CLP: Compact Latent Proxies for Representation-Level Compression of Reasoning Traces

Anonymous ACL submission

Abstract

Reasoning traces help large language models solve difficult problems, but long traces increase output tokens, latency, storage cost, and human inspection burden. Existing reasoning-trace compression methods mainly shorten the visible text through prompting, drafting, or rewriting. These approaches are useful, but they leave open whether a compact internal proxy can help shorten visible reasoning traces before text is fully decoded. In this paper, we propose CLP (Compact Latent Proxy), a lightweight representation-level compression module for frozen decoder-only language models. CLP pools top-layer token states into a small latent proxy, reconstructs token-aligned proxy states, and mixes proxy logits with the original backbone logits during decoding. This design does not claim that all reasoning is forced through the proxy. Instead, it provides a controlled and conservative proxy path that can support shorter visible traces while the base model remains available for answer stability. Across nine reasoning benchmarks, CLP achieves 66.6% mean accuracy and 62.0% mean trace compression, yielding the highest mean accuracy among the compared methods while maintaining substantial trace compression. The results position compact latent proxies as a base-supported representation-level route for improving the accuracy–compression trade-off of visible reasoning traces.

1 Introduction

Reasoning traces, such as Chain-of-Thought (CoT), have become a cornerstone for enhancing the reliability of Large Language Models (LLMs) on complex tasks (Wei et al., 2022; Wang et al., 2023; Yao et al., 2023). By making problem decomposition explicit and maintaining intermediate computational states, these traces allow models to navigate intricate reasoning paths more effectively. However, generating long-form rationales introduces significant costs, including increased decoding latency, higher token consumption, and larger storage requirements. In practical applications, these costs drive a joint objective: reducing the length of visible reasoning traces without suppressing the underlying computation required to ensure answer correctness. Existing methods for shortening reasoning traces primarily operate at the output level. Some control length via explicit token budgets or prompting, while others utilize draft-and-rewrite pipelines to compress long rationales (Han et al., 2025; Kang et al., 2024). While these behavioral-level approaches are effective at reducing text length, they rely on problem-specific complexity estimation—which is often unreliable when surface features do not reflect actual reasoning difficulty. Furthermore, methods that remove redundant tokens or steps through pruning can risk breaking the logical continuity of a reasoning chain, because locally unimportant tokens or steps may still support later decisions (Xia et al., 2025). Another direction involves latent or continuous reasoning, which attempts to move computation into continuous hidden spaces, thereby bypassing the need for full natural language expression (Hao et al., 2024; Shen et al., 2025). While these methods offer high compression potential, their internal representations can be difficult to structure or audit, making the resulting computation harder to interpret or control. Without an explicit structure for controlling what is retained, fully implicit reasoning can be difficult to audit and may lose answer-supporting information. In this paper, we propose CLP (Compact Latent Proxy), a representation-level compression module for frozen decoder-only backbones. CLP introduces a structured *proxy path* that routes answer-supporting signals through a restrictive latent bottleneck before the final next-token distribution is formed. The proxy path is not a second language model and not a replacement for the frozen backbone. Instead, it is a trainable auxiliary

route attached to the backbone's top-layer hidden states. It first compresses the token-level hidden sequence into a small set of latent proxy slots, then reconstructs token-aligned proxy states, and finally produces proxy logits that are mixed with the original base logits. In this sense, "proxy" means that the route provides a compact auxiliary signal for information that would otherwise contribute to a long visible trace. This design is intentionally conservative. The backbone path remains available during decoding, so the model does not need to force all reasoning into a tiny latent representation. The proxy path is used as a controlled compression interface: if the latent slots preserve answer-supporting cues, the mixed decoder can produce shorter traces without losing too much answer accuracy; if the proxy is insufficient, the base logits still provide stabilizing support. To encourage the proxy path to retain information useful for answer prediction, we implement a hybrid training objective. The objective combines proxy-logit causal language-modeling supervision with hidden-state reconstruction losses and auxiliary sufficiency, capacity, redundancy, and entropy terms; it is not a distribution-level distillation objective over backbone logits. Our experiments across nine reasoning benchmarks show that CLP reaches 66.6% mean accuracy and 62.0% mean trace compression, achieving the strongest mean accuracy while retaining substantial compression. These results show that compact latent proxies can improve the trade-off between final-answer accuracy and visible-trace compression, offering a controlled alternative to purely surface-level shortening or unconstrained latent methods.

The major contributions of this paper are as follows.

- We formulate visible reasoning-trace compression as a representation-level problem and define a base-supported proxy path over frozen decoder-only backbones.
- We propose CLP, a lightweight module that compresses top-layer hidden states into latent proxy slots, reconstructs token-aligned proxy states, and mixes proxy logits with base logits to guide shorter reasoning traces.
- We conduct experiments on nine reasoning benchmarks, showing that CLP improves the reported accuracy–compression trade-off while also identifying clear boundaries of proxy-only decoding and over-compressed traces.

2 Related Work

Explicit CoT compression. Chain-of-thought (CoT) prompting improves the reasoning ability of large language models by encouraging them to generate intermediate natural-language steps before producing the final answer (Wei et al., 2022; Kojima et al., 2022). As reasoning models become more capable, their intermediate traces can become increasingly verbose, which raises decoding cost, latency, and deployment overhead. This motivates recent work on efficient reasoning, where the goal is to preserve answer correctness while reducing unnecessary reasoning tokens (Sui et al., 2025). Some methods control the length of reasoning through prompts or token budgets. For example, TALE estimates a problem-specific reasoning budget and guides the model to reason within that budget (Han et al., 2025). Other methods construct shorter CoT traces for training. C3oT compresses long rationales into shorter rationales and trains the model to generate compressed chains while preserving key answer-supporting content (Kang et al., 2024). Self-training methods similarly exploit the model's own ability to produce shorter correct traces by selecting concise reasoning paths and using them for fine-tuning (Munkhbat et al., 2025). CoT-Valve further studies controllable length compression by tuning models to generate reasoning chains of different lengths according to task difficulty (Ma et al., 2025). These approaches are effective and practical, but they mainly operate on the visible text trace. CLP differs by applying compression to frozen hidden states before the model commits to a long visible trace.

Pruning and adaptive stopping for efficient reasoning. Another line of work reduces reasoning length during inference by identifying redundant tokens or steps. TokenSkip analyzes the semantic importance of tokens in CoT outputs and trains models to skip less important tokens, enabling controllable compression with limited accuracy loss (Xia et al., 2025). Step Entropy measures the information contribution of each reasoning step and uses low-entropy steps as candidates for compression or replacement with skip tokens (Li et al., 2025). Reinforcement-learning-based methods also encourage shorter reasoning. ThinkPrune trains long-thinking models under progressively tighter token

limits, forcing the model to consolidate reasoning while retaining essential steps (Hou et al., 2025). Inference-time early-stopping methods, such as ES-CoT, stop generation once the answer appears stable or the answer distribution becomes sufficiently confident (Mao et al., 2025). These methods reduce redundant decoding, but most of them still depend on an explicit natural-language reasoning trajectory. CLP is complementary to these methods: it compresses a proxy representation rather than pruning an already generated trace.

Latent and continuous reasoning. Recent studies question whether natural language is the most efficient medium for all reasoning operations. Coconut introduces continuous thoughts by feeding the last hidden state back into the model as the next input embedding, allowing reasoning to proceed in latent space rather than purely in token space (Hao et al., 2024). CCoT proposes dense continuous contemplation tokens as compressed representations of explicit reasoning chains, enabling additional computation with fewer visible reasoning tokens (Cheng and Van Durme, 2024). CODI distill explicit CoT reasoning into continuous space through self-distillation and aligns hidden activations between explicit and implicit reasoning modes (Shen et al., 2025). These works are closely related to our motivation: not all answer-supporting computation needs to be verbalized token by token. However, many latent-reasoning methods focus on replacing or augmenting CoT with continuous tokens. Our method differs by constructing a bottlenecked proxy representation from the model's internal hidden states and mixing proxy logits with base logits, which provides a controlled interface between the frozen backbone and the visible-trace compression path.

Bottlenecked representations and proxy reasoning. Representation bottlenecks have been widely used to encourage compact, task-relevant internal representations (Tishby et al., 2000; Alemi et al., 2017). In language models, adapter-style modules and parameter-efficient tuning methods show that small trainable components can steer a large frozen backbone without updating all model parameters (Houlsby et al., 2019; Hu et al., 2022). Our work follows this general principle but applies it to reasoning-trace compression. Instead of fine-tuning the full model or only shortening the final text, CLP builds a trainable proxy path over frozen top-layer token states. The proxy path compresses token-level hidden states into a small set of latent proxy slots, reconstructs a token-aligned proxy representation, and fuses it with local states for answer prediction. This design encourages the proxy to retain task-relevant signals under a compact bottleneck.

3 Method

3.1 Problem setup

Let x denote a reasoning prompt, and let y denote the generated rationale-plus-answer sequence. A frozen decoder-only backbone receives the current prefix and produces top-layer hidden states $H \in \mathbb{R}^{T\times d}$, where T is the prefix length and d is the hidden-state width of the backbone. The reasoning-trace compression goal is to generate a shorter visible trace while preserving final-answer correctness under the same evaluation protocol. In other words, the visible trace should be short enough to reduce decoding cost, but the final answer should remain reliable.

3.2 Overview

Figure 1 summarizes CLP. Given top-layer hidden states from a frozen decoder-only backbone, CLP first maps the token sequence into a small set of latent proxy slots. It then reconstructs token-aligned proxy states and produces proxy logits, which are mixed with the frozen backbone logits during decoding. Only proxy-side modules are trained with a hybrid objective that combines proxy-logit causal LM supervision, hidden-state reconstruction, and auxiliary regularization. This design keeps the backbone available for answer stability while allowing the proxy path to support shorter visible traces.

3.3 Compact latent proxy

CLP compresses the full top-layer hidden-state sequence into a small set of latent proxy slots:

$$R = \mathcal{B}(H), \qquad R \in \mathbb{R}^{K\times d_b}. \tag{1}$$

Here, $\mathcal{B}$ is the proxy encoder, R is the compact proxy, K is the number of latent slots, and d_b is the proxy width. Since $K \ll T$ in the compressed setting, this stage forms the central bottleneck of the method. Its purpose is not to copy all token-level states, but to retain a compact representation that is useful for final-answer prediction. In implementation, the proxy encoder normalizes H, projects it into the proxy width, and pools the sequence

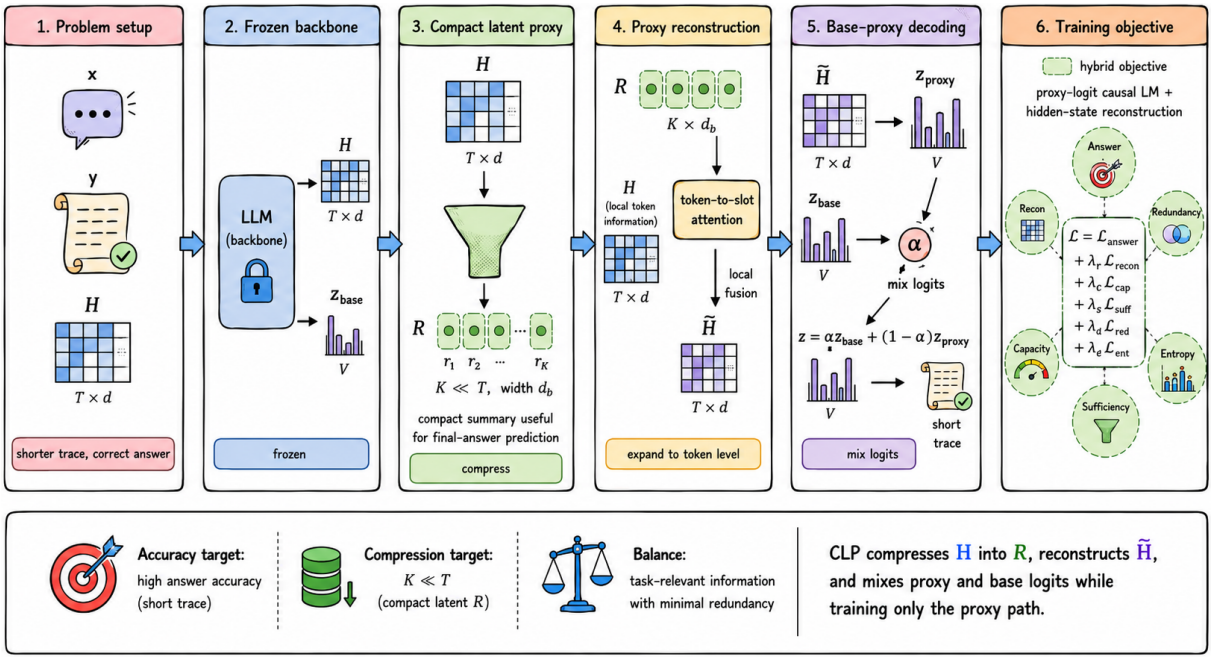

Figure 1: Overview of Compact Latent Proxy (CLP). The frozen backbone LLM provides top-layer token states H and base logits. CLP compresses H into latent slots R, reconstructs token-aligned proxy states $\widetilde{H}$, and mixes proxy logits with base logits for generation. Only the proxy-side modules are trained. The hybrid objective combines proxy-logit causal language-modeling supervision with hidden-state reconstruction losses, together with capacity, sufficiency, redundancy, and entropy regularization terms.

into latent slots. The detailed slot-pooling design, including the role of segment queries and global context, is given in Appendix A.

3.4 Proxy reconstruction

The compact proxy is expanded back into token-aligned proxy states:

$$\widetilde{H} = \mathcal{G}(H, R). \tag{2}$$

Here, $\mathcal{G}$ is the proxy reconstruction and local-fusion module, and $\widetilde{H}$ is the reconstructed token-level proxy sequence used for decoding. This step is needed because next-token prediction is token-aligned. The model therefore cannot use a single global proxy vector in the same way for every position. Instead, each token position attends to the latent slots and combines the retrieved proxy information with local token information. Additional reconstruction details are provided in Appendix A.

3.5 Base-proxy decoding

CLP computes proxy logits z^{proxy} from $\widetilde{H}$ and mixes them with the frozen backbone logits z^{base}:

$$z = \alpha z^{\text{base}} + (1-\alpha) z^{\text{proxy}}. \tag{3}$$

Here, z denotes the final mixed logits before softmax, and α is the base-logit weight. A larger α relies more on the frozen backbone, while a smaller α gives the proxy path more influence. This design keeps the method conservative. The proxy path can guide the model toward shorter traces, but the backbone remains available for answer stability. Thus, gains from CLP should be interpreted as evidence for a useful supported proxy path, not as evidence that the proxy alone replaces the full reasoning computation. The detailed mixing interpretation and sweep endpoints are reported in Appendix A.

3.6 Training objective

Only proxy-side parameters are trained; the backbone remains frozen. The training objective is

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\text{answer}} + \lambda_r \mathcal{L}_{\text{recon}} + \lambda_c \mathcal{L}_{\text{cap}} \\ + \lambda_s \mathcal{L}_{\text{suff}} + \lambda_d \mathcal{L}_{\text{red}} + \lambda_e \mathcal{L}_{\text{ent}}.\end{aligned} \tag{4}$$

Here, $\mathcal{L}_{\text{answer}}$ provides proxy-logit causal language-modeling supervision on the answer sequence, and $\mathcal{L}_{\text{recon}}$ is a hidden-state reconstruction loss that aligns reconstructed proxy states

with frozen hidden-state targets. The training objective is hybrid: it combines proxy-logit causal LM supervision with hidden-state reconstruction losses and auxiliary regularization. It is not a distribution-level distillation objective over backbone logits. $\mathcal{L}_{\text{cap}}$ maintains bottleneck pressure, $\mathcal{L}_{\text{suff}}$ encourages answer-supporting information to remain available in the proxy, and $\mathcal{L}_{\text{red}}$ and $\mathcal{L}_{\text{ent}}$ regularize slot usage. The coefficients λ_r, λ_c, λ_s, λ_d, and λ_e control the corresponding auxiliary terms. The main experiments use one fixed hyperparameter setting rather than tuning these coefficients separately for each dataset. The detailed meaning of each loss, the selected values, and the sensitivity evidence are moved to Appendix A to keep the main method concise.

4 Experimental Setup

Research questions. The experiments are organized around three main research questions:

- RQ1: Does CLP improve the accuracy–compression trade-off?
- RQ2: Which CLP components are necessary?
- RQ3: How much bottleneck capacity and base support are needed?

RQ1 asks whether CLP improves the accuracy–compression trade-off against all baselines; RQ2 tests which model components are necessary; and RQ3 studies the bottleneck capacity and base/proxy mixing weight.

Benchmarks. The evaluation uses the benchmark results summarized in the experiment records. The main comparison covers AQUA-RAT (Ling et al., 2017), ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), CommonsenseQA (Talmor et al., 2019), ECQA (Aggarwal et al., 2021), MathQA (Amini et al., 2019), OpenBookQA (Mihaylov et al., 2018), StrategyQA (Geva et al., 2021), and SVAMP (Patel et al., 2021). These tasks cover multiple-choice mathematics, science question answering, commonsense reasoning, explanation-oriented commonsense QA, and yes-or-no strategy reasoning. The external generalization table further uses GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and BIG-Bench Hard symbolic subsets (Suzgun et al., 2022).

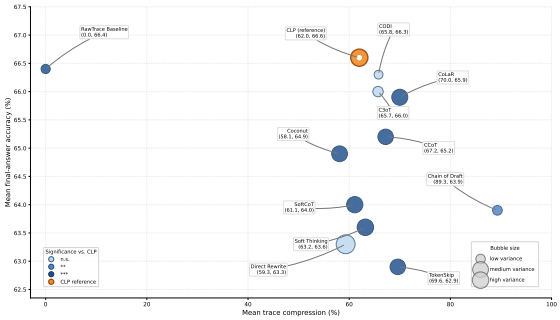

Figure 2: Mean accuracy–compression trade-off across compared methods. CLP is highlighted with a star. Bubble size indicates variance, and color depth indicates significance against CLP under a two-sided Wilcoxon signed-rank test. Full per-dataset results are reported in Appendix C.

Backbones. The main rows use Qwen2.5-1.5B-Instruct (Team, 2024). Robustness rows also report Qwen2.5-0.5B-Instruct, Qwen2.5-3B-Instruct (Team, 2024), Phi-3.5-mini-instruct (Abdin et al., 2024), and Llama-3.2-1B-Instruct (Dubey et al., 2024). These rows are separated from the main comparison so that backbone changes are not mixed with method comparisons.

Compared methods. The RawTrace baseline keeps the standard chain-of-thought trace (Wei et al., 2022). Direct Rewrite is a rewrite-style compression baseline related to self-refinement and post-hoc revision (Madaan et al., 2023). Chain of Draft (Xu et al., 2025a), C3oT (Kang et al., 2024), and CODI (Shen et al., 2025) are included as retained compressed-trace baselines. The broader same-category comparison also includes Coconut (Hao et al., 2024), CCoT (Cheng and Van Durme, 2024), CoLaR (Tan et al., 2025), Soft-CoT (Xu et al., 2025b), Soft Thinking (Zhang et al., 2025), and TokenSkip (Xia et al., 2025).

Metrics. The primary metrics are final-answer accuracy and trace compression. Trace compression is computed as $100 \times (1 - \text{compressed output tokens}/\text{raw trace tokens})$, so larger values mean shorter visible reasoning traces relative to RawTrace. Output tokens, latency, throughput, memory, and proxy overhead are reported as compute-cost measurements. The rationale audit reports average sufficiency, partial-support rate, strict-support rate, and misleading-rate. These audit fields follow the rationale-evaluation motivation of ERASER and the distinction between plausibility and faithfulness in explanation evaluation (DeYoung

et al., 2020; Jacovi and Goldberg, 2020). The audit is used as supporting evidence only; it is not presented as a proof of mechanistic faithfulness. For the experimental results, we report mean $\pm$ SD. Unless otherwise stated, significance is computed using a two-sided Wilcoxon signed-rank test over matched runs and datasets. The significance markers denote paired-test results: $^{*}p < .05$, $^{**}p < .01$, $^{***}p < .001$, and n.s. means not significant.

5 Results

5.1 RQ1: Does CLP improve the accuracy–compression trade-off?

Figure 2 reports the mean accuracy–compression trade-off across the compared methods. The x-axis shows mean trace compression, the y-axis shows mean final-answer accuracy, the bubble size indicates variance, and the color depth indicates the statistical significance of the difference from CLP. Overall, CLP lies in the high-accuracy region of the accuracy–compression frontier, achieving the strongest mean accuracy while still maintaining substantial compression. Specifically, CLP achieves 66.6% mean accuracy and 62.0% mean compression. This result indicates that CLP can reduce visible reasoning traces without moving into the low-accuracy region observed for several aggressive output-shortening methods.

Compared with RawTrace, CLP provides a clear compression benefit while preserving a comparable or slightly stronger accuracy level. RawTrace retains the full reasoning trace and therefore remains at 0.0% compression, whereas CLP substantially shortens the visible trace. Compared with aggressive compression methods such as Chain of Draft, TokenSkip, SoftCoT, and Soft Thinking, CLP achieves higher mean accuracy while avoiding the larger accuracy degradation caused by excessive trace shortening. Although some baselines obtain higher mean compression than CLP, they are positioned lower on the accuracy axis, indicating that stronger compression alone does not necessarily lead to a better overall trade-off.

To assess whether the observed differences are statistically reliable, we use a two-sided Wilcoxon signed-rank test over matched runs and datasets, with $p < 0.05$ as the primary significance threshold. The tests compare each method against CLP under the same matched evaluation setting. The tests show significant differences between CLP and RawTrace, Chain of Draft, SoftCoT, Soft Thinking, TokenSkip, Coconut, CCoT, and CoLaR at the 0.05 level. Differences from Direct Rewrite, C3oT, and CODI are not significant at this threshold, so these comparisons should be interpreted as statistically comparable rather than clear wins for CLP.

The variance encoding in Figure 2 further supports this interpretation. Methods with larger bubbles show greater variability across runs and datasets, while CLP remains near the top of the accuracy range with meaningful compression. Taken together, the mean results and the paired significance tests show that CLP does not simply maximize compression alone. Instead, it provides a balanced trade-off by achieving the highest mean accuracy among the compared methods while still reducing the length of visible reasoning traces.

The same trade-off also appears in the runtime measurements. As reported in Appendix C.7, CLP uses 76.7 mean output tokens and 1.70s mean latency, compared with 194.4 tokens and 3.28s for RawTrace. More aggressive compression methods such as Chain of Draft are faster, with 21.0 tokens and 0.95s latency, but they also show lower mean accuracy in the main comparison. Therefore, CLP should not be described as the fastest method; rather, it provides a practical cost-quality operating point by reducing a large portion of visible trace tokens while preserving stronger accuracy than the most aggressively compressed baselines.

Conclusion for RQ1. CLP provides the most favorable mean accuracy–compression balance in Figure 2. It keeps the strongest mean final-answer accuracy while moving away from the uncompressed RawTrace point, showing that the proposed proxy path reduces visible reasoning length without the larger accuracy degradation observed in more aggressive compression baselines.

5.2 RQ2: Are major CLP components effective?

Table 1 reports the component-level ablation results, using Full CLP as the reference point. The results show that the full architecture is needed for stable visible-trace compression. Removing the slot bottleneck, proxy reconstruction, or local fusion consistently weakens performance, indicating that compression, token-level reconstruction, and local context integration each contribute to decoding. In particular, removing reconstruction causes

Table 1: RQ2 component ablation results. Values are descriptive means over the component-ablation results.

Variant	Acc. (%)	Comp. (%)	Δ Acc. (pp)	Δ Comp. (pp)
Full CLP	66.6 ± 1.23	62.0 ± 1.67	0	0
No-slot / no bottleneck	65.3 ± 1.21***	50.0 ± 1.74	-1.3 ± 1.72	-12.0 ± 2.41
No reconstruction	62.2 ± 1.44***	64.0 ± 2.01	-4.4 ± 1.89	2.0 ± 2.62
No fusion	64.3 ± 1.30***	63.1 ± 1.67	-2.3 ± 1.79	1.1 ± 2.36
Matched dense adapter	65.7 ± 1.24***	54.1 ± 1.57	-0.9 ± 1.74	-8.0 ± 2.30
Random slot / shuffled slot	54.5 ± 1.16***	67.6 ± 1.70	-12.1 ± 1.69	5.5 ± 2.38
Proxy-only	7.1 ± 1.28***	99.0 ± 1.67	-59.5 ± 1.77	36.9 ± 2.37

the largest accuracy drop among the main component ablations, showing that the compact slots must be expanded back to token resolution before they can effectively support next-token prediction.

The comparison with the matched dense adapter further shows that the gain is not simply caused by adding trainable parameters. Although this variant has comparable capacity, it provides weaker compression than the structured bottleneck path. This supports the role of the latent-slot design rather than only the presence of an additional trainable module.

The Random slot / shuffled slot variant provides a slot-level intervention. It keeps the same overall decoding interface but breaks the learned correspondence between inputs and latent slots. The large accuracy drop indicates that the slot contents are functionally used by the decoder rather than being inactive auxiliary variables. This result supports the view that learned latent slots carry input-dependent task information under the base-supported decoding setup.

Finally, the Proxy-only variant clarifies the boundary of the method. Although it achieves very high compression, its accuracy collapses when base-model support is removed. This result should not be read as evidence that the proxy contains no useful task-relevant signal. Instead, it shows that the proxy path is not designed to operate as a standalone language model. The useful information in the proxy becomes effective primarily when combined with the calibrated base logits of the frozen backbone.

Conclusion for RQ2. The full CLP architecture is necessary: the slot bottleneck, proxy reconstruction, local fusion, and base-supported decoding each contribute to stable visible-trace compression. The shuffled-slot intervention shows that learned slots are functionally used, while proxy-only decoding confirms that CLP is a base-supported proxy mechanism rather than a standalone latent reasoner.

Table 2: RQ3 capacity sweep. Values are descriptive means over the capacity-sweep results.

Capacity	Mean Acc.	Mean Comp.	Mean Tokens
slots1_dim16	60.0 ± 1.13***	69.0 ± 1.96	63.1 ± 1.32
slots1_dim64	62.8 ± 1.21***	67.1 ± 1.34	67.0 ± 1.30
slots2_dim64	66.2 ± 1.36***	63.8 ± 2.04	73.2 ± 1.36
slots4_dim32	66.2 ± 1.24	63.7 ± 1.64	73.6 ± 1.73
slots4_dim64	65.7 ± 1.44***	63.3 ± 2.04	73.9 ± 1.35
slots8_dim64	65.8 ± 1.23***	63.8 ± 1.81	73.2 ± 1.72
slots16_dim64	65.4 ± 1.24***	62.7 ± 1.69	75.4 ± 1.63

Table 3: RQ3 base/proxy mixing sweep. α is the base-logit weight.

Mixing setting	Mean Acc.	Mean Comp.	Mean Tokens
alpha=0.75	64.1 ± 1.33***	66.0 ± 1.71	68.7 ± 1.51
alpha=0.9	66.6 ± 1.23	62.0 ± 1.67	76.6 ± 1.35
alpha=1	50.8 ± 1.10**	0	194.6 ± 2.90

5.3 RQ3: How much bottleneck capacity and base support are needed?

Table 2 shows that the capacity of the latent proxy has a non-monotonic effect. In each setting, `slotsX_dimY` means that the compact proxy contains X latent slots and each slot has width Y, corresponding to $R \in \mathbb{R}^{X \times Y}$. Very small settings, such as `slots1_dim16`, impose a strong bottleneck and therefore achieve high compression, but they lose accuracy because a single low-dimensional slot cannot retain enough answer-supporting task information. Increasing the number or width of slots improves accuracy until the proxy becomes expressive enough to carry the necessary task cues. However, very large slot settings do not continue to improve the accuracy–compression trade-off. Once the proxy capacity becomes too large, the bottleneck pressure is weakened and the generated traces become longer, which reduces the compression benefit without bringing clear additional accuracy gains. Thus, CLP works best in a middle-capacity regime where the proxy is compact enough to encourage compression but expressive enough to preserve task-relevant information.

Table 3 explains why the mixed-logit design is necessary. With $\alpha = 1$, the system returns to the raw base path and loses compression. The final $\alpha = 0.9$ setting keeps most of the base model's answer ability while allowing the proxy path to shorten the visible trace. This result narrows the interpretation of CLP: the proxy is most effective as a residual steering signal under strong base support rather than as an autonomous latent reasoning module.

Conclusion for RQ3. CLP works best in a middle-capacity and base-supported regime: the smaller proxies lead to lower accuracies, while the larger proxies weaken compression pressure. The mixed logits with strong base support provide the most reliable operating point.

6 Discussion

The main empirical message of CLP is that latent proxies improve visible-trace compression under base-model support, rather than replacing explicit reasoning. The frozen backbone still provides calibrated decoding support, while the proxy path supplies a compact auxiliary signal that can steer generation toward shorter traces.

The ablation and mixing results support this interpretation. Removing the slot bottleneck, reconstruction, fusion, or learned slot structure weakens performance, and the shuffled-slot variant shows that the latent slots are functionally used rather than inactive auxiliary variables. However, the proxy-only variant performs poorly, and the best mixing setting relies on strong base support. Thus, CLP is better understood as a residual steering mechanism for reasoning-trace compression than as an autonomous latent reasoning module.

The runtime results further indicate a cost-quality trade-off. CLP is not the fastest method, because more aggressive compression methods generate shorter outputs. Its advantage is that it reduces output length and latency relative to RawTrace while preserving stronger mean accuracy than the most aggressively compressed baselines.

Finally, CLP does not claim faithful mechanistic explanation. The rationale audit measures output-level support quality, not whether the visible rationale fully reflects internal computation. Stronger mechanistic claims would require additional causal tracing, probing, or trajectory-alignment analysis.

7 Limitations

The current evaluation has three main limitations. First, the method depends on a base-logit path. The proxy-only results show that the compact proxy is not a standalone reasoning engine. Second, the rationale audit evaluates visible support, not mechanistic faithfulness. A trace may look sufficient while the model relies on other internal information. Third, arithmetic and polarity-sensitive examples remain brittle. The failure cases show that exact calculation and yes/no direction can be lost even when the compressed trace keeps some useful cues. Future work should strengthen the evidence in three ways: further standardize all auxiliary comparisons under identical decoding and hardware settings, add verifier-based arithmetic checks for math tasks, and evaluate whether stronger sufficiency training reduces misleading compressed traces without sacrificing compression. A useful next step is also to report calibration-style diagnostics for when the model should choose a longer trace rather than force a short one.

8 Conclusion

This paper presents CLP, a compact latent proxy approach for reducing visible reasoning traces in frozen decoder-only language models. CLP routes answer-supporting signals through latent slots, reconstructs token-aligned proxy states, and mixes proxy logits with base logits while keeping the frozen backbone available for answer stability. Across the evaluated reasoning benchmarks, CLP improves the reported accuracy–compression trade-off and provides a practical cost-quality operating point rather than simply maximizing compression. Overall, the experimental results position CLP as a practical representation-level compression route. Future work should strengthen adaptive trace-length control and mechanistic analysis under stricter deployment settings.

Data availability.

An anonymized copy of the source code and experimental data is provided in the supplementary materials. The public repository will be released after the review period.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, and 1 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for commonsenseqa: New dataset and models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information

bottleneck. In *International Conference on Learning Representations*.

Aida Amini, Saadia Gabriel, Peter Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed Chain of Thought: Efficient Reasoning Through Dense Representations. *arXiv preprint arXiv:2412.13171*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized nlp models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training Large Language Models to Reason in a Continuous Latent Space. *arXiv preprint arXiv:2412.06769*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2024. C3oT: Generating shorter chain-of-thought without compromising effectiveness. *arXiv preprint arXiv:2412.11664*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. 2025. Making slow thinking faster: Compressing llm chain-of-thought via step entropy. *arXiv preprint arXiv:2508.03346*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. CoT-valve: Length-compressible chain-of-thought tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 6025–6035.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*.

Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. 2025. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25127–25152.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics*.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. 2025. CODI: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 677–693.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*.

Mirac Suzgun, Nathan Scales, Nathanael Sch"arli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and 1 others. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. 2025. Think silently, think fast: Dynamic latent compression of llm reasoning chains. *arXiv preprint arXiv:2505.16552*.

Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint physics/0004057*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. 2025. TokenSkip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025a. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025b. SoftCoT: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 23336–23351.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen, and Xin Eric Wang. 2025. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *arXiv preprint arXiv:2505.15778*.

A Supplementary Method Details

A.1 Details of the compact latent proxy

CLP maps the full hidden-state sequence H into a compact latent proxy:

$$R = \mathcal{B}(H), \qquad R \in \mathbb{R}^{K \times d_b}. \tag{5}$$

Here, $\mathcal{B}$ is the bottleneck or proxy encoder, R is the compact proxy representation, K is the number of latent slots, and d_b is the proxy width. Thus, instead of carrying the full token-level sequence $H \in \mathbb{R}^{T \times d}$ forward, the method compresses the information into K latent slots, each with dimension d_b. The implementation normalizes token features, builds segment-level queries, attends from those queries back to the token states, and mixes attended slots with a global context vector. This design limits the capacity of the proxy while allowing each slot to summarize information from the full prefix. The proxy is smaller than the full token-state sequence in two ways: it has fewer positions than the original prefix, and it can have a narrower feature width than the backbone state. The slot structure gives the module a controlled summary space. Each slot can focus on a different portion or aspect of the prompt, while the total number of slots limits how much detail can be carried forward. The global context vector prevents the slots from becoming purely local summaries, which is necessary when an answer depends on relations between distant parts of the prompt. CLP differs from a direct hidden-state adapter because the proxy first compresses the sequence into a small latent set. A generic adapter can add capacity at every token position, whereas the proxy must pass information through a restricted summary. The compression step provides the intended pressure of the method. If the proxy carries only redundant or irrelevant information, it should not improve decoding. If it preserves answer-supporting information, the mixed logits should recover part of the answer benefit while allowing shorter traces.

A.2 Details of proxy reconstruction

The compact proxy is then expanded into token-aligned proxy states:

$$\widetilde{H} = \mathcal{G}(H, R). \tag{6}$$

Here, $\mathcal{G}$ denotes the proxy reconstruction and local-fusion module. It takes the original token-level hidden states H and the compact proxy R as input, and produces $\widetilde{H}$, a token-aligned proxy state sequence that can be used for next-token decoding. The reconstruction stage compares normalized token features with normalized slot features, computes token-to-slot attention, and blends reconstructed proxy features with local token information. This fusion is necessary because a purely global slot summary can lose token-local cues that affect answer selection. Reconstruction converts the compact proxy back into a form that can interact with token-level decoding. Without this step, the model would need to use the same small set of proxy vectors for every next-token decision, which would make local contexts with the same global summary difficult to distinguish. Token-to-slot attention gives each token position a different view of the proxy, while the local token features keep the reconstructed state anchored to the current prefix. The reconstruction stage therefore plays a balancing role. It should not copy the original hidden state, because doing so would weaken the compression pressure. It also should not rely only on the latent slots, because that choice can remove local information needed for answer selection. The implemented fusion sits between these extremes by making the proxy available at token resolution while preserving a connection to the original prefix representation.

A.3 Details of base-proxy decoding

CLP computes proxy logits z^{proxy} from $\widetilde{H}$ and mixes them with the frozen backbone logits z^{base}:

$$z = \alpha z^{\text{base}} + (1 - \alpha) z^{\text{proxy}}. \tag{7}$$

Here, z^{base} denotes the logits produced by the frozen backbone, z^{proxy} denotes the logits produced by the proxy path, and z denotes the final mixed logits used for decoding. The scalar α is the base-logit weight. A larger α makes decoding rely more on the frozen backbone, while a smaller α makes decoding rely more on the proxy path. In the extreme cases, $\alpha = 0$ gives a pure proxy decoder, and $\alpha = 1$ gives the original base path without proxy influence. The mixed-logit design keeps the method conservative. The base model still contributes a strong next-token prior, and the proxy path adjusts that prior toward shorter visible traces when its reconstructed states contain task-relevant information. This choice also narrows the interpretation of the results. A gain from CLP is evidence that the compact path can improve the accuracy and compression trade-off under base support, not evidence

that all reasoning has been forced through the proxy alone. The base/proxy mixing sweep explains why the base-logit weight α must be reported. In the sweep, $\alpha = 0$ gives 99.1% compression but only 6.4% accuracy, whereas $\alpha = 0.9$ gives 66.6% accuracy and 62.0% compression. This supports the interpretation that the proxy path is useful as a compact supported route, but not as a complete replacement for the frozen backbone. When $\alpha = 1$, the system returns to the base path and loses compression. The effective operating region lies between these endpoints, where the proxy changes generation enough to reduce trace length without removing too much answer-supporting signal.

A.4 Details of the training objective and hyperparameters

Only proxy-side parameters are trained. The frozen backbone is used to provide the base hidden states and base logits, while the trainable part consists of the proxy encoder, slot pooling, proxy reconstruction, local fusion, and proxy-logit modules. The overall training objective is

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\text{answer}} + \lambda_r \mathcal{L}_{\text{recon}} + \lambda_c \mathcal{L}_{\text{cap}} \\ + \lambda_s \mathcal{L}_{\text{suff}} + \lambda_d \mathcal{L}_{\text{red}} + \lambda_e \mathcal{L}_{\text{ent}}.\end{aligned} \tag{8}$$

Here, $\mathcal{L}$ is the total training loss that is minimized during training. The term $\mathcal{L}_{\text{answer}}$ is the main answer loss, implemented as proxy-logit causal language-modeling supervision on the final answer tokens or answer sequence. It is the primary supervision signal and has an implicit weight of 1. The term $\mathcal{L}_{\text{recon}}$ is a hidden-state reconstruction loss, which aligns reconstructed proxy states with frozen hidden-state targets. Together, these terms make the objective hybrid: it combines proxy-logit causal LM supervision with hidden-state reconstruction losses. It is not a distribution-level distillation objective over backbone logits. The term $\mathcal{L}_{\text{cap}}$ is the capacity loss, which constrains the scale or capacity of the latent proxy. It prevents the latent slots from becoming an unrestricted memory that simply copies the original representation. The term $\mathcal{L}_{\text{suff}}$ is the sufficiency loss, which encourages the proxy to preserve answer-supporting information. This term is important because compression should not only shorten the trace but also retain the information needed to answer correctly. The term $\mathcal{L}_{\text{red}}$ is the redundancy loss, which penalizes different latent slots when they encode overly similar information. It encourages different slots to specialize in different aspects of the input. The term $\mathcal{L}_{\text{ent}}$ is the entropy or slot-usage loss, which discourages slot collapse by encouraging a more balanced use of the available slots. Specifically, λ_r controls the strength of the reconstruction loss, λ_c controls the strength of the capacity loss, λ_s controls the strength of the sufficiency loss, λ_d controls the strength of redundancy regularization, and λ_e controls the strength of entropy-based slot-usage regularization. A larger λ_r keeps the reconstructed proxy states closer to the frozen hidden-state targets. A larger λ_c enforces a more compact proxy but may remove answer-supporting information if it is too strong. A larger λ_s places more pressure on the proxy to retain information that is sufficient for the final answer. Larger λ_d and λ_e mainly affect how the latent slots are used: λ_d reduces duplicated slot content, while λ_e discourages degenerate usage in which only one or a few slots carry most of the signal. In our experiments, we do not tune these coefficients separately for each dataset. We first run a small validation sensitivity sweep over the three main auxiliary weights and then keep one fixed setting for all benchmark runs. The final setting is

$$\lambda_r = 0.3, \qquad \lambda_s = 0.5, \qquad \lambda_c = 0.1. \tag{9}$$

This setting is selected as a balanced accuracy–compression operating point rather than as the setting that maximizes accuracy alone. The reconstruction weight $\lambda_r = 0.3$ keeps the proxy aligned with the frozen hidden-state targets. The sufficiency weight $\lambda_s = 0.5$ encourages the proxy to preserve answer-supporting information. The capacity weight $\lambda_c = 0.1$ prevents the proxy from becoming too large while avoiding an overly aggressive bottleneck. The sensitivity results in Section B.3 support this choice. Removing reconstruction loss, i.e., setting $\lambda_r = 0$, lowers mean accuracy from 66.6 to 62.1, showing that reconstruction contributes to answer preservation. Removing sufficiency pressure, i.e., setting $\lambda_s = 0$, lowers mean accuracy to 62.8, showing that answer-supporting information is lost when the proxy is not explicitly encouraged to remain sufficient. For the capacity loss, a weaker setting such as $\lambda_c = 0.03$ slightly increases accuracy to 66.8 but reduces compression to 57.5, while a stronger setting such as $\lambda_c = 0.3$ improves compression to 66.2 but reduces accuracy to 64.6. Therefore, $\lambda_c = 0.1$ is used as the balanced setting.

The redundancy and entropy coefficients, λ_d and λ_e, are treated as fixed slot-regularization coeffi-

cients rather than as the main tuned hyperparameters. They are included in the general objective because they describe the full training pressure on the latent slots, but they are not swept in the main sensitivity table. The redundancy loss $\mathcal{L}_{\text{red}}$ is computed from the pairwise similarity between the latent proxy slots. Let $R = \{r_1, \ldots, r_K\}$ denote the K latent slots produced by the bottleneck, where each slot r_i summarizes part of the compressed proxy signal. We first normalize each slot as

$$\bar{r}_i = \frac{r_i}{\|r_i\|_2 + \epsilon}, \tag{10}$$

where ϵ is a small constant used only for numerical stability. The redundancy loss is then defined as the average squared cosine similarity between different slots:

$$\mathcal{L}_{\text{red}} = \frac{1}{K(K-1)} \sum_{i=1}^{K} \sum_{j \neq i} \left(\bar{r}_i^{\top} \bar{r}_j \right)^2 . \tag{11}$$

This loss becomes large when two latent slots encode highly similar information and becomes small when different slots are close to orthogonal. Therefore, it encourages the bottleneck to distribute information across complementary slots instead of repeatedly storing the same signal in multiple slots. In all experiments, we set $\lambda_d = 0.05$ and keep it fixed across all benchmark runs. This value is deliberately smaller than the main reconstruction and sufficiency weights, because $\mathcal{L}_{\text{red}}$ is intended to improve slot diversity rather than dominate answer learning. In the implementation, the capacity target used by $\mathcal{L}_{\text{cap}}$ is fixed to 1.0. This target is not treated as a primary hyperparameter in the paper; it simply defines the reference scale for the capacity penalty. The experiments then test whether these pressures produce a practical accuracy–compression trade-off gain.

B Supplementary Diagnostic Questions

This section analyzes three supplementary questions.

B.1 Supplementary Question A: Does the method remain useful across backbones and harder tasks?

Table 4 reports the backbone robustness rows. The absolute accuracy changes with backbone strength, but the compression level remains stable across the tested models. This suggests that the proxy mechanism is not tied to only one model size or model family. The stronger Qwen2.5-3B row improves accuracy with similar compression, while smaller or different-family backbones keep the same qualitative behavior.

Table 4: Supplementary backbone robustness. Values are descriptive means over the backbone-robustness results.

Backbone	Mean Acc.	Mean Comp.	Mean Latency
Qwen2.5-0.5B-Instruct	$62.0 \pm 1.13^{***}$	60.6 ± 1.53	1.18 ± 0.057
Qwen2.5-1.5B-Instruct	66.6 ± 1.23	62.0 ± 1.67	1.69 ± 0.062
Qwen2.5-3B-Instruct	$69.0 \pm 0.89^{***}$	62.6 ± 1.50	1.98 ± 0.062
Phi-3.5-mini-instruct	$65.4 \pm 1.21^{***}$	61.5 ± 1.86	1.75 ± 0.078
Llama-3.2-1B-Instruct	$63.4 \pm 1.18^{***}$	61.0 ± 1.68	1.54 ± 0.059

Table 5 evaluates long-reasoning and multi-hop settings. In the experimental results, the strongest accuracy reference is often RawTrace, while CLP keeps nonzero compression. CLP remains competitive on GSM8K, GSM-Hard, MATH, HotpotQA, 2WikiMultiHopQA, and BBH symbolic subsets, but the gains should be read as a compression trade-off rather than as uniform accuracy dominance. The gap is more visible on harder mathematical tasks, especially MATH, which is consistent with the failure analysis: exact calculations are more sensitive to missing intermediate steps than many commonsense or retrieval-style tasks.

Supplementary finding. The proxy mechanism remains useful across tested backbones and harder external tasks as a compression mechanism, but the strongest external accuracy reference can remain RawTrace, especially on exact mathematical reasoning where losing one intermediate step can change the final answer.

B.2 Supplementary Question B: Are shorter traces still useful and cheaper to decode?

Table 6 reports rationale-audit fields. CLP has the strongest average sufficiency and partial-support rate in the rationale-quality results. The misleading rate is lower than RawTrace and Direct Rewrite, although it remains high enough that the paper should not claim fully faithful rationales. This supports a careful interpretation: CLP tends to preserve answer-supporting cues while shortening traces, but the visible trace is still an imperfect explanation of the internal computation.

Table 7 reports runtime and compute metrics. CLP reduces the mean output length from 194.4 RawTrace tokens to 76.7 tokens and lowers mean latency from 3.28s to 1.70s, but it is not the fastest method because the proxy path introduces an av-

Table 5: Supplementary external long-reasoning and multi-hop generalization. CLP is compared with the strongest non-CLP accuracy baseline for each external task.

Task	CLP Acc.	CLP Comp.	Best Acc. Baseline	Acc.	Comp.	CLP Latency
2WikiMultiHopQA	49.1±1.01	53.0±0.86	RawTrace Baseline	49.0±0.86*	0	2.18±0.080
BBH symbolic subsets	56.4±1.24	50.2±1.26	RawTrace Baseline	57.0±0.63*	0	2.34±0.034
GSM-Hard	43.7±1.09	42.1±0.53	RawTrace Baseline	45.1±1.51*	0	3.07±0.040
GSM8K	60.5±1.17	48.0±0.78	RawTrace Baseline	62.1±1.37*	0	2.52±0.087
HotpotQA	51.8±1.63	55.1±2.02	RawTrace Baseline	51.9±0.64*	0	2.02±0.069
MATH	31.9±1.14	35.2±2.24	RawTrace Baseline	33.1±0.77*	0	3.94±0.069

Table 6: Supplementary rationale-quality audit. The audit is supporting evidence and does not establish mechanistic faithfulness.

Method	Avg. Suff.	Partial	Strict	Misleading
CLP	1.88 ± 0.049	65.1 ± 1.81	27.7 ± 1.71	33.9 ± 1.71
C3oT	$1.86 \pm 0.059^{\text{n.s.}}$	62.2 ± 1.50	25.7 ± 1.57	36.3 ± 1.35
CODI	$1.87 \pm 0.052^{\text{n.s.}}$	62.3 ± 1.60	28.4 ± 1.53	36.6 ± 1.62
Direct Rewrite	$1.85 \pm 0.057^{\text{n.s.}}$	61.0 ± 1.47	25.2 ± 1.42	37.5 ± 1.42
Chain of Draft	$1.87 \pm 0.044^{\text{n.s.}}$	58.2 ± 1.64	30.8 ± 1.76	39.9 ± 1.45
RawTrace Baseline	$1.82 \pm 0.050^{*}$	45.7 ± 1.65	35.3 ± 1.72	50.2 ± 1.65

Table 7: Supplementary runtime and compute results. Values are descriptive means over the runtime results.

Method	Tokens	Lat.	Thrpt.	Mem.	Proxy OH
CLP	76.6 ± 1.37	1.69 ± 0.066	0.75 ± 0.031	7.80 ± 0.034	0.08 ± 0.014
C3oT	$66.7 \pm 1.10^{\text{n.s.}}$	1.54 ± 0.070	0.79 ± 0.035	7.10 ± 0.032	0.04 ± 0.015
CODI	$66.1 \pm 1.43^{\text{n.s.}}$	1.54 ± 0.082	0.79 ± 0.038	7.20 ± 0.034	0.04 ± 0.012
Direct Rewrite	$83.1 \pm 1.79^{\text{n.s.}}$	1.70 ± 0.077	0.78 ± 0.033	7.00 ± 0.035	0
Chain of Draft	$21.0 \pm 0.95^{**}$	0.95 ± 0.065	1.27 ± 0.032	7.00 ± 0.038	0
RawTrace Baseline	$194.6 \pm 3.20^{***}$	3.28 ± 0.061	0.37 ± 0.033	7.00 ± 0.043	0
Coconut	$84.2 \pm 1.29^{***}$	1.75 ± 0.059	0.73 ± 0.035	7.60 ± 0.034	0.04 ± 0.019
CCoT	$66.8 \pm 1.39^{***}$	1.54 ± 0.063	0.83 ± 0.037	7.40 ± 0.036	0.04 ± 0.020
CoLaR	$60.9 \pm 1.31^{***}$	1.47 ± 0.074	0.88 ± 0.024	7.50 ± 0.032	0.04 ± 0.016
SoftCoT	$78.5 \pm 1.84^{***}$	1.68 ± 0.070	0.76 ± 0.040	7.30 ± 0.034	0.04 ± 0.017
Soft Thinking	$74.6 \pm 1.19^{***}$	1.64 ± 0.075	0.79 ± 0.036	7.30 ± 0.031	0.04 ± 0.014
TokenSkip	$61.8 \pm 0.97^{***}$	1.44 ± 0.070	0.89 ± 0.031	7.10 ± 0.033	0

erage overhead of 0.08. Chain of Draft has the shortest mean output length at 21.0 tokens and the lowest mean latency at 0.95s, but Figure 2 shows that its mean accuracy is lower than CLP. This separation matters: shorter output traces reduce visible rationale cost, while wall-clock speed also depends on proxy computation, decoding configuration, and implementation efficiency. The current implementation therefore supports a compression-and-quality trade-off claim more strongly than a universal speedup claim.

B.3 Supplementary Question C: How sensitive is CLP to training losses, and where does it fail?

Table 8 reports loss and hyperparameter sensitivity. Removing reconstruction loss or sufficiency loss lowers accuracy, which aligns with the component ablation: the proxy must both preserve useful information and remain aligned with answer-supporting behavior. Excessive capacity pressure can improve compression but risks removing information needed for correctness. These rows show

Table 8: Supplementary loss and hyperparameter sensitivity. Values are descriptive means over the sensitivity results.

Setting	λ_r	λ_s	λ_c	Mean Acc.	Mean Comp.
Final setting	0.3	0.5	0.1	66.6 ± 1.23	62.0 ± 1.67
No reconstruction loss	0	0.5	0.1	$62.1 \pm 1.32^{***}$	63.7 ± 1.47
Low reconstruction loss	0.1	0.5	0.1	$65.3 \pm 1.23^{***}$	62.8 ± 1.75
High reconstruction loss	0.8	0.5	0.1	$65.7 \pm 1.23^{***}$	59.8 ± 1.86
No sufficiency loss	0.3	0	0.1	$62.7 \pm 1.38^{***}$	62.6 ± 1.45
Low capacity penalty	0.3	0.5	0.03	$66.8 \pm 1.19^{***}$	57.4 ± 2.00
High capacity penalty	0.3	0.5	0.3	$64.6 \pm 1.12^{***}$	66.2 ± 1.55
No redundancy/entropy regularization	0.3	0.5	0.1	$65.2 \pm 1.38^{***}$	61.2 ± 1.76

why the final setting is a balanced operating point rather than an arbitrary configuration.

Table 17 reports the three representative failure cases used for this analysis, and Appendix C.10 gives the original JSON records together with case-level interpretation. We keep the detailed records in the appendix because these examples are diagnostic evidence rather than a new quantitative benchmark. The cases point to three boundaries. First, CLP can preserve the high-level principle but still omit the final exact arithmetic. Second, yes-or-no tasks can fail when the compressed trace keeps the relevant cue but reverses the final polarity. Third, very short traces can name the right operation while failing to execute it. These failures are consistent with the broader result pattern and motivate future work on verifiable arithmetic and stronger sufficiency constraints.

C Detailed Experimental Results and Analysis

C.1 Baseline Results

Table 9 provides the unified detailed comparison results between CLP and the behavioral baselines. These rows show that the main claim is not driven by one averaged score. In the experimental results, RawTrace has higher accuracy on some datasets, and Chain of Draft is the strongest compression baseline on most datasets. CLP is therefore interpreted as a balanced method rather than as the best row for every single metric. For example, ARC-Challenge and BoolQ show that CLP can re-

main highly accurate while keeping a compressed trace, whereas StrategyQA and SVAMP show cases where a baseline has higher accuracy. ECQA also clarifies the intended interpretation: the raw-trace baseline can have high accuracy on a dataset, but it has zero compression, so it does not solve the shorter-trace objective.

Table 9: Unified detailed CLP-versus-baseline results.

Dataset	Method	Acc. (%)	Comp. (%)	Δ Acc. (pp)	Δ Comp. (pp)
AQUA-RAT	CLP	46.5±0.65	45.1±0.75	0	0
AQUA-RAT	C3oT	44.8±1.77$^{n.s.}$	63.9±1.02	-1.7±1.88	18.8±1.27
AQUA-RAT	CODI	47.1±1.08$^{n.s.}$	67.8±1.05	0.6±1.26	22.8±1.29
AQUA-RAT	Direct Rewrite	43.9±1.17**	24.9±0.61	-2.6±1.34	-20.1±0.97
AQUA-RAT	Chain of Draft	42.1±0.70**	87.9±2.54	-4.4±0.96	42.8±2.65
AQUA-RAT	RawTrace Baseline	46.0±1.21$^{n.s.}$	0	-0.5±1.37	-45.1±0.75
AQUA-RAT	Coconut	44.0±1.44***	43.0±0.80	-2.5±1.58	-2.1±1.10
AQUA-RAT	CCoT	45.1±1.29***	51.3±2.62	-1.4±1.44	6.2±2.73
AQUA-RAT	CoLaR	45.8±0.94***	52.8±2.62	-0.7±1.15	7.7±2.72
AQUA-RAT	SoftCoT	44.1±0.77***	44.0±0.78	-2.4±1.01	-1.1±1.09
AQUA-RAT	Soft Thinking	42.9±1.55***	45.7±2.56	-3.6±1.68	0.6±2.66
AQUA-RAT	TokenSkip	41.9±0.60***	54.3±2.34	-4.6±0.89	9.2±2.46
ARC-Challenge	CLP	73.3±0.88	72.9±1.87	0	0
ARC-Challenge	C3oT	70.0±1.68$^{n.s.}$	62.9±1.13	-3.3±1.90	-10.0±2.18
ARC-Challenge	CODI	71.9±0.96$^{n.s.}$	63.3±2.29	-1.4±1.30	-9.7±2.95
ARC-Challenge	Direct Rewrite	71.0±1.52**	70.1±0.70	-2.3±1.75	-2.9±2.00
ARC-Challenge	Chain of Draft	69.9±1.78**	90.1±0.78	-3.4±1.99	17.2±2.03
ARC-Challenge	RawTrace Baseline	72.1±1.33$^{n.s.}$	0	-1.2±1.59	-72.9±1.87
ARC-Challenge	Coconut	71.9±1.22***	67.9±0.96	-1.3±1.50	-5.0±2.10
ARC-Challenge	CCoT	72.0±1.06***	76.8±2.01	-1.3±1.38	3.9±2.75
ARC-Challenge	CoLaR	73.2±1.53***	80.3±1.63	-0.1±1.77	7.3±2.48
ARC-Challenge	SoftCoT	70.9±0.57***	72.1±1.08	-2.3±1.05	-0.9±2.16
ARC-Challenge	Soft Thinking	71.0±0.90***	73.9±1.02	-2.3±1.26	0.9±2.13
ARC-Challenge	TokenSkip	69.9±1.52***	78.8±1.37	-3.4±1.75	5.9±2.32
BoolQ	CLP	76.5±0.89	71.7±1.11	0	0
BoolQ	C3oT	76.2±1.48$^{n.s.}$	64.3±1.98	-0.3±1.72	-7.5±2.27
BoolQ	CODI	73.0±0.54$^{n.s.}$	65.0±0.50	-3.4±1.04	-6.7±1.22
BoolQ	Direct Rewrite	71.8±1.05**	72.9±0.75	-4.6±1.37	1.2±1.34
BoolQ	Chain of Draft	71.9±0.68**	89.9±0.60	-4.5±1.12	18.2±1.26
BoolQ	RawTrace Baseline	74.9±0.61$^{n.s.}$	0	-1.5±1.08	-71.7±1.11
BoolQ	Coconut	74.9±1.10***	67.8±1.33	-1.6±1.42	-3.9±1.73
BoolQ	CCoT	75.1±0.72***	77.1±2.17	-1.4±1.14	5.4±2.44
BoolQ	CoLaR	75.9±0.74***	79.1±2.32	-0.6±1.16	7.3±2.57
BoolQ	SoftCoT	74.0±0.47***	70.8±1.90	-2.5±1.01	-1.0±2.20
BoolQ	Soft Thinking	74.0±1.72***	73.3±2.03	-2.4±1.94	1.6±2.32
BoolQ	TokenSkip	73.0±1.32***	78.9±2.38	-3.5±1.60	7.1±2.62
CommonsenseQA	CLP	68.4±1.70	62.7±2.08	0	0
CommonsenseQA	C3oT	67.1±0.65$^{n.s.}$	67.9±2.44	-1.3±1.82	5.2±3.21
CommonsenseQA	CODI	69.1±0.46$^{n.s.}$	64.0±0.69	0.7±1.76	1.3±2.19
CommonsenseQA	Direct Rewrite	66.1±1.36**	70.0±1.06	-2.3±2.18	7.3±2.33
CommonsenseQA	Chain of Draft	65.9±0.79**	90.1±1.26	-2.5±1.87	27.3±2.43
CommonsenseQA	RawTrace Baseline	68.1±1.60$^{n.s.}$	0	-0.3±2.34	-62.7±2.08
CommonsenseQA	Coconut	66.9±1.28***	58.7±1.89	-1.5±2.13	-4.0±2.81
CommonsenseQA	CCoT	67.1±0.91***	68.1±1.23	-1.3±1.93	5.3±2.41
CommonsenseQA	CoLaR	67.0±0.66***	71.0±1.50	-1.4±1.83	8.3±2.56
CommonsenseQA	SoftCoT	65.9±1.15***	62.0±1.23	-2.5±2.06	-0.8±2.41
CommonsenseQA	Soft Thinking	64.8±1.77***	64.1±0.79	-3.6±2.46	1.4±2.22
CommonsenseQA	TokenSkip	65.0±0.45***	70.0±0.68	-3.4±1.76	7.3±2.19
ECQA	CLP	69.1±1.08	64.0±0.78	0	0
ECQA	C3oT	74.0±0.45$^{n.s.}$	67.2±1.93	5.0±1.17	3.2±2.08
ECQA	CODI	72.1±0.93$^{n.s.}$	63.0±1.16	3.0±1.42	-1.0±1.40
ECQA	Direct Rewrite	66.9±1.03**	68.3±2.47	-2.2±1.49	4.3±2.60
ECQA	Chain of Draft	69.9±0.70**	90.0±2.43	0.9±1.28	26.0±2.55
ECQA	RawTrace Baseline	75.0±0.46$^{n.s.}$	0	5.9±1.17	-64.0±0.78
ECQA	Coconut	66.9±1.41***	59.9±1.03	-2.2±1.77	-4.1±1.29
ECQA	CCoT	66.8±1.56***	68.9±1.97	-2.3±1.90	4.9±2.12
ECQA	CoLaR	67.9±1.04***	72.2±1.85	-1.2±1.50	8.2±2.01
ECQA	SoftCoT	65.9±0.98***	63.0±2.57	-3.2±1.45	-1.0±2.69
ECQA	Soft Thinking	64.8±1.10***	64.9±0.85	-4.2±1.54	0.9±1.16
ECQA	TokenSkip	65.1±0.84***	71.0±0.83	-4.0±1.36	7.0±1.14
MathQA	CLP	50.1±0.96	24.8±2.41	0	0
MathQA	C3oT	47.9±1.18$^{n.s.}$	74.0±0.88	-2.2±1.53	49.2±2.57
MathQA	CODI	49.9±1.04$^{n.s.}$	67.9±0.54	-0.3±1.42	43.2±2.47
MathQA	Direct Rewrite	47.1±1.62**	24.8±1.43	-3.0±1.89	0
MathQA	Chain of Draft	45.2±1.47**	88.0±1.57	-5.0±1.76	63.2±2.88
MathQA	RawTrace Baseline	49.2±1.82$^{n.s.}$	0	-1.0±2.06	-24.8±2.41
MathQA	Coconut	48.1±1.41***	23.0±0.67	-2.0±1.71	-1.7±2.50
MathQA	CCoT	49.2±1.72***	33.0±2.31	-0.9±1.98	8.3±3.34
MathQA	CoLaR	49.1±0.95***	34.9±1.26	-1.1±1.36	10.1±2.72
MathQA	SoftCoT	46.9±1.06***	25.0±0.60	-3.2±1.44	0.3±2.48
MathQA	Soft Thinking	46.9±0.69***	27.1±0.78	-3.2±1.18	2.3±2.53
MathQA	TokenSkip	46.0±1.64***	36.0±0.72	-4.2±1.90	11.2±2.52
OpenBookQA	CLP	67.8±1.80	71.0±2.36	0	0
OpenBookQA	C3oT	62.2±1.75$^{n.s.}$	67.0±0.60	-5.5±2.51	-4.0±2.44
OpenBookQA	CODI	62.0±1.13$^{n.s.}$	65.1±1.16	-5.8±2.13	-5.9±2.63
OpenBookQA	Direct Rewrite	59.9±0.87**	68.8±2.29	-7.9±2.00	-2.2±3.29
OpenBookQA	Chain of Draft	63.0±0.48**	90.1±1.06	-4.8±1.87	19.1±2.59
OpenBookQA	RawTrace Baseline	62.1±1.77$^{n.s.}$	0	-5.7±2.52	-71.0±2.36
OpenBookQA	Coconut	66.0±0.62***	66.9±0.84	-1.8±1.91	-4.1±2.51
OpenBookQA	CCoT	66.1±1.41***	76.2±1.76	-1.7±2.29	5.1±2.95
OpenBookQA	CoLaR	67.0±1.41***	79.0±1.08	-0.7±2.29	8.0±2.60
OpenBookQA	SoftCoT	64.9±1.32***	70.1±0.71	-2.8±2.23	-1.0±2.47
OpenBookQA	Soft Thinking	65.3±1.80***	72.2±2.01	-2.5±2.55	1.2±3.11
OpenBookQA	TokenSkip	64.0±0.93***	77.9±0.51	-3.7±2.03	6.9±2.42
StrategyQA	CLP	73.4±1.16	75.5±1.78	0	0
StrategyQA	C3oT	75.9±1.77$^{n.s.}$	66.2±1.88	2.5±2.12	-9.3±2.58
StrategyQA	CODI	74.9±1.07$^{n.s.}$	68.1±0.82	1.5±1.58	-7.5±1.96
StrategyQA	Direct Rewrite	74.8±1.85**	73.7±1.98	1.4±2.18	-1.8±2.66
StrategyQA	Chain of Draft	73.9±0.82**	89.8±1.63	0.5±1.42	14.2±2.41
StrategyQA	RawTrace Baseline	75.0±0.85$^{n.s.}$	0	1.6±1.44	-75.5±1.78
StrategyQA	Coconut	72.0±0.98***	70.3±2.64	-1.5±1.52	-5.2±3.18
StrategyQA	CCoT	72.0±1.57***	78.9±1.27	-1.5±1.95	3.4±2.18
StrategyQA	CoLaR	72.7±1.85***	82.0±0.72	-0.7±2.18	6.4±1.92
StrategyQA	SoftCoT	71.1±0.73***	74.3±1.59	-2.3±1.37	-1.3±2.38
StrategyQA	Soft Thinking	70.9±0.60***	76.3±2.03	-2.5±1.31	0.8±2.70
StrategyQA	TokenSkip	70.0±0.69***	81.1±1.15	-3.4±1.35	5.6±2.12
SVAMP	CLP	74.4±1.41	70.5±0.68	0	0
SVAMP	C3oT	76.0±0.98$^{n.s.}$	57.8±1.68	1.6±1.72	-12.7±1.81
SVAMP	CODI	76.8±1.38$^{n.s.}$	67.8±1.59	2.4±1.97	-2.7±1.73
SVAMP	Direct Rewrite	68.1±1.03**	60.1±2.31	-6.3±1.74	-10.4±2.41
SVAMP	Chain of Draft	72.9±0.59**	87.7±1.94	-1.5±1.53	17.2±2.06
SVAMP	RawTrace Baseline	74.9±1.16$^{n.s.}$	0	0.5±1.82	-70.5±0.68
SVAMP	Coconut	73.0±0.50***	65.2±2.05	-1.4±1.50	-5.3±2.16
SVAMP	CCoT	73.1±1.72***	74.3±2.28	-1.3±2.23	3.9±2.38
SVAMP	CoLaR	74.1±0.84***	79.1±0.66	-0.3±1.64	8.6±0.95
SVAMP	SoftCoT	71.9±1.31***	68.8±1.62	-2.5±1.92	-1.7±1.75
SVAMP	Soft Thinking	72.0±1.85***	71.4±2.34	-2.4±2.33	0.9±2.43
SVAMP	TokenSkip	71.2±1.45***	78.1±2.15	-3.2±2.02	7.6±2.26

C.2 Component-Ablation Results

Table 10 gives the detailed results behind the component ablation. The pattern is consistent with the method design. Removing the slot bottleneck weakens the compression mechanism; removing reconstruction weakens the proxy's alignment with useful answer-supporting behavior; removing local fusion makes the proxy less able to interact with token-level information; and proxy-only decoding is the most fragile setting because it removes the stabilizing base path. The ablation is therefore not merely a list of alternatives. It tests whether each architectural constraint contributes to the intended supported-compression behavior, and the detailed rows show that the full CLP configuration is the most stable compromise across tasks.

Table 10: Detailed component-ablation records.

Dataset	Variant	Acc. (%)	Comp. (%)	Δ Acc. (pp)	Δ Comp. (pp)
AQUA-RAT	Full CLP	46.5±0.65	45.1±0.75	0	0
AQUA-RAT	No bottleneck	44.9±0.82***	33.0±1.36	-1.6±1.05	-12.1±1.56
AQUA-RAT	No recon.	41.9±1.69***	47.1±2.54	-4.6±1.81	2.0±2.65
AQUA-RAT	No fusion	44.0±1.81***	46.1±2.23	-2.5±1.92	1.0±2.36
AQUA-RAT	Dense adapter	45.3±1.46***	37.2±1.55	-1.2±1.60	-7.9±1.72
AQUA-RAT	Shuffled slot	34.0±1.03***	50.6±0.92	-12.5±1.22	5.5±1.19
AQUA-RAT	Proxy-only	5.9±1.07***	98.8±1.82	-40.6±1.26	53.7±1.97
ARC-Challenge	Full CLP	73.3±0.88	72.9±1.87	0	0
ARC-Challenge	No bottleneck	72.3±1.28***	60.7±2.54	-1.0±1.55	-12.2±3.15
ARC-Challenge	No recon.	69.2±1.30***	75.0±1.62	-4.1±1.57	2.0±2.47

ARC-Challenge	No fusion	71.3±0.99***	73.7±1.95	-2.0±1.32	0.8±2.70
ARC-Challenge	Dense adapter	72.6±0.55***	64.9±1.55	-0.7±1.04	-8.0±2.43
ARC-Challenge	Shuffled slot	61.4±0.99***	78.1±1.11	-11.8±1.33	5.2±2.17
ARC-Challenge	Proxy-only	7.2±1.25***	99.1±0.71	-66.1±1.53	26.2±2.00
BoolQ	Full CLP	76.5±0.89	71.7±1.11	0	0
BoolQ	No bottleneck	75.4±1.21***	60.1±1.36	-1.0±1.51	-11.7±1.76
BoolQ	No recon.	72.4±1.78***	73.6±2.36	-4.0±1.99	1.9±2.60
BoolQ	No fusion	74.4±0.94***	72.9±1.27	-2.1±1.30	1.1±1.68
BoolQ	Dense adapter	75.8±1.29***	63.7±1.44	-0.7±1.57	-8.0±1.81
BoolQ	Shuffled slot	64.4±1.49***	77.5±2.61	-12.0±1.74	5.8±2.84
BoolQ	Proxy-only	8.6±1.76***	98.8±1.82	-67.9±1.97	27.1±2.13
CommonsenseQA	Full CLP	68.4±1.70	62.7±2.08	0	0
CommonsenseQA	No bottleneck	67.0±0.61***	50.6±0.67	-1.4±1.81	-12.2±2.18
CommonsenseQA	No recon.	64.0±1.22***	64.3±1.64	-4.4±2.10	1.6±2.65
CommonsenseQA	No fusion	65.9±1.33***	63.8±2.50	-2.5±2.16	1.0±3.25
CommonsenseQA	Dense adapter	67.3±1.51***	54.7±1.07	-1.1±2.27	-8.1±2.34
CommonsenseQA	Shuffled slot	56.2±1.21***	68.2±1.54	-12.2±2.09	5.4±2.58
CommonsenseQA	Proxy-only	7.1±0.57***	99.2±1.94	-61.4±1.79	36.4±2.84
ECQA	Full CLP	69.1±1.08	64.0±0.78	0	0
ECQA	No bottleneck	68.3±1.68***	52.0±2.35	-0.8±1.99	-12.0±2.48
ECQA	No recon.	65.0±1.51***	65.9±1.48	-4.1±1.85	1.9±1.68
ECQA	No fusion	67.3±1.76***	65.2±0.71	-1.8±2.06	1.2±1.06
ECQA	Dense adapter	68.5±0.59***	56.0±1.05	-0.6±1.23	-8.0±1.31
ECQA	Shuffled slot	57.2±1.30***	69.5±0.79	-11.9±1.69	5.5±1.11
ECQA	Proxy-only	10.5±0.89***	99.1±1.07	-58.6±1.39	35.1±1.33
MathQA	Full CLP	50.1±0.96	24.8±2.41	0	0
MathQA	No bottleneck	48.3±0.88***	12.8±1.89	-1.8±1.31	-12.0±3.06
MathQA	No recon.	45.2±1.79***	27.1±2.54	-5.0±2.03	2.3±3.51
MathQA	No fusion	47.0±1.79***	26.1±0.65	-3.2±2.03	1.3±2.50
MathQA	Dense adapter	48.7±0.96***	17.1±0.94	-1.5±1.36	-7.6±2.59
MathQA	Shuffled slot	37.3±1.31***	30.8±1.86	-12.8±1.63	6.0±3.05
MathQA	Proxy-only	4.3±1.44***	98.3±2.40	-45.8±1.74	73.5±3.40
OpenBookQA	Full CLP	67.8±1.80	71.0±2.36	0	0
OpenBookQA	No bottleneck	66.7±1.23***	59.1±1.85	-1.1±2.18	-11.9±3.00
OpenBookQA	No recon.	63.6±1.31***	73.4±0.79	-4.2±2.23	2.4±2.49
OpenBookQA	No fusion	65.8±0.48***	72.3±1.58	-2.0±1.87	1.3±2.84
OpenBookQA	Dense adapter	67.2±0.77***	63.4±2.35	-0.6±1.96	-7.6±3.34
OpenBookQA	Shuffled slot	56.0±0.51***	76.8±1.02	-11.8±1.87	5.8±2.57
OpenBookQA	Proxy-only	6.6±1.38***	99.0±0.51	-61.2±2.27	28.0±2.42
StrategyQA	Full CLP	73.4±1.16	75.5±1.78	0	0
StrategyQA	No bottleneck	72.2±0.88***	63.3±1.39	-1.2±1.45	-12.2±2.25
StrategyQA	No recon.	69.3±1.25***	77.2±2.52	-4.2±1.70	1.7±3.09
StrategyQA	No fusion	71.3±1.25***	76.3±0.79	-2.1±1.70	0.8±1.94
StrategyQA	Dense adapter	72.6±1.62***	67.3±2.05	-0.8±1.99	-8.3±2.71
StrategyQA	Shuffled slot	61.6±1.50***	80.4±2.50	-11.8±1.89	4.9±3.06
StrategyQA	Proxy-only	9.1±1.01***	99.3±1.26	-64.3±1.53	23.7±2.18
SVAMP	Full CLP	74.4±1.41	70.5±0.68	0	0
SVAMP	No bottleneck	72.8±1.74***	58.5±1.45	-1.6±2.24	-11.9±1.60
SVAMP	No recon.	69.6±0.78***	72.7±1.88	-4.8±1.61	2.2±2.00
SVAMP	No fusion	71.8±0.59***	71.8±2.10	-2.6±1.53	1.3±2.21
SVAMP	Dense adapter	73.5±1.71***	62.3±1.61	-0.9±2.21	-8.1±1.74
SVAMP	Shuffled slot	62.0±0.70***	76.2±1.80	-12.4±1.57	5.7±1.93
SVAMP	Proxy-only	5.1±1.68***	99.2±2.36	-69.3±2.19	28.7±2.46

C.3 Slot-Capacity Results

Table 11 reports how performance changes as the number of latent slots varies. The results explain why the method should not use either an extremely small or an unbounded proxy. Too few slots can compress aggressively but risk discarding answer-supporting intermediate information. More slots usually make the proxy more expressive, but the gain is limited because the proxy then begins to behave less like a bottleneck and more like an auxiliary memory. The selected operating region therefore reflects a capacity trade-off: enough slots to preserve the main answer-supporting signal, but few enough to keep the representation compact and interpretable as a constrained proxy.

Table 11: Detailed slot-capacity records.

Dataset	Capacity Variant	Acc. (%)	Comp. (%)	Output Tokens
AQUA-RAT	slots1_dim16	39.3±1.77***	52.2±1.47	100.4±2.57
AQUA-RAT	slots1_dim64	42.3±1.83***	49.9±1.56	105.0±1.67
AQUA-RAT	slots2_dim64	46.0±1.62***	46.5±2.50	112.1±2.83
AQUA-RAT	slots4_dim32	46.1±0.92	46.8±1.22	111.8±2.35
AQUA-RAT	slots4_dim64	45.7±1.45***	46.4±2.56	112.3±2.48
AQUA-RAT	slots8_dim64	45.6±1.50***	46.7±0.85	111.6±2.45
AQUA-RAT	slots16_dim64	45.2±1.48***	45.7±1.24	114.0±0.84
ARC-Challenge	slots1_dim16	67.0±1.19***	79.9±2.34	34.4±0.45
ARC-Challenge	slots1_dim64	69.9±0.59***	78.0±1.84	37.6±1.14
ARC-Challenge	slots2_dim64	73.0±0.58***	74.5±1.75	43.3±0.74
ARC-Challenge	slots4_dim32	73.0±1.44	74.4±1.37	43.5±0.55
ARC-Challenge	slots4_dim64	72.7±1.62***	74.5±2.58	44.0±1.20
ARC-Challenge	slots8_dim64	72.6±1.17***	74.3±2.39	43.4±1.14
ARC-Challenge	slots16_dim64	72.3±0.95***	73.4±1.20	45.1±0.62
BoolQ	slots1_dim16	70.2±0.81***	78.9±2.50	33.8±0.69
BoolQ	slots1_dim64	72.9±1.75***	77.0±0.65	37.0±0.80
BoolQ	slots2_dim64	75.9±0.69***	73.9±2.03	42.1±0.51
BoolQ	slots4_dim32	76.2±1.04	73.4±0.86	42.5±0.54
BoolQ	slots4_dim64	75.6±1.74***	73.0±2.56	42.6±0.52
BoolQ	slots8_dim64	75.6±0.89***	73.7±1.85	42.2±0.81
BoolQ	slots16_dim64	75.3±1.37***	72.4±2.41	43.9±0.47
CommonsenseQA	slots1_dim16	61.7±1.24***	69.2±1.79	56.4±0.52
CommonsenseQA	slots1_dim64	64.5±1.00***	67.4±0.51	60.3±1.22
CommonsenseQA	slots2_dim64	68.0±1.50***	64.2±1.36	65.9±0.74
CommonsenseQA	slots4_dim32	67.8±0.58	64.2±1.00	66.4±0.58
CommonsenseQA	slots4_dim64	67.4±1.49***	63.9±1.56	66.7±0.80
CommonsenseQA	slots8_dim64	67.4±1.67***	64.2±1.92	66.1±1.51
CommonsenseQA	slots16_dim64	67.3±1.10***	63.5±2.40	67.9±1.73
ECQA	slots1_dim16	62.6±0.81***	71.3±1.69	49.1±0.75
ECQA	slots1_dim64	65.6±0.82***	69.2±1.74	52.4±0.49
ECQA	slots2_dim64	68.6±1.74***	65.8±2.34	58.1±1.17
ECQA	slots4_dim32	68.9±1.44	65.3±2.50	58.2±0.71
ECQA	slots4_dim64	67.9±1.83***	65.4±1.01	58.7±0.65
ECQA	slots8_dim64	68.2±1.11***	65.9±0.80	57.9±0.63
ECQA	slots16_dim64	67.8±0.61***	64.8±0.63	59.8±0.58
MathQA	slots1_dim16	43.0±1.65***	32.3±2.22	166.9±2.03
MathQA	slots1_dim64	45.8±0.78***	29.7±2.25	171.8±2.69
MathQA	slots2_dim64	49.6±0.93***	26.6±1.26	179.2±2.07
MathQA	slots4_dim32	49.7±0.88	26.5±0.75	180.3±4.14
MathQA	slots4_dim64	49.1±1.18***	26.2±1.65	180.2±1.51
MathQA	slots8_dim64	49.1±1.28***	26.8±0.91	179.4±3.54
MathQA	slots16_dim64	48.8±1.54***	25.4±1.85	182.5±4.12
OpenBookQA	slots1_dim16	61.7±0.73***	78.2±1.97	38.1±0.81
OpenBookQA	slots1_dim64	64.6±0.89***	76.4±0.69	41.6±1.02
OpenBookQA	slots2_dim64	67.4±1.27***	72.9±2.63	47.1±0.61
OpenBookQA	slots4_dim32	67.5±1.43	73.1±1.69	47.3±1.14
OpenBookQA	slots4_dim64	67.1±0.97***	72.5±2.18	47.8±1.20
OpenBookQA	slots8_dim64	67.2±0.47***	73.2±1.95	47.0±0.58
OpenBookQA	slots16_dim64	66.7±1.74***	71.9±0.84	49.2±1.08
StrategyQA	slots1_dim16	67.2±0.52***	82.2±1.40	33.6±1.22
StrategyQA	slots1_dim64	69.8±1.61***	80.4±0.97	37.5±0.47
StrategyQA	slots2_dim64	72.9±1.65***	77.1±1.77	43.5±0.67
StrategyQA	slots4_dim32	73.0±1.53	77.1±2.36	43.9±1.03
StrategyQA	slots4_dim64	72.5±0.50***	76.6±1.16	44.4±1.15
StrategyQA	slots8_dim64	72.6±0.86***	77.5±2.24	43.4±1.22
StrategyQA	slots16_dim64	72.4±1.27***	76.3±2.45	45.5±0.76
SVAMP	slots1_dim16	67.3±0.72***	77.3±1.93	55.0±1.13
SVAMP	slots1_dim64	70.0±0.88***	75.5±0.55	60.1±0.59
SVAMP	slots2_dim64	74.0±1.67***	72.3±2.26	67.8±0.89
SVAMP	slots4_dim32	73.7±1.53	72.1±1.98	68.3±0.63
SVAMP	slots4_dim64	73.1±1.65***	71.6±2.29	68.6±1.54
SVAMP	slots8_dim64	73.5±1.64***	72.0±2.38	67.9±1.24
SVAMP	slots16_dim64	73.1±0.55***	71.2±0.77	70.6±0.71

C.4 Base/Proxy Mixing Results

Table 12 provides the detailed evidence for the base/proxy logit-mixing choice. The endpoint rows are especially important. A proxy-only setting gives very high compression but loses too much accuracy, showing that the current proxy is not a complete standalone reasoning model. A base-only setting recovers the original decoding behavior and therefore loses the compression benefit. The useful region is between these endpoints, where the frozen backbone supplies a reliable language and answer prior while the compact proxy steers generation toward shorter traces. This supports the our claim that CLP is a supported proxy path, not a replacement for the backbone.

Table 12: Detailed base/proxy mixing records.

Dataset	Mixing Setting	Acc. (%)	Comp. (%)	Output Tokens
AQUA-RAT	alpha=0	5.7±1.64***	98.7±2.58	2.0±0.65
AQUA-RAT	alpha=0.75	43.9±0.80***	48.9±1.92	106.7±2.77
AQUA-RAT	alpha=0.9	46.5±0.65	45.1±0.75	115.3±1.95
AQUA-RAT	alpha=1	32.0±0.52**	0	210.4±3.36
ARC-Challenge	alpha=0	7.1±0.60***	99.0±1.07	1.9±0.57
ARC-Challenge	alpha=0.75	71.0±1.81***	77.0±0.90	39.3±0.90
ARC-Challenge	alpha=0.9	73.3±0.88	72.9±1.87	46.2±1.05
ARC-Challenge	alpha=1	49.2±1.36**	0	169.9±2.45
BoolQ	alpha=0	8.6±0.60***	99.4±2.04	1.9±1.00
BoolQ	alpha=0.75	74.3±1.10***	75.9±1.44	38.2±1.24
BoolQ	alpha=0.9	76.5±0.89	71.7±1.11	45.0±1.24
BoolQ	alpha=1	52.2±0.62**	0	160.1±1.38
CommonsenseQA	alpha=0	6.8±1.38***	99.0±2.62	1.9±0.94
CommonsenseQA	alpha=0.75	65.8±1.30***	66.5±0.74	61.9±1.02
CommonsenseQA	alpha=0.9	68.4±1.70	62.7±2.08	69.5±1.63
CommonsenseQA	alpha=1	52.5±1.41**	0	185.0±1.25
ECQA	alpha=0	10.2±1.30***	99.0±0.99	2.0±0.53
ECQA	alpha=0.75	66.4±1.66***	68.0±1.68	54.1±0.49
ECQA	alpha=0.9	69.1±1.08	64.0±0.78	60.9±0.73
ECQA	alpha=1	73.3±0.57**	0	169.7±3.58
MathQA	alpha=0	4.2±0.86***	98.7±1.03	1.9±0.74
MathQA	alpha=0.75	47.3±1.74***	29.0±2.25	173.6±2.44
MathQA	alpha=0.9	50.1±0.96	24.8±2.41	183.5±2.14
MathQA	alpha=1	31.1±1.80**	0	245.1±2.51
OpenBookQA	alpha=0	6.5±0.52***	99.0±0.89	2.1±0.57
OpenBookQA	alpha=0.75	65.5±0.83***	75.2±2.14	43.0±1.07
OpenBookQA	alpha=0.9	67.8±1.80	71.0±2.36	50.1±0.84
OpenBookQA	alpha=1	49.2±1.06**	0	175.7±4.55
StrategyQA	alpha=0	9.2±0.50***	99.1±0.57	2.0±0.94
StrategyQA	alpha=0.75	71.2±0.91***	79.1±2.35	39.0±0.67
StrategyQA	alpha=0.9	73.4±1.16	75.5±1.78	46.9±0.48
StrategyQA	alpha=1	47.4±0.71**	0	190.3±2.81
SVAMP	alpha=0	5.1±1.31***	99.1±0.64	2.1±0.93
SVAMP	alpha=0.75	71.9±1.35***	74.7±1.12	62.2±1.30
SVAMP	alpha=0.9	74.4±1.41	70.5±0.68	72.2±1.13
SVAMP	alpha=1	69.9±1.14**	0	244.9±2.71

C.5 Backbone-Robustness Results

Table 13 tests whether the observed behavior depends on one specific model instance. The rows compare smaller and larger Qwen variants as well as alternative instruction-tuned backbones. The absolute accuracy naturally changes with model strength, but the relevant pattern is whether a compact proxy can still reduce trace length without collapsing answer quality. The appendix rows show that the effect is not confined to a single backbone, while also making clear that the method is not backbone-independent in a theoretical sense. Stronger base models provide better support for the proxy, and weaker models leave less answer-supporting signal for the proxy to preserve.

Table 13: Detailed backbone-robustness records.

Dataset	Backbone	Acc. (%)	Comp. (%)	Latency (s)
AQUA-RAT	Qwen2.5-0.5B-Instruct	41.8±1.24***	43.5±1.07	1.45±0.026
AQUA-RAT	Qwen2.5-1.5B-Instruct	46.5±0.65	45.1±0.75	2.16±0.065
AQUA-RAT	Qwen2.5-3B-Instruct	48.8±0.59***	45.7±0.93	2.52±0.094
AQUA-RAT	Phi-3.5-mini-instruct	45.2±0.72***	44.5±2.47	2.23±0.096
AQUA-RAT	Llama-3.2-1B-Instruct	43.1±0.96***	44.0±0.77	1.97±0.069
ARC-Challenge	Qwen2.5-0.5B-Instruct	68.7±1.82***	71.2±0.64	0.96±0.070
ARC-Challenge	Qwen2.5-1.5B-Instruct	73.3±0.88	72.9±1.87	1.31±0.057
ARC-Challenge	Qwen2.5-3B-Instruct	75.9±1.03***	73.5±1.10	1.55±0.049
ARC-Challenge	Phi-3.5-mini-instruct	72.1±0.79***	72.1±1.83	1.38±0.056
ARC-Challenge	Llama-3.2-1B-Instruct	70.1±0.52***	72.0±1.63	1.21±0.089
BoolQ	Qwen2.5-0.5B-Instruct	71.8±0.96***	70.7±2.34	0.94±0.097
BoolQ	Qwen2.5-1.5B-Instruct	76.5±0.89	71.7±1.11	1.32±0.099
BoolQ	Qwen2.5-3B-Instruct	78.8±0.63***	72.6±0.67	1.55±0.056
BoolQ	Phi-3.5-mini-instruct	75.3±1.48***	71.5±1.75	1.38±0.054
BoolQ	Llama-3.2-1B-Instruct	73.4±1.84***	70.8±1.74	1.19±0.044
CommonsenseQA	Qwen2.5-0.5B-Instruct	63.7±0.74***	60.8±1.43	1.13±0.051
CommonsenseQA	Qwen2.5-1.5B-Instruct	68.4±1.70	62.7±2.08	1.60±0.060
CommonsenseQA	Qwen2.5-3B-Instruct	70.6±0.77***	62.9±2.38	1.88±0.050
CommonsenseQA	Phi-3.5-mini-instruct	66.9±1.11***	62.0±1.73	1.64±0.107
CommonsenseQA	Llama-3.2-1B-Instruct	65.0±0.79***	61.8±2.23	1.46±0.020
ECQA	Qwen2.5-0.5B-Instruct	64.5±1.58***	62.6±0.91	1.08±0.063
ECQA	Qwen2.5-1.5B-Instruct	69.1±1.08	64.0±0.78	1.49±0.060
ECQA	Qwen2.5-3B-Instruct	71.4±1.13***	64.7±0.93	1.75±0.077
ECQA	Phi-3.5-mini-instruct	67.6±1.48***	63.5±1.73	1.56±0.072
ECQA	Llama-3.2-1B-Instruct	65.8±1.27***	63.0±0.99	1.37±0.019
MathQA	Qwen2.5-0.5B-Instruct	45.6±0.99***	23.5±0.73	1.92±0.066
MathQA	Qwen2.5-1.5B-Instruct	50.1±0.96	24.8±2.41	2.97±0.056
MathQA	Qwen2.5-3B-Instruct	52.5±0.83***	25.6±1.44	3.47±0.055
MathQA	Phi-3.5-mini-instruct	48.8±0.52***	24.5±1.75	3.04±0.077
MathQA	Llama-3.2-1B-Instruct	46.8±1.53***	23.9±2.43	2.73±0.027
OpenBookQA	Qwen2.5-0.5B-Instruct	63.6±0.55***	69.8±0.85	0.99±0.015
OpenBookQA	Qwen2.5-1.5B-Instruct	67.8±1.80	71.0±2.36	1.37±0.034
OpenBookQA	Qwen2.5-3B-Instruct	70.5±0.66***	72.0±2.10	1.62±0.060
OpenBookQA	Phi-3.5-mini-instruct	66.8±0.69***	71.0±1.20	1.43±0.046
OpenBookQA	Llama-3.2-1B-Instruct	64.9±1.50***	70.0±1.88	1.25±0.095
StrategyQA	Qwen2.5-0.5B-Instruct	68.9±0.76***	74.1±2.58	0.97±0.026
StrategyQA	Qwen2.5-1.5B-Instruct	73.4±1.16	75.5±1.78	1.33±0.061
StrategyQA	Qwen2.5-3B-Instruct	75.7±1.33***	76.0±0.92	1.56±0.056
StrategyQA	Phi-3.5-mini-instruct	72.4±1.68***	74.9±1.11	1.39±0.109
StrategyQA	Llama-3.2-1B-Instruct	70.3±0.62***	74.3±1.22	1.22±0.039
SVAMP	Qwen2.5-0.5B-Instruct	69.8±0.86***	69.1±1.82	1.16±0.052
SVAMP	Qwen2.5-1.5B-Instruct	74.4±1.41	70.5±0.68	1.64±0.037
SVAMP	Qwen2.5-3B-Instruct	76.5±0.78***	70.8±1.99	1.92±0.048
SVAMP	Phi-3.5-mini-instruct	73.0±1.72***	69.7±2.65	1.69±0.059
SVAMP	Llama-3.2-1B-Instruct	70.9±0.87***	69.5±1.47	1.49±0.069

C.6 Rationale-Quality Audit Results

Table 14 separates answer accuracy from rationale diagnostics. This distinction is necessary because a short trace can be useful, partially useful, or misleading. The rows show that CLP often keeps a meaningful amount of rationale evidence while reducing trace length, but they also show why the paper avoids claiming guaranteed faithfulness. Some compressed traces cover the important cue but omit the decisive calculation or polarity mapping. These results therefore complement the quantitative tables: they indicate when compression remains diagnostically useful and where additional verification constraints are needed.

Table 14: Detailed rationale-quality-audit records.

Data	Method	Acc.	Comp.	Suff.	Partial	Strict	Mis.
AQUA	CLP	46.5±0.7	45.1±0.8	1.8±0.024	58.1±0.7	25.2±1.3	40.2±1.4
AQUA	C3oT	44.8±1.8$^{n.s.}$	63.9±1.0	1.8±0.1$^{n.s.}$	55.1±0.9	22.9±1.3	42.4±1.5
AQUA	CODI	47.1±1.1$^{n.s.}$	67.8±1.1	1.8±0.1$^{n.s.}$	55.3±0.9	25.9±2.2	42.7±1.7
AQUA	DRewrite	43.9±1.2**	24.9±0.6	1.7±0.046$^{n.s.}$	53.9±1.4	22.4±1.2	43.6±1.6
AQUA	CoD	42.1±0.7**	87.9±2.5	1.8±0.038$^{n.s.}$	51.3±0.9	28.4±2.3	46.0±0.7
AQUA	RawTrace	46.0±1.2$^{n.s.}$	0.0	1.7±0.020*	38.9±2.3	32.8±2.2	56.3±1.1
ARC-C	CLP	73.3±0.9	72.9±1.9	1.9±0.1	68.2±2.4	28.8±1.5	31.1±1.3
ARC-C	C3oT	70.0±1.7$^{n.s.}$	62.9±1.1	1.9±0.012$^{n.s.}$	65.1±1.7	26.9±1.7	33.3±1.3
ARC-C	CODI	71.9±1.0$^{n.s.}$	63.3±2.3	1.9±0.017$^{n.s.}$	65.2±1.4	29.8±0.6	33.8±1.5
ARC-C	DRewrite	71.0±1.5**	70.1±0.7	1.9±0.1$^{n.s.}$	64.1±1.7	26.7±1.4	34.7±0.6
ARC-C	CoD	69.9±1.8**	90.1±0.8	1.9±0.044$^{n.s.}$	61.5±2.0	32.0±1.2	37.1±0.8
ARC-C	RawTrace	72.1±1.3$^{n.s.}$	0.0	1.9±0.039*	48.6±0.7	36.6±1.7	47.5±2.1
BoolQ	CLP	76.5±0.9	71.7±1.1	1.9±0.039	69.3±2.4	29.1±0.6	31.0±0.7
BoolQ	C3oT	76.2±1.5$^{n.s.}$	64.3±2.0	1.9±0.1$^{n.s.}$	66.2±1.9	27.0±1.7	33.4±2.0
BoolQ	CODI	73.0±0.5$^{n.s.}$	65.0±0.5	1.9±0.031$^{n.s.}$	66.5±2.1	29.5±1.7	33.6±1.2
BoolQ	DRewrite	71.8±1.1**	72.9±0.8	1.9±0.046$^{n.s.}$	64.9±1.1	26.6±2.0	34.4±1.2
BoolQ	CoD	71.9±0.7**	89.9±0.6	1.9±0.021$^{n.s.}$	61.9±2.4	32.2±2.4	36.9±1.5
BoolQ	RawTrace	74.9±0.6$^{n.s.}$	0.0	1.9±0.1*	49.7±1.9	36.5±2.1	47.3±2.3
CSQA	CLP	68.4±1.7	62.7±2.1	1.9±0.1	69.3±2.4	28.8±2.1	30.1±1.9
CSQA	C3oT	67.1±0.7$^{n.s.}$	67.9±2.4	1.9±0.038$^{n.s.}$	66.3±1.4	27.3±1.5	32.3±0.8
CSQA	CODI	69.1±0.5$^{n.s.}$	64.0±0.7	1.9±0.024$^{n.s.}$	66.1±0.6	29.7±1.7	32.9±1.1
CSQA	DRewrite	66.1±1.4**	70.0±1.1	1.9±0.1$^{n.s.}$	65.0±0.6	26.4±1.7	33.5±1.6
CSQA	CoD	65.9±0.8**	90.1±1.3	1.9±0.1$^{n.s.}$	62.3±0.6	32.2±1.0	35.8±2.1
CSQA	RawTrace	68.1±1.6$^{n.s.}$	0.0	1.9±0.048*	49.8±1.8	36.7±1.0	46.3±0.9
ECQA	CLP	69.1±1.1	64.0±0.8	1.9±0.048	68.3±2.4	28.9±2.3	31.1±2.3
ECQA	C3oT	74.0±0.5$^{n.s.}$	67.2±1.9	1.9±0.040$^{n.s.}$	65.5±2.0	26.9±1.3	33.3±1.0
ECQA	CODI	72.1±0.9$^{n.s.}$	63.0±1.2	1.9±0.1$^{n.s.}$	65.5±1.9	29.5±2.0	33.6±2.2
ECQA	DRewrite	66.9±1.0**	68.3±2.5	1.9±0.019$^{n.s.}$	64.3±1.8	26.3±1.6	34.5±1.9
ECQA	CoD	69.9±0.7**	90.0±2.4	1.9±0.037$^{n.s.}$	61.1±0.9	31.8±2.3	36.9±1.8
ECQA	RawTrace	75.0±0.5$^{n.s.}$	0.0	1.8±0.015*	48.6±1.2	36.5±2.1	47.4±0.8

Data	Method	Acc.	Comp.	Suff.	Partial	Strict	Mis.
MathQA	CLP	50.1±1.0	24.8±2.4	1.8±0.1	55.9±1.6	24.0±1.5	41.0±2.1
MathQA	C3oT	47.9±1.2$^{n.s.}$	74.0±0.9	1.7±0.1$^{n.s.}$	53.1±0.6	22.0±2.1	43.4±0.6
MathQA	CODI	49.9±1.0$^{n.s.}$	67.9±0.5	1.8±0.1$^{n.s.}$	53.4±2.1	24.7±0.6	44.0±1.5
MathQA	DRewrite	47.1±1.6**	24.8±1.4	1.7±0.1$^{n.s.}$	52.2±1.5	21.7±1.9	44.8±2.0
MathQA	CoD	45.2±1.5**	88.0±1.6	1.8±0.026$^{n.s.}$	49.2±1.0	27.4±1.8	47.3±1.7
MathQA	RawTrace	49.2±1.8$^{n.s.}$	0.0	1.7±0.1*	36.8±1.2	31.6±1.1	57.1±1.5
OBQA	CLP	68.1±0.8	71.4±1.2	1.9±0.039	68.1±1.0	29.3±2.2	31.2±1.3
OBQA	C3oT	62.2±1.5	67.0±2.1	1.9±0.021$^{n.s.}$	65.2±0.8	27.0±0.9	33.5±1.2
OBQA	CODI	62.0±0.7	65.0±1.4	1.9±0.1$^{n.s.}$	65.2±1.1	29.8±1.6	33.6±2.1
OBQA	DRewrite	60.1±1.4	69.0±0.8	1.9±0.1$^{n.s.}$	63.9±1.3	26.6±0.7	34.5±1.3
OBQA	CoD	63.2±1.4	89.9±1.6	1.9±0.044$^{n.s.}$	61.2±1.8	32.1±0.7	37.1±0.7
OBQA	RawTrace	61.9±0.6	0.0	1.9±0.1*	48.6±2.4	36.6±2.4	47.5±2.3
SQA	CLP	73.3±0.9	75.5±1.0	1.9±0.048	66.9±1.4	29.2±1.8	32.0±1.7
SQA	C3oT	76.0±0.7	66.0±2.0	1.9±0.1$^{n.s.}$	64.3±0.6	27.1±1.6	34.5±1.2
SQA	CODI	75.2±1.4	67.9±2.4	1.9±0.1$^{n.s.}$	64.2±2.2	29.7±1.3	34.7±1.7
SQA	DRewrite	75.2±1.5	73.8±1.3	1.9±0.1$^{n.s.}$	63.0±0.6	26.5±0.6	35.6±1.1
SQA	CoD	74.1±0.8	90.0±1.3	1.9±0.036$^{n.s.}$	60.0±2.2	32.0±1.7	37.9±1.5
SQA	RawTrace	75.1±1.7	0.0	1.8±0.1*	47.5±1.5	36.5±1.1	48.2±0.8
SVAMP	CLP	74.4±1.4	70.5±0.7	1.9±0.018	62.1±0.6	26.0±1.5	37.9±2.1
SVAMP	C3oT	76.0±1.0	57.8±1.7	1.8±0.1$^{n.s.}$	59.4±2.4	24.0±1.6	40.4±2.0
SVAMP	CODI	76.8±1.4	67.8±1.6	1.8±0.042$^{n.s.}$	59.1±1.4	26.8±1.3	40.9±1.5
SVAMP	DRewrite	68.1±1.0	60.1±2.3	1.8±0.027$^{n.s.}$	57.9±2.4	23.5±0.7	41.7±1.3
SVAMP	CoD	72.9±0.6	87.7±1.9	1.8±0.1$^{n.s.}$	55.1±1.9	28.9±1.5	44.0±1.5
SVAMP	RawTrace	74.9±1.2	0.0	1.8±0.021*	42.6±0.9	33.5±0.9	54.3±2.0

Abbreviations: AQUA = AQUA-RAT; ARC-C = ARC-Challenge; CSQA = CommonsenseQA; OBQA = OpenBookQA; SQA = StrategyQA; DRewrite = Direct Rewrite; CoD = Chain of Draft.

C.7 Runtime Results

Table 15 reports output tokens, latency, throughput, memory, and proxy overhead. The rows show a consistent runtime pattern: RawTrace keeps the full output and therefore has the largest mean token count and latency, whereas high-compression methods such as Chain of Draft have the shortest outputs and fastest decoding. CLP sits between these extremes. It uses 76.7 mean output tokens and 1.70s mean latency, compared with 194.4 tokens and 3.28s for RawTrace and 21.0 tokens and 0.95s for Chain of Draft. This means CLP should not be described as the fastest method. Its advantage is instead a balanced setting: it cuts a large portion of visible trace tokens while preserving higher mean accuracy than the most aggressively compressed baselines. The table therefore supports a practical cost-quality trade-off claim rather than a universal runtime dominance claim.

Table 15: Detailed runtime results.

Data	Method	Tok.	Lat.	Thru.	Mem.	OH
AQUA	CLP	115.3±1.95	2.16±0.065	0.55±0.036	7.80±0.045	0.08±0.010
AQUA	C3oT	75.7±0.60$^{n.s.}$	1.64±0.089	0.73±0.026	7.10±0.050	0.04±0.009
AQUA	CODI	67.4±1.69$^{n.s.}$	1.55±0.034	0.77±0.053	7.20±0.020	0.04±0.009
AQUA	DRewrite	158.0±4.04$^{n.s.}$	2.60±0.091	0.46±0.047	6.99±0.039	0
AQUA	CoD	25.3±0.82**	1.00±0.020	1.20±0.013	7.01±0.048	0
AQUA	RawTrace	210.1±1.27***	3.47±0.023	0.35±0.029	7.00±0.025	0
AQUA	Coconut	119.7±0.85***	2.18±0.040	0.55±0.031	7.60±0.040	0.04±0.019
AQUA	CCoT	102.8±0.89***	1.97±0.025	0.61±0.040	7.40±0.044	0.04±0.022
AQUA	CoLaR	98.6±1.54***	1.92±0.039	0.62±0.027	7.50±0.030	0.04±0.020
AQUA	SoftCoT	117.5±1.39***	2.13±0.105	0.57±0.041	7.30±0.010	0.04±0.017
AQUA	SoftThink	113.2±1.77***	2.10±0.110	0.57±0.012	7.30±0.015	0.04±0.007
AQUA	TokenSkip	96.6±1.46***	1.86±0.018	0.65±0.043	7.10±0.026	0
ARC-C	CLP	46.2±1.05	1.31±0.057	0.90±0.023	7.80±0.033	0.08±0.013
ARC-C	C3oT	62.9±0.93$^{n.s.}$	1.50±0.043	0.80±0.053	7.10±0.018	0.04±0.007
ARC-C	CODI	62.9±0.49$^{n.s.}$	1.49±0.097	0.80±0.026	7.20±0.050	0.04±0.014
ARC-C	DRewrite	51.0±1.21$^{n.s.}$	1.31±0.037	0.91±0.027	7.00±0.042	0
ARC-C	CoD	17.0±0.52**	0.89±0.054	1.33±0.051	7.01±0.050	0
ARC-C	RawTrace	170.1±2.71***	2.99±0.017	0.40±0.018	7.00±0.049	0
ARC-C	Coconut	54.4±0.55***	1.39±0.043	0.86±0.054	7.60±0.041	0.04±0.013
ARC-C	CCoT	39.2±0.47***	1.21±0.033	0.99±0.033	7.40±0.035	0.04±0.016
ARC-C	CoLaR	33.9±0.49***	1.15±0.104	1.05±0.036	7.51±0.053	0.04±0.011
ARC-C	SoftCoT	47.7±0.60***	1.31±0.105	0.92±0.035	7.30±0.019	0.04±0.007
ARC-C	SoftThink	44.3±0.63***	1.28±0.109	0.95±0.040	7.30±0.033	0.04±0.008
ARC-C	TokenSkip	35.6±0.64***	1.14±0.091	1.06±0.019	7.10±0.032	0
BoolQ	CLP	45.0±1.24	1.32±0.099	0.91±0.008	7.80±0.020	0.08±0.017
BoolQ	C3oT	57.7±0.52$^{n.s.}$	1.43±0.058	0.84±0.026	7.10±0.036	0.04±0.018
BoolQ	CODI	56.0±1.13$^{n.s.}$	1.42±0.079	0.85±0.016	7.20±0.011	0.04±0.008
BoolQ	DRewrite	43.3±0.72$^{n.s.}$	1.21±0.099	0.98±0.016	7.00±0.028	0
BoolQ	CoD	16.0±0.72**	0.88±0.062	1.35±0.020	7.00±0.013	0
BoolQ	RawTrace	160.1±1.03***	2.86±0.077	0.42±0.032	7.00±0.055	0
BoolQ	Coconut	51.2±0.98***	1.35±0.049	0.90±0.046	7.60±0.014	0.04±0.018
BoolQ	CCoT	36.9±0.47***	1.18±0.038	1.02±0.012	7.41±0.047	0.04±0.023
BoolQ	CoLaR	33.5±0.83***	1.14±0.041	1.05±0.016	7.50±0.029	0.04±0.013
BoolQ	SoftCoT	46.3±0.93***	1.31±0.077	0.93±0.054	7.30±0.041	0.04±0.011
BoolQ	SoftThink	43.3±0.86***	1.25±0.083	0.95±0.012	7.30±0.014	0.04±0.011
BoolQ	TokenSkip	33.5±0.73***	1.10±0.089	1.09±0.021	7.10±0.014	0
CSQA	CLP	69.5±1.63	1.60±0.060	0.73±0.047	7.80±0.032	0.08±0.008
CSQA	C3oT	59.2±0.57$^{n.s.}$	1.45±0.039	0.83±0.028	7.10±0.016	0.04±0.009
CSQA	CODI	66.6±1.35$^{n.s.}$	1.53±0.107	0.78±0.038	7.19±0.047	0.04±0.009
CSQA	DRewrite	55.5±0.50$^{n.s.}$	1.37±0.078	0.88±0.021	7.00±0.033	0
CSQA	CoD	18.4±1.04**	0.91±0.060	1.30±0.018	7.00±0.037	0
CSQA	RawTrace	184.8±2.71***	3.17±0.026	0.37±0.049	7.00±0.036	0
CSQA	Coconut	75.7±1.52***	1.64±0.066	0.73±0.037	7.60±0.011	0.04±0.024
CSQA	CCoT	59.3±1.17***	1.44±0.076	0.83±0.036	7.39±0.043	0.04±0.024
CSQA	CoLaR	53.6±0.77***	1.38±0.103	0.87±0.011	7.50±0.027	0.04±0.008
CSQA	SoftCoT	70.4±0.66***	1.58±0.042	0.75±0.053	7.31±0.053	0.04±0.022
CSQA	SoftThink	66.5±1.32***	1.55±0.107	0.78±0.050	7.30±0.018	0.04±0.019
CSQA	TokenSkip	55.6±1.24***	1.37±0.016	0.88±0.031	7.10±0.030	0
ECQA	CLP	60.9±0.73	1.49±0.060	0.79±0.023	7.80±0.038	0.08±0.016
ECQA	C3oT	56.0±0.97$^{n.s.}$	1.42±0.092	0.85±0.039	7.10±0.037	0.04±0.018
ECQA	CODI	62.8±1.30$^{n.s.}$	1.49±0.076	0.80±0.017	7.20±0.020	0.04±0.016
ECQA	DRewrite	54.5±1.20$^{n.s.}$	1.36±0.046	0.89±0.027	7.00±0.024	0
ECQA	CoD	16.9±1.09**	0.90±0.082	1.33±0.025	6.99±0.047	0
ECQA	RawTrace	170.6±4.27***	2.97±0.114	0.40±0.024	7.01±0.054	0
ECQA	Coconut	68.0±0.61***	1.56±0.039	0.77±0.027	7.59±0.046	0.04±0.012
ECQA	CCoT	52.7±0.48***	1.37±0.019	0.87±0.038	7.40±0.030	0.04±0.016
ECQA	CoLaR	47.6±0.74***	1.31±0.050	0.92±0.011	7.50±0.028	0.04±0.022
ECQA	SoftCoT	63.0±0.62***	1.49±0.024	0.80±0.027	7.29±0.037	0.04±0.008
ECQA	SoftThink	59.6±0.78***	1.45±0.032	0.83±0.017	7.30±0.011	0.04±0.019
ECQA	TokenSkip	49.3±0.49***	1.29±0.111	0.93±0.010	7.09±0.042	0
MathQA	CLP	183.5±2.14	2.97±0.056	0.39±0.050	7.80±0.031	0.08±0.010
MathQA	C3oT	63.8±1.01$^{n.s.}$	1.50±0.024	0.80±0.028	7.10±0.047	0.04±0.020
MathQA	CODI	78.6±1.39$^{n.s.}$	1.68±0.114	0.71±0.029	7.19±0.053	0.04±0.019
MathQA	DRewrite	183.8±1.50$^{n.s.}$	2.91±0.045	0.42±0.043	7.00±0.031	0
MathQA	CoD	29.3±1.24**	1.04±0.100	1.14±0.021	7.00±0.019	0
MathQA	RawTrace	245.1±1.65***	3.90±0.048	0.31±0.029	7.00±0.016	0
MathQA	Coconut	188.6±2.77***	3.00±0.051	0.40±0.008	7.60±0.013	0.04±0.015
MathQA	CCoT	164.3±3.59***	2.72±0.054	0.44±0.050	7.40±0.028	0.04±0.017
MathQA	CoLaR	158.7±2.88***	2.65±0.039	0.44±0.040	7.50±0.029	0.04±0.018
MathQA	SoftCoT	184.1±4.53***	2.95±0.056	0.41±0.015	7.30±0.015	0.04±0.017
MathQA	SoftThink	178.6±1.36***	2.89±0.045	0.42±0.052	7.30±0.050	0.04±0.012
MathQA	TokenSkip	156.7±0.97***	2.58±0.041	0.46±0.039	7.10±0.035	0
OBQA	CLP	50.3±0.80	1.38±0.066	0.87±0.036	7.80±0.014	0.08±0.019
OBQA	C3oT	57.8±1.49$^{n.s.}$	1.43±0.019	0.84±0.035	7.10±0.024	0.04±0.009
OBQA	CODI	61.3±1.52$^{n.s.}$	1.48±0.077	0.81±0.049	7.20±0.020	0.04±0.008
OBQA	DRewrite	54.2±0.49$^{n.s.}$	1.35±0.113	0.90±0.050	7.00±0.053	0
OBQA	CoD	17.6±0.82**	0.90±0.090	1.32±0.044	7.00±0.016	0
OBQA	RawTrace	175.4±4.46***	3.04±0.046	0.39±0.043	7.01±0.054	0
OBQA	Coconut	57.7±0.50***	1.44±0.074	0.84±0.014	7.60±0.037	0.04±0.015
OBQA	CCoT	42.1±0.64***	1.24±0.100	0.96±0.030	7.40±0.014	0.04±0.022
OBQA	CoLaR	36.8±0.81***	1.18±0.093	1.02±0.021	7.50±0.011	0.04±0.017
OBQA	SoftCoT	52.4±1.24***	1.37±0.025	0.88±0.041	7.30±0.027	0.04±0.016
OBQA	SoftThink	49.2±1.12***	1.33±0.037	0.90±0.048	7.30±0.033	0.04±0.016
OBQA	TokenSkip	38.5±0.67***	1.15±0.087	1.03±0.029	7.10±0.036	0
SQA	CLP	46.9±0.89	1.35±0.075	0.89±0.010	7.80±0.030	0.08±0.014
SQA	C3oT	64.7±1.47$^{n.s.}$	1.52±0.101	0.79±0.017	7.10±0.022	0.04±0.023
SQA	CODI	60.8±1.53$^{n.s.}$	1.48±0.063	0.82±0.050	7.20±0.029	0.04±0.008
SQA	DRewrite	49.4±1.19$^{n.s.}$	1.30±0.089	0.93±0.034	7.00±0.013	0
SQA	CoD	19.0±1.07**	0.93±0.030	1.29±0.017	7.00±0.032	0
SQA	RawTrace	189.8±3.29***	3.24±0.074	0.37±0.017	7.00±0.035	0
SQA	Coconut	57.0±0.48***	1.42±0.055	0.83±0.037	7.60±0.017	0.04±0.025
SQA	CCoT	39.8±0.87***	1.22±0.040	0.98±0.019	7.40±0.032	0.04±0.014
SQA	CoLaR	34.0±1.08***	1.15±0.102	1.04±0.009	7.50±0.030	0.04±0.014
SQA	SoftCoT	49.3±1.04***	1.34±0.043	0.90±0.044	7.30±0.025	0.04±0.021
SQA	SoftThink	45.5±1.00***	1.30±0.045	0.93±0.014	7.31±0.040	0.04±0.012
SQA	TokenSkip	36.2±1.09***	1.13±0.050	1.06±0.051	7.10±0.023	0
SVAMP	CLP	72.2±1.13	1.65±0.037	0.73±0.020	7.80±0.051	0.08±0.014
SVAMP	C3oT	102.7±1.62$^{n.s.}$	1.97±0.098	0.62±0.047	7.10±0.020	0.04±0.014
SVAMP	CODI	78.3±2.01$^{n.s.}$	1.69±0.066	0.71±0.043	7.20±0.030	0.04±0.009
SVAMP	DRewrite	97.8±2.19$^{n.s.}$	1.88±0.056	0.64±0.010	7.00±0.040	0
SVAMP	CoD	29.3±1.00**	1.05±0.044	1.13±0.050	7.00±0.054	0
SVAMP	RawTrace	245.2±4.81***	3.89±0.056	0.31±0.037	7.00±0.046	0
SVAMP	Coconut	85.7±1.49***	1.76±0.094	0.68±0.037	7.60±0.053	0.04±0.024
SVAMP	CCoT	63.8±0.76***	1.51±0.111	0.81±0.054	7.40±0.040	0.04±0.024

Data	Method	Tok.	Lat.	Thru.	Mem.	OH
SVAMP	CoLaR	51.4±0.94***	1.35±0.041	0.88±0.022	7.49±0.040	0.04±0.012
SVAMP	SoftCoT	76.2±1.85***	1.65±0.093	0.73±0.033	7.31±0.050	0.04±0.024
SVAMP	SoftThink	71.1±1.45***	1.59±0.041	0.75±0.044	7.30±0.036	0.04±0.017
SVAMP	TokenSkip	53.8±1.03***	1.36±0.063	0.89±0.011	7.10±0.048	0

Abbreviations: AQUA = AQUA-RAT; ARC-C = ARC-Challenge; CSQA = CommonsenseQA; OBQA = OpenBookQA; SQA = StrategyQA; DRewrite = Direct Rewrite; CoD = Chain of Draft; SoftThink = Soft Thinking; Tok. = Tokens; Lat. = Latency; Thru. = Throughput; OH = Proxy overhead.

C.8 Loss and Hyperparameter Sensitivity Results

Table 16 expands the supplementary sensitivity analysis. The final setting $\lambda_r = 0.3$, $\lambda_s = 0.5$, and $\lambda_c = 0.1$ is a fixed operating point rather than a dataset-specific choice. The rows explain why this setting is reasonable. Removing reconstruction or sufficiency lowers accuracy because the proxy no longer has enough pressure to preserve answer-supporting behavior. Increasing the capacity penalty improves compression but can remove information needed for correctness. Lowering the penalty keeps more information but weakens the compression objective. The sensitivity table therefore supports the claim that the selected setting is a stable accuracy–compression compromise.

Table 16: Detailed loss and hyperparameter sensitivity records.

Dataset	Setting	λ_r	λ_s	λ_c	Accuracy (%)	Compression (%)
AQUA-RAT	Final setting	0.3	0.5	0.1	46.5±0.65	45.1±0.75
AQUA-RAT	No reconstruction loss	0	0.5	0.1	41.5±1.00***	46.3±1.76
AQUA-RAT	Low reconstruction loss	0.1	0.5	0.1	45.4±1.81***	45.7±1.60
AQUA-RAT	High reconstruction loss	0.8	0.5	0.1	45.6±0.46***	43.0±2.49
AQUA-RAT	No sufficiency loss	0.3	0	0.1	42.3±1.59***	45.3±1.39
AQUA-RAT	Low capacity penalty	0.3	0.5	0.03	46.7±1.38***	40.1±2.56
AQUA-RAT	High capacity penalty	0.3	0.5	0.3	44.3±1.06***	48.9±1.78
AQUA-RAT	No redundancy/entropy regularization	0.3	0.5	0.1	45.0±1.82***	44.0±2.59
ARC-Challenge	Final setting	0.3	0.5	0.1	73.3±0.88	72.9±1.87
ARC-Challenge	No reconstruction loss	0	0.5	0.1	69.2±1.80***	74.3±0.89
ARC-Challenge	Low reconstruction loss	0.1	0.5	0.1	71.9±1.63***	73.4±2.42
ARC-Challenge	High reconstruction loss	0.8	0.5	0.1	72.6±1.42***	70.4±2.56
ARC-Challenge	No sufficiency loss	0.3	0	0.1	69.7±1.44***	73.5±1.23
ARC-Challenge	Low capacity penalty	0.3	0.5	0.03	73.5±0.91***	68.5±2.50
ARC-Challenge	High capacity penalty	0.3	0.5	0.3	71.5±1.27***	77.3±2.11
ARC-Challenge	No redundancy/entropy regularization	0.3	0.5	0.1	72.0±0.94***	71.8±2.23
BoolQ	Final setting	0.3	0.5	0.1	76.5±0.89	71.7±1.11
BoolQ	No reconstruction loss	0	0.5	0.1	72.0±0.91***	73.6±1.25
BoolQ	Low reconstruction loss	0.1	0.5	0.1	75.1±1.33***	72.5±1.37
BoolQ	High reconstruction loss	0.8	0.5	0.1	75.7±0.90***	69.7±1.46
BoolQ	No sufficiency loss	0.3	0	0.1	73.1±1.32***	72.3±1.05
BoolQ	Low capacity penalty	0.3	0.5	0.03	76.8±1.62***	67.1±2.05
BoolQ	High capacity penalty	0.3	0.5	0.3	74.3±0.56***	76.2±0.50
BoolQ	No redundancy/entropy regularization	0.3	0.5	0.1	75.0±0.54***	71.1±0.96
CommonsenseQA	Final setting	0.3	0.5	0.1	68.4±1.70	62.7±2.08
CommonsenseQA	No reconstruction loss	0	0.5	0.1	64.0±1.73***	64.1±0.65
CommonsenseQA	Low reconstruction loss	0.1	0.5	0.1	67.1±1.31***	63.3±1.07
CommonsenseQA	High reconstruction loss	0.8	0.5	0.1	67.2±1.41***	60.0±2.07
CommonsenseQA	No sufficiency loss	0.3	0	0.1	64.6±1.59***	62.9±0.81
CommonsenseQA	Low capacity penalty	0.3	0.5	0.03	68.6±1.75***	57.9±2.11
CommonsenseQA	High capacity penalty	0.3	0.5	0.3	66.4±1.39***	66.5±1.94
CommonsenseQA	No redundancy/entropy regularization	0.3	0.5	0.1	66.9±1.26***	61.7±0.90
ECQA	Final setting	0.3	0.5	0.1	69.1±1.08	64.0±0.78
ECQA	No reconstruction loss	0	0.5	0.1	64.9±0.76***	65.9±1.39
ECQA	Low reconstruction loss	0.1	0.5	0.1	67.7±0.87***	65.0±1.99
ECQA	High reconstruction loss	0.8	0.5	0.1	68.3±1.63***	61.8±2.08
ECQA	No sufficiency loss	0.3	0	0.1	65.4±0.98***	64.8±1.94
ECQA	Low capacity penalty	0.3	0.5	0.03	69.2±0.57***	59.6±1.50
ECQA	High capacity penalty	0.3	0.5	0.3	66.8±1.20***	68.1±2.18
ECQA	No redundancy/entropy regularization	0.3	0.5	0.1	67.8±1.81***	63.2±2.12
MathQA	Final setting	0.3	0.5	0.1	50.1±0.96	24.8±2.41
MathQA	No reconstruction loss	0	0.5	0.1	45.0±0.78***	26.7±0.94
MathQA	Low reconstruction loss	0.1	0.5	0.1	48.7±0.98***	25.6±0.86
MathQA	High reconstruction loss	0.8	0.5	0.1	49.2±1.53***	22.8±1.10
MathQA	No sufficiency loss	0.3	0	0.1	45.8±1.59***	25.5±0.58
MathQA	Low capacity penalty	0.3	0.5	0.03	50.3±1.55***	20.6±0.91
MathQA	High capacity penalty	0.3	0.5	0.3	48.3±1.75***	29.1±0.79
MathQA	No redundancy/entropy regularization	0.3	0.5	0.1	48.5±1.37***	24.0±2.41
OpenBookQA	Final setting	0.3	0.5	0.1	67.8±1.80	71.0±2.36
OpenBookQA	No reconstruction loss	0	0.5	0.1	63.7±0.80***	73.0±2.17
OpenBookQA	Low reconstruction loss	0.1	0.5	0.1	66.6±0.82***	72.1±1.44
OpenBookQA	High reconstruction loss	0.8	0.5	0.1	67.3±0.76***	68.9±1.79
OpenBookQA	No sufficiency loss	0.3	0	0.1	64.5±0.98***	72.0±1.98
OpenBookQA	Low capacity penalty	0.3	0.5	0.03	68.3±0.46***	66.5±1.76
OpenBookQA	High capacity penalty	0.3	0.5	0.3	66.1±0.57***	75.5±0.65
OpenBookQA	No redundancy/entropy regularization	0.3	0.5	0.1	66.4±1.37***	70.6±1.41
StrategyQA	Final setting	0.3	0.5	0.1	73.4±1.16	75.5±1.78
StrategyQA	No reconstruction loss	0	0.5	0.1	69.4±1.62***	76.8±2.00
StrategyQA	Low reconstruction loss	0.1	0.5	0.1	72.1±0.70***	76.1±1.82
StrategyQA	High reconstruction loss	0.8	0.5	0.1	72.5±0.85***	73.0±1.15
StrategyQA	No sufficiency loss	0.3	0	0.1	69.8±1.41***	75.6±1.00
StrategyQA	Low capacity penalty	0.3	0.5	0.03	73.5±0.81***	70.6±1.32
StrategyQA	High capacity penalty	0.3	0.5	0.3	71.4±1.02***	79.3±1.46
StrategyQA	No redundancy/entropy regularization	0.3	0.5	0.1	72.2±1.55***	74.5±0.52
SVAMP	Final setting	0.3	0.5	0.1	74.4±1.41	70.5±0.68
SVAMP	No reconstruction loss	0	0.5	0.1	68.9±1.78***	72.3±1.47
SVAMP	Low reconstruction loss	0.1	0.5	0.1	72.8±1.17***	71.1±2.47
SVAMP	High reconstruction loss	0.8	0.5	0.1	73.2±1.52***	68.3±1.38
SVAMP	No sufficiency loss	0.3	0	0.1	69.7±1.40***	71.2±2.14
SVAMP	Low capacity penalty	0.3	0.5	0.03	74.4±0.90***	65.7±2.57
SVAMP	High capacity penalty	0.3	0.5	0.3	72.3±0.59***	74.7±1.44
SVAMP	No redundancy/entropy regularization	0.3	0.5	0.1	72.9±1.30***	69.9±1.41

C.9 Representative Failure-Case Results

Table 17 lists three diagnostic failures rather than aggregate benchmark rows. The cases were chosen because they expose different boundaries of reasoning-trace compression. MathQA tests whether the proxy preserves enough arithmetic detail after identifying the right principle. StrategyQA tests whether a retained cue is mapped to the correct yes/no polarity. SVAMP tests whether an operation label is enough when the actual numerical execution is missing. The shared pattern is that the trace is not empty or irrelevant; it often contains a useful clue. The failure occurs when the compressed trace does not preserve the final executable step. This is why the paper frames these records as boundary analysis and not as a separate quantitative benchmark.

C.10 Failure-Case Results and Analysis

The section gives the original results behind the three representative failure cases in Table 17. These results are used only for qualitative boundary analysis. They are not additional aggregate results.

MathQA: incomplete inclusion-exclusion arithmetic. The MathQA case shows a common reasoning-trace compression failure: the rationale preserves the correct high-level principle but does not carry the calculation to completion. The model identifies inclusion-exclusion and names the least common multiple of 11 and 35, but the final answer is 892 rather than the gold option E. The missing step is the concrete count: there are 100 multiples of 11, 31 multiples of 35, and 2 multiples of both, so the count divisible by either divisor is 129 and the count divisible by neither is 971. In the recorded output, CLP stops at the procedure description and emits an answer that corresponds to option C. This supports the limitation that the compact proxy can retain procedural intent without preserving enough arithmetic detail for exact execution.


```
{"accuracy":0.0,"answer_rationale_mismatch":false,"dataset":"
↪  mathqa",
"example_id":"mathqa-00000","final_answer_matches_gold":false,
"final_answer_present":true,"final_answer_text":"892","gold_a
↪  nswer":"E",
```

Table 17: Detailed failure-case records.

Dataset	Method	Observed Error / Metrics	Case Notes	Purpose
MathQA	CLP	accuracy=0; gold=E; final=892; sufficiency=1; trace_tokens=77; raw_trace_tokens=127	mathqa-00000: retains the inclusion-exclusion principle but does not finish the arithmetic	Used to explain method boundaries; not a quantitative table
StrategyQA	CLP	accuracy=0; gold=yes; final=No...; sufficiency=1; trace_tokens=39; raw_trace_tokens=167	strategyqa-00015: keeps the birth-year cue but flips the yes/no judgment	Used to explain method boundaries; not a quantitative table
SVAMP	CLP	accuracy=0; gold=198; final=96 pages; sufficiency=0; trace_tokens=21; raw_trace_tokens=108	svamp-00002: describes division but computes it incorrectly	Used to explain method boundaries; not a quantitative table


```
"high_sufficiency":false,"method_name":"CLP","misleading_wron
↪  g_proxy":true,
"missing_step_proxy":false,"partial_sufficiency":false,
"question":"what is the number of integers from 1 to 1100 (
↪  inclusive ) that are divisible by neither 11 nor by 35
↪  ?\nA. 884\nB. 890\nC. 892\nD. 910\nE. 945",
"question_feature_coverage":0.538462,"rationale_present":true,
"rationale_sufficiency_score":1,
"rationale_text":"To find the number of integers from 1 to 1100
↪  that are divisible by neither 11 nor by 35, we can use the
↪  principle of inclusion-exclusion. First, we find the number
↪  of integers divisible by 11, then by 35, and subtract the
↪  number divisible by both (11 and 35, which is 11 * 35 =
↪  385). We then subtract this from the total number of
↪  integers (1100) to get
the number divisible by neither 11 nor 35.",
"rationale_token_count":76,"raw_trace_tokens":127,
"source_file":"results\\mathqa\\mathqa_gate_records.jsonl",
"structural_evidence":true,"trace_compression_rate":0.3937007
↪  874015748,
"trace_tokens":77}
```


StrategyQA: polarity reversal under a retained cue. The StrategyQA case illustrates a polarity-sensitive failure. The rationale retains the useful cue that a student graduating high school in 2017 would have been born in 1999, and it also notes that 9/11 occurred in 2001. However, it converts that cue into the wrong yes-or-no judgment. Under the dataset label, the correct answer is yes because the student would have been about two years old at the time, which falls in the childhood-amnesia window described in the question context. The generated answer says no, so the error is not caused by total loss of the relevant clue. It is caused by a failure to map the retained cue to the correct polarity.


```
{"accuracy":0.0,"answer_rationale_mismatch":false,"dataset":"
↪  strategyqa",
"example_id":"strategyqa-00015","final_answer_matches_gold":f
↪  alse,
"final_answer_present":true,
"final_answer_text":"No, a student of the class of 2017 would
↪  not have amnesia about 9/11.",
"gold_answer":"yes","high_sufficiency":false,"method_name":"C
↪  LP",
"misleading_wrong_proxy":true,"missing_step_proxy":false,
"partial_sufficiency":false,
"question":"Context: Childhood amnesia is common, with most
↪  adults not remembering their lives before 2 or 3 years of
↪  age 9/11 occurred in 2001 Students graduating high-school
↪  in 2017 would have been born in 1999\nQuestion: Would a
↪  student of the class of 2017 have amnesia about 9/11?",
"question_feature_coverage":0.833333,"rationale_present":true,
"rationale_sufficiency_score":1,
"rationale_text":"Childhood amnesia is common, with most
↪  adults not remembering their lives before 2 or 3 years of
↪  age. The student of the class of 2017 would have been born
↪  in 1999, which is well before 9/11 occurred in 2001.",
"rationale_token_count":40,"raw_trace_tokens":167,
"source_file":"results\\strategyqa\\strategyqa_gate_records.j
↪  sonl",
"structural_evidence":true,"trace_compression_rate":0.7664670
↪  658682635,
"trace_tokens":39}
```


SVAMP: operation stated but not executed. The SVAMP case is the clearest example of an over-compressed trace. The rationale states the right operation: divide the total number of pages by the number of chapters. The trace is only 21 tokens compared with 108 raw-trace tokens, so it is highly compressed. However, the final answer is 96 pages instead of 198, and the rationale does not show the actual computation $594/3 = 198$. This case supports the boundary claim that a very short trace can preserve the operation label while losing the decisive intermediate execution step.


```
{"accuracy":0.0,"answer_rationale_mismatch":false,"dataset":"
↪  svamp",
"example_id":"svamp-00002","final_answer_matches_gold":false,
"final_answer_present":true,"final_answer_text":"96 pages.",
"gold_answer":"198","high_sufficiency":false,"method_name":"C
↪  LP",
"misleading_wrong_proxy":true,"missing_step_proxy":false,
"partial_sufficiency":false,
"question":"Frank was reading through his favorite book. The
↪  book had 3 chapters, each with the same number of pages. It
↪  has a total of 594 pages. It took Frank 607 days to finish
↪  the book. How many pages are in each chapter?",
"question_feature_coverage":0.176471,"rationale_present":true,
"rationale_sufficiency_score":0,
"rationale_text":"To find the number of pages in each chapter,
↪  we divide the total number of pages by the number of
↪  chapters.",
"rationale_token_count":21,"raw_trace_tokens":108,
"source_file":"results\svamp\\svamp_gate_records.jsonl",
"structural_evidence":true,"trace_compression_rate":0.8055555
↪  555555556,
"trace_tokens":21}
```


C.11 External Long-Reasoning Generalization Results

Table 18 evaluates longer and more difficult reasoning settings, including mathematical, multi-hop, and symbolic tasks. The external rows are important because they test whether the proxy only fits the main benchmark mix. After synchronizing output tokens with compression, CLP averages 48.9% accuracy, 158.5 output tokens, and 2.68s latency across these external rows. RawTrace remains the strongest accuracy reference at 49.7% on average, but it has zero compression and averages 292.5 output tokens and 4.46s latency. Chain of Draft is much shorter, averaging 65.7 tokens and 1.49s latency, but its average accuracy is lower at 45.2%. The results therefore show a sharper accuracy–compression tension: CLP reduces output length and remains competitive, but it does

not uniformly dominate the raw trace on external tasks and it is not the shortest method. Mathematical tasks are especially sensitive because losing a single intermediate calculation can change the final answer. The external evidence therefore supports cautious generalization: CLP transfers as a practical compression mechanism, but long-form exact reasoning still needs stronger verification or execution-aware constraints.

Table 18: Detailed external long-reasoning generalization records.

Task / Dataset	Method	Acc. (%)	Comp. (%)	Tokens	Lat. (s)
GSM8K	CLP	60.5±1.17	48.0±0.78	146.2±3.62	2.52±0.087
GSM8K	C3oT	59.8±1.44***	54.4±2.44	128.5±2.83	2.29±0.089
GSM8K	CODI	59.1±1.41**	54.9±1.02	125.9±2.01	2.25±0.052
GSM8K	Direct Rewrite	58.0±0.56***	45.3±2.09	154.2±2.20	2.55±0.027
GSM8K	Chain of Draft	56.9±0.64***	77.9±0.76	61.5±0.84	1.43±0.077
GSM8K	RawTrace Baseline	62.1±1.37*	0	279.5±4.82	4.31±0.083
MATH	CLP	31.9±1.14	35.2±2.24	263.4±1.73	3.94±0.069
MATH	C3oT	31.1±1.23***	42.8±2.23	231.4±3.21	3.52±0.059
MATH	CODI	31.2±1.81**	45.0±0.71	223.2±2.84	3.41±0.034
MATH	Direct Rewrite	28.8±1.23***	31.9±1.45	275.0±5.56	4.01±0.076
MATH	Chain of Draft	27.9±1.13***	71.8±2.12	113.3±1.02	2.06±0.073
MATH	RawTrace Baseline	33.1±0.77*	0	404.3±4.61	5.81±0.084
GSM-Hard	CLP	43.7±1.09	42.1±0.53	191.0±2.36	3.07±0.040
GSM-Hard	C3oT	43.0±1.21***	48.7±1.90	168.3±1.25	2.76±0.093
GSM-Hard	CODI	41.9±0.78**	52.0±2.52	158.6±1.37	2.62±0.106
GSM-Hard	Direct Rewrite	40.9±1.68***	39.0±2.51	201.4±2.81	3.13±0.093
GSM-Hard	Chain of Draft	39.9±1.47***	74.7±2.63	82.7±2.06	1.69±0.065
GSM-Hard	RawTrace Baseline	45.1±1.51*	0	331.2±7.92	4.91±0.038
HotpotQA	CLP	51.8±1.63	55.1±2.02	103.7±1.50	2.02±0.069
HotpotQA	C3oT	51.1±1.36***	58.2±1.83	96.8±2.20	1.91±0.093
HotpotQA	CODI	49.9±0.65**	59.9±2.02	91.9±1.86	1.83±0.098
HotpotQA	Direct Rewrite	50.2±1.58***	52.2±1.96	110.3±0.84	2.02±0.050
HotpotQA	Chain of Draft	48.0±0.66***	82.1±1.87	41.4±0.86	1.20±0.039
HotpotQA	RawTrace Baseline	51.9±0.64*	0	230.4±4.48	3.70±0.109
2WikiMultiHopQA	CLP	49.1±1.01	53.0±0.86	117.6±0.78	2.18±0.080
2WikiMultiHopQA	C3oT	47.9±1.10***	57.3±2.08	107.6±0.93	2.03±0.085
2WikiMultiHopQA	CODI	48.1±1.59**	58.3±1.64	105.2±2.18	2.00±0.070
2WikiMultiHopQA	Direct Rewrite	47.2±1.22***	50.0±0.61	124.7±2.69	2.20±0.087
2WikiMultiHopQA	Chain of Draft	44.9±1.11***	82.2±1.92	45.1±0.95	1.24±0.023
2WikiMultiHopQA	RawTrace Baseline	49.0±0.86*	0	249.0±6.12	3.94±0.088
BBH symbolic subsets	CLP	56.4±1.24	50.2±1.26	129.6±2.81	2.34±0.034
BBH symbolic subsets	C3oT	56.1±0.98***	53.8±2.09	119.8±1.45	2.18±0.042
BBH symbolic subsets	CODI	55.9±1.39**	55.9±1.54	114.5±1.61	2.11±0.033
BBH symbolic subsets	Direct Rewrite	55.1±0.87***	48.0±0.60	135.5±2.10	2.31±0.049
BBH symbolic subsets	Chain of Draft	52.8±1.11***	79.9±1.25	52.1±1.20	1.32±0.033
BBH symbolic subsets	RawTrace Baseline	57.0±0.63*	0	259.8±5.20	4.08±0.037