THE ILLUSION OF SUPERPOSITION IN LATENT CHAIN-OF-THOUGHT

Michael Rizvi-Martel
Mila & Université de Montréal
michael.rizvi-martel@mila.quebec

Marius Mosbach
Mila & McGill University
marius.mosbach@mila.quebec

ABSTRACT

Latent reasoning via continuous chain-of-thoughts (Latent CoT) has emerged as a promising alternative to discrete CoT reasoning. Operating in continuous space increases expressivity and has been hypothesized to enable *superposition*: the ability to maintain multiple candidate solutions simultaneously within a single representation. Despite theoretical arguments, it remains unclear whether language models actually leverage superposition when given continuous reasoning tokens. We investigate this question across two prominent latent CoT methods: Soft Thinking, a training-free approach that constructs reasoning embeddings as convex combinations of token embeddings, and Coconut, which fine-tunes models to reason with continuous latent thoughts. Using Logit Lens and entity-level probing to analyze internal representations, we find consistent evidence against superposition in both settings. For Soft Thinking, off-the-shelf models collapse superposed inputs to a single interpretation within the first few layers. For Coconut, the model learns to extract answers directly from the question embedding, achieving 96.6% accuracy without any latent tokens. Together, our results suggest that current latent CoT methods do not leverage superposition for reasoning both in training free and fine-tuned approaches.

1 INTRODUCTION

Chain-of-thought (CoT) reasoning has become the standard approach for tackling complex problems with large language models (LLMs), enabling them to break down problems by reasoning "step-by-step" (Wei et al., 2022). Various works have tried to further improve LLM reasoning through methods like self-consistency (Wang et al., 2022), tree-of-thoughts (Yao et al., 2023), and stream-of-search (Gandhi et al., 2024), and post-training LLMs on CoT data is now a crucial part of the LLM training pipeline (OpenAI, 2025; Guo et al., 2025). Exploring an alternative approach to reasoning, recent work proposed to have LLMs reason *directly in latent space* (Hao et al., 2024; Butt et al., 2025), which has been shown theoretically to be more expressive than discrete CoT Zhu et al. (2025). A compelling hypothesis for latent CoT's advantage is *superposition*: models could maintain multiple candidate solutions simultaneously, exploring several reasoning paths before committing to an answer (Zhu et al., 2025). This would represent a fundamental advantage over discrete CoT, which must commit to a single token at each step. However, there is little empirical evidence that LLMs leverage this capability, motivating the following research question:

Does superposition actually occur in latent CoT models?

We investigate this question across two complementary settings. First, we analyze Soft Thinking (Zhang et al., 2025), a training-free latent CoT method that creates superposition by computing linear combinations of input embeddings. Using Logit Lens (nostalgebraist, 2020) to probe internal representations, we find that off-the-shelf LLMs collapse superposed inputs to a single interpretation within the first few layers: entropy profiles are nearly identical to discrete CoT, KL divergences approach zero, and cosine similarities exceed 0.99. Second, we examine Coconut (Hao et al., 2024), a method that fine-tunes models to reason with continuous latent thoughts. Through entity-level probing, we show that the Coconut model learns to extract answers directly from the question representations, achieving 96.6% accuracy *without any latent tokens* and our belief evolution analysis

reveals no evidence of step-by-step reasoning during latent computation. Together, these results provide strong evidence that current latent CoT methods, whether training-free or fine-tuned, do not leverage superposition for reasoning.

2 Related Work

Latent Reasoning. Many works investigate the use of continuous tokens and latent representations in LLMs. Hao et al. (2024) show that finetuning LLMs to output a reasoning trace of continuous tokens provided considerable gains on logical reasoning tasks that require search during planning. Zhu et al. (2025) popularized the notion of "superposition" by showing theoretically that superposition in the latent state allows Transformer models to solve graph reachability tasks more efficiently. Follow-up work by Butt et al. (2025) proposes a novel method to train continuous CoTs via reinforcement learning which achieves comparable performance to discrete CoT on known math reasoning benchmarks. We base our analysis on the "Soft Thinking" approach by Zhang et al. (2025); a training-free method to generate latent CoTs based on convex combinations of embedding vectors. They report that their method offers a slight improvement on math benchmarks compared to discrete CoT baselines. Also close to our work, Deng et al. (2025) claim to have devised a training scheme which enables superposition in LLMs.

Alternative continuous thinking schemes have also been explored. Many works investigate the use of "filler" or "thinking" tokens; blank tokens which can be used to store intermediary computations (Pfau et al., 2024; Goyal et al., 2023; Herel & Mikolov, 2024). Moreover, there is a growing interest in "looped layers", a method which trains Transformers with recurrent attention layers Yang et al. (2023); McLeish et al. (2025). These methods also show promise in increasing reasoning abilities through additional latent computation.

Interpretability of Reasoning Models. Understanding the internal representations of NLP models has been a central research topic, even prior to LLMs (Adi et al., 2016; Linzen et al., 2016; Gulordava et al., 2018; Belinkov, 2022, *inter alia*). Recently, several works investigated how CoT reasoning changes internal model computations Yang et al. (2024); Dutta et al. (2024); Cywiński et al. (2025). However, to the best of our knowledge, no other works have attempted to understand the inner workings of latent CoT models from an interpretability perspective.

3 Method

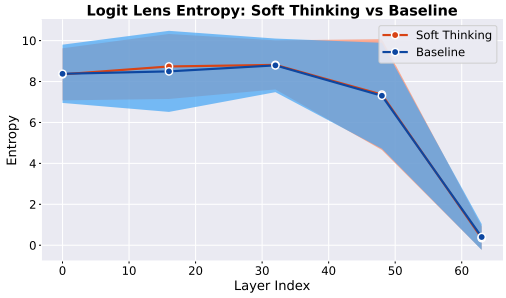
Let $\mathcal{M}$ be a transformer language model with L layers and vocabulary $\mathcal{V}$. We denote the *embedding matrix* by $E \in \mathbb{R}^{|\mathcal{V}| \times d}$, which maps discrete tokens to d-dimensional vectors, and the *unembedding matrix* by $U \in \mathbb{R}^{|\mathcal{V}| \times d}$, which projects hidden states back to the vocabulary space. For a token $k \in \mathcal{V}$, we write $e_k = E[k]$ for its embedding. Given an input sequence $\mathbf{x} = (x_1, \ldots, x_n)$, at each position i, $\mathcal{M}$ computes a hidden representation $h_i^{(\ell)} \in \mathbb{R}^d$ at each layer $\ell \in \{1, \ldots, L\}$.

CoT Generation. We consider a setting where, given an input query, a model generates a CoT followed by a final answer. A full sequence consists of *input tokens* $\mathbf{x} = (x_1, \ldots, x_n)$, *reasoning tokens* $\mathbf{r} = (r_1, \ldots, r_T)$, and *answer tokens* $\mathbf{y} = (y_1, \ldots, y_m)$. Generation proceeds autoregressively: at each step t, the model computes $p(r_t \mid \mathbf{x}, r_1, \ldots, r_{t-1})$ and selects a token according to some decoding strategy (we focus on greedy decoding). In *discrete* CoT, each $r_t \in \mathcal{V}$ is a vocabulary token with embedding e_{r_t} fed as input to the next step. The key difference with *latent* CoT is which point in embedding space is used: discrete CoT is constrained to the vocabulary manifold $\{e_k : k \in \mathcal{V}\}$, while latent CoT can use any point in embedding space.

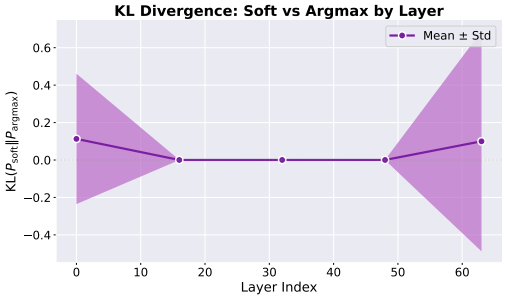
Soft Thinking. Soft Thinking (Zhang et al., 2025) constructs reasoning embeddings as convex combinations of token embeddings. At each step t, instead of selecting a discrete token, the method computes a distribution $p_t \in \Delta^{|\mathcal{V}|-1}$ over the vocabulary and forms the embedding

$$\tilde{e}_t = \sum_{k \in \mathcal{V}} p_{t,k}\, e_k, \tag{1}$$

which lies in the convex hull of vocabulary embeddings. According to Zhang et al. (2025), this method "naturally preserves a 'superposition' which retains the entire information in each step".



(a) Entropy comparison from entropy profile comparison experiment



(b) KL divergence for token intervention experiment

Figure 1: **Superposition collapses early in the forward pass of QwQ-32B. (a)** Shannon entropy shows identical patterns for Soft Thinking (orange) and discrete CoT (blue), both converging to near-zero entropy at the same rate. **(b)** KL divergence drops to $\sim 10^{-4}$ in middle layers, showing soft thinking tokens become functionally identical to discrete tokens within the first few layers. The uncertainty in soft thinking token embeddings does not propagate through the network.

Coconut. Coconut (Hao et al., 2024) takes a different approach: instead of constructing soft embeddings from vocabulary distributions, it feeds the model's own last hidden representation back as the next input embedding, enabling recurrent "reasoning in continuous latent space." The model is trained via a staged curriculum that progressively replaces discrete CoT tokens with these continuous latent thoughts. On ProsQA, a synthetic graph-traversal QA task, the authors report that latent tokens encode a breadth-first search (BFS) over the graph, citing logit-lens probing that reveals intermediate entities at latent positions. In Section Section 5, we revisit these claims and conduct an interpretability analysis on trained Coconut models.

Logit Lens. To understand how soft thinking tokens are processed, we employ Logit Lens (nostalgebraist, 2020), a technique for interpreting intermediate LLM computations. Normally, only the final layer representation $h_i^{(L)}$ is projected to vocabulary space via $p_i^{(L)} = \text{softmax}(U\, h_i^{(L)})$. Logit Lens applies this projection to intermediate representations, yielding $p_i^{(\ell)} = \text{softmax}(U\, h_i^{(\ell)})$ at any layer ℓ, revealing how predictions evolve across layers.

4 Do Off-the-Shelf Models Reason in Superposition?

If LLMs indeed leverage superposition, their internal representations when processing soft thinking tokens should differ meaningfully from discrete tokens, maintaining uncertainty and showing higher entropy at intermediate layers.

We test this hypothesis with two experiments: (1) a *side-by-side comparison* examining entropy profiles across layers when using latent CoT vs. discrete CoT, and (2) an *embedding-level intervention* measuring how changing a single token from soft thinking to discrete affects representations.

We use QwQ-32B (reasoning model) and Qwen2-1.5B (base; results in Appendix A) (Bai et al., 2023) and perform our analysis on 5 representative math problems of varying difficulty: 1 from GSM8K (Cobbe et al., 2021), 2 from MATH500 (Lightman et al., 2023), and 2 from AIME2024 (AMC, 2025)[1], applying Logit Lens at 5 strategic layers (see Appendix A.4 for the problem statements).

Benchmark results on the full datasets are reported in Appendix B.3.

4.1 COMPARING ENTROPY PROFILES IN LATENT VS. DISCRETE COT

We prompt both models with each problem, have them generate their respective CoTs, and at every 50 decoding steps apply Logit Lens to compute the Shannon entropy of the distribution over $\mathcal{V}$ at selected layers. If Soft Thinking reasons in superposition, we expect substantially higher entropy in its logit distributions compared to the discrete baseline.

Figure 1a shows entropy averaged over all CoT steps and problems. The entropy across layers is nearly *identical* for both approaches, with the same pattern: high entropy in early-to-middle layers collapsing to near-zero at the final layer. This is inconsistent with superposition. If the model truly maintained multiple solutions in parallel, Soft Thinking should exhibit higher entropy throughout; the indistinguishable profiles instead suggest that soft thinking tokens are processed like discrete CoT tokens, collapsing to a single interpretation early in the forward pass.

4.2 INTERVENING WITH DISCRETE TOKENS DURING SOFT THINKING

We perform an intervention experiment to test more directly whether internal representations differ when processing soft vs. discrete tokens. At every 50 steps of a Soft Thinking generation, we run two independent forward passes: one using the usual soft thinking token $\tilde{e}_t$, and one replacing it with the discrete argmax embedding $e_{\operatorname{argmax} p_t}$. For both, we apply Logit Lens at selected layers and compute KL divergence and cosine similarity between the resulting hidden representations. Only the intervened token changes; previous tokens in the KV cache remain soft thinking tokens.

Figure 1b shows KL divergence averaged over all thinking steps and problems. The divergence is strikingly small, dropping to approximately 10^{-4} in middle layers. Slightly larger divergence at the first layer reflects the initial embedding difference, while the small increase at the final layer corresponds to minor output logit differences. The near-zero divergence in middle layers indicates that soft thinking tokens and their discrete counterparts become functionally equivalent by the core computational layers; superposition encoded in the input is eliminated within the first few layers. Cosine similarity corroborates this: the average across all layers, steps, and problems is 0.9980 ± 0.0133. Moreover, Figure 2 shows that top predicted tokens are virtually identical for both approaches.

This finding is robust to the entropy of the soft distribution. Filtering to thinking steps where the soft distribution entropy exceeds the 75th percentile yields KL ≈ 0.022 for QwQ-32B (still negligibly small in absolute terms) and KL $\approx 5 \times 10^{-4}$ for Qwen2-1.5B. Even on the most ambiguous steps, the model collapses superposition just as rapidly.

Together, these results demonstrate that soft thinking tokens are not leveraged meaningfully. Superposition collapses almost immediately, and computation proceeds as if a discrete token were provided, constituting strong evidence against superposition in off-the-shelf LLMs.

5 DO FINE-TUNED MODELS REASON IN SUPERPOSITION?

The previous section showed that off-the-shelf LLMs do not leverage superposition when given superposed inputs. A natural follow-up question is whether models *trained* for latent reasoning behave differently. We investigate this using Coconut (Hao et al., 2024) (see Section 3 for background).

5.1 EXPERIMENTAL SETUP

We evaluate GPT-2 (124M) on ProsQA Hao et al. (2024), a synthetic graph-traversal QA task requiring multi-hop logical inference over defined relationships (e.g., *"Every dax is a wug. Every wug is a blicket. Rex is a dax. Is Rex a blicket or a gorple? Blicket."*). The CoT baseline is trained with standard chain-of-thought supervised fine-tuning. The Coconut model is trained with the staged curriculum of Hao et al. (2024), which progressively replaces CoT steps with continuous latent tokens. We probe by projecting hidden states at each reasoning position through the LM head. In order to avoid confounds stemming from contextualization, we probe latent tokens at their respective thinking steps, rather than at the end.

[1]We observe very consistent trends across examples and benchmarks, and hence, chose to focus on a small representative set of problems for our analysis.

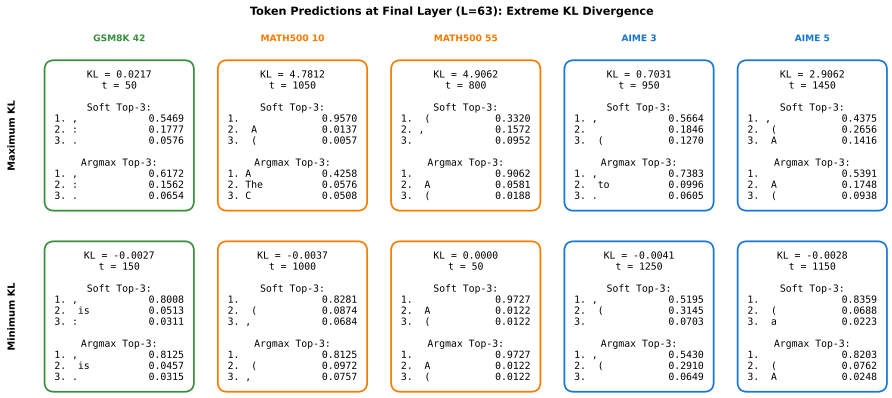


Figure 2: Top 3 tokens at the output layer for time steps with largest (top) and smallest (bottom) KL divergence. Each column represents a problem instance.

5.2 Probing Reveals No Step-by-Step Reasoning

We first examine belief about the target entity immediately after processing the question, before any reasoning tokens are produced. Here, contrary to the original paper's analysis, we take the normalized distribution over *all legal entities*. The original paper only looks at *valid next neighbor* entities. In order to better understand how belief evolves through the CoT, we track how the normalized entity distribution evolves across reasoning steps. At each step, graph entities are categorized into four groups: *correct next* (the right answer for the given step), *wrong neighbors*, *target*, and *other*.

As can be seen in Figure 3, the target entity dominates the entity distribution throughout the entire reasoning process. For CoT, the correct-next entity dominates early steps and the target takes over only at the final step, the expected signature of step-by-step multi-hop reasoning. Note that for CoT, the probing is done with teacher forcing whereas for Coconut, the latents are probed directly. This behavior suggests that the model learned a *shortcut solution*: it first obtains the correct answer in a single forward pass, then copies it over through the latent tokens.

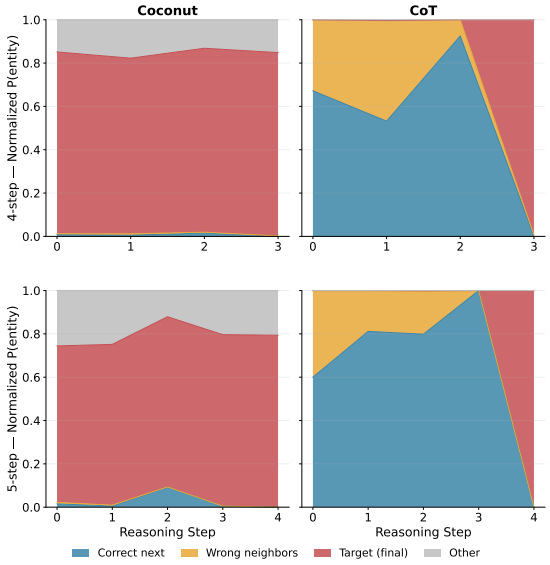


Figure 3: Step-aware entity belief (normalized among all entities). **Coconut (left column)**: target entity dominates from step 0 onward with no progression. **CoT (right column)**: correct-next entity dominates early steps, target takes over at the final step, the expected pattern for genuine multi-hop reasoning. Pattern holds across 4-step (top) and 5-step (bottom) examples.

5.3 Latent Tokens Are Unnecessary to Performance

Finally, we evaluate the trained Coconut model by feeding *only the question* (no latent tokens, no multi-pass recurrence) and greedily decoding the answer. As can be seen in Table 1, the model achieves **96.6% accuracy** under this regime; this suggests that the latent thinking only accounts for a 3% contribution in the performance of the Coconut model. We hypothesize that this increase in performance is due to an *echo chamber* phenomenon. For cases where $P(\text{target})$ is lower at step 0, it is possible the "latent thinking" procedure increases the probability.

Table 1: ProsQA accuracy under different reasoning conditions.

Condition	Accuracy
CoT (discrete, 5-hop)	85.3%
Coconut (6 latent tokens)	99.0%
Coconut (no latent tokens)	96.6%

6 Discussion and Conclusion

In conclusion, our experiments provide consistent evidence against reasoning in superposition across two complementary settings. For Soft Thinking, off-the-shelf LLMs process superposed inputs nearly identically to discrete tokens: entropy profiles match, KL divergences approach zero, and cosine similarities exceed 0.99. For Coconut, a fine-tuned model achieves 96.6% accuracy without any latent tokens, and entity-level probing reveals no step-by-step reasoning during latent computation.

Why does superposition collapse in pretrained models? As we argue above, this is not merely an inductive bias but a consequence of the pretraining objective: autoregressive training optimizes for discrete next-token prediction, creating representations that separate token identities. When presented with a superposed input, the model projects it onto the nearest discrete interpretation, precisely what the training objective rewards. Moreover, the Soft Thinking distribution p_t is itself very peaky across steps (see Figure 2 and the entropy heatmaps in Appendix A), meaning the input is already near-discrete and there is little superposition to exploit in the first place.

Is token-level superposition desirable? Beyond finding that models do not reason in superposition, we question whether token-level superposition is a desirable property in the first place. Many soft thinking tokens combine semantically unrelated items (see Figure 2): formatting characters, punctuation, or tokens with similar logits but unrelated meanings. A superposition of "(" and "{" represents syntactic uncertainty, not exploration of alternative reasoning paths. Meaningful parallel exploration likely requires superposition at a higher level of abstraction (over entire reasoning strategies, not individual tokens), which we consider an interesting avenue for future work.

Latent reasoning as flexibility. Despite our negative findings for current methods, latent reasoning remains a promising direction. The advantage of continuous embeddings may not be superposition but *flexibility*: the ability to express intermediate computations that do not correspond to natural language tokens, avoiding the discretization bottleneck. Future work should investigate whether latent reasoning provides benefits through other mechanisms, such as smoother optimization landscapes or more expressive intermediate representations.

Future work. It would be valuable to run similar experiments on other latent reasoning approaches, such as the RL-trained continuous CoTs of Butt et al. (2025). Moreover, our Coconut analysis is limited to ProsQA; testing on more complex tasks would clarify whether question-to-answer shortcuts are specific to this synthetic setting or a more general phenomenon.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*, 2016.

AMC. American invitational mathematics examination. https://artofproblemsolving.com/wiki/index.php/American_Invitational_Mathematics_Examination, 2025.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, March 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7/.

Natasha Butt, Ariel Kwiatkowski, Ismail Labiad, Julia Kempe, and Yann Ollivier. Soft tokens, hard truths. *arXiv preprint arXiv:2509.19170*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Bartosz Cywiński, Emil Ryd, Senthooran Rajamanoharan, and Neel Nanda. Towards eliciting latent knowledge from llms with mechanistic interpretability. *arXiv preprint arXiv:2505.14352*, 2025.

Jingcheng Deng, Liang Pang, Zihao Wei, Shichen Xu, Zenghao Duan, Kun Xu, Yang Song, Huawei Shen, and Xueqi Cheng. Latent reasoning in llms as a vocabulary-space superposition. *arXiv preprint arXiv:2510.15522*, 2025.

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *arXiv preprint arXiv:2402.18312*, 2024.

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D Goodman. Stream of search (sos): Learning to search in language. *arXiv preprint arXiv:2404.03683*, 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. *arXiv preprint arXiv:2310.02226*, 2023.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. Colorless green recurrent networks dream hierarchically. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1195–1205, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1108. URL https://aclanthology.org/N18-1108/.

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.

David Herel and Tomas Mikolov. Thinking tokens for language modeling. *arXiv preprint arXiv:2405.08644*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016.

Sean McLeish, Ang Li, John Kirchenbauer, Dayal Singh Kalra, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Jonas Geiping, Tom Goldstein, and Micah Goldblum. Teaching pretrained language models to think deeper with retrofitted recurrence. *arXiv preprint arXiv:2511.07384*, 2025.

nostalgebraist. interpreting gpt: the logit lens, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

OpenAI. OpenAI o3 and o4-mini System Card. Technical report, OpenAI, San Francisco, CA, April 2025. URL https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf. PDF available online.

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.

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

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. *arXiv preprint arXiv:2311.12424*, 2023.

Sohee Yang, Nora Kassner, Elena Gribovskaya, Sebastian Riedel, and Mor Geva. Do large language models perform latent multi-hop reasoning without exploiting shortcuts? *arXiv preprint arXiv:2411.16679*, 2024.

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.

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.

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*, 2025.

A Soft Thinking: Experimental Details

A.1 Models

We use two models from the Qwen family (Bai et al., 2023):

- **QwQ-32B**: A 32.5B-parameter reasoning model based on the Qwen2.5 architecture with 64 transformer layers and a hidden dimension of 5120. We use this as our primary model since it was trained for chain-of-thought reasoning.
- **Qwen2-1.5B**: A 1.5B-parameter base language model with 28 transformer layers and a hidden dimension of 1536. We use this smaller model to test whether our findings generalize across model scales.

Both models use the same tokenizer with a vocabulary of 151,643 tokens. All experiments use the models in `bfloat16` precision.

Table 2: Soft Thinking decoding hyperparameters.

Parameter	Symbol	Value
Temperature	τ	0.6
Top-k (sampling)	k	30
Soft top-k (embedding mix)	k_{soft}	15
Weighting scheme		Softmax
Max new tokens	$T_{\max}$	2048

A.2 Soft Thinking Configuration

We use the following decoding hyperparameters for all Logit Lens experiments, consistent across both models:

At each reasoning step t, the model computes logits over the vocabulary and selects the top-k_{soft} tokens. Their logits are normalized via softmax to obtain the mixing weights $p_t \in \Delta^{k_{\text{soft}}-1}$, and the soft thinking embedding is formed as $\tilde{e}_t = \sum_{i=1}^{k_{\text{soft}}} p_{t,i}\, e_{v_i}$ where $v_1, \ldots, v_{k_{\text{soft}}}$ are the top-k_{soft} tokens by logit value.

For the benchmark evaluation runs (Tables 4 and 5), we additionally vary the cold stop threshold $c \in \{0.0, 0.01, 0.1, 0.2\}$, which terminates soft thinking when the top token probability exceeds $1 - c$.

A.3 Logit Lens Setup

We apply Logit Lens at 5 evenly spaced probe layers: $\{0, \lfloor L/4 \rfloor, \lfloor L/2 \rfloor, \lfloor 3L/4 \rfloor, L-1\}$, where L is the number of transformer layers. This corresponds to layers $\{0, 7, 14, 21, 27\}$ for Qwen2-1.5B and $\{0, 16, 32, 48, 63\}$ for QwQ-32B.

Entropy profile comparison (Section 4.1). For each problem, we run two independent generations: one using Soft Thinking and one using standard discrete decoding (greedy argmax). Every 50 decoding steps, we apply Logit Lens at each probe layer and record the Shannon entropy of the resulting distribution over V. We also store the top-10 predicted tokens at each checkpoint for qualitative comparison.

Token-level intervention (Section 4.2). During Soft Thinking generation, we intervene every 50 steps by performing two fresh forward passes over the full sequence: one using the soft thinking embedding and one replacing it with the argmax embedding. Crucially, only the current token's embedding differs; the KV cache from previous (soft) tokens is shared. At each probe layer, we compute:

- KL divergence: $D_{\text{KL}}(p^{(\ell)}_{\text{soft}} \| p^{(\ell)}_{\text{argmax}})$ between the Logit Lens distributions.
- Cosine similarity: $\cos(h^{(\ell)}_{\text{soft}}, h^{(\ell)}_{\text{argmax}})$ between hidden representations.
- Entropy difference: $H(p^{(\ell)}_{\text{soft}}) - H(p^{(\ell)}_{\text{argmax}})$.
- Top-k token overlap: $|S^{(\ell)}_{\text{soft}} \cap S^{(\ell)}_{\text{argmax}}|/k$ for $k = 10$.

A.4 Evaluation Problems

We select 5 problems of varying difficulty from three standard math reasoning benchmarks. The same problems are used across all experiments and both models to enable direct comparison.

GSM8K, Problem 42 (Easy)

Grandma Jones baked 5 apple pies for the fireman's luncheon. She cut each pie into 8 pieces and set the five pies out on the buffet table for the guests to serve themselves. At the end of the evening, after the guests had taken and eaten their pieces of pie, there were 14 pieces of pie remaining. How many pieces were taken by the guests?

Answer: 26

MATH500, Problem 10 (Medium)

What is the least positive integer multiple of 30 that can be written with only the digits 0 and 2?

Answer: 2220

MATH500, Problem 55 (Medium)

Suppose that f is a polynomial such that $(x-1)\cdot f(x) = 3x^4 + x^3 - 25x^2 + 38x - 17$. What is the degree of f?

Answer: 3

AIME 2024, Problem 3 (Hard)

Let x, y and z be positive real numbers that satisfy the following system of equations:

$$\log_2\left(\frac{x}{yz}\right) = \frac{1}{2}, \quad \log_2\left(\frac{y}{xz}\right) = \frac{1}{3}, \quad \log_2\left(\frac{z}{xy}\right) = \frac{1}{4}.$$

Then the value of $\left|\log_2(x^4y^3z^2)\right|$ is $\frac{m}{n}$ where m and n are relatively prime positive integers. Find $m+n$.

Answer: 33

AIME 2024, Problem 5 (Hard)

Alice chooses a set A of positive integers. Then Bob lists all finite nonempty sets B of positive integers with the property that the maximum element of B belongs to A. Bob's list has 2024 sets. Find the sum of the elements of A.

Answer: 55

A.5 Compute

All experiments were run on a SLURM cluster using NVIDIA L40S GPUs (48 GB each). Table 3 summarizes the resource allocation for each experiment.

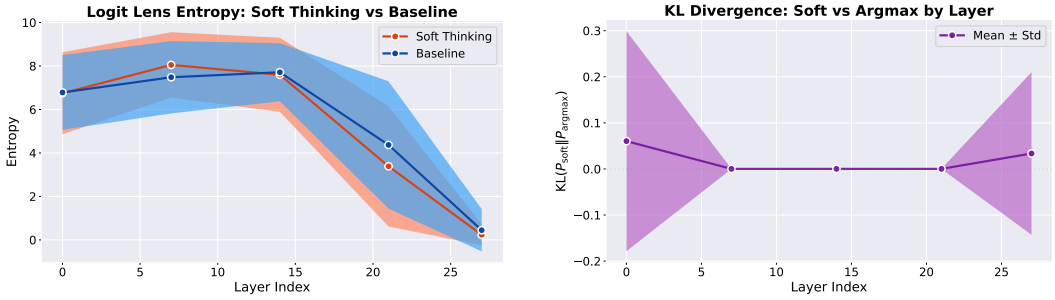
Table 3: Compute resources per experiment.

Experiment	Model	GPUs	RAM	Wall Time
Comparative (Logit Lens)	Qwen2-1.5B	1× L40S	32 GB	$<$ 1 h
Comparative (Logit Lens)	QwQ-32B	4× L40S	128 GB	$<$ 2 h
Convergence	Qwen2-1.5B	1× L40S	32 GB	$<$ 1 h
Convergence	QwQ-32B	4× L40S	128 GB	$<$ 3 h
Benchmark (MATH500)	QwQ-32B	4× L40S	128 GB	varies
Benchmark (AIME2024)	QwQ-32B	4× L40S	128 GB	varies
Coconut (ProsQA)	GPT-2	4× L40S	128 GB	$\sim$ 3 h

For QwQ-32B, we distribute the model across 4 GPUs using HuggingFace Accelerate's `device_map="balanced"` strategy with a per-GPU memory cap of 75% to avoid out-of-memory errors from uneven shard allocation.

B SOFT THINKING: ADDITIONAL RESULTS

B.1 AGGREGATE LOGIT LENS RESULTS (QWEN2-1.5B)



(a) Entropy comparison

(b) KL divergence by layer

Figure 4: **Logit Lens analysis for Qwen2-1.5B. (a)** Shannon entropy shows identical patterns for Soft Thinking (orange) and discrete CoT (blue), both converging to near-zero entropy at the same rate. **(b)** KL divergence between soft and argmax token representations drops to $\sim 10^{-4}$ in middle layers. Results mirror those of QwQ-32B (Figure 1), indicating superposition collapse is consistent across model scales.

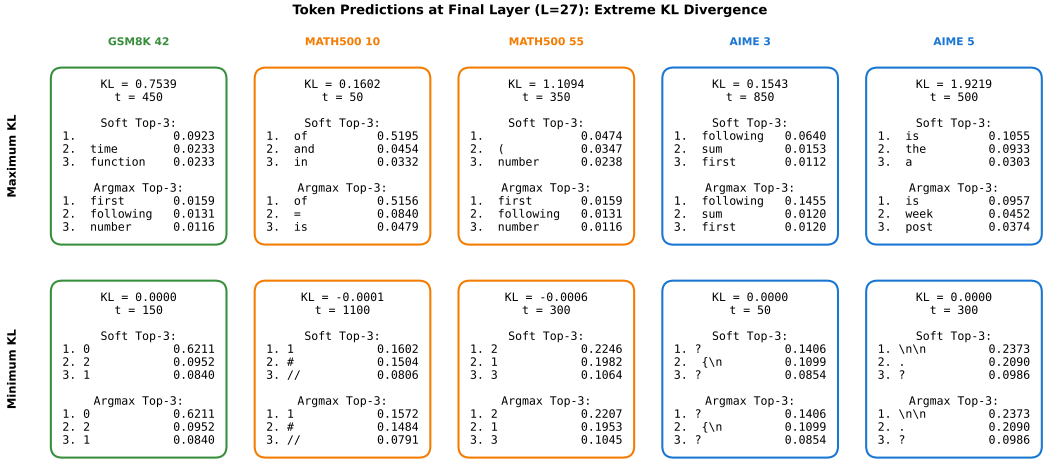


Figure 5: Top 3 predicted tokens at the output layer for time steps with largest (top) and smallest (bottom) KL divergence between soft and argmax representations in **Qwen2-1.5B**. Each column represents a problem instance.

B.2 PER-PROBLEM VISUALIZATIONS

B.2.1 TOKEN INTERVENTION

At every 50 decoding steps during Soft Thinking generation, we replace the current soft token embedding with the discrete argmax embedding and compare hidden-state representations through layers. Figures 6 and 7 show per-problem KL divergence heatmaps, Figures 8 and 9 show cosine similarity, and Figures 10 and 11 show absolute entropy difference.

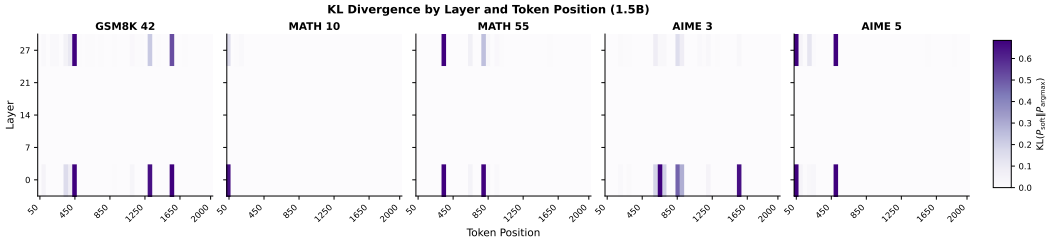


Figure 6: KL divergence $D_{\mathrm{KL}}(p^{(\ell)}_{\mathrm{soft}} \| p^{(\ell)}_{\mathrm{argmax}})$ by layer and token position for each problem (**Qwen2-1.5B**). Middle layers consistently show near-zero KL, confirming rapid superposition collapse.

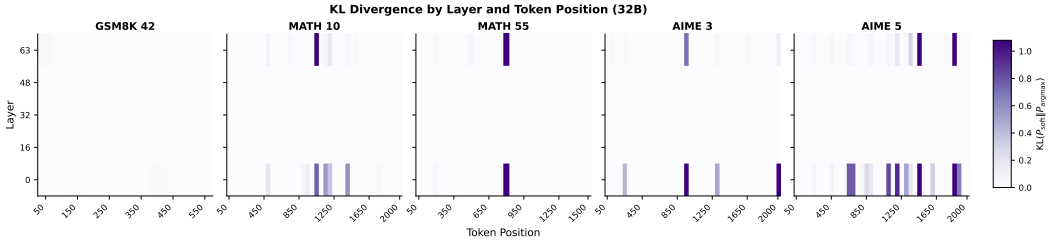


Figure 7: KL divergence by layer and token position for each problem (**QwQ-32B**). Same pattern as the 1.5B model: near-zero divergence in middle layers.

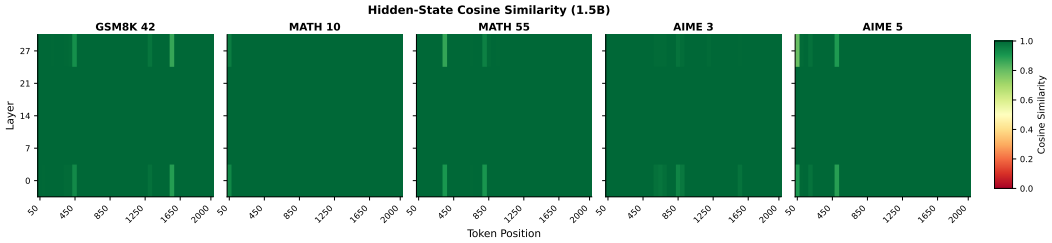


Figure 8: Cosine similarity between soft and argmax hidden states by layer and token position (**Qwen2-1.5B**). Values are near 1.0 across all layers and problems.

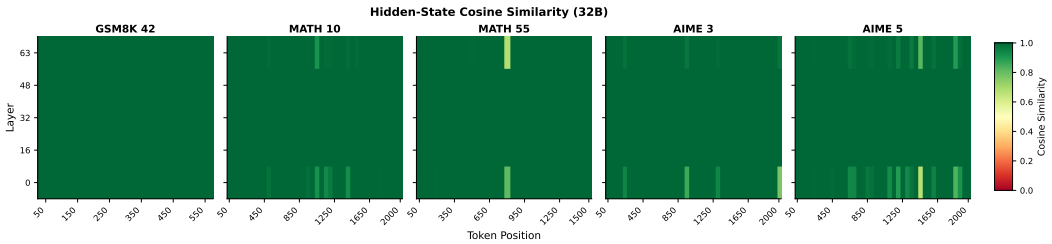


Figure 9: Cosine similarity between soft and argmax hidden states (**QwQ-32B**).

Figure 10: Absolute entropy difference $|H(p^{(\ell)}_{\text{soft}}) - H(p^{(\ell)}_{\text{argmax}})|$ by layer and token position (**Qwen2-1.5B**).

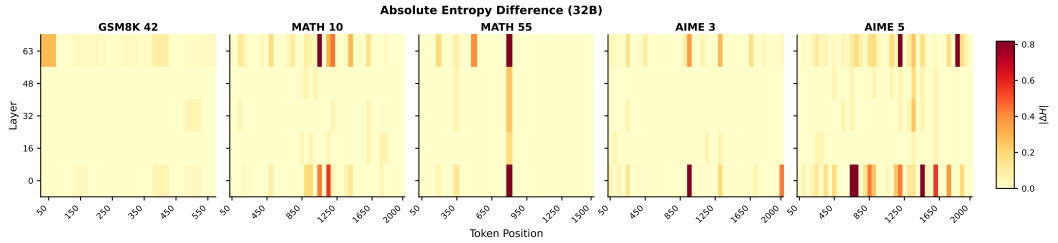


Figure 11: Absolute entropy difference by layer and token position (**QwQ-32B**).

B.2.2 Entropy Profiles

We compare full reasoning traces generated independently with Soft Thinking and discrete decoding. Figures 12 and 13 show entropy at each probe layer over the course of generation. In both models, entropy profiles are virtually indistinguishable between the two modes.

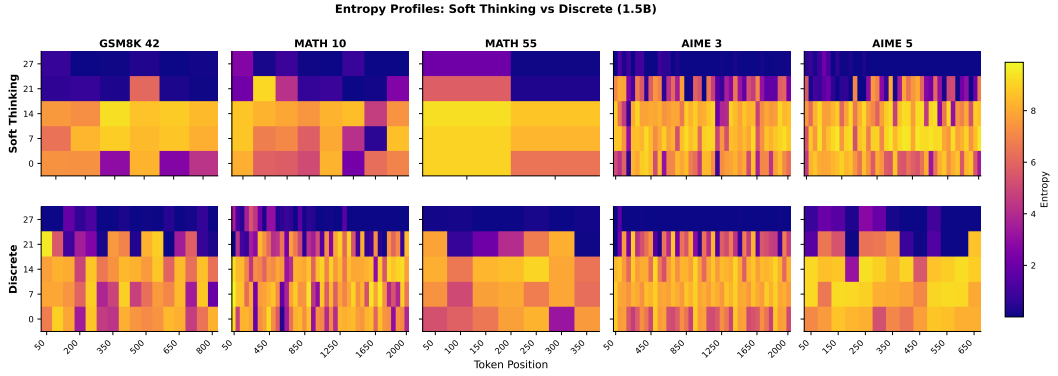


Figure 12: Per-problem entropy heatmaps for **Qwen2-1.5B**. Top row: Soft Thinking; bottom row: discrete decoding. Shared colorbar. Entropy patterns are nearly identical across both modes.

B.3 Benchmark Results

C CoCoNut: Experimental Details

C.1 ProsQA Task

ProsQA (Hao et al., 2024) is a synthetic graph-traversal QA task designed to evaluate multi-hop reasoning. Each example consists of a randomly generated directed graph over named entities, a starting node, and a target node reachable via a sequence of directed edges. The question provides the graph structure (as a list of edges) and asks for the entity reachable from a given starting node

Figure 13: Per-problem entropy heatmaps for **QwQ-32B**. Top row: Soft Thinking; bottom row: discrete decoding.

Table 4: MATH500 results with QwQ-32B (N=500). Discrete decoding uses standard greedy/sampling; Softmax and Uniform refer to the weighting scheme used to construct soft thinking embeddings.

Run	Decoding	max_topk	Cold Stop	Accuracy (%)	Avg Tokens
1	Discrete	–	0.0 (none)	97.08	4,326
2	Discrete	–	0.1	97.08	4,326
3	Discrete	–	0.2	**97.19**	4,307
4	Softmax	10	0.0 (none)	96.47	4,222
5	Softmax	10	0.1	96.84	4,056
6	Softmax	15	0.01	96.84	4,044
7	Softmax	15	0.1	96.80	4,003
8	Uniform	3	0.1	93.97	5,388

Table 5: AIME2024 results with QwQ-32B (N=30).

Run	Decoding	max_topk	Cold Stop	Accuracy (%)	Avg Tokens
1	Discrete	–	0.0 (none)	**77.29**	13,445
2	Discrete	–	0.1	**77.29**	13,445
3	Discrete	–	0.2	**77.29**	13,445
4	Softmax	15	0.01	75.83	12,445
5	Softmax	15	0.1	76.25	11,818

after a specified number of hops. The ground-truth chain-of-thought consists of the sequence of intermediate entities visited during the traversal.

We use the ProsQA dataset provided with the original Coconut codebase, which contains 17,886 training examples and 300 validation examples. Graph depths range from 3 to 6 hops, with the distribution concentrated at 4 and 5 hops.

C.2 Training Setup

We train GPT-2 (124M parameters, 12 layers, hidden dimension 768) on ProsQA using the Coconut training procedure of Hao et al. (2024). Table 6 summarizes the key hyperparameters.

Table 6: Coconut training hyperparameters on ProsQA.

Parameter	**CoT baseline**	**Coconut**
Base model	GPT-2 (124M)	GPT-2 (124M)
Learning rate	10^{-4}	10^{-4}
Optimizer	AdamW	AdamW
Weight decay	0.01	0.01
Batch size (per device)	16	16
Gradient accumulation steps	2	2
Total epochs	50	50
Epochs per stage	–	5
Latent tokens per step (c_{thought})	–	1
Max latent stage	–	6
Precision	FP32	FP32

Training follows a staged curriculum. At stage k (epochs $5k$ through $5(k+1)-1$), the first k chain-of-thought steps are replaced by k continuous latent tokens; the remaining steps are kept as discrete text. The model is trained to predict only the remaining CoT steps and the final answer (labels for question and latent positions are masked with -100). By stage 6 (epochs 30–34), all reasoning steps have been replaced by latent tokens, and the model must produce the answer using only continuous latent computation. The optimizer is reset at the start of each epoch (`reset_optimizer=True`).

Three special tokens are added to the vocabulary: `<|start-latent|>`, `<|latent|>`, and `<|end-latent|>`, all initialized from the embedding of the << token. During the forward pass, each `<|latent|>` token's embedding is replaced by the last hidden state from the preceding forward pass, implementing the continuous thought recurrence. A custom collator left-pads batches to align latent token positions across examples, enabling KV cache reuse in the multi-pass forward.

We use `torchrun` with 4 GPUs; FSDP wraps the model but does not shard GPT-2's layers (effectively acting as DDP at this model scale). The best CoT checkpoint is at epoch 49 (85.3% validation accuracy); the best Coconut checkpoint is at epoch 50 (99% validation accuracy).

C.3 Probing Methodology

Entity-level probing. At each reasoning position k (CoT step or latent token), we extract the hidden state at the final transformer layer and project it through the LM head to obtain a distribution over the full vocabulary. We then restrict this distribution to the set of graph entity tokens and renormalize, yielding a distribution over entities. Entities are grouped into four categories: *correct next* (the entity the model should traverse to at step k), *target* (the final answer), *wrong neighbors* (adjacent but incorrect entities), and *other*.

`per_pass` vs. `final_state` probing. As noted in Section 5.1, the original Coconut paper probes latent positions using the hidden states obtained after all forward passes are complete (`final_state`). At position k, this hidden state has been influenced by all subsequent latent passes via the accumulated KV cache; it is analogous to probing CoT step 1 after the model has already processed steps 2 through 5. We instead use `per_pass` probing: at latent step k, we probe the hidden state immediately after the k-th forward pass, before any subsequent passes occur. This gives

a fair comparison with CoT, where each step's representation reflects only the reasoning completed up to that point.

No-latent evaluation. In order to test whether latent tokens are necessary for the model's performance, we feed only the question tokens to the trained Coconut model (with no latent tokens and no multi-pass recurrence) and greedily decode the answer.

D COCONUT: ADDITIONAL RESULTS

D.1 ENTITY DISTRIBUTIONS

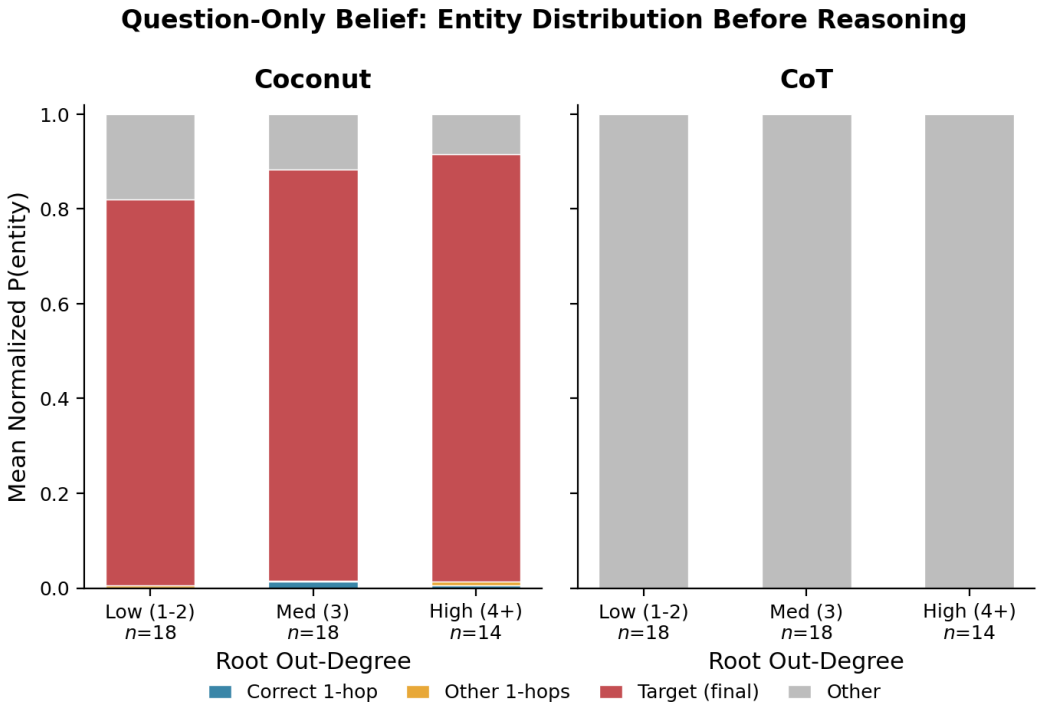


Figure 14: Entity distribution *before any reasoning*. Coconut (left) already assigns ∼82% normalized probability to the target. CoT (right) has near-zero target probability, with mass spread across non-target entities.

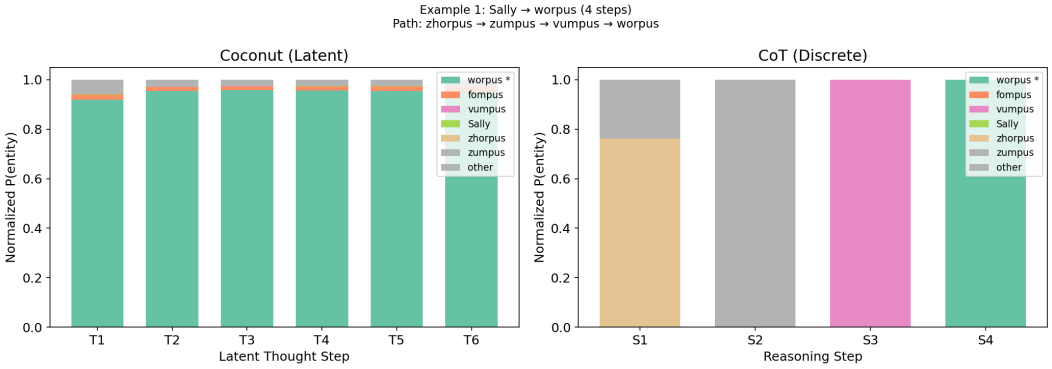


Figure 15: Per-step entity distribution for a 4-hop example. Coconut (left) locks onto the final target immediately. CoT (right) traverses intermediate entities step by step.

D.2 COCONUT LOGIT LENS ANALYSIS

As a complementary analysis, we apply Logit Lens (nostalgebraist, 2020) to both the Coconut and CoT models from Section 5. At each thinking position (latent tokens for Coconut, step-end tokens for CoT), we extract the hidden state at every transformer layer $\ell \in \{0, \ldots, 11\}$, project it through the final layer norm and LM head, and compute the Shannon entropy of the resulting vocabulary distribution.

Each point in Figure 16 shows the average entropy at layer ℓ, where the average is taken first across all thinking positions within an example, then across all examples ($n = 300$). Shaded regions show ± 1 standard deviation across examples.

For the CoT model, entropy is high in early and middle layers ($\sim$4 nats at layers 1–6), dropping sharply to near zero by layers 8–9. This indicates gradual refinement: early layers maintain broad uncertainty over the vocabulary, and the model progressively commits to specific tokens in later layers. The late-layer spike in CoT entropy (layers 10–11) likely reflects the model broadening over surface-form variations of the committed entity (e.g., capitalization, spacing).

For the Coconut model, entropy is uniformly low across all layers ($<$1.5 nats), indicating that the model's hidden states at latent positions are already near-deterministic even at early layers. The representations carry little information content that would suggest active computation. This is consistent with the entity-level probing results in Section 5.2: the model has already committed to the target entity before any latent reasoning begins.

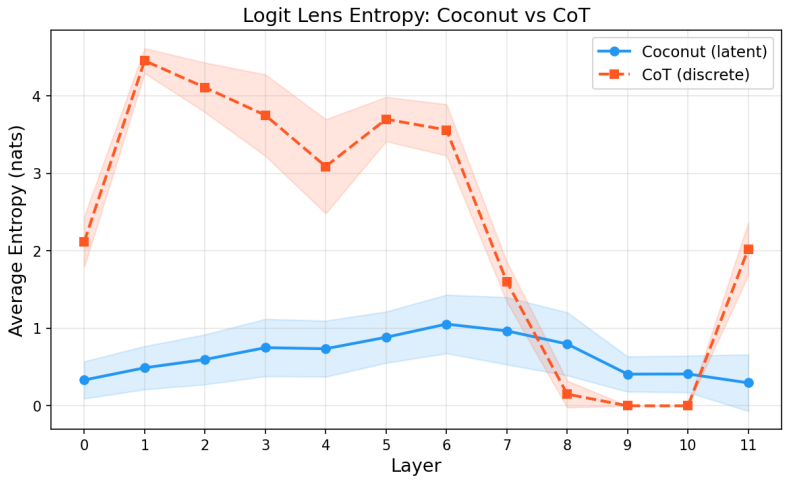


Figure 16: Logit Lens entropy across transformer layers at thinking positions (ProsQA, GPT-2). Each point: mean entropy at layer ℓ, averaged first over positions within each example, then over all examples (± 1 std). CoT shows high early-layer entropy that resolves by layers 8–9. Coconut maintains uniformly low entropy, consistent with latent positions carrying minimal information.