COPT: CONTRASTIVE ON-POLICY THINKING WITH CONTINUOUS SPACES FOR GENERAL AND AGENTIC REASONING

Anonymous authors
Paper under double-blind review

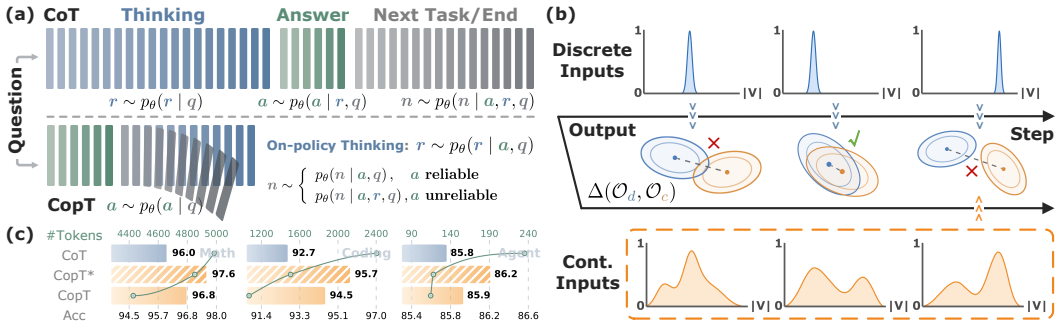


Figure 1: (a) Conceptual comparison between COT thinking and COPT on-policy thinking. (b) COPT contrasts the output distributions under discrete and continuous inputs. (c) COPT improves peak accuracy, marked by *, across mathematics, coding, and agentic reasoning tasks and substantially reduces tokens at matched accuracy.

ABSTRACT

Chain-of-thought (COT) is a standard approach for eliciting reasoning capabilities from large language models (LLMs). However, the common COT paradigm treats thinking as a prerequisite for answering, which can delay access to plausible answers and incur unnecessary token costs even when the model is able to identify an answer before extended thinking, a behavior known as performative reasoning. In this paper, we introduce COPT, a reformulated reasoning pipeline that reverses the usual order of thinking and answering. Instead of thinking before answering, COPT first elicits a draft answer and then invokes subsequent on-policy thinking conditioned on its own draft answer for reflection and correction. To assess whether the draft answer should be trusted, COPT recasts continuous embeddings as inference-time contrastive verifiers. Specifically, it contrasts the model's support for the same generated tokens under discrete-token inputs and continuous-embedding inputs, yielding a sequence-level reverse KL estimator for answer reliability. Our analysis shows that under certain assumptions, the expected estimate equals the mutual information between the unresolved latent state and the emitted answer token, explaining why it captures answer-relevant uncertainty rather than arbitrary uncertainty in the latent state. When the answer is deemed insufficiently reliable, COPT performs further on-policy thinking, where a second KL estimator dynamically controls draft-answer visibility, preserving useful partial information while reducing the risk of being misled by unreliable content. Across mathematics, coding, commonsense, and agentic reasoning tasks, COPT improves accuracy by up to 14% and reduces token usage by up to 57% at comparable or higher accuracy, without any additional training. We provide the code with the paper.

1 INTRODUCTION

Reasoning has become one of the central capabilities of large language models (LLMs), enabling them to solve increasingly complex tasks in mathematics (Google DeepMind, 2024; Jaech et al., 2024; OpenAI, 2025; Team et al., 2025), coding (Cao et al., 2026; Hui et al., 2024; Zhu et al.,

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.

2024; Roziere et al., 2023), and agentic (Anthropic, 2025b;a; Qwen Team, 2026; Patil et al., 2024) settings. A common approach for eliciting reasoning behavior is chain-of-thought (CoT), where LLMs generate intermediate natural-language steps before producing the final answer (Wei et al., 2022; Yao et al., 2023; Goyal et al., 2024; Pfau et al., 2024; Qwen Team, 2024; 2025). By making the thinking process explicit, CoT brings substantial improvements to complex tasks that demand advanced reasoning capabilities (Yang et al., 2025; Meta AI, 2025a;b; Guo et al., 2025; Agarwal et al., 2025; Abdin et al., 2025; Shi et al., 2025; Abouelenin et al., 2025).

A key limitation of the CoT paradigm is that it treats thinking as a prerequisite for answering. It works by first producing a thorough reasoning trace and only then arriving at the answer. However, recent work has revealed that, for many queries, LLMs exhibit performative reasoning (Boppana et al., 2026; Huang et al., 2026; Lindsey, 2026; Chen et al., 2025b), in which they insist on completing the reasoning process even when they have already internally identified a plausible answer.

We propose CoPT, a reversed reasoning paradigm. Rather than thinking before answering, an LLM first drafts an answer and then performs thinking for reflection and correction afterward. This reformulated paradigm provides earlier access to answers and avoids unnecessary token consumption when the model is able to identify a plausible answer before thorough thinking.

Reversing the usual order of thinking and answering raises two key challenges: when a draft answer should be trusted, and how it should be used during later thinking. We show that continuous embeddings, previously used for generation in latent CoT methods (Hao et al., 2024; Xu et al., 2025), can be recast as inference-time verifiers for this reversed reasoning setting. By contrasting the model's support for the same generated tokens under discrete-token and continuous-embedding inputs, they provide measurable criteria for draft reliability estimation and controlled utilization.

Latent CoT, where LLMs generate continuous embeddings instead of committing to discrete tokens (Hao et al., 2024; Shen et al., 2025; Zhu et al., 2025b; Xu et al., 2025; Tan et al., 2025), is a distinct line of work in parallel to explicit CoT. They are motivated by the observation that latent CoT offers higher representational bandwidth (Zhu et al., 2025c; Yu et al., 2026). Continuous embeddings can encode richer information by preserving uncertainty, whereas discrete tokens retain only the information carried by the sampled token at each step (Li et al., 2025; Chen et al., 2025a).

Instead of using continuous embeddings for generation during thinking in existing latent reasoning methods, CoPT keeps thinking explicit while recasting continuous embeddings as contrastive verifiers at inference. This allows it to retain the readability of explicit CoT while simultaneously leveraging uncertainty information, as in latent CoT. Meanwhile, it avoids issues that may arise when continuous embeddings are directly used for generation, such as unseen representations (Zhang et al., 2025), de-diversification (Wu et al., 2025), and drifting into noise (Shi et al., 2026).

To address the first challenge of determining when a draft answer should be trusted, CoPT introduces a contrastive mechanism with continuous spaces to estimate the reliability of the draft answer. Specifically, it contrasts the model's support for its own generated answer under two types of input representations: explicit inputs in discrete spaces and continuous embeddings constructed from next-token distributions and cached online along with explicit token generation. This contrast yields a sequence-level reverse KL estimator that indicates the reliability of the draft answer. If the draft answer appears sufficiently reliable, it will be accepted by the model directly. Otherwise, CoPT triggers a subsequent on-policy thinking process to either correct or support the answer.

The second challenge of how to use the draft answer arises once on-policy thinking is triggered. A draft answer deemed insufficiently reliable may still contain useful partial information, but exposing it throughout the entire later thinking process risks misleading the model. To control the visibility of the draft answer during on-policy thinking across thinking steps, CoPT periodically calculates a second KL estimator within each thinking chunk using a similar contrastive mechanism with continuous spaces. In this way, CoPT allows the model to use the draft answer when it appears helpful, while hiding it when the current thinking process becomes unstable.

Beyond the empirical results, we further provide a latent-state interpretation of the proposed contrastive estimator. Under a local mixture-prefix view, the continuous prefix preserves uncertainty over an unresolved latent reasoning state S, while the emitted answer token is denoted by A. We show that under a mixture-linear assumption (see Section 4), the expected estimate equals the mutual information $I(S; A)$, indicating that the estimator measures answer-relevant uncertainty rather

than the entropy of the latent state itself. This explains why the score grows only when uncertainty preserved by continuous embeddings changes the model's support for its own generated tokens, supporting its use for draft reliability estimation. Our contributions are summarized as follows:

- We propose CoPT, a training-free reasoning pipeline that enables LLMs to start with a draft answer and invoke on-policy thinking conditioned on it when necessary, thereby allowing earlier access to answers and selective correction afterward.
- We introduce a contrastive mechanism that measures the discrepancy between the model's support for the same generated tokens under discrete and continuous inputs, which helps identify potential errors in draft answers and modulates their exposure during the thinking process.
- We extensively validate the effectiveness of CoPT on mathematics, coding, and agentic reasoning tasks across multiple benchmarks, model architectures, and scales, demonstrating consistent gains over CoT baselines in both accuracy and token efficiency.

2 Related Work

Reasoning LLMs and explicit reasoning. Reasoning with explicit natural-language traces has become a standard way to improve the performance of LLMs on complex tasks (OpenAI, 2025; Anthropic, 2025b; Comanici et al., 2025). Early work elicits such behavior through prompting (Wei et al., 2022; Wang et al., 2022; Yao et al., 2023). More recent LLMs typically gain reasoning capabilities through reinforcement learning (Shao et al., 2024; Yu et al., 2025; Liu et al., 2025b) or multi-stage post-training that combines supervised fine-tuning with reinforcement learning (Liu et al., 2024; Shi et al., 2024; Yang et al., 2025; Ma et al., 2025; Yuan et al., 2025a;b). Representative open-source examples include DeepSeek-R1 (Guo et al., 2025) and Qwen3 (Yang et al., 2025), which show that large-scale RL and long-CoT post-training can elicit strong reasoning behaviors. Following works (Zeng et al., 2025a; Liu et al., 2025a; Yuan et al., 2026; Cao et al., 2026) further demonstrate the effectiveness of explicit reasoning across diverse mathematics, coding, and agentic tasks. Despite these advances, reasoning LLMs typically retain the standard thinking-before-answering order. In contrast, CoPT reverses this order by first eliciting a draft answer and invoking on-policy thinking conditioned on it when the answer appears insufficiently reliable.

Latent reasoning with continuous embeddings. A parallel line of work explores latent reasoning in continuous spaces, where LLMs operate on continuous embeddings instead of committing to discrete tokens (Hao et al., 2024; Su et al., 2025; Zhu et al., 2025b). These methods are motivated by the observation that continuous representations can encode information from the full next-token distribution, while discrete decoding retains only the sampled token. Latent reasoning is mainly achieved by adapting LLMs into continuous spaces via modified pretraining (Zeng et al., 2025b; Tack et al., 2025) or fine-tuning (Shen et al., 2025; Xu et al., 2025; Tan et al., 2025; Wei et al., 2025; Zhu et al., 2025a; Xia et al., 2026) objectives. Recent training-free methods (Wu et al., 2025; Xu et al., 2026) instead construct continuous embeddings directly during inference, such as Soft-Thinking (Zhang et al., 2025) and SwiReasoning (Shi et al., 2026). These prior latent reasoning methods mainly use continuous embeddings as a medium for generation. In contrast, CoPT recasts them as inference-time verifiers. This allows CoPT to use uncertainty information preserved by continuous embeddings as in latent reasoning while retaining the readability of explicit reasoning.

3 Methodology

As shown in Fig. 2, CoPT reformulates LLM reasoning into two reversed stages: a leading *draft-answer stage* and, when necessary, a trailing *on-policy thinking stage*. The key insight is to first elicit an early-stage answer at low cost, estimate its reliability with a normalized sequence-level reverse KL estimator, and selectively trigger on-policy thinking with dynamic access to the draft answer.

3.1 Reliability Estimator of the Draft Answer

Draft answer elicitation. Let p_θ denote the model with parameters θ. Let $E \in \mathbb{R}^{|\mathcal{V}| \times d}$ be the input embedding matrix of the model, where $\mathcal{V}$ is the vocabulary and d is the hidden size. For any token $v \in \mathcal{V}$, $E(v) \in \mathbb{R}^d$ denotes its embedding. Given a question token sequence $q = (q_1, \ldots, q_m)$,

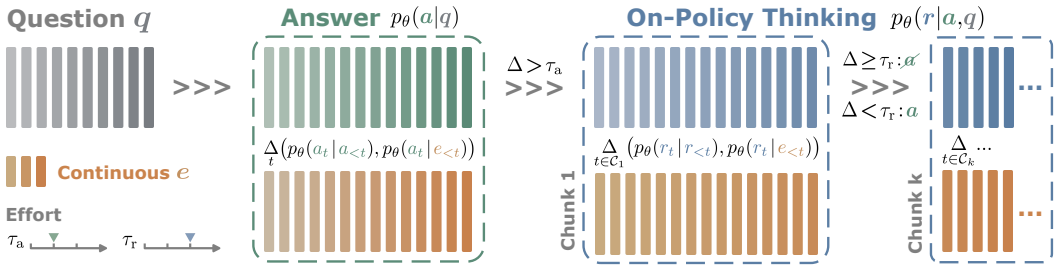


Figure 2: CopT starts with a draft answer and performs on-policy thinking conditioned on it. It contrasts the model's support for the same chosen tokens under discrete and continuous inputs to estimate draft reliability, and during thinking, chunk by chunk, to determine the visibility of the draft answer across time steps.

instead of allowing the model to think thoroughly, we force it to output `</think>` at the beginning and go straight into its answering mode.

Reliability estimation. To estimate how likely it is that a subsequent thorough thinking process will be required for correcting potential errors, we introduce a normalized sequence-level reverse KL estimator κ_{a}. Let the draft phase generate tokens $a = (a_1, \ldots, a_{T_a})$. During draft answer generation, for each generated token a_t, we cache two items calculated from the next-token distributions:

$$p_t := p_\theta(a_t \mid q, a_{<t}), \qquad e_t := \sum_{v\in\mathcal{V}} p_\theta(v \mid q, a_{<t}) E(v).$$

Here p_t is the chosen-token probability, and e_t is a continuous embedding obtained as the probability-weighted average over the vocabulary, which preserves uncertainty information per step.

After the draft answer a is completed, we calculate κ_{a} to estimate its reliability. More specifically, we compare the student discrete-prefix distribution induced by the original inputs against the teacher continuous-prefix distribution in which inputs are replaced with cached continuous embeddings.

For $t \in \{1, 2, \ldots T_a\}$, the student probability is simply p_t, and all the teacher probabilities are obtained in parallel using a single forward pass with the modified input embeddings. The teacher probabilities of the original answer tokens are gathered at the corresponding output positions:

$$p_t^e := p_\theta(a_t \mid q, e_{<t}).$$

This defines the continuous-prefix probability of the sampled draft answer as $p_\theta^e(a|q) = \prod_{t=1}^{T_a} p_t^e$.

We define the estimator

$$\kappa_{\mathrm{a}}(a_{1:T_a}) := \frac{1}{T_a} \sum_{t=1}^{T_a} \left[\log p_\theta(a_t \mid q, a_{<t}) - \log p_\theta(a_t \mid q, e_{<t})\right].$$

For any fixed draft length T_a, κ_{a} is an unbiased estimator of the normalized sequence-level reverse KL divergence between the two distributions:

$$\mathbb{E}_{a_{1:T_a}\sim p_\theta(\cdot|q)}\left[\kappa_{\mathrm{a}}(a_{1:T_a})\right] = \frac{1}{T_a} D_{\mathrm{KL}}(p_\theta(\cdot \mid q) \,\|\, p_\theta^e(\cdot \mid q)).$$

3.2 On-Policy Thinking Conditioned on the Draft Answer

On-policy thinking elicitation. A large κ_{a} indicates that answer context becomes substantially less supported with teacher-forced continuous embeddings, *i.e.*, the answer may be unreliable given additional uncertainty information. Let τ_{a} denote the reliability threshold. When $\kappa_{\mathrm{a}} > \tau_{\mathrm{a}}$, we force the model to output `<think>` after the draft answer and move into a subsequent thinking process.

Visibility controls for draft answers. For draft answers that are deemed insufficiently reliable, the goal of on-policy thinking is to use any beneficial information when necessary, while avoiding being misled by unreliable draft content. Let the on-policy thinking phase generate tokens

$$r = (r_{T_a+1}, r_{T_a+2}, \ldots, r_{T_a+T_r}).$$

We partition the thinking trajectory into chunks of length C. Let the k-th chunk start at position

$$s_k := T_a + 1 + (k-1)C,$$

and span positions $[s_k, s_k + C - 1]$. Let $m_k := \mathbb{1}\{a \text{ is visible in chunk } k\}$ denote the visibility mask for the k-th chunk, and define the visibility-conditioned draft input as

$$a^{(m_k)} := \begin{cases} a, & m_k = 1, \\ \varnothing, & m_k = 0. \end{cases}$$

For each generated token r_t in chunk k, we cache

$$p_t := p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:t-1}), \qquad e_t := \sum_{v \in \mathcal{V}} p_\theta(v \mid q, a^{(m_k)}, r_{T_a+1:t-1}) E(v).$$

Similarly, we calculate a second KL estimator $\kappa_{\mathrm{r}}^{(k)}$ on the current chunk whenever it reaches a predefined length C to decide whether the previous draft answer should become visible in the next chunk. [1] For $t \in \{s_k, \ldots, s_k + C - 1\}$, the student chosen-token probability is simply p_t, and all the teacher probabilities within the chunk are obtained in parallel using a single forward pass with the modified intra-chunk input embeddings:

$$p_t^e := p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}; e_{s_k:t-1}).$$

We define the estimator

$$\kappa_{\mathrm{r}}^{(k)}(r_{s_k:s_k+C-1}) := \frac{1}{C} \sum_{t=s_k}^{s_k+C-1} \big[\log p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:t-1}) - \log p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}; e_{s_k:t-1})\big].$$

For a fixed chunk length C, $\kappa_{\mathrm{r}}^{(k)}$ is an unbiased estimator of the normalized sequence-level reverse KL between the two chunk-level continuation distributions:

$$\mathbb{E}_{r_{s_k:s_k+C-1} \sim p_\theta(\cdot \mid q, a^{(m_k)}, r_{T_a+1:s_k-1})} \big[\kappa_{\mathrm{r}}^{(k)}(r_{s_k:s_k+C-1})\big] = \frac{1}{C} D_{\mathrm{KL}}\Big(p_\theta(\cdot \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}) \,\Big\|\, p_\theta^e(\cdot \mid q, a^{(m_k)}, r_{T_a+1:s_k-1})\Big).$$

$\kappa_{\mathrm{r}}^{(k)}$ estimates the reliability of the current chunk. A large $\kappa_{\mathrm{r}}^{(k)}$ suggests that the current chunk is unstable and more vulnerable to misleading information. Let τ_{r} denote the stability threshold. After each complete chunk, we update the visibility of the draft answer for the next chunk by

$$m_{k+1} = \begin{cases} 1, & \kappa_{\mathrm{r}}^{(k)} < \tau_{\mathrm{r}}, \\ 0, & \text{otherwise.} \end{cases}$$

4 Theoretical Analysis

In this section, we provide a theoretical interpretation to demonstrate the effectiveness of our CoPT method under certain assumptions. We focus on the reliability of our proposed reverse-KL estimator. Our analysis highlights a key property of the reverse-KL estimator: it measures *answer-relevant uncertainty*, rather than uncertainty over latent reasoning states themselves.

For convenience, we analyze a single answer position. Note that all probability distributions below are conditioned on the question (or equivalently, the prompt) q and the previous output prefix, which we omit when the context is clear. Let $\mathcal{S}$ be a finite set of latent reasoning states. A discrete output prefix (along with the prompt) commits the model to one latent state, while a continuous prefix may represent a superposition of several possible states.

[1] κ_{a} and κ_{r} are calculated on the already generated sequence, and therefore incur only small overhead once the corresponding chosen-token probabilities and continuous embeddings are cached online during generation.

Let $\mathcal{A}$ be a finite set of all possible answers (or equivalently, the next token). When the prefix is discrete, for each latent state $s \in \mathcal{S}$, let

$$P_s(a) = p_\theta(a \mid s), \quad \forall a \in \mathcal{A}$$

denote the next-token distribution induced by committing to s, where θ denotes the model parameters. When the prefix is continuous, we make the following assumption on the output distribution.

Assumption 1 (Mixture-linear continuous prefix)**.** *Let w be a distribution over $\mathcal{S}$ such that the discrete draft prefix commits to a latent state $S \sim w$, and then emits the answer $A \sim P_S$. Let e_w denote the corresponding continuous prefix which is determined by the distribution w. We assume the next-token distribution conditioned on a continuous prefix e_w is determined by*

$$p_\theta(a \mid e_w) = \bar{P}_w(a) := \sum_{s \in \mathcal{S}} w(s) P_s(a), \quad \forall a \in \mathcal{A}.$$

Note that for the emitted answer token A, the local reverse-KL contribution is

$$\kappa(S, A) = \log P_S(A) - \log \bar{P}_w(A).$$

We provide an empirical examination of Assumption 1 in Appendix B.8.

Theorem 1 (Reverse KL measures answer-relevant uncertainty)**.** *Under Assumption 1,*

$$\mathbb{E}_{S \sim w,\, A \sim P_S}[\kappa(S, A)] = \sum_{s \in \mathcal{S}} w(s) D_{\mathrm{KL}}\big(P_s \,\big\|\, \bar{P}_w\big) = I(S; A),$$

where $I(S; A)$ is the mutual information between the latent state S and the emitted answer token A under the joint distribution

$$\Pr(S = s, A = a) = w(s) P_s(a).$$

The proof is in Appendix Section D. Theorem 1 shows that CoPT does not penalize latent-state uncertainty by itself. Instead, it measures whether that uncertainty changes the next answer-token distribution. For example, the continuous prefix may represent a mixture over several possible states, $S \in \{s_1, s_2, s_3\}$, which can have high entropy. However, if all states induce the same next answer token or next-token distribution, then the emitted token A carries no information about which state was selected. In that case, $I(S; A) = 0$, and the expected reverse-KL contribution is zero. Thus, high uncertainty over latent states is harmless when all plausible states agree on the next answer.

Applying this argument token by token, if the mixture-prefix assumption holds at each answer position t, then the normalized draft score satisfies

$$\mathbb{E}[\kappa_{\mathrm{a}}] = \frac{1}{T_a} \sum_{t=1}^{T_a} I(S_t; A_t),$$

which is conditioned on the preceding context at each position. Therefore, κ_{a} estimates the average amount of answer-relevant uncertainty in the draft answer.

5 Experiments

5.1 Experimental Settings

Models. We evaluate CoPT on pure Transformer-based Qwen3 models (Yang et al., 2025) and hybrid Gated-DeltaNet Qwen3.5 models (Qwen Team, 2026) at 2B, 8B, and 35B scales. This selection allows us to validate the effectiveness of CoPT across model families, scales, and architectures, including pure Transformer, hybrid, dense, and sparse mixture-of-experts models.

Domains and Benchmarks. We evaluate CoPT on 10 benchmarks spanning four domains: math and STEM reasoning (GSM8K (Cobbe et al., 2021), Math500 (Hendrycks et al., 2021), AIME 2024 (HuggingFaceH4, 2024), AIME 2025 (Yentinglin, 2025), GPQA Diamond (Rein et al., 2024)); coding reasoning (HumanEval (Chen et al., 2021), LeetCode-Contest (Guo et al., 2024)); commonsense reasoning (CommonsenseQA (Talmor et al., 2019)); single-turn and multi-turn agentic reasoning (BFCL v4 (Patil et al., 2025), ZebraArena (Zhao et al., 2026)). More benchmark details are provided in Appendix C.1 and hyperparameter details are provided in Appendix C.2.

Table 1: Comparison on mathematics, coding, commonsense, and STEM reasoning with the Qwen3-8B model. CopT improves accuracy by $+1.67$ points on average while reducing tokens by 15.1%.

Mathematics	**GSM8K**		**Math500**		**AIME24**		**AIME25**	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	95.75	2138	96.00	4985	75.83	12077	67.50	12924
CoT (Greedy)	95.83	2240	96.40	5311	70.00	11680	60.00	13292
CopT (Ours)	95.91 (+0.16)	1636 (-23.5%)	96.80 (+0.80)	4428 (-11.2%)	76.67 (+0.84)	11367 (-5.9%)	68.75 (+1.25)	12508 (-3.2%)
Coding, Commonsense & STEM	**HumanEval**		**LeetCode-Contest**		**CommonsenseQA**		**GPQA Diamond**	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	92.68	2368	43.18	13350	83.95	1083	59.60	8123
CoT (Greedy)	93.90	2627	47.73	12670	84.03	1113	56.57	7909
CopT (Ours)	94.51 (+1.83)	1023 (-56.8%)	50.00 (+6.82)	12690 (-4.9%)	85.09 (+1.14)	1051 (-3.0%)	60.10 (+0.50)	7115 (-12.4%)

5.2 Main Results on Mathematics, Coding, Commonsense, and STEM Reasoning

Tab. 1 reports accuracy and token usage on mathematics, coding, commonsense, and STEM reasoning benchmarks with Qwen3-8B. Across all eight benchmarks, CopT achieves higher accuracy than both standard CoT and greedy CoT, while consistently using fewer tokens than standard CoT, demonstrating improved accuracy-efficiency trade-offs across diverse reasoning domains.

On mathematical reasoning, CopT improves GSM8K accuracy by $+0.16\%$ while reducing generation length by 23.5%, and improves Math500 accuracy by $+0.80\%$ with 11.2% fewer tokens. On the more challenging AIME benchmarks, CopT improves AIME24 by $+0.84\%$ and AIME25 by $+1.25\%$, while still reducing generation length by 5.9% and 3.2%, respectively. This suggests that CopT can substantially reduce unnecessary thinking on relatively easier mathematical problems while preserving sufficient reasoning effort for harder ones. The same trend extends beyond mathematics. On HumanEval, CopT improves accuracy by $+1.83\%$ while reducing generation length by 56.8%, yielding the largest token reduction among the evaluated benchmarks. On the more challenging LeetCode-Contest benchmark, CopT achieves the largest accuracy gain of $+6.82\%$ while still using 4.9% fewer tokens. CopT also improves CommonsenseQA by $+1.14\%$ with 3.0% fewer tokens, and GPQA Diamond by $+0.50\%$ with 12.4% fewer tokens.

Overall, the results show two complementary benefits of the draft-first reasoning paradigm. When a draft answer is sufficiently reliable, CopT can avoid unnecessary extended thinking and substantially reduce generation length. When additional reasoning is needed, CopT can selectively invoke on-policy thinking to refine the draft, yielding larger accuracy gains on more challenging tasks.

5.3 Comparison with Continuous-Generation and Early-Truncation Methods

Tab. 2 first compares CopT with training-free methods that directly use continuous embeddings for generation, Soft-Thinking (Zhang et al., 2025) and SwiReasoning (Shi et al., 2026), with the Qwen3-8B model. Compared with Soft-Thinking, CopT improves average accuracy by $+3.36$ percentage points while using 9.6% fewer tokens. Compared with SwiReasoning, CopT achieves comparable average accuracy while reducing token usage by 10.7%. Importantly, these methods directly use continuous embeddings as the generation medium, such that their intermediate reasoning is not fully represented in natural language. In contrast, CopT allows users to inspect the complete reasoning process and uses continuous embeddings only as inference-time verifiers while still benefiting from uncertainty information preserved in continuous embeddings.

We further compare with early-truncation methods, including DEER (Yang et al., 2026), EAT (Hosseini et al., 2026), and CODE-Stop (Wang et al., 2026). To enable comparison at similar or lower token budgets, we set the visibility threshold to $\tau_r = 1.0$ for CopT in this comparison while keeping all other settings unchanged. CopT achieves the highest average accuracy of 85.07%, compared with 83.17%, 83.18%, and 83.88%, respectively, while also using the fewest tokens on average. Early-truncation methods decide whether an ongoing thinking-first CoT trajectory can be stopped, while CopT first produces an explicit draft answer and invokes additional thinking only when that

Table 2: Comparison with training-free continuous-generation and early-truncation methods.

Method	GSM8K		Math500		AIME24		Average	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
w/ continuous-generation methods (× fully explicit readability)								
Soft-Thinking	95.38	2073	96.00	4934	67.92	12271	86.43	6426
SwiReasoning	96.06	2218	96.80	4807	79.17	12491	90.68	6505
CopT (Ours)	95.91 (+0.53)	1636 (-21.1%)	96.80 (+0.80)	4428 (-10.3%)	76.67 (+8.75)	11367 (-7.4%)	89.79 (+3.36)	5810 (-9.6%)
w/ early-truncation methods								
DEER	95.20	1071	92.60	2732	61.70	8796	83.17	4200
EAT	94.77	1098	90.20	3407	64.58	8120	83.18	4208
CODE-Stop	92.20	1059	93.60	3246	65.83	8162	83.88	4156
CopT (Ours)	95.38 (+0.18)	956 (-10.7%)	94.00 (+1.40)	2589 (-5.2%)	65.83 (+4.13)	7926 (-9.9%)	85.07 (+1.90)	3824 (-9.0%)

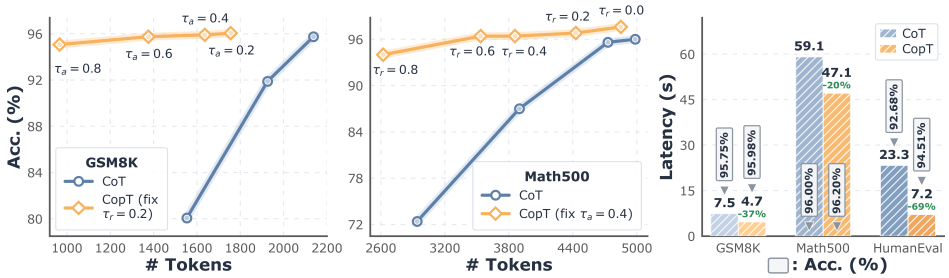


Figure 3: Left and center: Controllable reasoning effort by τ_a and τ_r. Right: CopT reduces average per-sample latency at comparable or higher accuracy, measured on a single H200 GPU.

draft is insufficiently reliable. The subsequent on-policy thinking is conditioned on the draft answer, which can be retained, revised, or rejected, with dynamical visibility control during correction.

5.4 Controllable Reasoning Effort and Latency Reduction

Fig. 3 studies how CopT controls reasoning effort and how token savings translate into real latency reduction. The left and center panels show the accuracy-token trade-off obtained by sweeping τ_a and τ_r. These curves show that CopT can reduce generation length when less thinking is needed, while allocating more on-policy thinking to obtain higher accuracy when the thresholds trigger stronger verification and correction. For simplicity, we use a fixed visibility threshold of $\tau_r = 0.2$ for reasoning effort control across all 10 benchmarks. The right panel reports average latency per sample at comparable or higher accuracy. The reductions are consistent with the token-efficiency gains: CopT reduces latency by 37% on GSM8K, 20% on Math500, and 69% on HumanEval. These results show that CopT reduces actual generation latency while preserving reasoning accuracy.

5.5 Experimental Results on Agentic Reasoning

Tab. 3 evaluates CopT on agentic reasoning benchmarks, including the non-live and live splits of BFCL v4 and multi-turn ZebraArena with one missing clue. We use Qwen3.5 models for these evaluations given their stronger support for agentic and tool-use settings (Qwen Team, 2026). On BFCL v4, CopT maintains slightly higher accuracy while reducing generation length by roughly half across model scales. Specifically, CopT improves accuracy by $+0.24$ points with 51.3% fewer tokens on Qwen3.5-2B, and by $+0.16$ points with 51.1% fewer tokens on Qwen3.5-35B-A3B.

The gains become substantially larger on the multi-turn ZebraArena benchmark. CopT improves accuracy by $+1.32$, $+14.41$, and $+27.63$ points on the small, medium, and large splits, respectively. At the same time, it reduces generation length by 0.9%, 29.6%, and 22.4%. The improvement grows markedly with task difficulty, suggesting that the benefit can accumulate across multiple rounds of interaction: reliable intermediate drafts can avoid unnecessary thinking, while unreliable drafts can trigger additional on-policy reasoning and correction before errors propagate to later turns.

Table 3: Comparison on agentic reasoning BFCL v4 (single-turn) and ZebraArena (multi-turn).

Method	Qwen3.5-2B		Qwen3.5-35B-A3B							
	BFCL v4 (non-live and live)		BFCL v4 (non-live and live)		ZebraArena (multi-turn)					
					Small		Medium		Large	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	77.53	234	85.77	235	93.71	3357	75.00	7217	59.21	8070
CopT (Ours)	77.77 (+0.24)	114 (-51.3%)	85.93 (+0.16)	115 (-51.1%)	95.03 (+1.32)	3326 (-0.9%)	89.41 (+14.41)	5082 (-29.6%)	86.84 (+27.63)	6262 (-22.4%)

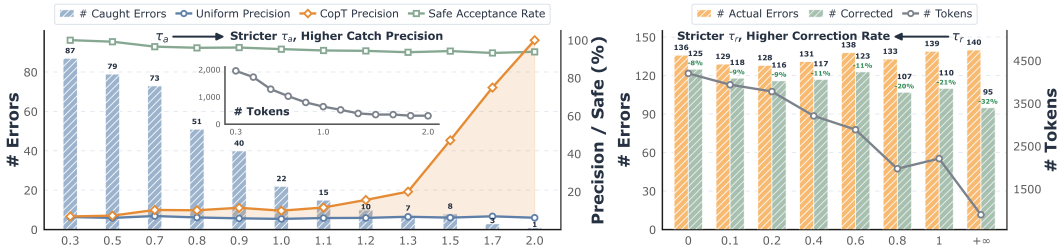


Figure 4: Left: The estimator κ_a identifies erroneous drafts more precisely than uniform selection. Right: The threshold τ_r trades off correction rate and token usage by controlling draft visibility.

5.6 Ablation Studies on Design Choices

Draft answer reliability estimation. We examine whether the draft-answer reliability estimator κ_a can effectively identify unreliable draft answers on GSM8K. Fig. 4 (left) compares CopT with a uniform selection under different thresholds τ_a. For each threshold, we report the number of caught errors, the precision defined as the fraction of presumed errors that are truly erroneous, and the safe acceptance rate defined as the fraction of directly accepted answers that are correct. As τ_a becomes stricter, CopT selects fewer draft answers as unreliable, but the selected subset becomes increasingly concentrated on truly erroneous drafts. In contrast, uniform selection yields a much lower and nearly flat precision, indicating that uniform allocation of additional thinking fails to distinguish unreliable answers from reliable ones. These results confirm that κ_a provides a meaningful reliability estimate.

Visibility control for draft answers. We study the effect of the visibility estimator κ_r during on-policy thinking on Math500. Fig. 4 (right) varies the threshold τ_r and reports the number of actual draft errors and the number of errors that are successfully corrected. Smaller τ_r corresponds to stricter visibility control, exposing the draft answer less often. As τ_r becomes stricter, CopT increases reasoning effort with less reliance on the potentially wrong draft answer, and corrects a larger fraction of erroneous draft answers. These results show that the estimator κ_r plays a direct role in reflection and correction. If the draft answer is always exposed, unreliable content may influence the on-policy thinking process and prevent the model from correcting its initial mistake. By contrast, dynamic visibility control allows CopT to hide the draft answer when the current thinking chunk appears unstable, while still allowing access to the draft when it provides useful partial information.

6 Conclusion

This paper introduces CopT, a training-free LLM reasoning pipeline that reverses the usual order of thinking and answering. CopT first elicits a draft answer and then invokes on-policy thinking conditioned on the draft when it appears unreliable. CopT recasts continuous embeddings, previously used for generation in latent CoT methods, as inference-time verifiers by contrasting the model's support for the same generated tokens under discrete-token and continuous-embedding inputs. Experiments on math, coding, and agentic reasoning tasks show that CopT consistently improves both accuracy and token efficiency, suggesting that on-policy thinking with contrastive verification from continuous embeddings provides a practical path toward more cost-effective LLM reasoning.

AI USE STATEMENT

In this work, we used generative AI tools, including LLMs and coding agents, to assist with language polishing, improving clarity and readability, and code debugging. The core intellectual contributions of this work originate from the human authors, who formulated the research questions, developed the proposed method, designed the experiments, and analyzed the results. The authors retain final authority over all research claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper studies an inference-time method for reasoning LLMs. By improving accuracy and token efficiency, COPT may make reasoning LLMs more accessible and cost-effective for beneficial applications, such as scientific problem solving, programming assistance, and tool-using agents. At the same time, stronger and cheaper reasoning may also increase the usefulness of LLMs in harmful applications if deployed without proper safeguards. These risks are not specific to COPT, but follow from improving the capability of reasoning LLMs. COPT does not introduce a new class of societal risks beyond those associated with the underlying models.

REPRODUCIBILITY STATEMENT

This paper includes implementation details in Appendix C and empirical examination for Assumption 1 in Appendix B.8. The code is provided with the paper.

REFERENCES

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, Lingjiao Chen, Gustavo de Rosa, Suriya Gunasekar, Mojan Javaheripi, Neel Joshi, et al. Phi-4-reasoning technical report. *arXiv preprint arXiv:2504.21318*, 2025.

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*, 2025.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Anthropic. System card: Claude opus 4 & claude sonnet 4, 2025a. URL https://www.anthropic.com/claude-4-system-card.

Anthropic. Claude opus 4.5 system card. https://assets.anthropic.com/m/64823ba7485345a7/Claude-Opus-4-5-System-Card.pdf, November 2025b. System card.

Siddharth Boppana, Annabel Ma, Max Loeffler, Raphael Sarfati, Eric Bigelow, Atticus Geiger, Owen Lewis, and Jack Merullo. Reasoning theater: Disentangling model beliefs from chain-of-thought. *arXiv preprint arXiv:2603.05488*, 2026.

Ruisheng Cao, Mouxiang Chen, Jiawei Chen, Zeyu Cui, Yunlong Feng, Binyuan Hui, Yuheng Jing, Kaixin Li, Mingze Li, Junyang Lin, et al. Qwen3-coder-next technical report. *arXiv preprint arXiv:2603.00729*, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders,

Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

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*, 2025a.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, et al. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025b.

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.

Google DeepMind. Ai achieves silver-medal standard solving international mathematical olympiad problems. https://deepmind.google/discover/blog/ai-solves-imo-problems-at-silver-medal-level/, 2024.

Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *The Twelfth International Conference on Learning Representations*, 2024.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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.

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

Parsa Hosseini, Sumit Nawathe, Mahdi Salmani, Meisam Razaviyayn, and Soheil Feizi. Early stopping for large reasoning models via confidence dynamics. *arXiv preprint arXiv:2604.04930*, 2026.

Zixuan Huang, Xin Xia, Yuxi Ren, Jianbin Zheng, Xuanda Wang, Zhixia Zhang, Hongyan Xie, Songshi Liang, Zehao Chen, Xuefeng Xiao, et al. Does your reasoning model implicitly know when to stop thinking? *arXiv preprint arXiv:2602.08354*, 2026.

HuggingFaceH4. Aime 2024 (american invitational mathematics examination 2024). Hugging Face dataset, 2024. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, et al. Qwen2.5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

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*, 2025.

Jack Lindsey. Emergent introspective awareness in large language models. *arXiv preprint arXiv:2601.01828*, 2026.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 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*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, et al. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*, 2025.

Meta AI. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, 2025a.

Meta AI. Llama 3.3 model card. https://github.com/meta-llama/llama-models/blob/main/models/llama3_3/MODEL_CARD.md, 2025b.

OpenAI. Openai o3 and o4-mini, 2025. URL https://openai.com/index/introducing-o3-and-o4-mini.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37:126544–126565, 2024.

Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Jacob Pfau, William Merrill, and Samuel R Bowman. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*, 2024.

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

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. *arXiv preprint arXiv:2502.21074*, 2025.

Dachuan Shi, Chaofan Tao, Anyi Rao, Zhendong Yang, Chun Yuan, and Jiaqi Wang. CrossGET: Cross-guided ensemble of tokens for accelerating vision-language transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 44960–44990. PMLR, 2024.

Dachuan Shi, Yonggan Fu, Xiangchi Yuan, Zhongzhi Yu, Haoran You, Sixu Li, Xin Dong, Jan Kautz, Pavlo Molchanov, and Yingyan Celine Lin. LaCache: Ladder-shaped KV caching for efficient long-context modeling of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 54892–54903. PMLR, 2025.

Dachuan Shi, Abedelkadir Asi, Keying Li, Xiangchi Yuan, Leyan Pan, Wenke Lee, and Wen Xiao. Swireasoning: Switch-thinking in latent and explicit for pareto-superior reasoning llms. In *International Conference on Learning Representations*, volume 2026, pp. 137060–137093, 2026.

DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qinqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. *arXiv preprint arXiv:2502.03275*, 2025.

Jihoon Tack, Jack Lanchantin, Jane Yu, Andrew Cohen, Ilia Kulikov, Janice Lan, Shibo Hao, Yuandong Tian, Jason Weston, and Xian Li. Llm pretraining with continuous concepts. *arXiv preprint arXiv:2502.08524*, 2025.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 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: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, 2019.

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

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Xi Wang, James McInerney, Lequn Wang, and Nathan Kallus. Entropy after for reasoning model early exiting. *arXiv preprint arXiv:2509.26522*, 2026.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, and et al. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 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.

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, and Hua Wu. Llms are single-threaded reasoners: Demystifying the working mechanism of soft thinking. *arXiv preprint arXiv:2508.03440*, 2025.

Kejing Xia, Mingzhe Li, Lixuan Wei, Zhenbang Du, Xiangchi Yuan, Dachuan Shi, Qirui Jin, and Wenke Lee. Metastate: Persistent working memory enhances reasoning in discrete diffusion language models. *arXiv preprint arXiv:2603.01331*, 2026.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*, 2025.

Zhongxing Xu, Zhonghua Wang, Zhe Qian, Dachuan Shi, Feilong Tang, Ming Hu, Shiyan Su, Xiaocheng Zou, Wei Feng, Dwarikanath Mahapatra, et al. Thinking in uncertainty: Mitigating hallucinations in mlrms with latent entropy-aware decoding. *arXiv preprint arXiv:2603.13366*, 2026.

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.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weipinng Wang. Dynamic early exit in reasoning models. In *International Conference on Learning Representations*, volume 2026, pp. 88170–88210, 2026.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Yentinglin. Aime 2025 (american invitational mathematics examination 2025). Hugging Face dataset, 2025. URL https://huggingface.co/datasets/yentinglin/aime_2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Xinlei Yu, Zhangquan Chen, Yongbo He, Tianyu Fu, Cheng Yang, Chengming Xu, Yue Ma, Xiaobin Hu, Zhe Cao, Jie Xu, et al. The latent space: Foundation, evolution, mechanism, ability, and outlook. *arXiv preprint arXiv:2604.02029*, 2026.

Xiangchi Yuan, Xiang Chen, Tong Yu, Dachuan Shi, Can Jin, Wenke Lee, and Saayan Mitra. Mitigating forgetting between supervised and reinforcement learning yields stronger reasoners. *arXiv preprint arXiv:2510.04454*, 2025a.

Xiangchi Yuan, Chunhui Zhang, Zheyuan Liu, Dachuan Shi, Soroush Vosoughi, and Wenke Lee. Superficial self-improved reasoners benefit from model merging. *arXiv preprint arXiv:2503.02103*, 2025b.

Xiangchi Yuan, Dachuan Shi, Chunhui Zhang, Zheyuan Liu, Shenglong Yao, Soroush Vosoughi, and Wenke Lee. Behavior knowledge merge in reinforced agentic models. *arXiv preprint arXiv:2601.13572*, 2026.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025a.

Boyi Zeng, Shixiang Song, Siyuan Huang, Yixuan Wang, He Li, Ziwei He, Xinbing Wang, Zhiyu Li, and Zhouhan Lin. Pretraining language models to ponder in continuous space, 2025b.

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

Wanjia Zhao, Ludwig Schmidt, James Zou, Vidhisha Balachandran, and Lingjiao Chen. Zebraarena: A diagnostic simulation environment for studying reasoning-action coupling in tool-augmented llms. *arXiv preprint arXiv:2603.18614*, 2026.

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Emergence of superposition: Unveiling the training dynamics of chain of continuous thought. *arXiv preprint arXiv:2509.23365*, 2025a.

Hanlin Zhu, Shibo Hao, Zhiting Hu, Jiantao Jiao, Stuart Russell, and Yuandong Tian. Reasoning by superposition: A theoretical perspective on chain of continuous thought. *arXiv preprint arXiv:2505.12514*, 2025b.

Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*, 2024.

Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfa Huang, Dawei Zhu, Hao Wang, Kaiwen Xue, Xuanliang Zhang, Yong Shan, et al. A survey on latent reasoning. *arXiv preprint arXiv:2507.06203*, 2025c.

CONTENTS OF APPENDIX

A Limitations **17**

B Supplementary Experiments **17**

B.1 Peak Accuracy with Draft-Visibility Control 17
B.2 Detailed Results on BFCL v4 Benchmark 17
B.3 Ablation on Draft-Answer Visibility 18
B.4 Ablation on Selective On-Policy Thinking 18
B.5 Ablation on Alternative Confidence Estimators 19
B.6 Ablation on Answer-Content Granularity 19
B.7 Wall-Clock Speedups and Verification Overhead 20
B.8 Empirical Examination of the Mixture-Linearity Assumption 20

C Supplementary Details **22**

C.1 Benchmark Details 22
C.2 Implementation Details 23
C.3 Broader Related Work 23
C.4 Sequence Distributions for the KL Estimators 23

D Additional Derivations for Answer-Relevant Uncertainty **24**

E Case Studies Comparing CoT and CopT **26**

A LIMITATIONS

While CoPT reformulates LLM reasoning in a training-free manner, its current design has a few limitations. First, the proposed KL estimators are evaluated on the realized trajectory available at inference time, rather than averaged over multiple continuations sampled from the model distribution. This matches our per-instance reasoning control setting, but may lead to higher variance than multi-sample estimates. We mitigate this by averaging over tokens, while future work may study more adaptive or lower-variance estimators. Second, CoPT requires access to next-token probabilities. This is natural for open-weight models and inference systems that expose logits, but may be less convenient for closed APIs that only return text outputs. Future work may explore API-compatible variants that approximate the contrastive reliability using limited observable information.

B SUPPLEMENTARY EXPERIMENTS

B.1 PEAK ACCURACY WITH DRAFT-VISIBILITY CONTROL

Table 4: Peak-accuracy results corresponding to Fig. 1(c). CopT uses the default visibility threshold $\tau_r = 0.2$, while CopT* changes only τ_r to 0.

Method	Math500		HumanEval		BFCL v4	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	96.00	4985	92.68	2368	85.77	235
CopT	96.80	4428	94.51	1023	85.93	115
CopT*	97.60 (+1.60)	4851 (-2.7%)	95.73 (+3.05)	1462 (-38.3%)	86.21 (+0.44)	118 (-49.8%)

Fig. 1(c) additionally reports the peak accuracy denoted as CopT*. While the main results use a shared visibility threshold of $\tau_r = 0.2$, CopT* simply changes this threshold to $\tau_r = 0$, with all other settings unchanged. This stricter visibility control places less reliance on the draft answer during subsequent on-policy thinking, providing higher accuracies.

As shown in Tab. 4, the stricter visibility setting further improves accuracy across mathematics, coding, and agentic reasoning. CopT* improves over standard CoT by $+1.60$, $+3.05$, and $+0.44$ points on Math500, HumanEval, and BFCL v4, respectively, while still using fewer tokens than CoT. These results complement the default reasoning effort setting and show that the visibility threshold provides a controllable trade-off between peak accuracy and generation efficiency.

B.2 DETAILED RESULTS ON BFCL V4 BENCHMARK

Table 5: Per-split accuracy comparison on the BFCL v4 benchmark.

Model	Method	Live		Non-Live		Overall	
		Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
Qwen3.5-2B	CoT	72.83	231	83.04	237	77.53	234
	CopT (Ours)	**72.91** (+0.08)	**121** (-47.6%)	**83.48** (+0.44)	**106** (-55.3%)	**77.77** (+0.24)	**114** (-51.3%)
Qwen3.5-35B-A3B	CoT	82.01	228	90.17	242	85.77	235
	CopT (Ours)	**82.24** (+0.23)	**123** (-46.1%)	**90.26** (+0.09)	**106** (-56.2%)	**85.93** (+0.16)	**115** (-51.1%)

Tab. 5 reports per-split results on the live and non-live splits of BFCL v4. Across both model scales, CoPT improves accuracy on both splits while reducing token usage. Overall accuracy increases by 0.24 and 0.16 percentage points for Qwen3.5-2B and Qwen3.5-35B-A3B, respectively, with 51.3% and 51.1% fewer tokens.

B.3 Ablation on Draft-Answer Visibility

We further examine how κ_r-based visibility control affects the correction of erroneous draft answers and the accuracy-token trade-off of on-policy thinking.

Correction of initially incorrect drafts. Fig. 4 (right) varies the visibility threshold τ_r under the same setup. We define the uncorrected-draft error rate as the fraction of initially incorrect drafts that remain incorrect after on-policy thinking. A larger τ_r exposes the draft more often, while $\tau_r = +\infty$ keeps it always visible. When the draft is always visible, 95 of the 140 initially incorrect drafts are corrected, leaving 45 incorrect. The correction rate is therefore 67.9%, and the uncorrected-draft error rate is $45/140 = 32.1\%$. At $\tau_r = 0$, 125 of 136 initially incorrect drafts are corrected. The correction rate increases to 91.9%, while the uncorrected-draft error rate decreases to $11/136 = 8.1\%$. Thus, stricter visibility control is associated with a 24.1 point increase in correction rate and a corresponding decrease in uncorrected-draft error rate.

Fixed versus dynamic visibility. Always hiding the draft is not necessarily the best accuracy-computation choice, since the draft may contain useful partial information. Tab. 6 compares the two fixed visibility settings with dynamic κ_r-based visibility.

Table 6: Accuracy and generation length under fixed and dynamic draft-answer visibility. Dynamic visibility balances error correction and reuse of useful draft information.

Draft visibility	Math500		HumanEval	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens
Always visible	90.80	1,546	90.85	665
Always hidden	96.40	4,734	**95.12**	1,833
Dynamic (κ_r)	**96.80**	4,428	94.51	1,023

Relative to always-visible thinking, dynamic visibility improves accuracy by 6.00 points on Math500 and 3.66 percentage points on HumanEval. Compared with always hiding the draft, dynamic visibility reduces generation length by 6.5% on Math500 and 44.2% on HumanEval, while achieving 0.40 percentage points higher accuracy on Math500 and only 0.61 percentage points lower accuracy on HumanEval.

These results show that exposing the draft is neither uniformly beneficial nor uniformly harmful. Unconditional exposure is associated with more uncorrected draft errors, while unconditional hiding prevents reuse of useful draft information and increases computation. Dynamic κ_r-based visibility provides an intermediate accuracy-token trade-off.

B.4 Ablation on Selective On-Policy Thinking

To examine the role of κ_a-based estimator, we compare CopT with a draft-first baseline that invokes answer-conditioned thinking after every draft. The comparison is a component ablation rather than a matched-token comparison: it tests whether answer-first generation followed by an unconditional second reasoning phase can reproduce the results of CopT.

Table 7: Comparison with a draft-first baseline that always invokes answer-conditioned thinking.

Method	GSM8K		AIME25	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens
CoT	95.75	2,138	67.50	12,924
Draft-first + always rethink	94.16	618	36.67	6,312
CopT (Ours)	**95.91**	1,636	**68.75**	12,508

As shown in Tab. 7, always rethinking achieves 94.16% accuracy on GSM8K, which is 1.59 percentage points below CoT and 1.75 percentage points below CopT. On the more challenging AIME25

benchmark, its accuracy drops to 36.67%, which is 30.83 percentage points below CoT and 32.08 percentage points below CoPT.

These results show that generating a draft and indiscriminately invoking another reasoning phase does not reproduce the performance of CoPT. The answer-first ordering is therefore not sufficient by itself. The comparison highlights the importance of the κ_{a}-based decision in determining when additional thinking is likely to be useful.

B.5 Ablation on Alternative Confidence Estimators

Table 8: Comparison of reliability estimators on Math500 under default- and low-effort settings.

Estimator	Default Effort		Low Effort ($\tau_r = 1.0$)		Average	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens	Acc. (%)	# Tokens
Probability confidence	96.0	4623	93.2	**2585**	94.60	3604
Entropy	96.2	4842	93.4	2803	94.80	3823
CopT (Ours)	**96.8**	**4428**	**94.0**	2589	**95.40**	**3509**

Tab. 8 compares the reliability estimator κ_a with confidence signals that are obtained directly from the draft logits, including answer-token probability confidence and token entropy. We evaluate all estimators under both the default reasoning-effort setting and a lower-effort setting with $\tau_r = 1.0$. Under the default setting, κ_a achieves the highest accuracy of 96.8% while also using fewer tokens than both probability confidence and entropy. The same trend holds in the low-effort regime, where κ_a reaches 94.0% accuracy at a similar token budget. Averaged across the two settings, κ_a achieves 95.40% accuracy with 3,509 tokens, compared with 94.60%/3,604 tokens for probability confidence and 94.80%/3,823 tokens for entropy. We note that a large κ_a does not imply a correspondence with semantic incorrectness. Instead, κ_a measures how strongly the model's support for the realized draft changes when the discrete prefix is replaced by its continuous representation, and uses this discrepancy as an inference-time signal for answer-conditioned on-policy thinking could be useful.

B.6 Ablation on Answer-Content Granularity

Table 9: Ablation on the granularity of κ_{a} estimation. The default setting computes κ_{a} over the whole draft answer, while the answer-content setting computes it only on the extracted answer span.

Granularity	GSM8K		Math500	
	Acc. (%)	# Tokens	Acc. (%)	# Tokens
Whole draft answer (default)	95.38	956	96.80	4428
Answer content only	**96.36** (+0.98)	**885** (-7.4%)	**97.40** (+0.60)	**4019** (-9.2%)

Tab. 9 studies the effect of the granularity used for κ_a estimation. In the default setting, we compute the KL-based reliability estimator over the whole draft answer. We find that a finer-grained variant can further improve performance when the final answer span is easy to identify. For example, in mathematics tasks, the final answer is usually enclosed by `\boxed{}`, which allows us to extract the answer tokens and compute κ_a only on this compact region. This setting improves GSM8K accuracy from 95.38% to 96.36% while reducing the average generation length from 956 to 885 tokens. On Math500, it improves accuracy from 96.80% to 97.40% and reduces the average generation length from 4428 to 4019 tokens.

The results suggest that more precise reliability estimation can be useful when a task provides a clear and compact answer region. However, such extraction is task-dependent and is not always available for reasoning tasks beyond mathematics. To keep the method general across domains, all results in this paper use the default setting, where κ_a is computed over the whole draft.

B.7 Wall-Clock Speedups and Verification Overhead

The continuous-prefix verification passes do not generate tokens, but there is still a computational cost to be measured when assessing practical efficiency. We therefore examine both end-to-end inference latency and the cost of an individual contrastive-estimator invocation.

End-to-end inference latency. Tab. 10 provides the numerical results underlying the latency comparison in Fig. 3. We evaluate Qwen3-8B on a single NVIDIA H200 GPU and report average wall-clock latency. The measurements cover the complete CopT inference procedure, including draft generation, contrastive reliability estimation, on-policy thinking, and chunk-level visibility checks.

Table 10: End-to-end latency and accuracy with Qwen3-8B on a single H200 GPU.

Benchmark	CoT (s)	CopT (s)	Reduction	CoT Acc. (%)	CopT Acc. (%)
GSM8K	7.45	4.70	**36.9%**	95.75	95.98
Math500	59.12	47.13	**20.3%**	96.00	96.20
HumanEval	23.33	7.15	**69.4%**	92.68	94.51

The results demonstrate that the reductions in generated tokens translate into measured end-to-end speedups after accounting for the additional verification passes. Specifically, CopT reduces latency by 36.9%, 20.3%, and 69.4% on GSM8K, Math500, and HumanEval, respectively, while maintaining comparable or higher accuracy.

Contrastive-estimator overhead. The contrastive estimator evaluates cached continuous embeddings in a parallel forward pass, rather than generating the corresponding sequence autoregressively. To quantify this cost, we benchmark a single draft-answer estimator κ_{a} invocation on Qwen3-8B with a 256-token input prefix and 256 subsequent decoding steps.

Table 11: Mean latency measurements for full generation with and without the estimator.

Measurement	Mean latency (ms)
One autoregressive decoding step	19.52
One contrastive-estimator forward pass	25.29
Full generation without the estimator	5,018.59
Full generation with the estimator	5,043.88

As shown in Tab. 11, one estimator invocation takes approximately $1.30\times$ as long as one autoregressive decoding step, but is needed only once for the entire 256-token sequence. In this representative setting, it adds 25.29 ms, or 0.50%, to the full-generation latency. Together, these results show that the small verification overhead can be outweighed by reduced generation, yielding substantial wall-clock speedups.

B.8 Empirical Examination of the Mixture-Linearity Assumption

Assumption 1 provides a mixture-linear model of continuous-prefix behavior for the theoretical analysis in Sec. 4. We empirically examine whether this assumption provides a useful local approximation of continuous-prefix behavior. We evaluate Qwen3-8B along answer-first draft trajectories on 128 GSM8K and 128 Math500 prompts.

Local mixture construction. At each observed draft position with prefix x, we define

$$w(v) = p_\theta(v \mid x), \qquad P_v(y) = p_\theta(y \mid x, v),$$

where $w(v)$ is the next-token distribution and P_v is the next-token output distribution after appending a discrete token v. Using candidate tokens as observable local-state proxies, we compare the

continuous-prefix output distribution with the corresponding hard-state mixture:

$$e_w = \sum_{v \in \mathcal{V}} w(v) E(v), \quad P_{\text{soft}} = p_\theta(\cdot \mid x, e_w),$$
$$P_{\text{mix}} = \sum_{v \in \mathcal{V}} w(v) P_v.$$

Here $E(v)$ is the token embedding.

Controlled finite-mixture evaluation. To make the comparison tractable, we retain the top-K tokens $V_K = \text{TopK}(w)$ and renormalize their probabilities:

$$\widetilde{w}(v) = \frac{w(v)}{\sum_{u \in V_K} w(u)}, \qquad v \in V_K.$$

We then evaluate

$$P_{\text{mix}}^{(K)} = \sum_{v \in V_K} \widetilde{w}(v) P_v,$$
$$P_{\text{soft}}^{(K)} = p_\theta\left(\cdot \,\middle|\, x, \sum_{v \in V_K} \widetilde{w}(v) E(v)\right).$$

We use $K = 16$ and $P_{\text{mix}}^{(K)}$ is an exact mixture over the renormalized top-K distribution. We measure their discrepancy using total variation (TV) distance and Jensen–Shannon (JS) divergence.

Table 12: Local mixture-linearity evaluation with Qwen3-8B. TV and JS compare the continuous-prefix output with the hard-state mixture.

Benchmark	Evaluated positions	Top-16 retained mass	Average TV	Average JS
GSM8K	38,752	99.99%	0.0550	0.0214
Math500	46,748	99.95%	0.0673	0.0271

Tab. 12 shows that the top-16 states retain more than 99.95% of the probability mass on both benchmarks. The mean TV and JS values indicate that the continuous-prefix and finite-mixture output distributions are close on average, although the approximation is not exact.

Comparison with committed hard states. To assess whether the continuous representation is specifically mixture-like, rather than merely close because the next-token distribution is nearly deterministic, we compare its distance to the mixture against the probability-weighted dispersion of committed hard states:

$$D_{\text{hard}}(x) = \sum_{v \in V_K} \widetilde{w}(v) D_{\text{TV}}\left(P_v, P_{\text{mix}}^{(K)}\right).$$

For a set of evaluated positions $\mathcal{X}$, we report the ratio of mean TV distances

$$R_{\mathcal{X}} = \frac{\mathbb{E}_{x \in \mathcal{X}}\left[D_{\text{TV}}\left(P_{\text{soft}}^{(K)}(x), P_{\text{mix}}^{(K)}(x)\right)\right]}{\mathbb{E}_{x \in \mathcal{X}}[D_{\text{hard}}(x)]}.$$

A value $R_{\mathcal{X}} < 1$ means that the continuous-prefix distribution is closer to the mixture on average than a hard-token state drawn from $\widetilde{w}$. We report this ratio for all evaluated positions and for the nondegenerate subset satisfying $D_{\text{hard}}(x) \geq 0.01$.

As shown in Tab. 13, the mean continuous-prefix-to-mixture distance is 74.6% and 75.2% of the mean hard-state dispersion on GSM8K and Math500, respectively. On nondegenerate positions, these ratios remain 72.8% and 73.9%. Thus, even when candidate hard states induce meaningfully different output distributions, the mean continuous-prefix approximation error is approximately 26%–27% lower than the corresponding hard-state dispersion.

Table 13: Continuous-prefix approximation error relative to probability-weighted hard-state dispersion. $R_{\mathcal{X}}$ is computed as the ratio of mean TV distances.

Benchmark	$R_{\mathcal{X}}$	Nondegenerate positions	$R_{\mathcal{X}}$ (Nondegenerate)
GSM8K	0.746	11,450 (29.5%)	0.728
Math500	0.752	14,704 (31.5%)	0.739

Conclusion. The analysis tests a single next-token transition at each observed draft position, using candidate tokens as observable local-state proxies, and the results support mixture linearity as an empirically meaningful local approximation of continuous-prefix behavior.

C Supplementary Details

C.1 Benchmark Details

We evaluate CopT on 10 reasoning benchmarks, covering GSM8K (Cobbe et al., 2021), Math500 (Hendrycks et al., 2021), AIME 2024 (HuggingFaceH4, 2024), AIME 2025 (Yentinglin, 2025), GPQA Diamond (Rein et al., 2024) for mathematics and STEM reasoning; HumanEval (Chen et al., 2021), LeetCode-Contest (Guo et al., 2024), for coding reasoning; CommonsenseQA (Talmor et al., 2019) for commonsense reasoning; and BFCL v4 (Patil et al., 2025), ZebraArena (Zhao et al., 2026) for agentic reasoning.

- **GSM8K**: We evaluate on the test set of 1,319 grade-school math problems. The benchmark tests whether a model can solve natural-language arithmetic questions that often require several reasoning steps. 🤗: https://huggingface.co/datasets/openai/gsm8k.
- **Math500**: We use the curated set of 500 problems from the MATH benchmark. The problems span multiple high-school competition mathematics areas, including algebra, geometry, number theory, precalculus, and intermediate algebra. 🤗: https://huggingface.co/datasets/HuggingFaceH4/MATH-500.
- **AIME 2024**: The benchmark contains 30 problems from the 2024 American Invitational Mathematics Examination, covering both AIME I and AIME II. Each problem requires a concise numeric answer and is designed to test competition-level mathematical reasoning. 🤗: https://huggingface.co/datasets/HuggingFaceH4/aime_2024.
- **AIME 2025**: The benchmark contains 30 problems from the 2025 American Invitational Mathematics Examination, covering both AIME I and AIME II. Each problem requires a concise numeric answer and continues the focus on competition-style math reasoning with challenging questions that test symbolic and logical skills. 🤗: https://huggingface.co/datasets/yentinglin/aime_2025.
- **GPQA Diamond**: We use the Diamond subset of GPQA, which consists of 198 expert-verified multiple-choice STEM questions. The questions mainly cover mathematics, physics, chemistry, biology, and computer science, and are intended to be difficult for non-experts. The problems are designed to evaluate expert-level factual knowledge and reasoning ability. 🤗: https://huggingface.co/datasets/hendrydong/gpqa_diamond_mc.
- **HumanEval**: We use the 164 hand-written Python programming tasks from HumanEval. Each task provides a function signature and docstring, and correctness is measured by executing the generated function against unit tests. 🤗: https://huggingface.co/datasets/openai/openai_humaneval.
- **LeetCode-Contest**: The benchmark contains programming contest problems of different difficulty levels collected from LeetCode contests. We evaluate on its hard set and model outputs are judged by whether the generated solutions pass all the associated tests. 🤗: https://huggingface.co/datasets/TechxGenus/LeetCode-Contest.

- **CommonsenseQA**: We use the test split of 1,221 multiple-choice questions designed to evaluate commonsense reasoning. The questions are constructed from ConceptNet relations, with each example containing one correct answer and multiple distractor options. 🤗: https://huggingface.co/datasets/tau/commonsense_qa.
- **BFCL v4**: The Berkeley Function Calling Leaderboard v4 evaluates the ability of LLMs to invoke functions and tools accurately in realistic agentic settings. We use all subtasks from the non-live and live splits, which contain 2,501 problems in total. : https://github.com/shishirpatil/gorilla.
- **ZebraArena**: A diagnostic simulation environment for evaluating multi-turn agentic reasoning in tool-augmented LLMs. It is built on Zebra logic puzzles under a missing-clues setting, where models must query the environment for hidden facts or relations and then solve a uniquely verifiable constraint satisfaction problem. : https://github.com/wanjiaZhao1203/ZebraArena.

Following the evaluation setup of Qwen3 and Qwen3.5, we allocate a large maximum output budget to allow sufficient reasoning. Specifically, we set the maximum generation length to 32,768 tokens for GSM8K, Math500, GPQA Diamond, HumanEval, LeetCode-Contest, CommonsenseQA, and BFCL v4, and 38,912 tokens for AIME 2024 and AIME 2025. For ZebraArena, we set the maximum generation length to 32,768 tokens for the small split, 65,536 tokens for the medium split, and 98,304 tokens for the large split. We repeat evaluations eight times and report the average accuracy for both CoPT and baselines on the AIME 2024 and AIME 2025 benchmarks.

C.2 Implementation Details

All experiments are conducted on a single NVIDIA H200 GPU. We use the default generation settings of Qwen3 (Yang et al., 2025) and Qwen3.5 (Qwen Team, 2026) models, including temperature of 0.6, top-p of 0.95, top-k of 20, and min-p of 0 for all experiments. The chunk size of CoPT is set to $C = \max\big(1, \lfloor \frac{T_a}{4} \rfloor\big)$, which balances timely estimation with low additional cost. By default, the draft answer is invisible to the first chunk of on-policy thinking, since the earliest value of κ_{r} is available only after the first chunk is completed. We set the maximum draft answer length to 512 for the multi-turn ZebraArena benchmark and 1,024 for all other benchmarks. For reasoning effort control with the reliability threshold τ_a, we use shared settings based on the length. We set $\tau_a = 0.4$ for tasks below 5K tokens (GSM8K, Math500, CommonsenseQA, and BFCL v4), $\tau_a = 1.0$ for tasks above 5K tokens (AIME24, AIME25, GPQA Diamond, and ZebraArena), and $\tau_a = 0.0$ for coding tasks, which typically require long spans of precise output sequences.

C.3 Broader Related Work

Adaptive early-truncation methods. Another line of research improves reasoning efficiency by adaptively terminating explicit CoT generation when further reasoning is considered unnecessary (Yang et al., 2026; Wang et al., 2026; Hosseini et al., 2026). Although these methods share the goal of reducing unnecessary thinking, CoPT differs from them in both motivation and trajectory structure. First, CoPT obtains an explicit draft answer before allocating an extended reasoning trajectory, while early-truncation methods decide whether an ongoing CoT prefix is already sufficient for answering. Second, when additional thinking is triggered, CoPT reasons from a realized hypothesis that can be retained, revised, or rejected, so the resulting trajectory is generally not a prefix continuation of the original thinking-first CoT. Third, CoPT performs draft-specific verification by contrasting the model's support for the realized draft under discrete-token and continuous-embedding inputs, and dynamically controls whether the draft remains visible during subsequent correction. In contrast, early-truncation methods primarily evaluate whether an ongoing reasoning prefix can be terminated.

C.4 Sequence Distributions for the KL Estimators

We define the sequence distributions used by the KL estimators for fixed generated lengths.

For $\kappa_{\mathbf{a}}$ in the draft answer stage. For a fixed draft length T_a, the discrete-prefix continuation distribution over the draft answer is

$$p_\theta(a_{1:T_a} \mid q) := \prod_{t=1}^{T_a} p_\theta(a_t \mid q, a_{<t}).$$

The corresponding continuous-prefix continuation distribution is

$$p_\theta^e(a_{1:T_a} \mid q) := \prod_{t=1}^{T_a} p_\theta(a_t \mid q, e_{<t}),$$

where the discrete prefix preceding each draft token is replaced by the cached continuous embeddings.

For $\kappa_{\mathbf{r}}$ in the on-policy thinking stage. We partition the on-policy thinking trajectory into chunks of length C. Let the k-th chunk start at position

$$s_k := T_a + 1 + (k-1)C,$$

and span positions $[s_k, s_k + C - 1]$. Let m_k denote the visibility state of the draft answer for the k-th chunk.

Conditioned on the visibility-controlled draft-answer input $a^{(m_k)}$ and the prefix $r_{T_a+1:s_k-1}$ preceding the current chunk, the student chunk-level continuation distribution is

$$p_\theta(r_{s_k:s_k+C-1} \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}) := \prod_{t=s_k}^{s_k+C-1} p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:t-1}).$$

The corresponding continuous-prefix intra-chunk continuation distribution is

$$p_\theta^e(r_{s_k:s_k+C-1} \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}) := \prod_{t=s_k}^{s_k+C-1} p_\theta(r_t \mid q, a^{(m_k)}, r_{T_a+1:s_k-1}; e_{s_k:t-1}),$$

where the already generated content inside the current chunk is replaced by the cached continuous embeddings $e_{s_k:t-1}$, while the prefix before the current chunk remains discrete.

D Additional Derivations for Answer-Relevant Uncertainty

In this section, we provide the full derivation and several consequences of Theorem 1. The main message is that the reverse-KL estimator measures uncertainty that affects the next answer token, rather than uncertainty over latent reasoning states themselves.

Proof of Theorem 1. Recall the mixture-linear prefix model under Assumption 1. The local reverse-KL contribution is

$$\kappa(S, A) = \log P_S(A) - \log \bar{P}_w(A).$$

Taking expectation gives

$$\begin{aligned}\mathbb{E}[\kappa(S, A)] &= \sum_{s \in \mathcal{S}} w(s) \sum_a P_s(a) \left[\log P_s(a) - \log \bar{P}_w(a)\right] \\ &= \sum_{s \in \mathcal{S}} w(s) D_{\mathrm{KL}}\big(P_s \,\big\|\, \bar{P}_w\big)\,.\end{aligned}$$

Moreover, under the joint distribution

$$\Pr(S = s, A = a) = w(s) P_s(a),$$

the marginal distribution of A is

$$\Pr(A = a) = \sum_s w(s) P_s(a) = \bar{P}_w(a).$$

Therefore,

$$\begin{aligned} I(S;A) &= \sum_{s,a} w(s) P_s(a) \log \frac{\Pr(S=s, A=a)}{\Pr(S=s)\Pr(A=a)} \\ &= \sum_{s,a} w(s) P_s(a) \log \frac{P_s(a)}{\bar{P}_w(a)} \\ &= \sum_{s} w(s) D_{\mathrm{KL}}\big(P_s \,\big\|\, \bar{P}_w\big)\,. \end{aligned}$$

Hence,

$$\mathbb{E}[\kappa(S,A)] = I(S;A),$$

which completes the proof.

□

Harmless latent-state uncertainty. If all latent states in the support of w induce the same next-token distribution,

$$P_s = P_\star \qquad \text{for all } s \in \mathrm{supp}(w),$$

then

$$\bar{P}_w = \sum_s w(s) P_s = P_\star.$$

Therefore,

$$\kappa(S,A) = \log P_\star(A) - \log P_\star(A) = 0$$

almost surely. Thus, the reverse-KL score is zero even if the latent-state entropy $H(S)$ is large.

This formalizes the intuition that a continuous prefix may encode a superposition of many states and still be reliable. If all plausible states agree on the next-token distribution, then the uncertainty is irrelevant to the answer.

Stability under approximately equivalent states. More generally, suppose all state-conditioned distributions are close to a common distribution $P_\star$. Then

$$\mathbb{E}[\kappa(S,A)] \le \sum_s w(s) D_{\mathrm{KL}}(P_s \,\|\, P_\star)\,.$$

To see this, note that

$$\begin{aligned} \sum_s w(s) D_{\mathrm{KL}}(P_s \| P_\star) &= \sum_s w(s) \sum_a P_s(a) \log \frac{P_s(a)}{P_\star(a)} \\ &= \sum_s w(s) \sum_a P_s(a) \left[\log \frac{P_s(a)}{\bar{P}_w(a)} + \log \frac{\bar{P}_w(a)}{P_\star(a)} \right] \\ &= \sum_s w(s) \sum_a P_s(a) \log \frac{P_s(a)}{\bar{P}_w(a)} + \sum_s w(s) \sum_a P_s(a) \log \frac{\bar{P}_w(a)}{P_\star(a)} \\ &= \sum_s w(s) D_{\mathrm{KL}}(P_s \| \bar{P}_w) + \sum_a \left[\sum_s w(s) P_s(a) \right] \log \frac{\bar{P}_w(a)}{P_\star(a)} \\ &= \sum_s w(s) D_{\mathrm{KL}}(P_s \| \bar{P}_w) + \sum_a \bar{P}_w(a) \log \frac{\bar{P}_w(a)}{P_\star(a)} \\ &= \sum_s w(s) D_{\mathrm{KL}}(P_s \| \bar{P}_w) + D_{\mathrm{KL}}(\bar{P}_w \| P_\star). \end{aligned}$$

Since KL divergence is nonnegative,

$$\sum_s w(s) D_{\mathrm{KL}}\big(P_s \,\big\|\, \bar{P}_w\big) \le \sum_s w(s) D_{\mathrm{KL}}(P_s \,\|\, P_\star)\,.$$

By Theorem 1,

$$\mathbb{E}[\kappa(S, A)] = \sum_s w(s) D_{\mathrm{KL}}\big(P_s \,\big\|\, \bar{P}_w\big)\,,$$

which proves the claim.

Deterministic-token case. Suppose each latent state s deterministically implies one answer token

$$g(s) \in \mathcal{A},$$

so that

$$P_s(a) = \mathbb{1}\{a = g(s)\}.$$

Define the marginal probability of the answer token induced by the soft prefix state as

$$\rho(a) = \Pr(g(S) = a) = \sum_{s:g(s)=a} w(s).$$

Then

$$\bar{P}_w(a) = \sum_s w(s)\mathbb{1}\{a = g(s)\} = \rho(a).$$

Since $A = g(S)$ deterministically,

$$\kappa(S, A) = \log 1 - \log \rho(g(S)) = -\log \rho(g(S)).$$

Taking expectation gives

$$\mathbb{E}[\kappa(S, A)] = \sum_a \rho(a)(-\log \rho(a)) = H(g(S)).$$

Therefore, under the deterministic-answer setting, the reverse-KL score is exactly the entropy of the induced answer token $g(S)$ instead of the entropy of the latent state S.

E Case Studies Comparing CoT and CoPT

We compare representative GSM8K cases from Qwen3-8B. Thinking excerpts show the beginning and end of each trace, and square brackets summarize omitted content.

1 Accepting a correct draft directly — **93.9% fewer tokens**

Question. Jared is trying to increase his typing speed. He starts with 47 words per minute (WPM). After some lessons the next time he tests his typing speed it has increased to 52 WPM. If he continues to increase his typing speed once more by 5 words, what will be the average of the three measurements?
Gold answer: 52 WPM.

CoT — 3,268 tokens

THINKING / BEGINNING

Okay, let me try to figure out this problem. So, Jared is trying to increase his typing speed. He starts at 47 words per minute (WPM). Then after some lessons, he tests again and it's 52 WPM. Now, the question says if he continues to increase his speed once more by 5 words, what will be the average of the three measurements?

[Repeatedly recalculates the average and reconsiders whether the final increase gives 52 or 57 WPM.]

THINKING / END

Therefore, I think the answer is 52. Therefore, the average of the three measurements is 52 WPM.

Final answer: **52 (correct)**.

CoPT — 200 tokens

DRAFT / BEGINNING

We are given the following information:
Initial typing speed: 47 WPM

Computes $(47 + 52 + 57)/3 = 52$.

DRAFT / END

$\boxed{52}$

SUBSEQUENT THINKING / 0 TOKENS

None. The draft is returned directly.

Final answer: **52 (correct)**.

2 Verifying a correct draft briefly — 90.5% fewer tokens

Question. 200 pounds of carrots are to be distributed to 40 restaurants in a certain city. Each restaurant is to receive 2 pounds of carrots. How many pounds of carrots will not be used?
Gold answer: 120 pounds.

CoT — 3,717 tokens

THINKING / BEGINNING

Okay, let's see. The problem says there are 200 pounds of carrots to be distributed to 40 restaurants. Each restaurant is supposed to get 2 pounds. The question is asking how many pounds of carrots will not be used. Hmm, so I need to figure out the total amount of carrots that will be given out and then subtract that from the total available to find the leftover.

[Repeats the subtraction, considers distributing all 200 pounds, and searches for alternative interpretations.]

THINKING / END

Alternatively, maybe the problem is in the way the carrots are distributed. For example, if they have to distribute the carrots in some specific way, but the problem doesn't mention any constraints. Therefore, I think my initial answer is correct.

Final answer: 120 (correct).

CoPT — 354 tokens

DRAFT / BEGINNING

We are given the following:
Total pounds of carrots: 200 pounds
Uses $40 \times 2 = 80$ and $200 - 80 = 120$.

DRAFT / END

$\boxed{120}$

THINKING / BEGINNING

Okay, let's see. The problem says there are 200 pounds of carrots to be distributed to 40 restaurants.

[Checks $40 \times 2 = 80$ and $200 - 80 = 120$.]

THINKING / END

That seems right. So the answer is 120 pounds of carrots will not be used.

Final answer: 120 (correct).

3 Avoiding an incorrect revision — 92.2% fewer tokens

Question. Robyn sends sixteen emails a day. Seven are work emails, and two-thirds of the remainder are to family. One-third of the other emails are to her boyfriend. How many emails a day does she send to her boyfriend?
Gold answer: 1 email.

CoT — 7,386 tokens

THINKING / BEGINNING

Okay, let me try to figure out how many emails Robyn sends to her boyfriend each day. So, the problem says she sends sixteen emails a day. Out of these, seven are work emails. Then, two-thirds of the remainder are to family. And one-third of the other emails are to her boyfriend. Hmm, let me break this down step by step.

[Initially obtains 1; then assumes the named categories must account for all 16 emails and changes the interpretation.]

THINKING / END

This seems to be the most logical interpretation, even though the wording is slightly awkward. Therefore, the answer is 3. But initially, I thought it was 1, but that led to inconsistency. Therefore, I think the correct answer is 3.

Final answer: 3 (incorrect).

CoPT — 579 tokens

DRAFT / BEGINNING

We are told that Robyn sends 16 emails a day, and we are to determine how many of those are sent to her boyfriend.

Computes $16 - 7 = 9$, $\frac{2}{3} \times 9 = 6$, then $\frac{1}{3} \times (9 - 6) = 1$.

DRAFT / END

Robyn sends 1 email per day to her boyfriend.

THINKING / BEGINNING

Okay, let me try to figure out how many emails Robyn sends to her boyfriend each day.

[Checks the same sequence of subtractions and fractions.]

THINKING / END

Then one-third of those 3 is 1. Yep, that seems right. So the answer is 1.

Final answer: 1 (correct).

4 Repairing an incorrect draft **78.3% fewer tokens**

Question. Maria was charging her power bank before going to the park when she disconnected it and noticed the power bank wasn't fully charged yet. Once at the park, her friends asked her if they could charge their phones. While charging her friends' phones, she noticed that her power bank was losing 9% of the total capacity each hour. 5 hours later the battery started to lose instead 7% of the total capacity each hour for about 3 hours. In the end, the charge remaining was 28%. What was the charge of the power bank when Maria went out to the park?
Gold answer: 94%.

CoT 16,508 tokens

THINKING / BEGINNING
Okay, let me try to figure out this problem step by step. So, Maria had a power bank that she was charging before going to the park. She disconnected it and noticed it wasn't fully charged yet.
[Explores exponential decay; rejects fixed losses after incorrectly assuming an initial charge of 100%.]

THINKING / END
Therefore, the correct interpretation is that the power bank loses 9% of its current charge each hour, leading to exponential decay. Therefore, the answer is approximately 55.8%. Therefore, the charge of the power bank when Maria went out to the park was approximately 55.8%.

Final answer: **55.8% (incorrect)**.

CoPT 3,579 tokens

DRAFT / BEGINNING
We are given that:
Maria disconnected her power bank before going to the park and noticed it wasn't fully charged yet.
Uses exponential decay,
$C(0.91)^5(0.93)^3 = 0.28$.

DRAFT / END
$\boxed{55.77\%}$

THINKING / BEGINNING
Okay, let me try to figure out this problem.
[Reinterprets "total capacity" as fixed losses; calculates $28 + 5 \times 9 + 3 \times 7 = 94$ and checks the wording.]

THINKING / END
Therefore, I think the answer is 94%.

Final answer: **94% (correct)**.

These four cases illustrate complementary behaviors of CoPT. In Case 1, the correct draft is returned directly without subsequent on-policy thinking, while Case 2 shows that a brief verification can confirm the answer without an extended trace. Case 3 highlights a failure of prolonged CoT: after initially obtaining the correct answer, it introduces an unsupported assumption that the named email categories must account for all emails and revises the answer incorrectly. CoPT retains the correct result. Case 4 demonstrates that CoPT can also recover from an erroneous draft: subsequent thinking replaces exponential decay with fixed losses measured against total battery capacity, whereas CoT retains the incorrect model. Across these selected examples, CoPT produces correct final answers with 78.3–93.9% fewer output tokens than CoT, illustrating both reduced unnecessary reasoning and the ability to verify or revise a draft when necessary.