

LEARNING EFFICIENT LATENT REASONING WITH ABSTRACT CHAIN-OF-THOUGHT

Keshav Ramji*, Tahira Naseem & Ramón Fernandez Astudillo
IBM Research AI

ABSTRACT

While long, explicit chains-of-thought (CoT) have proven effective on complex reasoning tasks, they are costly to generate during inference. Recent latent reasoning methods have focused on leveraging continuous representations, yet have primarily explored the pre-training regime and are not performant at scale, lagging behind verbalized CoT. We propose **Abstract Chain-of-Thought**, a discrete latent reasoning post-training mechanism in which the language model produces a short sequence of tokens from a reserved codebook in lieu of a natural language CoT, prior to generating a response. To make previously unseen “abstract” tokens useful, we introduce a policy iteration-style warm-up loop that alternates between (i.) bottlenecking from a verbal CoT via masking and performing supervised fine-tuning, and (ii.) self-distillation by training the model to generate abstract tokens from the prompt alone via constrained decoding from the codebook. After warm-up, we optimize the generation of abstract sequences with warm-started reinforcement learning under constrained decoding. For the Qwen3-8B and Granite-3.3-8B language models, Abstract-CoT achieves up to $12\times$ fewer reasoning tokens while achieving comparable performance across mathematical reasoning, instruction-following, and multi-hop reasoning in natural language. We also find emergent non-uniform usage over the codebook, resembling power law dynamics seen in pre-training. Our findings highlight the potential for post-training recipes that facilitate latent reasoning for inference efficiency while inducing new representational dynamics through the introduction of a new “reasoning language”.

1 INTRODUCTION

Large language models (LLMs) increasingly rely on long, explicit chains-of-thought (CoTs) to solve complex, multi-step reasoning problems. Despite its effectiveness, verbalized CoT (Wei et al., 2022; Kojima et al., 2022) is an expensive mechanism, increasing latency and cost at inference while bloating the length of traces during reinforcement learning (RL). These drawbacks have motivated approaches to compress or internalize natural language CoT with more efficient intermediate representations (Cheng & Durme, 2024; Deng et al., 2024). Simultaneously, approaches focusing on pause or filler tokens (Goyal et al., 2024; Pfau et al., 2024) suggest that their addition facilitates deliberate internalized thinking through its activations. Recent works such as Coconut (Hao et al., 2025) have also sought to enable reasoning mechanisms through continuous concept spaces, suggesting that principled methods for internalized recurrence can balance efficiency and expressivity.

In this work, we study a simple question: can we replace long verbalized rationales with a short sequence of discrete abstract tokens that function as a latent scratchpad, while retaining the benefits of CoT in response generation? In a similar vein, is it possible to enable latent reasoning on an existing instruction-tuned model? We propose **Abstract Chain-of-Thought (Abstract-CoT)**: instead of generating natural-language reasoning, we induce the model to emit a bounded-length sequence of tokens from a reserved abstract codebook of distinguishable filler tokens. Abstract-CoT is designed to be token-efficient and non-verbal, producing short intermediate traces while offering an alternative to rationales generated in natural language.

However, adding previously unseen tokens creates a cold-start problem, as their embeddings are randomly initialized and meaningless initially. While these tokens appear semantically uninformative,

*Correspondence to keshav.ramji@ibm.com.

our recipe aims to learn to produce a sequence of these tokens, inducing new pathways between a prompt and response. We address this with a policy iteration warm-up loop that can be viewed as discrete latent variable modeling under an explicit information bottleneck via masking. Each iteration alternates between bottlenecked supervised fine-tuning (SFT) and self-distillation to remove dependence on the verbal CoT at training time. Subsequently, we apply reinforcement learning with a generative reward model to induce exploration over abstract token sequences and refine the abstract generation policy. Our findings demonstrate substantial gains in token efficiency while matching or outperforming verbal chain-of-thought. We summarize our contributions below:

1. **Abstract Chain-of-Thought:** We propose Abstract-CoT, a mechanism for reasoning through a codebook of reserved tokens introduced entirely in LLM post-training.
2. **Warm-up via Policy Iteration:** We warm up the embeddings of the reserved tokens by alternating bottlenecked SFT and self-distillation, yielding an on-policy abstract generator.
3. **Warm-started RL for abstract policies:** We optimize generation of abstract traces using GRPO, with constrained decoding to the abstract codebook.
4. **Token Efficiency:** Abstract-CoT achieves large reductions in reasoning tokens (up to $12\times$) while matching performance on MATH-500, AlpacaEval, and HotpotQA. We also observe power-law dynamics over the codebook, suggesting learning of meaningful concepts.

2 RELATED WORK

The works closest to ours can be categorized into four bins: (1) approaches that introduce filler tokens to enhance expressivity and computational width during reasoning through its internal activations, (2) techniques that learn to compress or replace segments of the verbal CoT, (3) methods for reasoning purely in continuous latent concept spaces (and similar "soft" thinking / tokens approaches), and (4) RL to introduce inference-time budget control. We focus on the methods most closely related to the motivation behind our work and our approach; we do not claim our literature search to be exhaustive across the complete space of latent and efficient reasoning topics. We refer the reader to related surveys such as (Chen et al., 2025; Zhu et al., 2025).

2.1 FILLER TOKENS

Works on filler tokens augment the token sequence with special tokens that do not primarily serve as human-readable natural language text (i.e. are semantically uninformative), but instead expand the model’s effective computation in the forward pass. Goyal et al. (2024) introduce `<pause>` tokens, showing that explicitly allocating intermediate tokens to allow for "thinking time" can improve reasoning. Mu et al. (2023) propose *gist tokens* which serve as a learned bottleneck, summarizing longer contexts into a small set of activations that can be cached and reused, introducing contextual "slots" that can hold task-relevant information. Other works suggest that such tokens can be used to expand expressivity limits (Pfau et al., 2024; Merrill & Sabharwal, 2025; London & Kanade, 2025), and used to cache preceding context for long-context retrieval (Shah et al., 2025). Works such as Jang et al. (2025) increase inference-time compute via token-space interventions, exploring the settings in which allocating additional intermediate representations can boost capabilities.

While our abstract tokens are introduced specifically as a latent reasoning trace (rather than as prompt compression), they share a similar essence that a small number of lightweight extra positions can be trained to store or carry additional (non-linguistic) information. This also relates our work to parameter-efficient methods that seek to optimize continuous prompt embeddings to steer generation (Lester et al., 2021; Li & Liang, 2021).

2.2 CoT COMPRESSION, DISTILLATION, AND DISCRETE CODEBOOKS

Compression, distillation, and partially removing the textual rationale (often through staged curricula) are some of the key mechanisms used to target the verbosity and cost of verbalized CoT. Early works such as Hsieh et al. (2023) demonstrated that explicit step-by-step rationales can be distilled to smaller models. Recent methods include seeking to directly shorten verbalized CoT via multi-round refinement (Yan et al., 2025) and learning to skip intermediate reasoning tokens in a controllable fashion while retaining generation quality (Xia et al., 2025).

Approaches that compress parts of the rationale into a learned discrete or quantized representation are closest to the motivation behind our discrete codebook. [Su et al. \(2025\)](#) combines latent tokens (learned via vector quantization) with remaining text tokens, inducing an efficiency-interpretability trade-off. Complementary works perform step-wise compression of CoT into latent tokens ([Zhang et al., 2025a](#)) as well as gradually internalizing explicit steps into implicit computation ([Deng et al., 2024](#)) in a curriculum fashion. By contrast, our abstract tokens are not a quantized reconstruction of a teacher rationale, but rather, an introduced reserved discrete codebook, with the model trained to use it as a compact reasoning language under constrained decoding. This allows the model to potentially explore *other reasoning pathways* aside from the teacher CoT.

2.3 CONTINUOUS AND HYBRID LATENT REASONING

Beyond discrete codebooks, several recent methods replace parts of the textual rationale with continuous latent thought states. Coconut ([Hao et al., 2025](#)) replaces some CoT tokens with continuous latent vectors derived from hidden states and trains the language model with a curriculum that gradually increases the latent segment and replaces verbalized CoT segments. CODI ([Shen et al., 2025](#)) similarly compresses CoT into a continuous space, using self-distillation to align latent trajectories with those induced by explicit rationales. System-1.5 reasoning ([Wang et al., 2025](#)) introduces dynamic shortcuts, traversing between language and latent spaces while aiming to reduce unnecessary verbal reasoning and retain controllability.

Related “soft” thinking approaches propagate non-discrete intermediate representations by feeding mixtures or distributions over embeddings as subsequent inputs ([Xu et al., 2025](#); [Zhang et al., 2025b](#)). Recent works such as [Butt et al. \(2025\)](#) study training and optimization stability when such soft tokens are treated as decision variables through RL. Hybrid methods such as HybridCoT ([Anonymous, 2026](#)) explicitly interleave latent and text tokens to balance efficiency with partial interpretability. In our work, we operate fully in the discrete token space, retaining compatibility with standard decoding tools, but still aim to obtain the efficiency associated with latent reasoning.

2.4 RL FOR BUDGET CONTROL

A complementary direction focuses on controlling inference-time cost by explicitly optimizing the reasoning budget, which is often operationalized as the length of intermediate reasoning traces. Recent work applies RL to learn when to expend additional reasoning steps as opposed to answering early; for example, by learning adaptive chain-of-thought triggering policies under compute constraints ([Lou et al., 2025](#)), or by pruning or shortening intermediate reasoning via training-time objectives that directly reward efficiency ([Hou et al., 2026](#)). Other recent approaches optimize a length-accuracy trade-off with RL objectives, either by allocating token budgets dynamically ([Kleinman et al., 2025](#)) or by explicitly optimizing for consistency with user-specified length constraints ([Aggarwal & Welleck, 2025](#)).

While our RL stage is most closely aligned with this line of work, it differs in the action space: instead of optimizing over free-form textual CoT length, we optimize over sequences constrained to a reserved discrete codebook. This enables strong control over the intermediate sequence while aiming to avoid the brittleness of length control in open-ended natural language.

3 LATENT REASONING WITH ABSTRACT CHAIN-OF-THOUGHT

3.1 PROBLEM SETUP AND NOTATION

Let x , c , and y denote a prompt, gold verbal chain-of-thought (CoT), and the target answer, respectively. We assume training data $\mathcal{D} = \{(x_i, c_i, y_i)\}_{i=1}^N$, where c_i is only available during the warm-up phase. Let π_θ be a causal decoder-only LLM with parameters θ and base vocabulary \mathcal{V} .

Abstract Codebook. We extend the tokenizer with a set of M previously unseen (reserved) tokens in the abstract codebook:

$$\mathcal{V}_{\text{abs}} = \{\langle \text{TOKEN_A} \rangle, \langle \text{TOKEN_B} \rangle, \dots, \langle \text{TOKEN_Z} \rangle, \langle \text{TOKEN_AA} \rangle, \dots\},$$

along with two delimiters `<beginabstract>` and `<endabstract>`, marking the abstract reasoning segment¹. Thus, an abstract chain-of-thought is a token sequence $z = (z_1, \dots, z_m) \in \mathcal{V}_{abs}^m$, formatted as:

$$\tilde{z} = \langle \text{beginabstract} \rangle z_1 z_2 \dots z_m \langle \text{endabstract} \rangle$$

We denote the maximum length of the abstract sequence by $m \leq m_{\max}$; at inference-time, the model receives x and must generate \tilde{z} and y without access to c . Let \mathcal{Z} denote the positions of the full abstract sequence \tilde{z} (including `<beginabstract>` and `<endabstract>`), and let $\mathcal{Z}_{\text{abs}} \subseteq \mathcal{Z}$ denote the positions of the m codebook tokens $z_1, \dots, z_m \in \mathcal{V}_{\text{abs}}$.

We view the abstract trace z as a discrete latent variable, mediating reasoning. Ideally, we would like to maximize the marginal likelihood:

$$\log \pi_{\theta}(y | x) = \log \sum_{z \in \mathcal{V}_{\text{abs}}^*} \pi_{\theta}(z | x) \pi_{\theta}(y | x, z) \quad (1)$$

for a sequence of length $\leq m_{\max}$, but the sum over discrete traces is intractable. Therefore, Abstract Chain-of-Thought uses a bootstrapping procedure that alternates (i) proposing a trace z (via a teacher policy, or constrained to the current model), and (ii) updating the model given the proposed trace, followed by distillation to learn to propose traces from x alone.

3.2 WARM-UP VIA POLICY ITERATION

The abstract tokens start with randomly initialized embeddings, so the model initially cannot exploit the bottleneck in the absence of a prior that enforces specific concept mappings. Therefore, we perform an **abstract embedding warm-up** with a policy iteration loop over iterations $t = 1, \dots, T$; each iteration produces a dataset of abstract trajectories $\tilde{z}^{(t)}$ and updates θ via SFT.

Constrained Decoding. We use $\pi_{\theta}^{\text{abs}}$ to denote the policy restricted to an allowed token set $\mathcal{A} = \mathcal{V}_{\text{abs}} \cup \{\langle \text{endabstract} \rangle\}$ which can be generated. At each step i , with context $h = x \cup \{\langle \text{beginabstract} \rangle\} \cup \tilde{z}_{<i}$, $\pi_{\theta}^{\text{abs}}(a | h) = \frac{\pi_{\theta}(a|h) \mathbb{1}[a \in \mathcal{A}]}{\sum_{u \in \mathcal{A}} \pi_{\theta}(u|h)}$. We enforce a hard cap m_{\max} of codebook tokens that may be generated; if $m = m_{\max}$, we force the end delimiter to be generated, then allow the response to be generated without constrained decoding.

(1) Bottlenecked SFT with Abstract Tokens. Given (x, c, y) , we construct $\tilde{z}^{(t)}$ using a policy ϕ_t . In the first iteration, we use random initialization, sampling a random number of abstract tokens per CoT step ($\text{rand}(1, \frac{|s|}{2})$ for $|s|$ tokens in step s) and choosing the specific tokens uniformly from \mathcal{V}_{abs} . We include ablations with other initialization schemes (besides uniformly at random) in Appendix A.1. In subsequent iterations ($t \geq 2$), abstract sequences are generated on-policy: $\tilde{z}^{(t)} \sim \pi_{\theta}^{\text{abs}}(\cdot | x, c)$ under constrained decoding.

We form a single concatenated training sequence $s = [x ; c ; \tilde{z} ; y]$ and define a block-structured attention mask that enforces an information bottleneck. Let indices be partitioned into prompt (\mathcal{X}), verbal CoT (\mathcal{C}), abstract sequence (\mathcal{Z}) and answer (\mathcal{Y}). The **abstract tokens attend to the prompt and the verbal CoT**; that is:

$$A_{i,j} = 1 \quad \forall i \in \mathcal{Z}, j \in \mathcal{X} \cup \mathcal{C} \cup \mathcal{Z}_{\leq i}$$

Crucially, the **answer attends only to the prompt and the abstract tokens, not to the verbal CoT**:

$$A_{i,j} = \begin{cases} 1 & i \in \mathcal{Y}, j \in \mathcal{X} \cup \mathcal{Z} \cup \mathcal{Y}_{\leq i} \\ 0 & i \in \mathcal{Y}, j \in \mathcal{C} \end{cases}$$

All other entries follow standard causal masking, ensuring any dependence of y on c must pass through the limited number of abstract-token positions \mathcal{Z} . We then optimize a masked SFT objective

¹We use alphabetical names `<TOKEN_A>`, ..., `<TOKEN_Z>`; for $M > 26$, we continue with two letter identifiers (AA-ZZ), which may be similarly extended for larger codebooks.

Algorithm 1 : Policy Iteration Warm-Up for Abstract-CoT**Require:** Training data $\mathcal{D} = \{(x, c, y)\}$, abstract codebook \mathcal{V}_{abs} , iterations T

- 1: Initialize $\theta^{(0)}$ from base instruction-tuned model; add new token embeddings for \mathcal{V}_{abs}
- 2: **for** $t = 1$ **to** T **do**
- 3: Generate abstract traces $\tilde{z}^{(t)} \leftarrow \phi_t(x, c; \theta^{(t-1)})$ (random if $t=1$, else constrained decoding)
- 4: Update $\theta^{(t)} \leftarrow \arg \min_{\theta} \mathbb{E}_{(x,c,y) \sim \mathcal{D}} [\mathcal{L}_{\text{SFT}}(\theta; x, c, \tilde{z}^{(t)}, y)]$
- 5: Distill: generate $\tilde{z} \sim \pi_{\theta^{(t)}}^{\text{abs}}(\cdot | x)$; update $\theta^{(t)} \leftarrow \arg \min_{\theta} \mathbb{E} [\mathcal{L}_{\text{Distill}}(\theta; x, \tilde{z}, y)]$
- 6: **end for**
- 7: **return** $\theta^{(T)}$

that trains on the abstract sequence² and the answer while hiding the verbal CoT with bottleneck attention mask A :

$$\mathcal{L}_{\text{SFT}}(\theta; x, c, \tilde{z}, y) = - \sum_{j \in (\mathcal{Z}_{\text{abs}} \cup \mathcal{Y})} \log \pi_{\theta}(s_j | s_{<j}; A), \quad (2)$$

(2) Self-Distillation Without Verbal CoT. The bottlenecked SFT stage exploits c to shape the hidden states at abstract-token positions, but our target policy ultimately should produce abstract tokens from the prompt alone; this motivated the loss computation on $\tilde{z}^{(t)}$. We create a distillation dataset by generating $\tilde{z} \sim \pi_{\theta}^{\text{abs}}(\cdot | x)$ via constrained decoding (with $m \leq m_{\text{max}}$) and pairing it with the gold answer y : $\mathcal{D}_{\text{distill}}^{(t)} = \{(x_i, \tilde{z}_i, y_i)\}_{i=1}^N$. We train with standard causal SFT on $[x; \tilde{z}; y]$, where s_j spans the abstract and response tokens:

$$\mathcal{L}_{\text{Distill}}(\theta; x, \tilde{z}, y) = - \sum_{j \in (\mathcal{Z} \cup \mathcal{Y})} \log \pi_{\theta}(s_j | x, \tilde{z}, y_{<j}) \quad (3)$$

3.3 REINFORCEMENT LEARNING FROM WARM-START

After the warm-up phase, we optimize the abstract-token policy with RL; we refer to this as warm-started RL. In practice, this is implemented in a similar manner as the warm-up self-distillation phase: (1) generate \tilde{z} under a guided-choice constraint, and (2) append `<endabstract>` and decode y unconstrained. Our default GRPO (Shao et al., 2024) updates include log-probabilities for both the abstract trace and the answer tokens, improving response quality after RL in addition to shaping the intermediate abstract sequence³. We use a generative reward model – specifically, gpt-oss-20b (OpenAI, 2025) – to score outputs in our experiments, for our recipe to generalize to non-verifiable, natural language settings.

For each prompt x , we sample a group of K trajectories $\{(\tilde{z}_k, y_k)\}_{k=1}^K$ by first drawing $\tilde{z}_k \sim \pi_{\theta}^{\text{abs}}(\cdot | x)$, then $y_k \sim \pi_{\theta}(\cdot | x, \tilde{z}_k)$, and computing rewards $\hat{R}_k = \hat{R}(x, \tilde{z}_k, y_k)$. We define advantages:

$$A_k = \frac{\hat{R}_k - \text{mean}(\hat{R}_{1:K})}{\text{std}(\hat{R}_{1:K}) + \epsilon}$$

We update θ via applying GRPO to an action space over (\tilde{z}, y) :

$$\mathcal{J}(\theta) = \mathbb{E}_x \left[\frac{1}{K} \sum_{k=1}^K A_k \left(\sum_{t \in \mathcal{Z}_{\text{abs}}} \log \pi_{\theta}^{\text{abs}}(z_{k,t} | x, z_{k,<t}) + \sum_{t \in \mathcal{Y}} \log \pi_{\theta}(y_{k,t} | x, \tilde{z}_k, y_{k,<t}) \right) - \beta \text{KL} \left(\pi_{\theta}^{\text{abs}}(\tilde{z} | x) \pi_{\theta}(y | x, \tilde{z}) \parallel \pi_{\theta_{\text{ref}}}^{\text{abs}}(\tilde{z} | x) \pi_{\theta_{\text{ref}}}(y | x, \tilde{z}) \right) \right]. \quad (4)$$

where $\pi_{\theta_{\text{ref}}}$ is the reference policy (the warm-started model). KL regularization is applied over both the abstract and the response distributions.

²Optionally, $\tilde{z}^{(t)}$ can be treated as fixed in this stage (without backpropagating on the abstract tokens), which would update the embeddings solely through gradients flowing from the answer loss.

³Alternatively, updates can be isolated to just the abstract tokens along with a fixed decoding rule.

3.4 ABSTRACT CHAIN-OF-THOUGHT AS A DISCRETE LATENT BOTTLENECK

Our masking design ensures that the answer *cannot* directly attend to the verbal CoT. Instead, the answer attends to the abstract sequence, which attends to the verbal chain-of-thought as guidance. Let $H_{Z_{\text{abs}}}$ denote the hidden states at the abstract-token positions Z_{abs} produced from the prefix $[x; c; \tilde{z}]$ following masking of the verbal CoT. The only dependence of answer generation on the verbal CoT is through $H_{Z_{\text{abs}}}$, which induces conditional Markov structure:

$$C \rightarrow H_{Z_{\text{abs}}} \rightarrow Y \text{ (conditioned on } X \text{ and } Z)$$

By the data processing inequality, any dependence between the verbal CoT and the answer must be bounded by the information that can be transmitted through the abstract segment, yielding:

$$I(C; Y \mid X, Z) \leq I(C; H_{Z_{\text{abs}}} \mid X, Z) \quad (5)$$

Since $H_{Z_{\text{abs}}}$ scales linearly with the number of abstract tokens m , increasing m_{max} increases the channel capacity from c to y during warm-up, while the self-distillation and RL phases of the Abstract-CoT pipeline tune the model’s inference-time thinking budget.

4 RESULTS

4.1 EXPERIMENTAL SETUP

Datasets. We train using subsampled data (600k samples) from the Dolci-Think-SFT dataset (AI2, 2025b) for the policy iteration warm-up loop and from Dolci-Think-RL (AI2, 2025a) for reinforcement learning; these were used in the development of the Olmo 3 Think models (Olmo et al., 2025). The former contains prompts paired with gold verbal CoT and gold answers, while we only use prompts and gold responses from the latter. We report results on MATH-500 (Hendrycks et al., 2021), AlpacaEval-LC-2.0 (Dubois et al., 2024), and HotpotQA (500 samples; Yang et al. (2018)), spanning math (verifiable), general instruction-following, and multi-hop question-answering (non-verifiable) settings.

Models. We consider two instruction-tuned models, Qwen3-8B (Yang et al., 2025) and Granite-3.3-8B-Instruct (Granite Team, 2024). We evaluate models without their ”thinking mode” with standard CoT prompting, and report both performance and intermediate token (reasoning/rationale) length. The ”thinking mode” is disabled to ensure controlled comparison. For Abstract-CoT, we extend each tokenizer with the abstract codebook, train new embeddings during warm-up, and decode constrained abstract traces up to m_{max} tokens. We use $T = 3$ policy iteration warm-up rounds, with ablation results included ($T \in \{1, 2, 3\}$) in Appendix A.2.

Baselines and Methods. We include the following baselines:

- **Baseline (no thinking):** Direct answer generation from the prompt.
- **Pause Tokens (Goyal et al., 2024):** Insert m_{max} `<pause>` tokens before generation.
- **SFT w/o CoT:** Supervised fine-tuning on (x, y) pairs.
- **SFT w/ CoT:** Supervised fine-tuning on (x, c, y) trajectories.
- **SFT + RL:** SFT with CoT followed by RL on the answer (no abstract codebook).

The variants of our method analyzed are: **RL-only** (cold-start), **Warm-up** (warm-up only), and **Warm-up + RL** (warm-started GRPO). We use an abstract codebook of $M = 32$ tokens, and constrain abstract generation to at most $m_{\text{max}} = 128$. Each phase of warm-up (bottlenecked SFT, self-distillation) consists of 3 epochs of training, and RL is performed for 1M episodes. All training (SFT and RL) are performed on $8 \times$ NVIDIA H100 GPUs.

Table 1: Main results on MATH-500 (accuracy), AlpacaEval (win-rate), and HotpotQA (F1). We include the average number of generated tokens per prompt during evaluation, between reasoning tokens for each method and response tokens. This includes verbal-CoT tokens (baselines), abstract codebook tokens in \mathcal{V}_{abs} for Abstract-CoT, and pause-token count for the Pause Tokens baseline. The top results per model for each column are bolded, with the second best results underlined.

Method	MATH-500		AlpacaEval		HotpotQA	
	Accuracy	Tokens	Win-rate	Tokens	F1	Tokens
Qwen3-8B						
Baseline	82.4	1205	52.4	322	51.1	527
Pause Tokens	78.6	142	46.7	212	49.0	173
SFT (no CoT)	85.8	394	54.9	298	51.3	331
SFT (CoT)	89.8	1522	57.0	460	54.8	679
SFT+RL	92.6	1671	<u>58.4</u>	496	58.1	735
Abstract-CoT (RL-only)	82.2	118	50.2	229	48.7	149
Abstract-CoT (Warm-up)	87.4	173	55.6	251	53.3	174
Abstract-CoT (Warm-up + RL)	<u>90.6</u>	144	60.3	225	<u>57.9</u>	171
Granite-3.3-8B						
Baseline	67.2	1093	49.6	381	41.6	554
Pause Tokens	64.6	158	40.5	241	41.3	179
SFT (no CoT)	70.8	526	48.4	362	42.2	343
SFT (CoT)	76.2	1762	52.5	469	<u>45.8</u>	598
SFT+RL	<u>78.4</u>	1813	<u>54.1</u>	447	46.9	769
Abstract-CoT (RL-only)	64.8	132	50.3	278	38.9	134
Abstract-CoT (Warm-up)	71.6	189	52.8	246	40.8	199
Abstract-CoT (Warm-up + RL)	79.0	151	57.4	237	45.1	178

4.2 ABSTRACT-CoT ENABLES EFFICIENT REASONING

Our findings show that Abstract-CoT nears or exceeds the performance of post-training with verbalized CoT (SFT + RL) for both models and all benchmarks, as illustrated in Table 1. We find that both models achieve meaningful gains on AlpacaEval (+1.9 pts for Qwen3, +3.3 pts for Granite-3.3). We measure token efficiency as the ratio of verbal CoT tokens to abstract tokens, averaged over evaluation prompts. For c_{verbal} (the model-generated verbal rationale from a verbal CoT baseline) and m , we denote the compression ratio $= \frac{\mathbb{E}[|c_{\text{verbal}}|]}{\mathbb{E}[m]}$. We achieve substantial gains in token efficiency: **11.6-12.0** \times (MATH-500), **1.9-2.2** \times (AlpacaEval), and **4.3** \times (HotpotQA). In Figure 1, we plot the token usage and performance on MATH-500 and AlpacaEval, which highlights the efficacy of our method in token-efficient reasoning.

Notably, performing cold-start RL or warm-up alone is insufficient. While cold-start often underperforms the base instruction-tuned model, warm-up manages to outperform the SFT without CoT baseline, yet lags behind SFT with verbal CoT (except for Granite-3.3 on AlpacaEval). However, in tandem, these methods succeed, suggesting that the warm-up phase is indeed effective as a warm-start to bootstrap RL. We also observe that pause token fine-tuning, despite reducing the number of generated tokens to similar levels as Abstract-CoT, performs worse than the baseline in all settings, corroborating findings in prior works (Goyal et al., 2024; Hao et al., 2025) that pause fine-tuning without pause pre-training as an initialization proves ineffective.

4.3 DISTRIBUTION OF ABSTRACT TOKEN USAGE

We analyze the distribution of abstract token frequencies during RL training (in Figure 2), and observe an interesting phenomenon: RL shapes the distribution of tokens by virtue of their usage in abstract token sequences, resulting in a power law-like distribution over the tokens (as seen in Figure 3). While we initialize the token distribution for the warm-up as uniformly random, subsequent generation and training with on-policy sequences augments this distribution, resulting in an initial shape where a small number of tokens (4) are clearly invoked more than the rest, with some tokens being barely used. Then, we clearly see one token (specifically, <TOKEN_C>) begin to be used

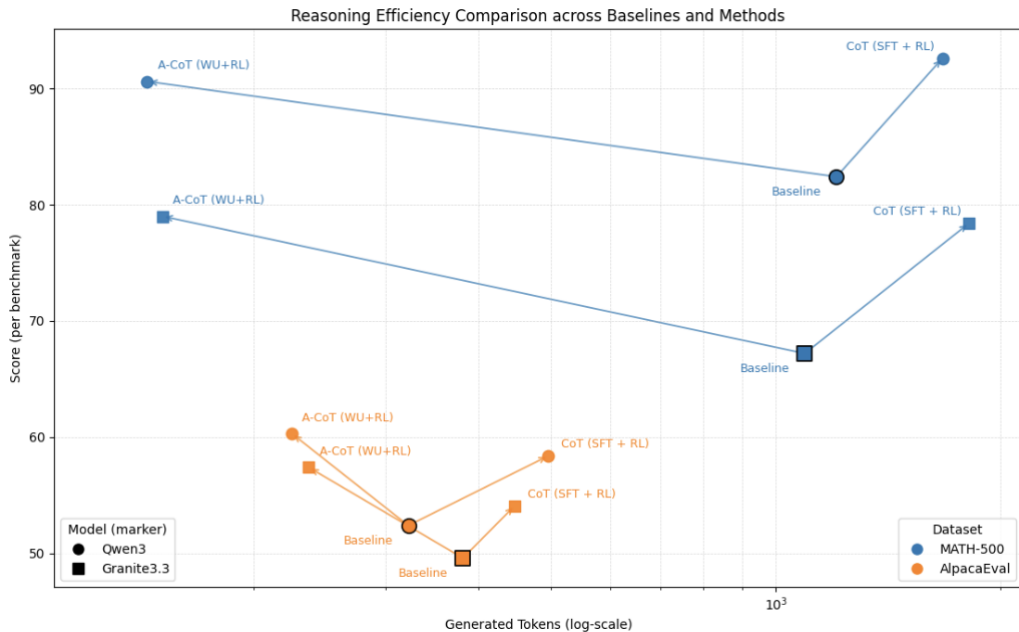


Figure 1: Plot of average generated (reasoning/rationale + response) tokens (log-scale) vs. benchmark scores (MATH-500 accuracy and AlpacaEval-LC-2.0 win-rate). We compare verbal CoT (SFT + RL) and Abstract-CoT (A-CoT for brevity) with policy iteration warm-up and warm-started RL (denoted by WU + RL). Compared to the baseline (initial model), Abstract-CoT drastically reduces the number of generated tokens while achieving similar or better performance on both benchmarks.

much more than the rest, suggesting that it is commonly used across diverse settings. On the other hand, some tokens with low frequency after warm-up began to appear in abstract sequences more often, such as the `<TOKEN_R>` token, suggesting the efficacy of the embedding learning stage in promoting token usage across the codebook and the value of using a diverse corpus in warm-up.

This behavior presents promise for future analysis on concept-level understanding, to study the mechanisms of concept learning with abstract tokens and how those concept representations evolve as a result of exploration during RL. While we do not perform compression of verbalized chain-of-thought, it is possible that similar concepts are learned and stored in the abstract tokens in a many-to-one fashion – a hypothesis that merits further study. New phenomena may also be unveiled as the size of the codebook M is scaled, dependent on the duration of RL training, the number of rollouts per prompt (we use 8 rollouts by default) and the diversity of the prompts.

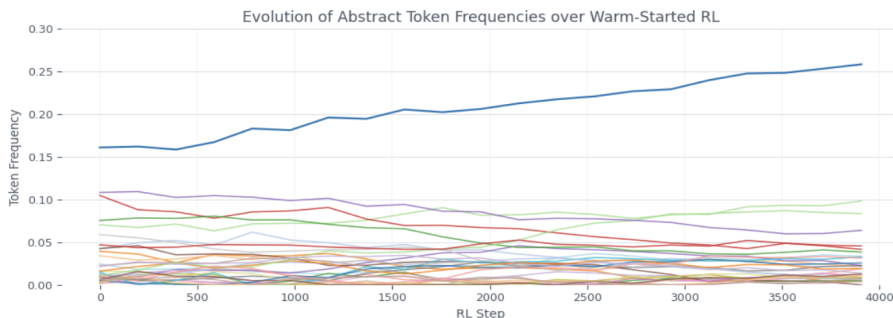


Figure 2: Evolution of the abstract token distribution over 1M episodes (≈ 3900 steps) of warm-started RL. We observe a clear divergence of the top token from the rest as the distribution of token usage in abstract sequences is shaped; simultaneously, some tokens that were sparsely used in the warm-up phase saw meaningful re-usage in RL, resulting in a gradual increase in frequency.

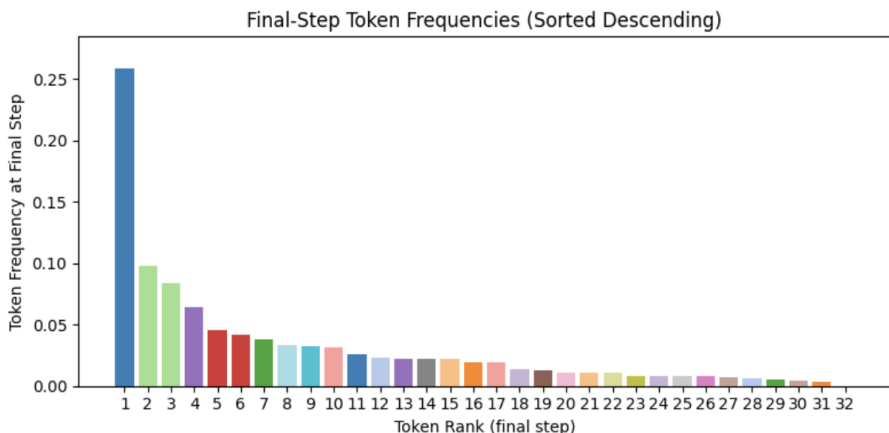


Figure 3: Distribution of final-step token frequencies from Figure 2, which demonstrates a power law characterization akin to Zipf’s law, induced purely through post-training. This provides an indication that a ”reasoning language” is being learned through token re-use across diverse contexts.

5 DISCUSSION

Abstract-CoT crucially lies between two extremes: fully verbal CoT (which is human-interpretable but expensive during inference) and fully implicit reasoning (efficient but challenging to control). The discrete abstract codebook offers a middle ground, yielding short, bounded traces while exposing a structured intermediate segment (between prompt and response) that can be analyzed. We attribute much of our work’s success to the warm-up phase: notably, the cold-start RL results with constrained decoding to the abstract codebook suggest that a more substantial burn-in (in terms of data volume and training strategy) is required to learn a prompt-response mapping through the abstract sequence. At the same time, the role of scale is evidenced by the RL results on top of multiple iterations of the policy iteration warm-up (600k SFT prompts + 102k RL prompts, 1M episodes).

Because Abstract-CoT exposes a short, explicit intermediate segment of filler tokens, it offers a practical path for inference scaling in strategies or regimes where only the final response is required. While Abstract-CoT obfuscates human-readability of the intermediate process, the abstract tokens provide an intermediate interface for chain-of-thought monitorability analysis (Korbak et al., 2025) and auditability for analysis of semantic concepts, beyond aggregate statistics.

As small budgets may be insufficient for long-horizon reasoning, budget-adaptive mechanisms with Abstract-CoT (with difficulty-aware abstract sequence lengths) offers exciting potential for more challenging regimes. While our findings on other initialization methods (Appendix A.1) proved ineffective compared to uniformity over the vocabulary, it is possible that a hierarchical structure over the codebook could enable re-usable subroutines for particular tasks, offering another mechanism to shape a (possibly larger) abstract vocabulary.

6 CONCLUSION

We introduce Abstract Chain-of-Thought, a discrete latent reasoning mechanism that post-trains language models to generate a short sequence of new ”abstract tokens”. A policy iteration warm-up loop learns abstract-token embeddings and sequence generation, while warm-started RL further improves performance through exploration. We find that Abstract Chain-of-Thought can produce substantially shorter sequences (12×) while achieving similar performance as post-training with verbalized CoT – these results also generalize across model families. Our findings suggest that filler tokens can be effectively introduced and used in post-training alone, offering a meaningful vocabulary-augmentation strategy to introduce a ”reasoning language” without requiring continued pre-training to be performed. Our method presents a new outlet for both inducing and analyzing latent reasoning behavior in LLMs, with fertile ground for future research in this direction.

REFERENCES

- Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. In *Second Conference on Language Modeling*, 2025. URL <https://openreview.net/forum?id=4jdIxXBNve>.
- AI2. Dolci-Think-RL-7B. Hugging Face Datasets, 2025a. URL <https://huggingface.co/datasets/allenai/Dolci-Think-RL-7B>. Accessed: 2026-02-08.
- AI2. Dolci-Think-SFT-7B. Hugging Face Datasets, 2025b. URL <https://huggingface.co/datasets/allenai/Dolci-Think-SFT-7B>. Accessed: 2026-02-08.
- Anonymous. Hybridcot: Interleaving latent and text chain-of-thought for efficient reasoning, 2026. URL <https://openreview.net/forum?id=4mfGbmzTwu>.
- Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. Soft tokens, hard truths, 2025. URL <https://arxiv.org/abs/2509.19170>.
- 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, 2025. URL <https://arxiv.org/abs/2505.16782>.
- Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations, 2024. URL <https://arxiv.org/abs/2412.13171>.
- 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>.
- Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpaca-eval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 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. URL <https://openreview.net/forum?id=ph04CRkPdC>.
- IBM Granite Team. Granite 3.0 language models, October 2024. URL <https://github.com/ibm-granite/granite-3.0-language-models/>.
- Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language model to reason in a continuous latent space, 2025. URL <https://openreview.net/forum?id=tG4SgayTtk>.
- 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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL <https://openreview.net/forum?id=7Bywt2mQsCe>.
- Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of LLMs via reinforcement learning. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL <https://openreview.net/forum?id=V51gPuluQD>.
- Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step: Outperforming larger language models with less training data. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.
- Yoonna Jang, Kisu Yang, and Isabelle Augenstein. Expanding computation spaces of large language models at inference time, 2025. URL <https://arxiv.org/abs/2509.24884>.
- Michael Kleinman, Matthew Trager, Alessandro Achille, Wei Xia, and Stefano Soatto. e1: Learning adaptive control of reasoning effort, 2025. URL <https://arxiv.org/abs/2510.27042>.

- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22*, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.
- Tomek Korbak, Mikita Balesni, Elizabeth Barnes, Yoshua Bengio, Joe Benton, Joseph Bloom, Mark Chen, Alan Cooney, Allan Dafoe, Anca Dragan, Scott Emmons, Owain Evans, David Farhi, Ryan Greenblatt, Dan Hendrycks, Marius Hobbhahn, Evan Hubinger, Geoffrey Irving, Erik Jenner, Daniel Kokotajlo, Victoria Krakovna, Shane Legg, David Lindner, David Luan, Aleksander Madry, Julian Michael, Neel Nanda, Dave Orr, Jakub Pachocki, Ethan Perez, Mary Phuong, Fabien Roger, Joshua Saxe, Buck Shlegeris, Martín Soto, Eric Steinberger, Jasmine Wang, Wojciech Zaremba, Bowen Baker, Rohin Shah, and Vlad Mikulik. Chain of thought monitorability: A new and fragile opportunity for ai safety, 2025. URL <https://arxiv.org/abs/2507.11473>.
- Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.
- Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. NuminaMath. [<https://huggingface.co/AI-MO/NuminaMath-CoT>] (https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.
- Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021.
- Charles London and Varun Kanade. Pause tokens strictly increase the expressivity of constant-depth transformers. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=eG5oh811WZ>.
- Chenwei Lou, Zewei Sun, Xinnian Liang, Meng Qu, Wei Shen, Wenqi Wang, Yuntao Li, Qingping Yang, and Shuangzhi Wu. Adacot: Pareto-optimal adaptive chain-of-thought triggering via reinforcement learning, 2025. URL <https://arxiv.org/abs/2505.11896>.
- William Merrill and Ashish Sabharwal. Exact expressive power of transformers with padding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=0labxStFcy>.
- Jesse Mu, Xiang Lisa Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=2DtxPCL3T5>.
- Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2025. URL <https://arxiv.org/abs/2512.13961>.
- OpenAI. gpt-oss-120b gpt-oss-20b model card, 2025. URL <https://arxiv.org/abs/2508.10925>.

- Jacob Pfau, William Merrill, and Samuel R. Bowman. Let’s think dot by dot: Hidden computation in transformer language models. In *First Conference on Language Modeling*, 2024. URL <https://openreview.net/forum?id=NikbrdtYvG>.
- Alok N. Shah, Khush Gupta, Keshav Ramji, and Pratik Chaudhari. Language modeling with learned meta-tokens, 2025. URL <https://arxiv.org/abs/2509.16278>.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.
- 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 Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.36. URL <https://aclanthology.org/2025.emnlp-main.36/>.
- DiJia Su, Hanlin Zhu, Yingchen Xu, Jiantao Jiao, Yuandong Tian, and Qingqing Zheng. Token assorted: Mixing latent and text tokens for improved language model reasoning. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.
- Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. System-1.5 reasoning: Traversal in language and latent spaces with dynamic shortcuts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=MNdUv07wAu>.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.
- Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. Tokenskip: Controlling chain-of-thought compression for efficient reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.
- Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23336–23351, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1137. URL <https://aclanthology.org/2025.acl-long.1137/>.
- JianZhi Yan, Le Liu, Youcheng Pan, Shiwei Chen, Zike Yuan, Yang Xiang, and Buzhou Tang. From long to lean: Performance-aware and adaptive chain-of-thought compression via multi-round refinement. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question

answering. 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. 2369–2380, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL <https://aclanthology.org/D18-1259/>.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Da, Da Zheng, Huajun Chen, and Ningyu Zhang. Lightthinker: Thinking step-by-step compression. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025a.

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, 2025b. URL <https://arxiv.org/abs/2505.15778>.

Rui-Jie Zhu, Tianhao Peng, Tianhao Cheng, Xingwei Qu, Jinfan 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>.

A ABLATIONS

A.1 WARM-UP INITIALIZATION SCHEMES

For the first iteration of the warm-up stage, given the embeddings for the abstract codebook are randomly initialized, we consider a teacher policy (schedule) for the abstract tokens as a prior for the first bottlenecked SFT phase. This selects which symbol from the codebook is selected for each token, once the number $m \leq m_{\max}$ of abstract tokens to generate has been decided (by random number generation corresponding to each step in the verbal CoT, as described in Section 3.2). We ablate three candidate schedules: (1) uniform at random selection; (2) a "heavy-tailed" distribution similar to Zipf's law, with a prior that the highest frequency tokens appear $2\times$ more frequently than the next two tokens, and so on; and (3) a cyclic, alphabetical schedule that uses the tokens in alphabetical order (if $m > M$, we restart from $\langle \text{TOKEN}_A \rangle$). All experiments are performed on Granite-3.3-8B-Instruct with a subset of the NuminaMath-CoT (Li et al., 2024) dataset, using 10k samples per stage, and $M = 5$ abstract tokens with $m_{\max} = 256$.

Table 2: Ablation of initialization schemes on MATH-500. "Bottlenecked SFT" denotes one warm-up SFT pass using a fixed initialization policy; "Policy Iteration (1 iter.)" denotes one full warm-up cycle (bottlenecked SFT + self-distillation). Each stage is trained for 3 epochs.

Method	MATH-500
Uniform Random	
Bottlenecked SFT	22.4
Policy Iteration (1 iter.)	58.8
Heavy-Tailed	
Bottlenecked SFT	21.2
Policy Iteration (1 iter.)	58.6
Alphabetical (cyclic)	
Bottlenecked SFT	20.8
Policy Iteration (1 iter.)	56.6

We find that the methods perform comparably in both setups, with cyclic performing the worse by ~ 1.6 - 2.2 pts after both stages. The poor performance of bottlenecked SFT (the first phase within our policy iteration loop) alone – due to forcing the model to generate responses while only attending to the abstract sequence – led us to introduce the self-distillation phase, where we induce the model to generate its own sequences of abstract tokens via constrained decoding. This was also a key motivator behind computing loss on the abstract sequence \tilde{z} in addition to the response y during the bottlenecked SFT phase, to facilitate a baseline for self-distillation to improve on. We find that on-policy generated sequences do appear to be useful in meaningfully increasing performance by $2.2 - 2.6\times$. Nonetheless, with a small volume of domain-specific data (and small M), this still lags well-behind the baseline for the Granite LLM, leading us to study increasing the number of iterations T for the Policy Iteration loop.

A.2 SCALING WARM-UP ITERATIONS

We include the complete results across 3 iterations of the warm-up, followed by RL, in Table 3. For most experiments, each iteration of warm-up gains over the previous ones, both in isolation and when used as a warm-start for RL. Given our findings with domain-specific data (NuminaMath) that the warm-up phase lags ($\approx 8 - 10$ pts.) behind the baseline, these experiments use the substantially larger and more diverse Dolci-Think- $\{\text{SFT}, \text{RL}\}$ corpora, as described in Section 4.1, and increases the number of iterations given the finding that on-policy-generated sequences boost performance (shown in Appendix A.1). This motivates the choice to abandon the token selection policy ϕ used for initialization after the first iteration, instead using on-policy generated sequences in the bottlenecked SFT phase from the second iteration onward. Our findings suggest that high-quality, diverse (multi-task) data with increased scale helps to facilitate both embedding learning and effective abstract sequence generation.

Table 3: We vary the policy iteration warm-up rounds ($T \in \{1, 2, 3\}$), denoted by PI- $\{T\}$.

Method	MATH-500	AlpacaEval	HotpotQA
Qwen3-8B			
Abstract-CoT (PI-1)	86.2	47.9	50.8
Abstract-CoT (PI-2)	87.0	52.2	52.1
Abstract-CoT (PI-3)	87.4	55.6	53.3
Abstract-CoT (PI-1+RL)	88.4	52.8	49.4
Abstract-CoT (PI-2+RL)	90.0	58.9	55.1
Abstract-CoT (PI-3+RL)	90.6	60.3	57.9
Granite-3.3-8B			
Abstract-CoT (PI-1)	66.2	42.8	34.2
Abstract-CoT (PI-2)	69.8	49.0	36.7
Abstract-CoT (PI-3)	71.6	52.8	40.8
Abstract-CoT (PI-1+RL)	74.4	48.2	38.6
Abstract-CoT (PI-2+RL)	78.8	55.0	43.4
Abstract-CoT (PI-3+RL)	79.0	57.4	45.1

B GENERATIVE REWARD MODEL PROMPT

The following prompt is used with `gpt-oss-20b` as a generative reward model in GRPO training with the "medium" thinking mode.

```
You are an expert evaluator assessing the quality of AI
assistant responses. Your task is to score a response on a
scale from 0 to 10.
```

```
## Scoring Criteria
```

```
Evaluate the response across these dimensions:
```

```
**Helpfulness (Does it address the user's needs?）**
```

- Directly answers the question or completes the task
- Provides appropriate level of detail
- Anticipates follow-up needs

```
**Accuracy (Is the information correct?）**
```

- Factually correct information
- No hallucinations or fabrications
- Appropriate caveats when uncertain

```
**Clarity (Is it easy to understand?）**
```

- Well-organized and structured
- Clear language appropriate to context
- Proper formatting when needed

```
**Relevance (Does it stay on topic?）**
```

- Addresses the actual query
- Avoids unnecessary tangents
- Appropriate scope and focus

```
**Safety & Harmlessness**
- No harmful, offensive, or inappropriate content
- Respectful and unbiased
- Considers ethical implications

## Scoring Scale

**10 - Exceptional** : Perfect or near-perfect response that
excels across all criteria
**9 - Excellent** : Outstanding response with only trivial
imperfections
**8 - Very Good** : Strong response that fully addresses the
query with minor areas for improvement
**7 - Good** : Solid response that meets expectations but has
some room for improvement
**6 - Above Average** : Decent response with noticeable
limitations
**5 - Average** : Adequate response that addresses the query
but has clear gaps
**4 - Below Average** : Partially helpful but with significant
issues in one or more criteria
**3 - Poor** : Minimally useful with major problems across
multiple criteria
**2 - Very Poor** : Severe deficiencies, barely addresses the
query
**1 - Extremely Poor** : Almost completely fails to address the
query
**0 - Unacceptable** : Completely fails to address the query,
contains harmful content, or is entirely inappropriate

Your Task

Provide your evaluation as a JSON object with the following
structure:

```json
"score": <number between 0-10>,
"reasoning": "<2-4 sentences explaining your score,
highlighting key strengths and weaknesses>"
```

---

Conversation Context
{CONVERSATION_HISTORY}

Assistant Response to Evaluate
{RESPONSE_TO_SCORE}
```

C EXAMPLES OF ABSTRACT CHAIN-OF-THOUGHT BEHAVIOR

Table 4: Example of mathematical problem solving with Abstract-CoT.

| | |
|---------------|---|
| Prompt | How many five-digit integers are divisible by 5 and have their digits sum to 20? |
| Abstract-CoT: | <pre> <beginabstract> <TOKEN_O> <TOKEN_M> <TOKEN_C> <TOKEN_AC> <TOKEN_A> <TOKEN_Q> <TOKEN_AD> <TOKEN_C> <TOKEN_R> <TOKEN_S> <TOKEN_C> <TOKEN_M> <TOKEN_U> <TOKEN_O> <TOKEN_V> <TOKEN_D> <TOKEN_G> <TOKEN_C> <TOKEN_Q> <TOKEN_A> <TOKEN_AD> <TOKEN_C> <TOKEN_AD> <TOKEN_M> <TOKEN_M> <TOKEN_C> <TOKEN_R> <TOKEN_R> <TOKEN_C> <TOKEN_X> <TOKEN_R> <TOKEN_C> <TOKEN_L> <TOKEN_AD> <TOKEN_C> <TOKEN_P> <TOKEN_C> <TOKEN_R> <TOKEN_Z> <TOKEN_M> <TOKEN_M> <TOKEN_C> <TOKEN_F> <TOKEN_R> <TOKEN_R> <TOKEN_AD> <TOKEN_C> <TOKEN_E> <TOKEN_C> <TOKEN_M> <TOKEN_R> <TOKEN_M> <TOKEN_C> <TOKEN_Y> <TOKEN_C> <TOKEN_R> <TOKEN_C> <TOKEN_C> <TOKEN_B> <TOKEN_N> <TOKEN_AD> <TOKEN_AD> <TOKEN_C> <TOKEN_AA> <TOKEN_U> <TOKEN_S> <TOKEN_R> <TOKEN_C> <TOKEN_Z> <TOKEN_AD> <TOKEN_C> <TOKEN_M> <TOKEN_R> <TOKEN_Q> <TOKEN_C> <TOKEN_C> <TOKEN_R> <TOKEN_AD> <TOKEN_M> <TOKEN_D> <TOKEN_M> <TOKEN_R> <TOKEN_O> <TOKEN_R> <TOKEN_A> <TOKEN_C> <endabstract> </pre> |
| Answer | Answer: There are 1116 five-digit integers divisible by 5 whose digits sum to 20. |

Table 5: An example of instruction-following with Abstract-CoT.

| | |
|---------------|--|
| Prompt | I'm starting high school and I procrastinate a lot. Can you give me a simple plan to stop procrastinating that I can actually follow, even on busy weeks? |
| Abstract-CoT: | <pre> <beginabstract> <TOKEN_T> <TOKEN_C> <TOKEN_L> <TOKEN_F> <TOKEN_C> <TOKEN_AD> <TOKEN_T> <TOKEN_C> <TOKEN_M> <TOKEN_R> <TOKEN_Q> <TOKEN_V> <TOKEN_L> <TOKEN_C> <TOKEN_F> <TOKEN_AE> <TOKEN_F> <TOKEN_T> <TOKEN_AD> <TOKEN_M> <TOKEN_R> <TOKEN_C> <TOKEN_L> <TOKEN_T> <TOKEN_C> <TOKEN_F> <TOKEN_L> <TOKEN_L> <TOKEN_C> <TOKEN_F> <TOKEN_AE> <TOKEN_C> <TOKEN_T> <TOKEN_AA> <TOKEN_M> <TOKEN_R> <TOKEN_C> <TOKEN_L> <endabstract> </pre> |
| Answer | <p>Starting high school can be exciting, but procrastination can feel overwhelming. Here's a simple, flexible plan to help you manage your time and stay on track:</p> <ol style="list-style-type: none"> 1. Start Small: The 2-Minute Rule: if a task takes less than 2 minutes, do it immediately (e.g., answer an email, organize your desk). This prevents small tasks from piling up. 2. Prioritize with the "Most Important Task First" (MIT): Every day, identify one task that's the most important (e.g., finishing a big project, studying for a test). Do this task first (even if it's short). 3. Break Tasks into Smaller Steps: If a task feels too big, split it into tiny steps (e.g., "Write a 2-page essay" → "Outline 1 paragraph today"). Check off each step as you go. 4. Use the "15-Minute Rule": If you're stuck, promise yourself: "'I'll work on this for 15 minutes. If I still don't want to, I'll stop.'" This tricks your brain into starting, and often you'll keep going!" 5. Schedule Daily "Focus Time": Block 30-60 minutes each day for focused work (e.g., homework, studying). If you're busy, shorten the time or do it in chunks. <p>You've got this! Start with one or two strategies, and adjust as you go. Over time, you'll build habits that make high school manageable and even enjoyable!</p> |