STRUCTURAL PROCESS SUPERVISION FOR LATENT CHAIN-OF-THOUGHT REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent reasoning approaches enhance token-level efficiency and robustness by replacing verbose, explicit chain-of-thought (CoT) tokens with compact continuous-space embeddings. However, existing methods lack efficient process supervision over these embeddings, which often leads to representation collapse and uneven information distribution. To address this, we propose Prototype-Mediated Process Supervision (PMPS), which introduces learnable reasoning prototypes as semantic anchors to provide structural process-level supervision for latent reasoning. PMPS projects latent embeddings and explicit CoT embeddings into a shared prototype space, achieving many-to-many soft alignment between unequal-length representations through prototype assignment. Meanwhile, we introduce a Progressive Sequential Alignment (PSA) module to further guide training: positional priors initially encourage sequential alignment structure, then gradually relax to permit adaptive matching. Experimental results show that PMPS compresses output token length to under 50% of explicit CoT on GSM8K-Aug. Compared to leading baseline SIM-CoT, our method achieves average accuracy gains of 2.51% across different model families. On GPT-2, accuracy of PMPS even surpasses CoT-SFT. On larger models and a more challenging task, PMPS consistently attains the highest accuracy among all latent reasoning methods with comparable output length.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated remarkable reasoning capabilities on complex tasks such as mathematical reasoning and logical inference (Achiam et al., 2023; Grattafiori et al., 2024; Comanici et al., 2025; Yang et al., 2025; Liu et al., 2025; Zeng et al., 2026), particularly when employing Chain-of-Thought (CoT) prompting techniques (Wei et al., 2022). However, explicit CoT reasoning suffers from multiple limitations, including low inference efficiency and constrained expressiveness (Chen et al., 2025; Li et al., 2025b; Zhu et al., 2025), motivating researchers to explore more efficient and flexible reasoning paradigms.

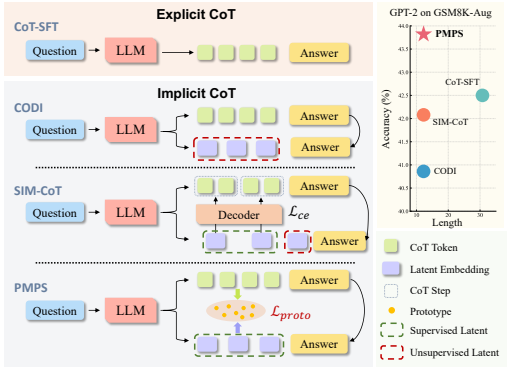


Figure 1: **Comparison of different CoT methods.** Our proposed PMPS provides process supervision over all latent embeddings.

To overcome the limitations of explicit CoT, researchers have proposed latent reasoning methods that perform reasoning in latent space rather than language space (Li et al., 2025a; Zhang et al., 2025; Tan et al., 2025). Compared to explicit CoT, latent reasoning offers three advantages: **(1) Efficiency**: Replacing verbose CoT tokens with compact hidden states substantially reduces inference cost. **(2) Expressiveness**: Continuous high-dimensional representations transcend the discrete token bottleneck, encoding richer semantic information that natural language cannot faithfully capture(Deng et al., 2024; Hao et al., 2024; Deng et al., 2026). **(3) Controllability**: The reasoning length and implicit CoT are controllable by presetting the number of latent tokens. Despite these advantages, existing latent reasoning approaches still suffer from critical limitations. As shown in Figure 1, CODI (Shen et al., 2025) optimizes

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

latent embeddings solely via the final answer, which leaves intermediate reasoning unguided and often causes *representation collapse*. SIM-CoT (Wei et al., 2025) mitigates this collapse via an auxiliary decoder to establish a one-to-one alignment between fixed-number latent embeddings and variable-length CoT steps. This design, which introduces a decoder of identical architecture to the base model, not only increases training overhead but also induces information bottlenecks, as excessive reasoning steps are compressed while surplus embeddings remain unsupervised. Empirical experiment (Figure 4) reveals two prevalent *representation collapse* in existing methods: *odd-even collapse*, where embeddings at odd and even positions collapse into two alternating groups; and *tail collapse*, where the last few embeddings become nearly identical.

To address the above limitations, we propose **Prototype-Mediated Process Supervision (PMPS)**, a novel framework that provides structural process supervision for latent reasoning. We introduce learnable reasoning prototypes as semantic anchors and project both latent and explicit CoT embeddings into a unified prototype space, imposing structured semantic organization on the latent representations. Through Sinkhorn-Knopp (Cuturi, 2013) assignments, we establish soft many-to-many alignment between latent and CoT representations, and formulate a bidirectional cross-prediction loss that compels each latent embedding to capture differentiated reasoning semantics. Furthermore, we introduce a **Progressive Sequential Alignment (PSA)** module that applies a Gaussian positional prior to impose sequential structure on the matching, then gradually relaxes it via cosine annealing to allow adaptive alignment. Importantly, all components are used only during training, incurring only a small amount of training overhead and no additional inference cost.

Extensive experiments demonstrate the effectiveness of PMPS. On GSM8K-Aug, PMPS reduces the reasoning length by over 50% relative to explicit CoT on GPT-2 while improving accuracy by 3.96%, and it surpasses the CODI and SIM-CoT baselines by 4.33% and 2.86%, respectively. PMPS consistently achieves superior accuracy over all latent reasoning baselines across different model families, scales, and the more challenging task. In summary, our main contributions can be summarized as follows:

- We propose PMPS, the first framework to introduce structural process-level supervision for latent reasoning. PMPS establishes a soft alignment between unequal-length latent and explicit CoT embeddings via a prototype-based bidirectional cross-prediction objective, effectively mitigating *representation collapse*.
- To preserve the intrinsic sequential reasoning structure of both CoT and latent embeddings, we introduce PSA, which employs a Gaussian positional prior with cosine annealing to guide early sequential alignment and gradually relaxes the constraint during training, enabling adaptive matching while balancing structural bias and flexibility.
- Extensive experiments across diverse benchmarks and model scales demonstrate that PMPS consistently outperforms leading latent reasoning baselines while preserving the inference efficiency, confirming both strong effectiveness and broad generalizability.

2 Related Work

2.1 Latent Reasoning in LLMs

Chain-of-thought (CoT) prompting (Wei et al., 2022) improves LLM reasoning, but the inference cost increases substantially with verbose reasoning length, motivating reasoning in continuous hidden-state space. iCoT (Deng et al., 2024) and Coconut (Hao et al., 2024) progressively replace explicit CoT tokens with latent embeddings via curriculum learning. CODI (Shen et al., 2025) stabilizes training through teacher–student distillation at the answer position. SIM-CoT (Wei et al., 2025) enforces rigid one-to-one assignment between CoT steps and latent embeddings with an auxiliary decoder. CoLaR (Tan et al., 2025) and Latent-SFT (Deng et al., 2026) compress a contiguous sequence of CoT tokens into a single latent embedding. However, these methods either lack direct reasoning supervision or enforce an overly rigid correspondence between the CoT and latent embeddings, which limits the expressiveness of latent representations. We propose PMPS to address these limitations via a structural prototype-mediated soft alignment between the CoT and latent embeddings in latent space, requiring *no auxiliary decoder* and *no hard CoT partition*.

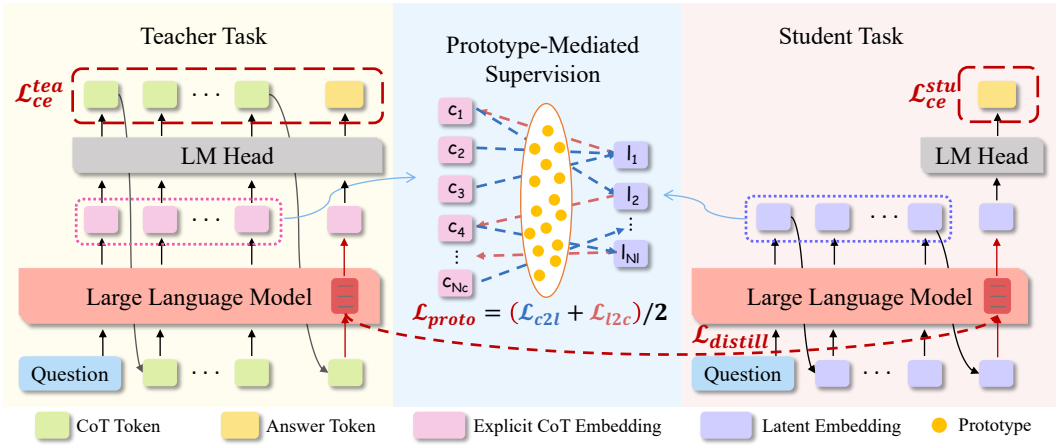


Figure 2: **Overview of PMPS framework.** During training, PMPS adopts the conventional parallel teacher-student self-distillation architecture and leverages the explicit CoT embeddings from the teacher task to supervise the latent embeddings from the student task via a bidirectional cross-prediction loss in the prototype space. At inference, only the student task is executed, performing latent reasoning without any additional overhead.

2.2 Process Supervision and Prototype-Based Representation Learning

Process Reward Models (Lightman et al., 2024) show that step-level correctness labels yield better reasoning verifiers than outcome-only rewards, with follow-up work automating such supervision via Monte Carlo tree search (Wang et al., 2024). However, existing process supervision methods require explicit, discrete reasoning steps, which are incompatible with continuous latent embeddings. In parallel, SwAV (Caron et al., 2020) and PCL (Li et al., 2020), originally from visual self-supervised learning, show that learnable prototypes can effectively structure embedding spaces without negative pairs. Knowledge distillation approaches (Sun et al., 2019; Wang et al., 2020) align teacher–student representations but typically assume matched sequence lengths. Our work bridges these by leveraging the prototype-based assignment and cross-prediction objective to provide many-to-many structural supervision across the unequal number of latent and CoT embeddings, complemented by answer-position distillation from the teacher to ensure outcome-level alignment.

3 Methodology

In this section, we present the technical details of our proposed framework. We first review the conventional self-distillation framework. We then introduce PMPS, which provides structural process-level supervision for latent embeddings. Finally, we describe the PSA module, which imposes a curriculum-scheduled positional prior to guide sequential alignment structure during training. The overview of our framework is presented in Figure 2 and Appendix A.1.

3.1 Preliminaries

To obtain high-quality explicit CoT embeddings that align with the model's intrinsic distribution as supervision signals, we adopt the widely used self-distillation framework (Zhang et al., 2019; Liao et al., 2023; Liu et al., 2026) as the foundational architecture. Within this framework, the model is jointly optimized for two tasks: explicit CoT reasoning (teacher task) and latent reasoning (student task). Inspired by CODI (Shen et al., 2025), we align the hidden states of teacher and student task at the answer token positions. The objective function for this base architecture is formulated as:

$$\mathcal{L}_{\text{base}} = \mathcal{L}_{ce}^{stu} + \mathcal{L}_{ce}^{tea} + \lambda_d \cdot \mathcal{L}_{dist} \tag{1}$$

where $\mathcal{L}_{ce}^{stu}$ and $\mathcal{L}_{ce}^{tea}$ are cross-entropy losses for the student's answer prediction and the teacher's full-sequence prediction, $\mathcal{L}_{dist}$ is a Smooth L1 distillation loss aligning teacher and student hidden states at the answer token, and λ_d is the weighting coefficient for distillation loss. However, relying solely on this base architecture lacks process-level supervision for the latent embeddings, which can lead to latent homogenization and representation collapse.

3.2 PROTOTYPE-MEDIATED PROCESS SUPERVISION

To mitigate the representation collapse induced by the absence of process supervision, we propose to leverage the CoT embeddings from the teacher task as supervision signals. These explicit CoT embeddings are well-suited targets for two reasons: (1) the $\mathcal{L}_{ce}^{tea}$ loss ensures that they encode high-quality, semantically meaningful reasoning information, and (2) since both the explicit CoT embeddings and latent embeddings are generated by the same model, they share the same hidden-state space, providing a common representational basis for alignment.

However, direct one-to-one distillation is infeasible because the number of latent embeddings N_l is far smaller than the number of CoT tokens N_c. We therefore propose Prototype-Mediated Process Supervision (PMPS), which introduces K learnable prototype vectors $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_K\} \in \mathbb{R}^{K\times d}$ as shared semantic anchors to mediate a soft many-to-many alignment between the two sets of representations, imposing structured semantic organization on the latent space. Let $\mathbf{H}_l = \{\mathbf{l}_1, \ldots, \mathbf{l}_{N_l}\} \in \mathbb{R}^{N_l\times D}$ and $\mathbf{H}_c = \{\mathbf{c}_1, \ldots, \mathbf{c}_{N_c}\} \in \mathbb{R}^{N_c\times D}$ denote the latent embeddings and the explicit CoT embeddings, respectively, where D is the hidden dimension and d is the prototype dimension.

Projection and Scoring. A shared two-layer MLP with batch normalization serves as the projection head $g(\cdot)$, projecting both $\mathbf{H}_l$ and $\mathbf{H}_c$ from the original hidden space into the prototype space, followed by ℓ_2 normalization:

$$\mathbf{z} = g(\mathbf{h})/\|g(\mathbf{h})\|_2 \tag{2}$$

where $\mathbf{h}$ denotes either the latent embedding or the explicit CoT embedding. Each unit feature vector $\mathbf{z}$ is scored against the ℓ_2-normalized prototypes via inner product $s_k = \mathbf{z}^\top \mathbf{p}_k$, yielding score matrices $\mathbf{S}_l \in \mathbb{R}^{N_l\times K}$ and $\mathbf{S}_c \in \mathbb{R}^{N_c\times K}$.

Prototype Assignment. To obtain soft assignments while preventing prototype collapse, we apply the Sinkhorn-Knopp algorithm (Cuturi, 2013) to enforce an equipartition constraint. Given a score matrix $\mathbf{S} \in \mathbb{R}^{M\times K}$, where M denotes the number of tokens, we initialize $\mathbf{Q}^{(0)} \propto \exp(\mathbf{S}/\epsilon)$, with ϵ serving as the regularization parameter. We then iteratively alternate row and column normalization:

$$\mathbf{Q}^{(t)} \leftarrow \mathrm{Norm}_{\mathrm{row}}\Big(\mathrm{Norm}_{\mathrm{col}}\Big(\mathbf{Q}^{(t-1)}\Big)\Big) \tag{3}$$

The converged $\mathbf{Q} \in \mathbb{R}^{M\times K}$ nearly satisfies that each row sums to 1 (a valid per-token distribution) and each column sums to M/K (uniform prototype utilization) after three iterations. This procedure is applied independently to both $\mathbf{S}_l$ and $\mathbf{S}_c$ with stop-gradient operation following SwAV (Caron et al., 2020), producing the final soft assignment codes $\mathbf{Q}_l \in \mathbb{R}^{N_l\times K}$ and $\mathbf{Q}_c \in \mathbb{R}^{N_c\times K}$. Detailed Sinkhorn-Knopp algorithm is show in Appendix A.2.

Many-to-Many Matching. Leveraging the prototype assignments as a bridge, we compute the soft matching matrix between latent embeddings and explicit CoT embeddings:

$$\mathbf{M}_{c2l} = \mathbf{Q}_c \cdot \mathbf{Q}_l^\top/\tau_m \in \mathbb{R}^{N_c\times N_l} \tag{4}$$

where temperature τ_m controls matching sharpness. We then apply softmax normalization along the last dimensions to obtain bidirectional matching weights:

$$\mathbf{A}_{c2l} = \mathrm{softmax}(\mathbf{M}_{c2l}), \quad \mathbf{A}_{l2c} = \mathrm{softmax}(\mathbf{M}_{c2l}^\top) \tag{5}$$

This formulation naturally induces a many-to-many alignment: a single latent embedding can attend to multiple explicit CoT embeddings for compression, while a single explicit CoT embedding can contribute to multiple latent embeddings for semantic sharing.

Bidirectional Cross-Prediction Loss. Alongside the stop-gradient assignment codes $\mathbf{Q}$, we compute gradient-carrying prototype predictions $\boldsymbol{\phi}_l = \mathrm{softmax}(\mathbf{S}_l/\tau)$ and $\boldsymbol{\phi}_c = \mathrm{softmax}(\mathbf{S}_c/\tau)$, where τ is a prediction temperature. For each explicit CoT embedding j, we aggregate the matched latent predictions weighted by $\mathbf{A}_{c2l}$ and minimize the cross-entropy with its assignment code:

$$\mathcal{L}_{c2l} = -\frac{1}{N_c}\sum_{j=1}^{N_c}\sum_{k=1}^{K} Q_c^{(j,k)} \log\left(\sum_{i=1}^{N_l} A_{c2l}^{(j,i)} \cdot \phi_{l,i}^{(k)}\right) \tag{6}$$

where $Q_c^{(j,k)}$ is the (j,k)-th entry of $\mathbf{Q}_c$, $A_{c2l}^{(j,i)}$ is the matching weight from the j-th explicit CoT embedding to the i-th latent embedding, and $\phi_{l,i}^{(k)}$ is the predicted probability of the i-th latent

embedding for the k-th prototype. Symmetrically, the latent-to-CoT direction yields:

$$\mathcal{L}_{l2c} = -\frac{1}{N_l} \sum_{i=1}^{N_l} \sum_{k=1}^{K} Q_l^{(i,k)} \log \left(\sum_{j=1}^{N_c} A_{l2c}^{(i,j)} \cdot \phi_{c,j}^{(k)} \right) \tag{7}$$

The overall process supervision loss is $\mathcal{L}_{proto} = \frac{1}{2}(\mathcal{L}_{c2l} + \mathcal{L}_{l2c})$. The CoT-to-latent direction ($\mathcal{L}_{c2l}$) ensures that the latent embeddings collectively explain each explicit CoT embedding, while the latent-to-CoT direction ($\mathcal{L}_{l2c}$) encourages each latent embedding to capture diverse CoT semantics.

3.3 Progressive Sequential Alignment

Explicit CoT inherently follows a progressive process: early tokens build up the reasoning chain, while final tokens converge toward the answer. However, the unconstrained matching in the base PMPS allows any latent token to align with any CoT position, which risks failing to capture the progressive reasoning structure. Conversely, enforcing a rigid segmented mapping (e.g., uniformly partitioning the CoT into N_l segments for strict assignment) is overly restrictive: uniform partitioning introduces noisy supervision that undermines the flexibility of semantic-driven matching. To navigate between these two extremes, we introduce Progressive Sequential Alignment (PSA), which injects sequential structure into the matching through a soft Gaussian positional prior and gradually relaxes it via curriculum-scheduled decay. Concretely, we augment $\mathbf{M}_{c2l}$ with a bias penalizing positionally distant matches:

$$\hat{M}_{c2l}^{(j,i)} = M_{c2l}^{(j,i)} - \alpha \cdot \left(\frac{j-1}{N_c - 1} - \frac{i-1}{N_l - 1} \right)^2 \tag{8}$$

where ($u_j = (j-1)/(N_c - 1)$) and ($v_i = (i-1)/(N_l - 1)$) are normalized positions, and (α) controls the prior strength. This bias acts as a multiplicative Gaussian prior ($\exp[-\alpha(u_j - v_i)^2]$), favoring matches at similar relative positions (Su & Hua, 2017; Yang et al., 2018). As the latent index increases, the preferred CoT region shifts forward, encouraging sequential alignment while retaining semantic flexibility without enforcing strict monotonicity. The biased scores replace (M_{c2l}) in both directions of the matching normalization. To balance structural guidance with representational flexibility, we adopt cosine annealing to decay α from $\alpha_{\max}$ to 0 over training:

$$\alpha(t) = \frac{\alpha_{\max}}{2} \left(1 + \cos \left(\pi \cdot \frac{t}{T} \right) \right) \tag{9}$$

where t and T are the current and total training steps. Early in training (α large), the strong prior helps latent embeddings establish sequential structure and stabilizes matching. Late in training ($\alpha \to 0$), the constraint is fully released, allowing adaptive alignment where reasoning steps may span or consolidate across latent tokens. PSA modifies only the matching scores before softmax, introducing no learnable parameters and incurring zero inference overhead.

3.4 Overall Training Objective

The overall training objective combines the base loss with our proposed process supervision loss:

$$\mathcal{L}_{total} = \mathcal{L}_{base} + \lambda_{proto} \cdot \mathcal{L}_{proto} \tag{10}$$

where λ_{proto} is the weighting coefficient for the process supervision loss.

The teacher task, prototype layer, projection head, and PSA module are used exclusively during training. At inference time, the model architecture and computational cost are identical to the single student task. This design ensures that all accuracy improvements from process supervision come at zero additional inference cost.

4 Experiments

4.1 Experimental Setup

Training and Evaluation Data. Following prior work on latent reasoning (Shen et al., 2025; Wei et al., 2025), we primarily train and evaluate on **GSM8K-Aug** (Deng et al., 2023), an augmented

Table 1: **Performance comparison on in-domain dataset and out-of-domain datasets.** Experiments are conducted on three different backbones, reporting answer accuracy (Acc.) and output token length (Len.). Explicit CoT method is marked with . The best and second best accuracy among latent reasoning approaches are highlighted in **bold** and underline, respectively.

Method	In-domain		Out-of-domain					Average	
	GSM8K-Aug		GSM-Hard	SVAMP	MultiArith	Average			
	Acc. (%)	Len.	Acc. (%)	Acc. (%)	Acc. (%)	Acc. (%)	Len.	Acc. (%)	Len.
GPT-2									
CoT-SFT	42.50	30.76	9.25	40.00	92.22	47.16	25.08	45.99	26.50
Coconut	36.50	15.40	7.58	34.67	82.20	41.48	11.87	40.24	12.75
CoLaR-2	11.07	14.43	2.58	15.00	41.11	19.56	10.89	17.44	11.78
CODI	40.86	12.23	9.40	40.00	92.22	47.21	12.89	45.62	12.72
SIM-CoT	42.08	12.24	9.40	43.00	93.89	48.76	12.58	47.09	12.49
PMPS(Ours)	**44.35**	12.24	**10.46**	**48.33**	**96.67**	**51.82**	12.58	**49.95(+2.86)**	12.49
Qwen2.5-0.5B-Instruct									
CoT-SFT	59.82	42.68	16.91	61.67	97.78	58.79	38.91	59.05	39.85
Coconut	15.69	12.53	3.26	26.33	22.78	17.46	13.01	17.02	12.89
CoLaR-2	29.57	17.92	6.44	37.00	77.78	40.41	16.04	37.70	16.51
Latent-SFT	43.21	13.09	9.86	50.66	91.67	50.73	15.60	48.85	14.97
CODI	39.35	14.43	9.33	48.33	83.89	47.18	15.03	45.23	14.88
SIM-CoT	42.08	14.45	10.16	51.67	92.22	51.35	15.03	49.03	14.91
PMPS(Ours)	**46.70**	14.43	**10.84**	**57.33**	**95.00**	**54.39**	15.68	**52.47(+3.44)**	15.37
Llama-3.2-1B-Instruct									
CoT-SFT	63.00	32.51	15.09	66.33	98.33	59.92	27.07	60.69	28.43
Coconut	40.03	16.44	8.42	47.33	83.89	46.55	13.80	44.92	14.46
CoLaR-2	42.15	12.70	8.95	55.33	91.67	51.98	13.41	49.53	13.24
Latent-SFT	48.82	12.83	10.54	50.33	90.56	50.48	9.65	50.06	10.44
CODI	53.90	13.21	12.28	56.00	**97.78**	55.35	11.46	54.99	11.90
SIM-CoT	56.03	13.20	12.36	60.33	97.22	56.64	13.47	56.49	13.41
PMPS(Ours)	**56.56**	13.19	**13.42**	**63.67**	97.22	**58.10**	13.51	**57.72(+1.23)**	13.43

version of the GSM8K (Cobbe et al., 2021) where CoT annotations are structured arithmetic expressions rather than natural language. To assess the generalization ability of our method, we additionally evaluate on three out-of-domain (OOD) benchmarks: (1) **GSM-Hard** (Gao et al., 2023), (2) **SVAMP** (Patel et al., 2021), (3) **MultiArith** (Roy & Roth, 2015). Detailed information is provided in Appendix B.1. We adopt two evaluation metrics: **Accuracy (Acc.)** measures the correctness of the final numerical answer, and **Length (Len.)** measures the total number of generated tokens.

Baselines. We compare our approach against six representative baselines across the following four categories: (1) Explicit CoT methods: **CoT-SFT**; (2) Curriculum learning-based methods: **Coconut** (Hao et al., 2024); (3) Token compression-based methods: **CoLaR-2** (Tan et al., 2025) and **Latent-SFT** (Deng et al., 2026); (4) Self-distillation-based methods: **CODI** (Shen et al., 2025) and **SIM-CoT** (Wei et al., 2025). Notably, the latter three categories constitute latent reasoning methods. Detailed descriptions of all baselines are provided in Appendix B.2.

Implementation Details. We conduct experiments on three model families: GPT-2 (Radford et al., 2019), Qwen2.5-0.5B-Instruct (Yang et al., 2024), and LLaMA-3.2-1B-Instruct (Grattafiori et al., 2024). For a fair comparison, all methods except Coconut adopt LoRA (Hu et al., 2022) fine-tuning with the same rank of 128; Coconut uses full-parameter fine-tuning as in its original implementation. For SIM-CoT, we adopt the CODI-based variant for a direct comparison under the same backbone. All other training hyperparameters follow the respective original papers and formal repositories. For PMPS, we set $N_l = 6$ for GPT-2 and Qwen2.5-0.5B-Instruct, and $N_l = 8$ for LLaMA-3.2-1B-Instruct. More detailed configurations are provided in Appendix C.

4.2 Main Result

Table 1 presents the main results on one in-domain benchmark (GSM8K-Aug) and three out-of-domain benchmarks (GSM-Hard, SVAMP, MultiArith), with three backbone models from different model families: GPT-2, Qwen2.5-0.5B-Instruct, and LLaMA-3.2-1B-Instruct.

Table 2: **Scalability to a larger backbone model.** We report answer accuracy (Acc.) and output token length (Len.) on the four aforementioned benchmarks using LLaMA-3.2-3B-Instruct. Explicit CoT method is marked with . The best and second best accuracy among latent reasoning approaches are highlighted in **bold** and underline, respectively.

Method	In-domain		Out-of-domain					Average	
	GSM8K-Aug		GSM-Hard	SVAMP	MultiArith	Average			
	Acc. (%)	Len.	Acc. (%)	Acc. (%)	Acc. (%)	Acc. (%)	Len.	Acc. (%)	Len.
CoT-SFT	71.27	32.95	20.52	78.00	100.00	66.17	27.89	67.45	29.15
Coconut	11.98	12.25	2.35	14.67	13.33	10.12	12.23	10.58	12.24
CoLaR-2	57.39	12.89	13.87	69.00	96.67	59.85	9.88	59.23	10.63
CODI	54.28	13.20	12.96	67.00	**100.00**	59.99	13.48	58.56	13.41
SIM-CoT	61.41	13.19	14.03	**71.67**	97.22	60.97	13.53	61.08	13.45
PMPS(Ours)	**63.46**	13.17	**14.40**	71.33	**100.00**	**61.91**	13.48	**62.30(+1.22)**	13.40

Table 3: **Scalability to more challenging task.** We conduct experiments on MATH using LLaMA-3.2-3B-Instruct and Qwen3-4B-Instruct. The best accuracy among latent approaches is highlighted in **bold**.

Method	Llama-3.2-3B		Qwen3-4B	
	Acc. (%)	Len.	Acc. (%)	Len.
COT-SFT	21.00	119.69	38.60	132.26
CoLaR-2	6.20	54.27	18.40	54.30
CODI	8.80	14.85	17.60	17.61
PMPS(Ours)	**11.00**	14.95	**19.60**	17.60

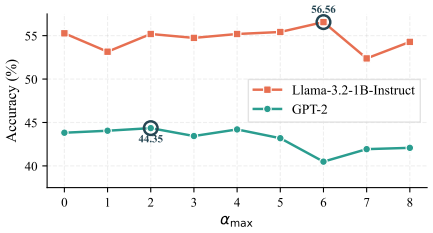


Figure 3: **Ablation Study on PSA maximum enforcement strength.** The best configurations are marked with circles.

Overall performance. On GPT-2, PMPS outperforms the two leading latent reasoning baselines, CODI and SIM-CoT, by 3.49% and 2.27% on GSM8K-Aug, and by 4.33% and 2.86% in average accuracy. Remarkably, PMPS even exceeds the explicit reasoning upper bound CoT-SFT by 3.96% in average accuracy while using only 47.1% of the output tokens, demonstrating that latent reasoning with effective process supervision can surpass explicit CoT at significantly higher inference efficiency. On Qwen2.5-0.5B-Instruct and LLaMA-3.2-1B-Instruct, PMPS consistently outperforms all latent reasoning baselines (+3.44% and +1.23%). These gains across diverse model architectures and scales confirm the effectiveness of our proposed structural process supervision method.

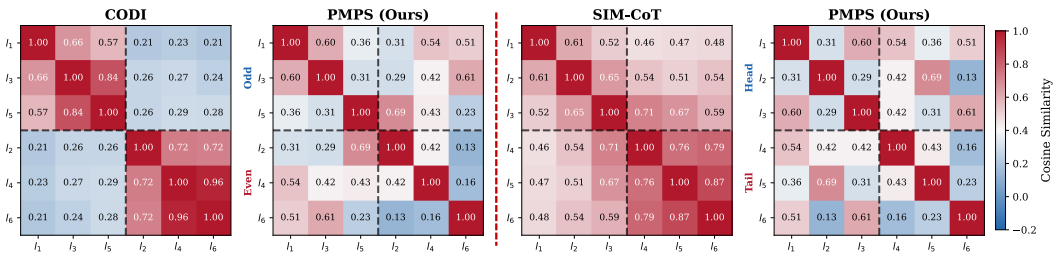
Out-of-domain (OOD) generalization. The OOD results show that PMPS generalizes robustly to unseen benchmarks: PMPS achieves an average accuracy of 51.82%, 54.39%, and 58.10% across three backbone models, surpassing the strongest latent baseline SIM-CoT by 3.06%, 3.04%, and 1.46%. The superior performance on OOD benchmarks confirms that our proposed PMPS helps internalize genuine arithmetic reasoning rather than shallow pattern matching.

Inference efficiency. Compared to explicit CoT-SFT, PMPS reduces the output token length to under 50% across all three models (47.1%, 38.6% and 47.2%, respectively) while achieving comparable or superior accuracy. Critically, PMPS maintains output token length comparable to CODI and SIM-CoT, verifying that the accuracy improvements stem entirely from superior training-time structural process supervision rather than increased inference computation. Moreover, this structural supervision is lightweight: the projection head and prototypes only introduce less than 2M extra parameters ($< 1\%$ of the base model), whereas SIM-CoT's auxiliary decoder effectively doubles the trainable parameters during training.

4.3 Scaling to Larger Models and Harder Task

To further validate the scalability of PMPS, we conduct two additional groups of experiments: scaling to a larger backbone model and training on a more challenging dataset.

Larger model. As shown in Table 2, we scale up the backbone model to LLaMA-3.2-3B-Instruct (Grattafiori et al., 2024) and evaluate performance across the aforementioned benchmarks. PMPS achieves the highest average accuracy among all latent reasoning baselines, outperforming CODI



(a) Latent Similarity of CODI and PMPS (b) Latent Similarity of SIM-CoT and PMPS

Figure 4: **Average pairwise latent cosine similarity heatmaps.** The latent embeddings of CODI and SIM-CoT exhibit pronounced odd-even and tail collapse, respectively, whereas PMPS shows neither pattern of collapse.

and SIM-CoT by +3.74% and +1.22%, respectively, while maintaining comparable output length. Notably, PMPS demonstrates marked improvements over CODI on the in-domain GSM8K-Aug benchmark (+9.18%) and reduces the generation token length by over 54% relative to explicit CoT method. These results confirm that our proposed structural prototype-based alignment in PMPS remains highly effective as the model capacity scales up.

More challenging task. As presented in Table 3, we conduct training and evaluation on MATH, which consists of highly challenging competition-level problems with substantially longer reasoning chains. We employ two larger backbone models: LLaMA-3.2-3B-Instruct and Qwen3-4B-Instruct-2507 (Yang et al., 2025). Detailed experiment configurations are illustrated in Appendix C.2. On LLaMA-3.2-3B-Instruct and Qwen3-4B-Instruct-2507, PMPS respectively achieves an accuracy gain of +2.20% and +2.00% compared to CODI, while significantly compressing the output length by 87.51% and 86.69% compared to explicit CoT. Notably, the CoT annotations in MATH are formulated as free-form solutions without explicit step delimiters, which precludes the application of methods relying on strict one-to-one latent-to-step alignment (e.g., Coconut, SIM-CoT). In contrast, PMPS operates via soft prototype matching and remains agnostic to specific formatting, enabling its direct application to any CoT data.

4.4 Latent Embedding Analysis

To further investigate the reasons behind PMPS's superior performance, we directly analyze the pairwise cosine similarity among the N_l=6 latent embeddings on randomly sampled 100 GSM8K-Aug test instances using GPT-2. The analysis reveals that CODI and SIM-CoT suffer from distinct patterns of **representation collapse**, whereas PMPS maintains well-diversified representations free from either collapse mode. Additionally, we conduct the latent embedding interpretability and comparision experiments, which are provided in Appendix D.

PMPS resolves the odd-even collapse in CODI. Figure 4 (a) presents the average pairwise cosine similarity matrices for CODI (left) and PMPS (right), where tokens are grouped into odd-indexed (l_1, l_3, l_5) and even-indexed (l_2, l_4, l_6) subsets. For CODI, two homogeneous diagonal blocks emerge: within-odd similarity averages 0.693, within-even reaches 0.800, yet cross-group similarity drops to 0.250. This pattern persists across all 100 samples. This phenomenon arises because CODI's $\mathcal{L}_{dist}$ constrains only the answer-position hidden state, leaving the latent embeddings unsupervised. The autoregressive mechanism then converges to a degenerate solution where each token diverges from its predecessor, forming two alternating representations. Our PMPS avoids this collapse through two complementary mechanisms: the Sinkhorn equipartition constraint assigns different latent tokens to distinct prototypes, and the bidirectional cross-prediction loss forces each token to encode unique CoT semantics. As shown in the right panel, PMPS exhibits no block-diagonal separation between odd and even groups, confirming preserved representation diversity.

PMPS eliminates the tail collapse in SIM-CoT. Figure 4 (b) compares the average pairwise cosine similarity matrices of SIM-CoT (left) and PMPS (right). For SIM-CoT, similarity increases from head to tail, with the tail group (l_4, l_5, l_6) forming a prominent high-similarity block (average 0.805) that far exceeds the head group (average 0.591). This positive gap is observed in 85% of test

samples. This stems from SIM-CoT's rigid one-to-one matching, which binds each latent token to exactly one CoT step. In our training data, 99.0% of samples have fewer steps than N_l=6, leaving the tail latent tokens without corresponding supervision and collapsing into redundant representations. PMPS resolves this through soft many-to-many matching under the equipartition constraint, enabling each latent token to flexibly aggregate CoT content regardless of position. As shown in the right panel, PMPS displays no tail-concentrated block, with the tail group similarity even lower than head group, demonstrating that all six latent tokens encode diverse information.

4.5 Ablation Study

In this section, we conduct ablation studies to investigate the effects of different PMPS components, the PSA maximum enforcement strength, the number of latent embeddings and prototypes, and supervision loss weighting coefficient λ_{proto}. The number of latent embeddings and prototypes results, and λ_{proto} results are provided in Appendix E.

Ablation on PMPS components. The results in Table 4 isolate the contribution of each component. Removing (L_{l2c}) reduces average accuracy by 3.78%, as this objective updates the shared projection head and prototypes by matching predictions from CoT representations to latent assignment codes, thereby shaping the shared prototype space that mediates process supervision. Removing (L_{c2l}) decreases average accuracy by 2.05%, as this objective directly supervises latent representations by encouraging their aggregated prototype predictions to match the corresponding CoT assignment codes. Together, these results support the complementary roles of direct latent supervision through (L_{c2l}) and shared prototype-space learning through (L_{l2c}), with their combination yielding the best performance. Removing PSA leads to a consistent decline (-0.76% on Average), validating that the curriculum-scheduled positional prior provides necessary sequential structure guidance that pure semantic matching alone cannot recover. Removing prototypes (details in Appendix C.3) also yields a substantial drop(-3.49% on Average), confirming that the categorical structure imposed by prototypes, where alignment is mediated through shared clusters, is essential for establishing meaningful many-to-many correspondences between the latent and explicit CoT embeddings.

Table 4: **Ablation Study on PMPS components using GPT-2.** Different modules are removed to assess their contributions to the final performance.

Method	GSM8K-Aug Acc. (%)	OOD Average Acc. (%)	Average Acc. (%)
PMPS	**44.35**	**51.82**	**49.95**
- w/o $\mathcal{L}_{l2c}$	39.42	48.42	46.17(-3.78)
- w/o $\mathcal{L}_{c2l}$	42.23	49.79	47.90(-2.05)
- w/o PSA	43.82	50.98	49.19(-0.76)
- w/o prototype	41.70	48.04	46.46(-3.49)

Ablation on maximum enforcement strength. Figure 3 studies the effect of $\alpha_{\max}$ on GPT-2 and Llama-3.2-1B-Instruct. Both models benefit from a moderate $\alpha_{\max}$: GPT-2 peaks at $\alpha_{\max} = 2$ (44.35%), while Llama-3.2-1B achieves its best at $\alpha_{\max} = 6$ (56.56%). Excessively strong enforcement hurts performance on both models, as overly rigid positional constraints disrupt the flexibility of semantic-driven matching. Notably, the optimal $\alpha_{\max}$ scales with model capacity: GPT-2 (124M) favors a lighter prior, while Llama-3.2-1B (1B) benefits from stronger guidance, suggesting that larger models can better leverage structural priors before cosine annealing releases the constraint. *More experiments such as interpretability cases, hyperparameter study are given in Appendix D, E.*

5 Conclusion

We propose PMPS, a novel training paradigm for latent reasoning that performs structural process-level supervision on latent embeddings via soft alignment. Experimental results demonstrate that across diverse model families, model sizes, and task difficulties, our method consistently achieves SOTA performance among latent reasoning approaches while reducing the output token length by more than 50% compared to explicit CoT. These findings highlight the strong generalization, robustness, and scalability of our approach. Furthermore, we conduct an in-depth analysis from the latent representation perspective, offering insights into why PMPS consistently outperforms existing baselines. However, the current evaluation of PMPS is mainly limited to mathematical reasoning. Future work should further explore the effectiveness of PMPS on a broader range of reasoning tasks.

6 AI USE STATEMENT

In this work, we used generative AI tools solely to assist with language editing, including improving grammar, clarity, and readability of the manuscript. We did not use these tools for any other research or writing tasks, including those requiring disclosure under the ICLR 2027 AI policy. All AI-assisted edits were reviewed by the authors to ensure that they accurately preserved the intended meaning and technical content. The authors take full responsibility for the final content of the paper.

7 ETHICS STATEMENT

This work investigates latent reasoning using publicly available mathematical reasoning datasets. Our experiments do not involve human-subject studies or the collection of sensitive personal data. To the best of our knowledge, the scope and experimental setting of this study do not raise specific ethical concerns under the ICLR Code of Ethics.

8 REPRODUCIBILITY STATEMENT

To support reproducibility, we describe PMPS and its training procedure, including Sinkhorn-Knopp assignment, in Section 3 and Appendix A. The experimental setup, benchmarks, and baselines are presented in Section 4.1 and Appendix B, while implementation details, hyperparameters, and hardware configurations are provided in Appendix C. Additional analyses and ablation studies are included in Appendices D and E.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Xinghao Chen, Anhao Zhao, Heming Xia, Xuan Lu, Hanlin Wang, Yanjun Chen, Wei Zhang, Jian Wang, Wenjie Li, and Xiaoyu Shen. Reasoning beyond language: A comprehensive survey on latent chain-of-thought reasoning. *arXiv preprint arXiv:2505.16782*, 2025.

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

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Jingcheng Deng, Liang Pang, Zihao Wei, Shicheng Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Llm latent reasoning as chain of superposition, 2026. URL https://arxiv.org/abs/2510.15522.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023.

Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step, 2024. URL https://arxiv.org/abs/2405.14838.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International conference on machine learning*, pp. 10764–10799. PMLR, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. *arXiv preprint arXiv:2509.24251*, 2025a.

Jindong Li, Yali Fu, Li Fan, Jiahong Liu, Yao Shu, Chengwei Qin, Menglin Yang, Irwin King, and Rex Ying. Implicit reasoning in large language models: A comprehensive survey. *arXiv preprint arXiv:2509.02350*, 2025b.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020.

Yusheng Liao, Shuyang Jiang, Yiqi Li, Yu Wang, and Yanfeng Wang. Self-improvement of non-autoregressive model via sequence-level distillation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 14202–14212, 2023.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Yang Liu, Ming Ma, Xiaomin Yu, Pengxiang Ding, Han Zhao, Mingyang Sun, Siteng Huang, and Donglin Wang. Ssr: Enhancing depth perception in vision-language models via rationale-guided spatial reasoning. *Advances in Neural Information Processing Systems*, 38:123926–123958, 2026.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 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: human language technologies*, pp. 2080–2094, 2021.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Subhro Roy and Dan Roth. Solving general arithmetic word problems. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1743–1752, 2015.

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

Bing Su and Gang Hua. Order-preserving wasserstein distance for sequence matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model compression. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4323–4332, 2019.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. Think silently, think fast: Dynamic latent compression of llm reasoning chains. *Advances in Neural Information Processing Systems*, 38:4646–4668, 2025.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Xilin Wei, Xiaoran Liu, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Jiaqi Wang, Xipeng Qiu, and Dahua Lin. Sim-cot: Supervised implicit chain-of-thought. *arXiv preprint arXiv:2509.20317*, 2025.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Baosong Yang, Zhaopeng Tu, Derek F. Wong, Fandong Meng, Lidia S. Chao, and Tong Zhang. Modeling localness for self-attention networks. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4449–4458, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1475. URL https://aclanthology.org/D18-1475/.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3713–3722, 2019.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *Advances in Neural Information Processing Systems*, 38:168990–169012, 2025.

Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, Tianle Cai, Taylor Kergan, Assel Kembay, Andrew Smith, Chenghua Lin, Binh Nguyen, Yuqi Pan, Yuhong Chou, Zefan Cai, Zhenhe Wu, Yongchi Zhao, Tianyu Liu, Jian Yang, Wangchunshu Zhou, Chujie Zheng, Chongxuan Li, Yuyin Zhou, Zhoujun Li, Zhaoxiang Zhang, Jiaheng Liu, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A survey on latent reasoning, 2025. URL https://arxiv.org/abs/2507.06203.

Algorithm 1 Training PMPS with Progressive Sequential Alignment

Require: Dataset $\mathcal{D}$; model f_θ; projector g_ω; prototypes P; latent length N_l.
Require: Training steps T; temperatures τ, τ_m; Sinkhorn parameters ϵ, J; PSA strength $\alpha_{\max}$; process loss weight λ_{proto}.

for $t = 0, \ldots, T-1$ **do**
  Sample $\mathcal{B} = \{(x_b, c_b, y_b)\}_{b=1}^{B}$.
  Run the teacher task on (x_b, c_b, y_b); compute $\mathcal{L}_{\text{ce}}^{\text{tea}}$ on (c_b, y_b).
  In a separate no-gradient teacher pass, extract CoT predictor states H_c and answer-anchor targets U^{tea}.
  Run N_l recurrent latent steps on each x_b; collect $H_l^b = [l_1^b; \ldots; l_{N_l}^b]$.
  Teacher-force the student answer y_b; obtain $\mathcal{L}_{\text{ce}}^{\text{stu}}$ and answer-anchor states U^{stu}.
  Compute $\mathcal{L}_{\text{base}}$ from both task losses and distillation between U^{stu} and U^{tea}.
  $P \leftarrow \text{NormalizeRows}(P)$.
  for $v \in \{l, c\}$ **do**
    $Z_v \leftarrow \text{NormalizeRows}(g_\omega(H_v))$; $S_v \leftarrow Z_v P^\top$.
    $Q_v^{\text{valid}} \leftarrow \textsc{Sinkhorn}(\text{ValidRows}(S_v), \epsilon, J)$.
    Restore Q_v^{valid} to the per-example layout Q_v; set padded entries to zero.
    $\phi_v \leftarrow \text{softmax}_{\text{row}}(S_v/\tau)$.
  end for
  $\alpha_t \leftarrow \frac{\alpha_{\max}}{2}[1 + \cos(\pi t/T)]$.
  for $b = 1, \ldots, B$ **do**
    Construct the positional distance matrix D^b using Eq. 8 with strength $\alpha(t)$.
    $\hat{M}^b \leftarrow Q_c^b (Q_l^b)^\top / \tau_m - \alpha_t D^b$.
    $A_{c2l}^b \leftarrow \text{softmax}_{\text{row}}(\hat{M}^b)$; $A_{l2c}^b \leftarrow \text{softmax}_{\text{row}}((\hat{M}^b)^\top)$.
    $\widetilde{\phi}_c^b \leftarrow A_{c2l}^b \phi_l^b$; $\widetilde{\phi}_l^b \leftarrow A_{l2c}^b \phi_c^b$.
  end for
  $\mathcal{L}_{c2l} \leftarrow -\frac{1}{\sum_b N_c^b} \sum_{b,j,k} Q_{c,jk}^b \log \widetilde{\phi}_{c,jk}^b$.
  $\mathcal{L}_{l2c} \leftarrow -\frac{1}{BN_l} \sum_{b,i,k} Q_{l,ik}^b \log \widetilde{\phi}_{l,ik}^b$; $\mathcal{L}_{\text{proto}} \leftarrow \frac{1}{2}(\mathcal{L}_{c2l} + \mathcal{L}_{l2c})$.
  $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{base}} + \lambda_{\text{proto}} \mathcal{L}_{\text{proto}}$.
  Update θ, ω, P using $\nabla_{\theta,\omega,P} \mathcal{L}_{\text{total}}$.
end for
return Trained student model.

A Method Supplement

A.1 Overall PMPS Training

Algorithm 1 summarizes the training procedure of PMPS with Progressive Sequential Alignment (PSA). Each training example contains a question x, an explicit reasoning chain c, and an answer y. The teacher and student tasks share the same language model f_θ. The prototype projector g_ω and the prototype matrix $P \in \mathbb{R}^{K \times d}$ are shared across the two representation views.

All matching operations and loss sums in Algorithm 1 are restricted to valid positions. In the padded implementation, invalid matching destinations are masked before softmax, and padded source positions are excluded from loss reduction. Each directional loss is averaged over its valid target tokens across the minibatch, rather than averaging the per-example losses. Mixture probabilities are clamped below at 10^{-12} before taking logarithms.

Gradient flow and inference. Both the teacher targets and the Sinkhorn codes are detached, so the matching weights are also treated as fixed targets during each update. Gradients from $\mathcal{L}_{\text{proto}}$ flow through the prototype predictions to g_ω and P, and through the student representations to the language model. They do not flow through H_c to the teacher computation, while the shared language model still receives gradients from the teacher cross-entropy. At inference, only the student computation is retained. The prototype projector g_ω and P are discarded; the latent recurrence projector remains part of the student model.

A.2 Sinkhorn-Knopp Details

Algorithm 2 computes the detached prototype codes used in Algorithm 1. Its input $S \in \mathbb{R}^{M \times K}$ contains the scores of all valid tokens from one representation view, where M is the valid token

Algorithm 2 Sinkhorn–Knopp Assignment over Valid Minibatch Tokens

Require: Scores $S \in \mathbb{R}^{M \times K}$ with $M > 0$; regularization ϵ; iterations J.
Ensure: Detached assignments $Q \in \mathbb{R}^{M \times K}$.

function SINKHORN(S, ϵ, J)
  Disable gradient tracking for the following operations.
  $\delta \leftarrow 10^{-8}$; $R \leftarrow \exp(S/\epsilon)^\top$ ▷ $R \in \mathbb{R}^{K \times M}$
  $R \leftarrow R/(\sum_{k,m} R_{km} + \delta)$.
  for $r = 1, \ldots, J$ **do**
    $a_k \leftarrow \sum_{m=1}^{M} R_{km}$ for all k.
    $R_{km} \leftarrow R_{km}/[K(a_k + \delta)]$ for all k, m.
    $b_m \leftarrow \sum_{k=1}^{K} R_{km}$ for all m.
    $R_{km} \leftarrow R_{km}/[M(b_m + \delta)]$ for all k, m.
  end for
  return $Q \leftarrow MR^\top$.
end function

count and K is the number of prototypes. The procedure is applied independently to the latent and explicit CoT views.

Balanced assignment. Exponentiating the scores with regularization parameter ϵ produces a positive assignment matrix. In the transposed matrix R, rows index prototypes and columns index tokens. The two normalization steps alternately impose prototype mass $1/K$ and token mass $1/M$. After transposing and rescaling by M, we obtain the assignment matrix $Q \in \mathbb{R}^{M \times K}$.

$$Q\mathbf{1}_K = \mathbf{1}_M, \qquad Q^\top \mathbf{1}_M = \frac{M}{K}\mathbf{1}_K, \tag{11}$$

Here, $\mathbf{1}_n$ denotes the all-ones vector of length n. The first constraint normalizes each token's assignment distribution, while the second enforces uniform aggregate mass across prototypes. In practice, these constraints are approximately satisfied due to the finite number of iterations and numerical stabilization.

Implementation details. We use $J = 3$ iterations and $\epsilon = 0.05$. With a finite number of iterations, token-wise normalization is approximately satisfied after the final token-normalization step, whereas uniform prototype usage is approximate rather than exact. Padding tokens are removed before assignment and restored as zero rows afterward. In distributed training, the current implementation applies Sinkhorn within each device's minibatch, without gathering scores across devices. No gradients are propagated through this procedure; the returned codes supervise the differentiable prototype predictions.

B Benchmarks and Baselines

B.1 Benchmarks

GSM8K-Aug. GSM8K-Aug expands the original GSM8K training set from approximately 7.5K to 385K examples through GPT-4-based data generation, while retaining the standard 1,319-example test split. Unlike the natural language reasoning chains in GSM8K, the chain-of-thought annotations in GSM8K-Aug are represented as structured sequences of arithmetic expressions (e.g., `<<12*3=36>><<9*2=18>><<36+18=54>>`), where each expression is logically derived from preceding steps. This format facilitates training latent reasoning models by providing concise yet faithful step-by-step supervision.

GSM-Hard. GSM-Hard is a more challenging variant of GSM8K, created by replacing the numerical values in the original problems with larger numbers that require more complex arithmetic operations. This dataset is designed to evaluate the arithmetic robustness and compositional reasoning capabilities of language models under increased computational difficulty.

SVAMP. SVAMP (Simple Variations on Arithmetic Math word Problems) is a challenge set of elementary-level math word problems that tests sensitivity to structural variations in problem statements. By systematically varying question types, equation structures, and entity associations, SVAMP exposes models that rely on shallow heuristics rather than genuine mathematical reasoning.

MultiArith. MultiArith is a benchmark consisting of multi-step arithmetic word problems that require two or more mathematical operations to solve. It evaluates a model's ability to correctly decompose complex problem narratives into sequential reasoning steps involving basic arithmetic operations such as addition, subtraction, multiplication, and division.

MATH. MATH is a comprehensive benchmark comprising 12,500 competition-level mathematics problems spanning seven subjects including algebra, geometry, number theory, and combinatorics. Problems are annotated with difficulty levels from 1 to 5, and each solution requires multi-step reasoning with formal mathematical derivations, making it a rigorous testbed for advanced mathematical reasoning capabilities.

B.2 BASELINES

CoT-SFT. CoT-SFT is a standard baseline that directly fine-tunes the language model on (Question, Chain-of-Thought, Answer) triples via supervised fine-tuning, where the model learns to generate explicit step-by-step reasoning tokens before producing the final answer. It serves as the upper-bound reference for explicit reasoning performance, against which latent reasoning methods are compared.

Coconut. Coconut proposes a multi-stage curriculum learning strategy to train latent reasoning: starting from standard CoT data, it progressively replaces explicit language reasoning steps with continuous thoughts (the last hidden states of the LLM fed back directly as input embeddings), stage by stage, until all reasoning steps are internalized into the continuous latent space. This curriculum-based approach allows the model to gradually transition from explicit language reasoning to fully latent reasoning, avoiding the difficulty of learning continuous thought representations from scratch.

CoLaR-2. CoLaR-2 dynamically compresses explicit reasoning chains into the latent space through a two-stage training approach: first, supervised fine-tuning with an auxiliary next compressed embedding prediction objective that merges consecutive token embeddings using a variable compression factor; second, reinforcement learning that leverages the latent head's non-deterministic nature to explore diverse reasoning paths. In our experiments, we set the compression rate to 2. This framework allows the model to adjust reasoning speed at inference time by simply prompting the desired compression factor, achieving significant reductions in reasoning chain length with minimal performance degradation.

Latent-SFT. Latent-SFT frames latent reasoning as a "chain of superposition", where each latent token carries higher entropy and encodes a superposition of multiple reasoning trajectories rather than merely compressing a single reasoning path. It introduces a unified framework addressing challenges at three levels: constraining hidden states within the pre-trained vocabulary space (Latent-Vocab), constructing semantically compact and sufficient latent chains via Induction-Supervision Masking (Latent-Chain), and employing stochastic Gumbel-Softmax optimization to guide the model toward generalizable solutions (Latent-Optim).

CODI. CODI compresses explicit Chain-of-Thought reasoning into a continuous latent space through a self-distillation framework, jointly training a teacher task (explicit CoT) and a student task (implicit CoT) while aligning their hidden states at a designated token position.

Sim-CoT Sim-CoT addresses the latent instability problem in implicit CoT methods where latent representations become homogeneous and lose semantic diversity as the number of reasoning tokens increases by introducing step-level supervision via an auxiliary decoder during training that aligns each implicit token with its corresponding explicit reasoning step. The auxiliary decoder is removed at inference time, preserving the token efficiency of implicit CoT with no added overhead, while also providing interpretability through per-step projection onto the explicit reasoning vocabulary.

C Implementation Details

In this section, we present the detailed hyperparameter configurations used in GSM8K-Aug experiments, the MATH experiments in Table 3, and the component ablations in Table 4. Across all experiments, we adopt the AdamW optimizer with a weight decay of 0.1 and a warmup ratio of 0.03. For CoT-SFT, CoLaR-2, CODI, SIM-CoT, and our proposed PMPS, we apply LoRA-based fine-tuning with the LoRA rank set to 128 and the LoRA alpha set to 32. All models are trained on 8 NVIDIA A100 GPUs, and all evaluations are conducted on a single NVIDIA A100 GPU.

C.1 Experimental Settings on GSM8K-Aug

CoT-SFT. For GPT-2, the batch size is set to 256 and the learning rate to 8×10^{-4}, trained for 40 epochs. For Qwen2.5-0.5B-Instruct and Llama-3.2-1B-Instruct, the batch size is set to 512 with learning rates of 8×10^{-4} and 9×10^{-4}, respectively, trained for 20 epochs.

Coconut. All models use a batch size of 128 and a learning rate of 1×10^{-4}. Each stage replaces one CoT step with two latent embeddings, with a maximum of 5 stages, trained for 20 epochs in total. All other parameters remain consistent with the original paper.

CoLaR-2. All models are configured with a compression rate of 2, a batch size of 256, and a learning rate of 1×10^{-4}, trained for 50 epochs. All other parameters follow the original paper.

Latent-SFT. In Stage 2, the Gumbel temperature and noise scale are set to 1.0 for all models. All models are configured with a batch size of 512 and a learning rate of 3×10^{-4}, trained for 50 epochs. All other stages and parameters remain consistent with the original paper.

CODI and Sim-CoT. For GPT-2, Qwen2.5-0.5B-Instruct, and Llama-3.2-1B-Instruct, the learning rates are set to 3×10^{-3}, 8×10^{-4}, and 9×10^{-4}, respectively, with corresponding distillation loss factors of 1.0, 1.0, and 20.0. All other parameters follow the original paper.

PMPS. For training on GSM8K-Aug, the regularization parameter ϵ is set to 0.05, the matching temperature τ_m to 0.3, the prediction temperature τ to 0.1, and the prototype dimension d to 128. To improve prototype utilization and stability, Sinkhorn assignment is performed at the batch level rather than on individual samples, with the number of iterations set to 3. For GPT-2 and Qwen2.5-0.5B-Instruct, λ_{proto} is set to 0.5, the batch size to 256, the numbers of latents and prototypes to 6 and 24 respectively, and the maximum enforcement strength to 2. For Llama-3.2-1B-Instruct, λ_{proto} is set to 0.3, the batch size to 512, the number of latents and prototypes to 8 and 32, respectively, and the maximum enforcement strength to 6. All remaining parameters are consistent with CODI.

C.2 Experimental Settings on MATH

For experiments on MATH in Table 3, we use Llama-3.2-3B-Instruct and Qwen3-4B-Instruct-2507 as the backbone models. We retain the complete solution text as the explicit CoT supervision without segmenting it into predefined reasoning steps.

CoT-SFT. Both Llama-3.2-3B-Instruct and Qwen3-4B-Instruct-2507 models are trained with learning rate of 8×10^{-4} and batch size of 256 for 50 epochs.

CoLaR-2. Both models are trained with learning rate of 1×10^{-4} for 100 epochs. Other configurations are the same with GSM8K-Aug experiments.

PMPS and CODI. Both methods use $N_l = 8$ latent embeddings and a latent recurrence projector with hidden dimension 3072 and trained for 100 epochs. For Llama-3.2-3B-Instruct, we use a learning rate of 9×10^{-4} and a batch size of 256. For Qwen3-4B-Instruct-2507, we use a learning rate of 5×10^{-4} and a batch size of 256. For PMPS, the λ_{proto} is set to 0.3 and the number of prototypes is set to 32.

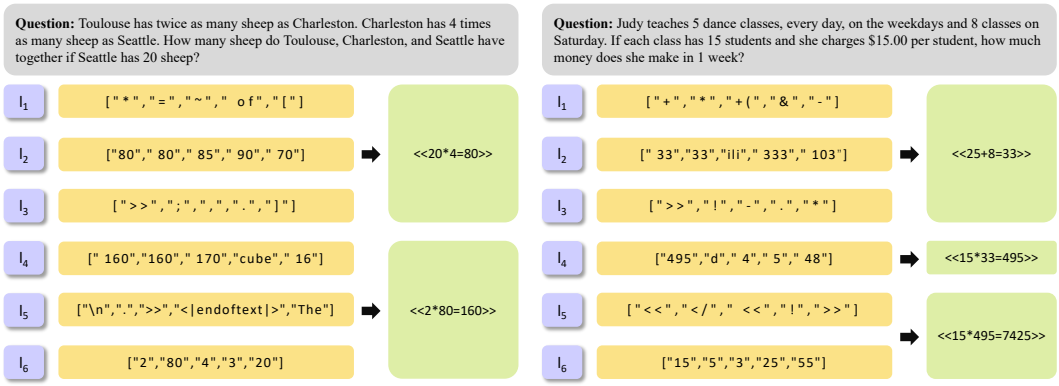


Figure 5: **Interpretability cases of PMPS.** We decode the latent embeddings and visualize the top-5 tokens with the highest probabilities, aligning them with the corresponding CoT steps.

Question: Janet buys a brooch for her daughter. She pays $500 for the material to make it and then another $800 for the jeweler to construct it. After that, she pays 10% of that to get it insured. How much did she pay?
Answer: <<500+800=1300>> <<1300*.1=130>> <<1300+130=1430>>

	Latent 1	Latent 2	Latent 3	Latent 4	Latent 5	Latent 6	Answer	
CODI	>> , .	130 140 135	. on >>	130 133 140	. on </	130 133 13	1330.0	✗
SIM-CoT	. 13 and	. , is	130 130 13	1300 13 13	1300 13 13	1300 13 13	1330.0	✗
PMPS	+ . ,	1300 1200 13	. >> ,	130 130 135	* . The	13 10 .	1430.0	✓

Figure 6: **Case comparision of CODI, SIM-CoT and PMPS.** All methods use GPT-2 with six latent steps. Each column lists the top-3 tokens obtained by projecting the corresponding latent state through the language model head. Colored backgrounds highlight recurring readouts across CODI's odd/even positions and SIM-CoT's final three positions.

C.3 Experimental Settings for Component Ablations

We conduct the component ablations in Table 4 using GPT-2 trained on GSM8K-Aug and evaluate on GSM8K-Aug, GSM-Hard, SVAMP, and MultiArith.

Removing a directional loss. For w/o $\mathcal{L}_{l2c}$, we remove the latent-to-CoT cross-prediction term and retain only $\mathcal{L}_{c2l}$ for process supervision. Conversely, w/o $\mathcal{L}_{c2l}$ removes the CoT-to-latent term and retains only $\mathcal{L}_{l2c}$. Both variants retain the shared projection head, prototypes, Sinkhorn assignments, and PSA. All other settings follow the reference configuration.

Removing PSA. For w/o PSA, we set the positional enforcement strength to zero throughout training, so the matching weights depend only on the prototype assignment similarities. Both directional losses and all other settings remain unchanged.

Removing prototypes. For w/o prototype, we remove the learnable prototypes and Sinkhorn assignment and replace prototype-mediated supervision with direct matching in the projected embedding space. We retain a shared two-layer MLP with hidden dimension 512 and output dimension 128, followed by ℓ_2 normalization. Pairwise cosine similarities between the projected CoT and latent representations, scaled by $\tau_m = 0.3$ and augmented with the positional bias, are normalized in each direction to obtain soft many-to-many matching weights. We replace the prototype cross-entropy objective with the mean of the two directional cosine reconstruction losses, each defined as one minus the cosine similarity between the target and its reconstruction and averaged over valid target positions in the minibatch. This variant preserves bidirectional soft matching while removing the prototype-based representation and balanced assignment mechanism.

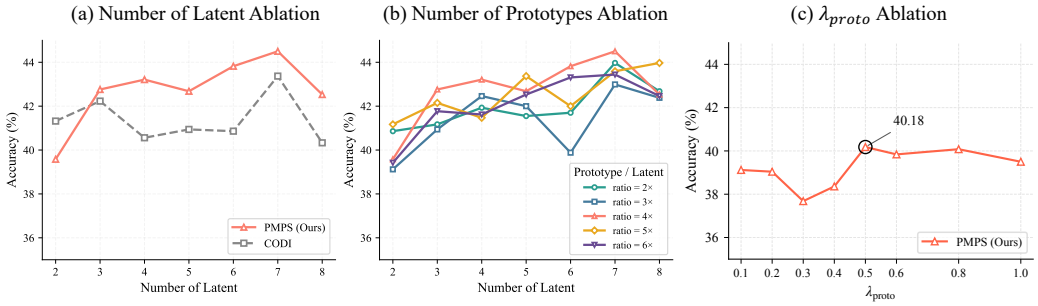


Figure 7: **Ablation studies of PMPS.** (a) Number of latent tokens, with CODI as the baseline. (b) Prototype-to-latent ratio across different latent counts. (c) Process-supervision weight (λ_{proto}) Ablation. All results are conducted with GPT-2 on GSM8K-Aug. In (c), all other hyperparameters are fixed at non-optimal settings; the best accuracy in this sweep therefore differs from the Table 1 result obtained with the final configuration.

D Interpretability Experiment

To qualitatively examine whether the latent reasoning embeddings encode semantically meaningful information, we conduct a qualitative vocabulary-readout analysis on GSM8K test examples using GPT-2 with ($N_l = 6$) latent steps. For each method, we project each latent state through its language model head and rank vocabulary tokens by their resulting probabilities. We report the top-5 tokens for the PMPS examples in Figure 5 and the top-3 tokens for the comparison with CODI and SIM-CoT in Figure 6. PMPS receives supervision through teacher CoT representations, without direct token-level reconstruction targets for its intermediate latent states. Following CODI, the final answer-revealing CoT step is excluded from the teacher reasoning chain used for distillation to avoid exposing the answer verbatim before the distillation anchor.

D.1 Semantic Content of PMPS Latent States

Figure 5 shows that the PMPS readouts contain numbers and symbols associated with the reference reasoning. In the sheep example, (l_2) and (l_4) assign high probability to (80) and (160), respectively, matching the intermediate quantities in the reference CoT. In the dance-class example, the corresponding positions expose (33) and (495). These examples illustrate that task-relevant intermediate values are accessible from the latent states through vocabulary projection.

D.2 Comparison with Baselines

Figure 6 compares the three methods on the same example, whose reference solution computes ($500 + 800 = 1300$), ($1300 \times 0.1 = 130$), and ($1300 + 130 = 1430$). CODI exhibits recurring readouts within its odd- and even-indexed positions: the even positions repeatedly favor (130), while (l_3) and (l_5) share punctuation and "on". SIM-CoT displays identical rendered top-3 token lists at (l_4), (l_5), and (l_6), dominated by (1300) and (13). These patterns are qualitatively consistent with the odd–even and tail similarity patterns, respectively, reported in Section 4.4. In contrast, PMPS exposes (1300) at (l_2) and (130) at (l_4), together with arithmetic operators such as "+" and "*" at other positions, and produces the correct answer (1430). Both baselines output (1330).

These examples complement the aggregate similarity analysis in Figure 4 by illustrating task-relevant content and repeated readout patterns. Vocabulary projections provide a partial view of the latent states; they do not establish a one-to-one mapping to explicit CoT steps or a causal explanation of the final predictions.

E Ablation Study

Ablation on number of latent tokens. Figure 7(a) compares PMPS (with a prototype-to-latent ratio of 4×) against the CODI baseline across varying numbers of latent tokens. PMPS outperforms

CODI at all latent counts except for 2, where the limited number of latent tokens constrains the capacity of prototype-based alignment, leaving insufficient room for soft matching to take effect. The best performance is achieved at 6 and 7 latent tokens. Notably, CODI suffers a sharp accuracy drop at 8 latent tokens, falling to the lowest among all configurations, indicating that its answer-level distillation alone is insufficient to supervise a larger set of latent representations. PMPS, however, still maintains competitive accuracy at 8 latent tokens, demonstrating that the prototype-based alignment provides more stable supervision and enables the method to scale up effectively.

Ablation on number of prototypes. Figure 7(b) examines the impact of the prototype-to-latent ratio by varying it across $\{2\times, 3\times, 4\times, 5\times, 6\times\}$. Among all ratios, the $4\times$ setting consistently yields the best overall performance across different latent counts. A ratio that is too small (e.g., $2\times$ or $3\times$) limits the expressiveness of the prototype space, providing insufficient semantic granularity for diverse reasoning patterns. Conversely, an excessively large ratio (e.g., $6\times$) introduces redundant prototypes that may dilute the clustering signal. Based on these results, we adopt a prototype-to-latent ratio of $4\times$ as the default setting in all other experiments.

Ablation on λ_{proto}. Figure 7(c) varies λ_{proto} from 0.1 to 1.0 while keeping all other hyperparameters fixed. Accuracy varies non-monotonically, reaching its highest observed value of 40.18% at $\lambda_{proto} = 0.5$. Increasing the weight therefore does not consistently improve accuracy. Importantly, the other hyperparameters in this sweep are not set to their optimal values, so its best result differs from the main-table result and should be interpreted as sensitivity under this fixed configuration.