does not by itself explain why one exposure pattern outperforms another. Applying the chain rule once more through $s_{ij} = s_{\theta_r}(D_i, D_j)$ gives the corresponding retriever-parameter gradient while holding the selected candidate set fixed:

$$\begin{aligned} \nabla_{\theta_r} \mathcal{L}_{i,\mathcal{A}} &= \sum_{j \in \mathcal{S}_i} \frac{\partial \mathcal{L}_{i,\mathcal{A}}}{\partial s_{ij}} \nabla_{\theta_r} s_{ij} \\ &= -\sum_{j \in \mathcal{S}_i} g_{ij}^{\mathcal{A}} \nabla_{\theta_r} s_{ij}. \end{aligned} \tag{15}$$

No matter how many source tokens, attention heads, or language-model interactions contribute to token prediction, the retriever receives supervision through one scalar score channel per candidate document.

The scope is deliberately local. We fix the selected candidates, detach top-K_s retrieval and corpus construction, and assume finite temperature $\tau > 0$ and differentiability of the language model with respect to routing weights. The derivation does not prove convergence, does not claim differentiability through retrieval, and does not explain why any particular bottleneck size should be structurally optimal. Its role is to identify the score channel that constrains the retriever update.

Exposure as a controlled intervention. FULL ATTENTION lets the language model access all source-token states. This can improve its evidence capacity, but it does not increase the dimensionality of the retriever update. The retriever is not told that one token was useful, another was incidental, and another carried the real evidence; it is told only whether the document score should increase or decrease relative to other candidates. Thus full token access and fine-grained retriever supervision are different things.

Changing $\mathcal{A}$ changes $U_{ij}^{\mathcal{A}}$, and therefore changes the centered document credit received by the retriever under the same candidates and routing rule. A simple construction illustrates why this intervention is measurable. Consider two exposure patterns $\mathcal{A}$ and $\mathcal{B}$, a target D_i, and two candidate documents D_j, D_k with the same routing scores and NTP objective, but with $U_{ij}^{\mathcal{A}} > U_{ik}^{\mathcal{A}}$ and $U_{ij}^{\mathcal{B}} < U_{ik}^{\mathcal{B}}$. One candidate may contain local continuation cues that help NTP under one exposure pattern; another may contain evidence that better supports the retrieval

relation but is only useful when preserved by a different pattern. This is not a predictive theorem; it is the measurement motivation for comparing source exposure operators.

E.3 Language-Model Loss Diagnostics

This subsection records token-level language-model loss diagnostics used in the mechanism discussion. These are auxiliary runs, not matched endpoints. Because the loss-diagnostic rows do not come from the same endpoint evaluation bucket as Table 6 and the full run-level hyperparameter ledger is not included here, CoIR retrieval values and loss-versus-retrieval figures are omitted.

Table 26: Auxiliary NTP/training loss diagnostics.

Method	Train loss	Mixed NTP loss
FULL ATTENTION	2.3740	2.635
EOS STATE	2.5396	2.669
MEAN STATE	2.5438	2.676
APPENDED LATENTS ($K = 1$)	2.5253	2.636
APPENDED LATENTS ($K = 8$)	2.4611	2.625
EXTRACTED LATENTS ($D = 8$)	2.5389	2.644
QUERY-SELECTED EVIDENCE top-$k = 8$	2.3092	2.513

E.4 Qualitative CoIR Case Study

This appendix gives the detailed case study behind the source exposure interpretation. It is deliberately post-hoc: qrels are used only after the clean retrieval runs have been produced. The case miner computes ranks, disagreement buckets, and lexical overlap proxies automatically, then emits query, positive, and top-ranked negative snippets from predefined disagreement buckets. The case entries in this catalog are programmatically selected rather than manually browsed or cherry-picked. The short explanations below are manual interpretations written after export for those emitted cases. They are not used as the statistical estimator; the aggregate counts and threshold sweeps should be read together with this catalog.

The main qualitative pattern is not that FULL ATTENTION is simply worse. Full token access preserves decisive local details when exact APIs, rare identifiers, constants, return paths, or error strings are the real evidence. The same access can also make the language model sensitive to locally useful tokens that lower NTP loss without identifying the qrel-positive document. The cases are consistent with a cautious bottleneck hypothesis: compact or structured exposure may reduce sensitivity to some incidental local matches while retaining operation-level intent, root cause, or whole-function semantics. This is an interpretation of post-hoc disagreements, not a mechanism proof; the case study helps explain why token-level loss and retriever quality can diverge on some CoIR tasks, without implying that every local match is spurious.

Table 27: Representative automatically mined disagreement cases.

Case	Full	Comparison	Reading
AppsRet. `q7056`	miss	App. lat. rank 1	structured succeeds
COIR CSN `q176529`	rank 10	App. lat. rank 1	structured succeeds
StackOverflowQA `q18496`	miss	App./Ext. lat. rank 1	structured succeeds
CSN-CC `q175016`	rank 1	App. lat. rank 10	full-token succeeds

In AppsRetrieval `q7056`, full token access follows a near-duplicate loop/control-flow shape, while the compact row recovers the intended algorithmic operation. In COIR CodeSearchNet `q176529`, the structured row promotes the comment about the overall operation above helper-function evidence that is locally plausible but not the qrel target. In StackOverflow QA `q18496`, the positive answer concerns an AspectJ instrumentation root cause, while full access is distracted by generic exception/thread cues. In CodeSearchNet CC `q175016`, FULL ATTENTION succeeds because the gold continuation depends on exact event-handling and JSON unmarshalling details; compression removes useful local evidence rather than only noise.

E.4.1 Automatic Mining Protocol and Aggregate Counts

We form disagreement buckets from qrel-positive ranks, not from manual labels. FULL ATTENTION is the baseline, and the comparison side is the best rank among EOS STATE, MEAN STATE, APPENDED LATENTS $K = 1$, APPENDED LATENTS, EXTRACTED LATENTS $D = 8$, and QUERY-SELECTED EVIDENCE top-$k = 8$. We use this oracle comparison only to mine disagreement cases; endpoint claims use individual operators. Lexical overlap flags are automatic lexical proxies over the retrieved positive and top-ranked negative, including identifier/API overlap, rare-token overlap, exact substring overlap, and near-duplicate line evidence. These flags identify local-evidence alignment, not semantic-error labels by definition. The catalog below is produced by the fixed miner before manual inspection; manual explanations only interpret the automatically emitted cases and do not affect which cases enter the catalog. The statistics in Tables 28–30 are also computed before manual inspection.

Here "comparison" is the best rank among EOS STATE, MEAN STATE, APPENDED LATENTS $K = 1$, APPENDED LATENTS, EXTRACTED LATENTS $D = 8$, and QUERY-SELECTED EVIDENCE top-$k = 8$. All counts are query-level at top-10.

Table 28: Disagreement counts from the case-study miner across the completed CoIR case tasks. Task names are abbreviated.

Task	Q	Comp. resc.	Comp. bet.	Full resc.	Full bet.	Agree fail
AppsRet.	3765	587	346	46	79	2295
COIR CSN	8122	859	1669	85	279	1742
CFB-MT	13277	697	1569	250	621	3594
CFB-ST	31306	1299	5169	213	1187	1946
CSN-CC	8122	831	1711	63	181	1643
CTO Contest	221	12	36	0	7	19
CTO DL	180	24	65	0	4	11
CosQA	500	104	127	6	21	133
SOQA	1994	135	225	1	13	79
Text2SQL	5851	331	1716	54	248	453
All micro	73338	4879	12633	718	2640	11915

Table 29: Automatic lexical overlap proxy rates at the default threshold 0.03.

Bucket	Q	Full al.	Comp. al.	Full only	Comp. only	Gap
Struct. fav.	17512	32.3%	18.2%	2872	402	0.0512
FULL ATTENTION fav.	3358	21.5%	29.9%	145	425	-0.0409
Agree succ.	40553	3.8%	3.7%	84	67	-0.0002
Agree fail	11915	46.6%	43.7%	1115	762	0.0047

Table 30: Threshold sensitivity for the automatic lexical overlap proxy on directional disagreement buckets.

Bucket	Th.	Q	Full al.	Comp. al.	Paired p
Struct. fav.	0.01	17512	39.6%	22.1%	$< 10^{-300}$
Struct. fav.	0.03	17512	32.3%	18.2%	$< 10^{-300}$
Struct. fav.	0.05	17512	27.5%	15.7%	$< 10^{-300}$
Struct. fav.	0.10	17512	21.0%	12.0%	8.1×10^{-288}
FULL ATTENTION fav.	0.01	3358	26.7%	36.6%	5.6×10^{-39}
FULL ATTENTION fav.	0.03	3358	21.5%	29.9%	7.3×10^{-33}
FULL ATTENTION fav.	0.05	3358	18.5%	25.6%	8.0×10^{-28}
FULL ATTENTION fav.	0.10	3358	14.4%	21.0%	6.2×10^{-29}

A row is lexically aligned when the top-ranked negative has stronger lexical overlap evidence than the qrel-positive under the automatic proxy. Structured-favored rows are cases where the comparison side ranks the positive higher than FULL ATTENTION; FULL ATTENTION-favored rows are the reverse. Across thresholds, the sign pattern is stable: when structured exposure is favored, the FULL ATTENTION side has more lexically aligned negatives; when FULL ATTENTION is favored, the comparison side has more lexically aligned negatives.

E.4.2 Illustrative Reading

The case study is intentionally illustrative. The representative rows and automatic lexical overlap summaries above show plausible trade-offs between exact local evidence and compact source exposure, and they help interpret where the aggregate CoIR differences may come from. They are not used as endpoint estimates, and they do not prove a mechanism: the matched quantitative ablations, memory measurements, and replay checks remain the evidence for the paper's main claims.

F Reproducibility Details

F.1 Experimental Setup Details

This subsection expands the condensed experimental protocol in Section 5.

Benchmarks and metrics. CoIR is our main in-domain benchmark; we report nDCG@10 on its ten subtasks (Li et al., 2025). BEIR is a cross-domain direction check: models are trained only on code and evaluated on its 13-task natural-language suite without BEIR-specific fine-tuning (Thakur et al., 2021).

Candidate-batch terminology. We reserve `top_k` for raw data construction: `top_k=4` keeps four companion source documents per target before batching, and is not the softmax candidate count. *Docs* denotes the total materialized source-document pool for the row before target–source self-masking. It is not the raw `top_k` companion count and is not inferred by multiplying the 16-document local row by the number of devices. The 256-document CSGR rows are configured directly as 256-document share-group pools. The fixed-K materialized candidate-batch sweep in Table 20 varies this pool from 16 to 256 documents under the same seed, model, data, optimizer, and source-exposure capacity; the 16-document row is the local row, and the larger rows are CSGR rows. *Cand.* denotes the routed candidates left for each target after self-masking, namely Docs-1 in this protocol. The softmax normalization set $\mathcal{S}_i$ in Eq. 1 is this after-mask routed set, so the 16-doc and 256-doc rows normalize over 15 and 255 candidates, respectively. This change is the operating-point intervention itself.

Training protocol. Matched runs use the public Revela code corpus for one epoch with LoRA rank 256, DeepSpeed ZeRO-2, bf16, `top_k=4` raw companions, gradient accumulation 4, and 8×80GB GPUs. For the fixed-K materialized candidate-batch sweep, only the materialized document batch and its after-mask candidate/normalization set are varied. The language

model is always Llama-3.2-1B; only the separate retriever backbone varies over Qwen2.5-0.5B, Llama-3.2-1B/3B, and Qwen3-8B. Source exposure rows share data, optimizer, steps, routing temperature, NTP objective, evaluation code, and random seed unless stated otherwise. FULL ATTENTION is our internal REVELA-style reproduction under this protocol: it keeps the full source-token KV interface, uses the same NTP objective, and disables compression slots. The 0.5B/1B/3B/8B CSGR backbone endpoint rows use 256 docs with K=16 and are reported as larger-backbone endpoint checks. Command-level hyperparameters are in Appendix F.2.

Baselines. Following CoIR and REVELA reporting (Li et al., 2025; Cai et al., 2026), we include BM25, supervised or weakly supervised embedding models, zero-shot APIs, and NTP-trained baselines. Published scores provide context. They are separate from our FULL ATTENTION row, which is the matched full-token REVELA-style reproduction used for causal comparisons with CSM, stop-gradient sharing, and SGR. Because the training corpus and CoIR both contain code and StackOverflow-derived material, Appendix F.3 reports a lexical overlap audit against CoIR queries, corpus documents, and qrel positives. The audit finds 250 any-hit items among 2.61M checked eval items (0.0096%), with 24 hits among 157k qrel-positive documents (0.0153%); we therefore keep published rows as reporting context, while noting that exact and near-duplicate lexical overlap appears rare and concentrated in StackOverflowQA. As a coarse sensitivity check, excluding the entire StackOverflowQA task leaves the means at 55.92 for FULL ATTENTION, 57.80 for 1B CSGR, 59.49 for 3B CSGR, and 62.57 for 8B CSGR (Appendix Table 33).

Terminology and provenance. We use *published external context* for copied baselines and *ours* for runs under our protocol. Internal attribution, support, and larger-backbone rows report their seed provenance in the relevant tables; Appendix F.4 records audit fields.

F.2 Implementation Details

For reproducibility, the implementation record includes tokenizer, source length L, backbone LM, retriever encoder, optimizer, learning rate schedule, temperature τ, training steps, hardware, checkpoint selection, random seeds, and evaluation commands. Run-name conventions and matched-run provenance are summarized below.

Environment. The main local environment uses Python 3.8.20, PyTorch 2.4.1+cu121, Transformers 4.45.2, DeepSpeed 0.15.2, PEFT 0.13.2, Datasets 3.1.0, MTEB 1.14.22, `coir-eval` 0.7.0, and `pytrec-eval-terrier` 0.5.6. Training uses DeepSpeed ZeRO-2, bf16, and 8×80GB GPUs. The anonymous artifact includes the Conda environment file, DeepSpeed config, source patch, and sanitized launch scripts.

Data and splits. Following REVELA, training uses the public Revela code training corpus, a JSONL file of 358,763 batches built from CodeRAG-Bench StackOverflow posts, online tutorials, and library documentation. The local checksum for the training JSONL used by the reproduction package is:

```
3ca23d7d70e75fc674851e1b259a5d4d
83bb4aac48130f42af513b205aa9acb7
```

CoIR labels are not used during training. Evaluation uses the official CoIR query, corpus, and qrel files for the ten tasks in Table 7; qrels are used only for nDCG@10 evaluation, contamination auditing, and post-hoc case-study grouping. We follow the REVELA/CoIR protocol and do not claim semantic benchmark-corpus decontamination; the lexical audit in Appendix F.3 measures exact, normalized, and near-duplicate overlap.

F.3 Benchmark Contamination Audit

Because the training corpus is built from StackOverflow posts, tutorials, and documentation, and CoIR also contains code-search and StackOverflow-derived tasks, absolute endpoint interpretation requires an explicit overlap audit. The matched ablations are still internally comparable because every row uses the same training corpus and evaluation pipeline; the audit below addresses whether benchmark endpoints are likely to be dominated by direct lexical reuse.

The audit compares the Revela training JSONL `/path/to/revela_code_training_corpus/training.batchaware_g1024_dpp16.jsonl` against the local CoIR HuggingFace dataset cache for the test split. For each CoIR task, it checks three benchmark sides: qrel-restricted queries, all corpus documents, and qrel-positive corpus documents. It uses raw exact 128-bit content hashes, normalized hashes after NFKC normalization, lowercasing, and

non-alphanumeric whitespace collapse, and 64-bit SimHash near-duplicate screening at Hamming distance ≤ 3. The minimum normalized length is 32 characters. SimHash screening in this low-cost run is applied to queries and qrel positives, with pair-specific sampled-anchor confirmation requiring at least two shared 5-gram anchors. The run scanned 7.33M training units, used 7.25M after length filtering, and checked 2.61M eval items. The emitted artifacts are `docs/generated/contamination_training_vs_coir_20260525.summary.csv`, `.details.csv`, `.flagged_ids.csv`, `.tasks.csv`, and `.meta.json`.

Table 31: Lexical contamination audit summary.

Benchmark side	Eval items	Exact	Norm.	SimHash	Any / rate
Queries	161,969	13	41	77	78 / 0.0482%
Corpus docs	2,294,004	52	148	–	148 / 0.0065%
Qrel positives	157,026	4	9	22	24 / 0.0153%
All checked items	2,612,999	69	198	99	250 / 0.0096%

Table 32: Highest-overlap audit rows.

Task	Side	Eval items	Any	Any rate
StackOverflowQA	queries	1,994	42	2.1063%
StackOverflowQA	corpus docs	19,917	112	0.5623%
StackOverflowQA	qrel positives	1,994	6	0.3009%
CodeFeedbackST	queries	31,300	22	0.0703%
CodeFeedbackMT	queries	13,277	7	0.0527%
CodeFeedbackST	qrel positives	31,197	6	0.0192%
CSN-CC/java	corpus docs	176,476	26	0.0147%

Table 33 is a coarse sensitivity check: no ranking is recomputed, no filtered training is performed, and the table simply removes the task where lexical overlap is most concentrated.

Table 33: CoIR mean after removing StackOverflowQA.

Row	CoIR mean	StackOverflowQA	Mean w/o StackOverflowQA
FULL ATTENTION 1B	58.81	84.85	55.92
CSGR 1B (K=16)	61.10	90.76	57.80
CSGR 3B (K=16)	62.67	91.25	59.49
CSGR 8B (K=16)	65.61	93.00	62.57

The audit does not show broad direct contamination: only 250 of 2.61M eval items are flagged by any configured lexical rule, and only 24 qrel-positive documents are flagged. The overlap is concentrated in StackOverflowQA, where both the training corpus and benchmark derive from StackOverflow-like material. This supports using the matched ablations as causal evidence and suggests that the reported endpoints are not primarily explained by exact or near-duplicate lexical reuse. The remaining caveat is scope: this is a bounded lexical audit, not a semantic memorization test or a contamination-filtered retraining run. In the metadata, no SimHash buckets are pruned by the maximum-bucket-size guard (`near_bucket_skipped=0`, `near_bucket_members_skipped=0`), but the sampled-anchor prefilter skips broad anchors to keep pair-specific confirmation tractable.

Training command template. All source exposure rows share the command template below; the exposure-specific variables are listed after the block. Absolute paths are anonymized here, and no API keys or private logging credentials are required for reproduction.

Trainable parameter scope. Unless otherwise stated, training is joint rather than retriever-only. The dense base weights of both the retriever backbone and the Llama-3.2-1B language-model predictor are kept frozen, and optimization updates their LoRA adapters on the target modules listed in the command template, together with the source-exposure parameters introduced by the selected operator. For CSM these additional parameters are the shared appended compression-slot embeddings; FULL ATTENTION, EOS STATE, MEAN STATE, and QUERY-SELECTED EVIDENCE add no trainable source-interface parameters, and the cross-attention gate is fixed (`-trainable_cross_attn_gate False`). FULL ATTENTION is the internal REVELA-style full-token reproduction used for matched causal attribution, not a copied published score. The same language-model adapter parameters are used in the source encoding pass and the mixed prediction pass, so SGR replay returns source-memory gradients to that source pass while the mixed-prediction backward contributes its token-loss gradients. CoIR and BEIR evaluation use only the trained retriever encoder to embed queries and corpus documents; the language model and compression slots are training-time components and are not queried at retrieval evaluation time.

Matched endpoint comparisons in this manuscript use `-seed 42`; the block above is the reported local-control command template. The fixed-K materialized candidate-batch sweep varies the materialized source-document batch reported as *Docs*.

Table 34: Training command template shared by source exposure rows. Absolute paths are anonymized.

```
export REPO_ROOT=/path/to/anonymous/revela
export DATA_PATH=/path/to/revela_code_training_
  corpus/training.jsonl
export MODEL_PATH=/path/to/meta-llama/Llama-3.2-1B
export REFERENCE_MODEL_PATH=/path/to/meta-llama/
  Llama-3.2-1B
export OUTPUT_DIR=/path/to/output/${RUN_NAME}
export GPU_LIST=0,1,2,3,4,5,6,7
export WANDB_DISABLED=true
export LORA_TARGETS=q_proj,k_proj,v_proj,o_proj,\
down_proj,up_proj,gate_proj

cd ${REPO_ROOT}
PYTHONPATH=${REPO_ROOT}/src \
deepspeed --include localhost:${GPU_LIST} \
  --module tevatron.llm_retriever.driver.train \
  --deepspeed deepspeed/ds_zero2_config.json \
  --output_dir ${OUTPUT_DIR} \
  --model_name_or_path ${MODEL_PATH} \
  --reference_model_name_or_path \
    ${REFERENCE_MODEL_PATH} \
  --lora --lora_r 256 --lora_alpha 64 \
  --lora_dropout 0.1 \
  --lora_target_modules ${LORA_TARGETS} \
  --bm25_retrieval_file ${DATA_PATH} \
  --add_passage_prefix True \
  --add_query_prefix True \
  --passage_prefix "passage:  " \
  --query_prefix "query:  " \
  --first_half False --bf16 True --pooling eos \
  --append_eos_token True \
  --normalize True --temperature 0.01 \
  --attn_temperature 0.0001 \
  --exclude_diagonal True --disable_v_norm True \
  --inbatch_attn_layer_strategy all \
  --cross_attn_gate 1.0 \
  --trainable_cross_attn_gate False \
  --source_memory_mode ${SOURCE_MEMORY_MODE} \
  --compression_tokens ${COMPRESSION_TOKENS} \
  --detail_slot_count ${DETAIL_SLOT_COUNT} \
  --legacy_query_evidence_mode \
    ${LEGACY_QUERY_EVIDENCE_MODE} \
  --query_token_top_k ${QUERY_TOKEN_TOP_K} \
  --query_router_loss_beta 0.0 \
  --query_router_temperature 1.0 \
  --query_token_selection_mode heuristic_only \
  --query_evidence_lambda 1.0 \
  --per_device_train_batch_size 4 \
  --train_group_size 4 \
  --learning_rate 1e-4 --passage_max_len 157 \
  --num_train_epochs 1 \
  --gradient_accumulation_steps 4 \
  --logging_steps 1 \
  --warmup_steps 100 \
  --dataloader_num_workers 4 --top_k 4 --seed 42 \
  --save_steps 500 --run_name ${RUN_NAME} \
  --report_to none \
  --log_retriever_metrics True \
  --log_system_metrics True \
  --log_ntp_losses True
```

`top_k=4` controls how many unique source documents are retained from each raw training pack before batching; it is not the final source batch size reported in the operating-point table. Each target masks its own source, so the actual routed candidate count is one fewer: 15 in the 16-document local row and 255 in the 256-document global row. Throughout the tables, *Docs* is the total materialized source-document pool before this self-mask, *Cand.* is the after-mask routed candidate count, and the softmax normalization set is the same after-mask candidate set unless explicitly stated otherwise. The candidate and softmax-normalization set therefore change as the intended operating-point intervention; they are not a second intervention in the fixed-K materialized candidate-batch sweep.

The main operator settings are: FULL ATTENTION uses `SOURCE_MEMORY_MODE=full`, `COMPRESSION_TOKENS=0`, `DETAIL_SLOT_COUNT=0`, and disables legacy query evidence, giving the internal full-token REVELA-style reproduction. EOS STATE uses `SOURCE_MEMORY_MODE=eos`; MEAN STATE uses `SOURCE_MEMORY_MODE=mean`; APPENDED LATENTS uses `SOURCE_MEMORY_MODE=compression` with `COMPRESSION_TOKENS=K`; EXTRACTED LATENTS uses `SOURCE_MEMORY_MODE=detail` with `DETAIL_SLOT_COUNT=D`; and QUERY-SELECTED EVIDENCE top-k uses `SOURCE_MEMORY_MODE=full`, legacy query evidence set to `topk_full_kv`, and `QUERY_TOKEN_TOP_K=k`.

Checkpointing and evaluation. Main-line local-control runs with `top_k=4` and 16 materialized docs train for one epoch, 2803 optimizer steps in the code-corpus setup, saving every 500 steps and at the final checkpoint. CoIR evaluation encodes queries and corpus with batch size 64, evaluates the ten CoIR tasks independently, and reports the unmerged clean nDCG@10 mean only when all ten task summaries are present. The qualitative case study uses completed CoIR rankings and qrel-positive ranks post hoc; the scripts that mine disagreement buckets are included in the artifact.

Anonymous supplement contents. The anonymous supplement is available at https://anonymous.4open.science/r/anonymous-csgr-revela-9B03/. It includes code modifications, training and evaluation scripts, environment files, DeepSpeed config, run metadata schema, sanitized command templates, and generated case-study metadata. It does not redistribute Llama weights, raw third-party training data, private checkpoints, W&B keys, or machine-local paths. Users must obtain gated base models and public datasets under their original

terms.

F.4 Terminology and Run Provenance

Paper terminology is defined by the source exposure operator, not by historical experiment names. FULL ATTENTION exposes raw source-token KV states and serves as our internal full-token REVELA-style reproduction; CSM/APPENDED LATENTS exposes compression slots inserted into the source sequence; EXTRACTED LATENTS exposes externally extracted latent memories; QUERY-SELECTED EVIDENCE exposes selected source-token KV states; and CSGR with g=256 denotes CSM+SGR with 256 materialized docs.

For provenance, the tables record only the fields needed to identify a test: exposure operator, capacity (K, D, or top-k), routed-layer setting, `top_k`, materialized docs when relevant, checkpoint/evaluation status, seed, hardware/status notes, and whether the clean CoIR or BEIR task set is complete. Internal run ids, directory paths, and implementation flag strings are not paper terminology and are omitted here.

All internal appendix tables should be read as clean evaluation rows for an exposure operator unless explicitly marked as diagnostics. The endpoint rows tied to Table 6 are the complete matched `top_k=4`, 16-doc main line with clean CoIR evaluations at seed 42. Auxiliary generalization checks and support sweeps are also seed-42 diagnostics unless their table explicitly states a different provenance; they are not used to estimate uncertainty.

F.5 Assets, Licenses, and Terms

Table 35 summarizes the external assets used in the paper. The anonymous supplement releases only our code modifications, scripts, metadata schemas, and derived evaluation or case-study metadata needed to reproduce the analyses. It does not re-host gated model weights, raw benchmark corpora, raw training corpora, or proprietary API outputs. When an asset aggregates multiple upstream sources, the table records the aggregate package when available and notes that the original source terms still apply.

Table 35: External assets and release treatment.

Asset	Release and terms treatment
Code and local modifications	Release code, patches, scripts, configs, and metadata; remove secrets and machine-local paths.
Llama-3.2 backbones	Do not redistribute model weights; users obtain gated Llama access under Meta's terms.
Qwen2.5-0.5B retriever backbone	Used only for the 0.5B larger-backbone check; users obtain upstream weights under the original model terms.
Qwen3-8B retriever backbone	Used only for the 8B larger-backbone endpoint; users obtain upstream weights under the original model terms.
Revela training corpus	Provide path/checksum and preparation assumptions; do not redistribute raw corpus text.
CodeRAG-Bench sources	Users obtain upstream StackOverflow, tutorial, and documentation data under the original terms.
CoIR and `coir-eval`	Use official benchmark/qrels for evaluation; do not repackage raw CoIR text or qrels.
External baselines	Reproduce only cited aggregate scores; do not redistribute baseline weights, API outputs, or proprietary responses.
BEIR	Use only for zero-shot evaluation; do not redistribute raw BEIR data.
Case-study metadata	Release ids, scripts, summaries, and derived metrics; omit raw copyrighted text unless permitted.