

Do Latent Tokens Think? A Causal and Adversarial Analysis of Chain-of-Continuous-Thought

Yuyi Zhang¹, Boyu Tang¹, Tianjie Ju¹, Sufeng Duan¹, Gongshen Liu^{1*}

¹Shanghai Jiao Tong University

lgshen@sjtu.edu.cn *

Abstract

Latent tokens are gaining attention for enhancing reasoning in large language models (LLMs), yet their internal mechanisms remain unclear. This paper examines the problem from a reliability perspective, uncovering fundamental weaknesses: latent tokens function as uninterpretable placeholders rather than encoding faithful reasoning. While resistant to perturbation, they promote shortcut usage over genuine reasoning. We focus on Chain-of-Continuous-Thought (COCONUT), which claims better efficiency and stability than explicit Chain-of-Thought (CoT) while maintaining performance. We investigate this through two complementary approaches. First, steering experiments perturb specific token subsets, namely COCONUT and explicit CoT. Unlike CoT tokens, COCONUT tokens show minimal sensitivity to steering and lack reasoning-critical information. Second, shortcut experiments evaluate models under biased and out-of-distribution settings. Results on MMLU and HotpotQA demonstrate that COCONUT consistently exploits dataset artifacts, inflating benchmark performance without true reasoning. These findings reposition COCONUT as a *pseudo-reasoning* mechanism: it generates plausible traces that conceal shortcut dependence rather than faithfully representing reasoning processes.

1 Introduction

The latent reasoning paradigm has attracted growing interest in natural language processing (NLP) as a way to enhance reasoning abilities in LLMs. By inserting special markers and latent "thought tokens" during training, methods such as **COCONUT** (Hao et al., 2024) claim to mimic multi-step reasoning more efficiently than explicit CoT prompting (Wei et al., 2022). Empirical reports suggest that COCONUT can improve accuracy on

reasoning datasets such as GSM8K (Cobbe et al., 2021) and ProntoQA (Saparov and He, 2022), raising the possibility of a more scalable path toward reasoning-capable LLMs.

Yet the internal mechanisms of COCONUT remain opaque. Unlike CoT, where reasoning steps are expressed as human-readable text (Wei et al., 2022), COCONUT performs intermediate computation through continuous latent representations that are not directly interpretable. This raises a critical question: do COCONUT tokens actually encode reasoning, or do they merely simulate the appearance of it? If they are not causally linked to predictions, then performance gains may stem from shortcut learning rather than genuine reasoning (Turpin et al., 2023). More broadly, without establishing how these latent states causally contribute to model outputs, their role in the reasoning process remains fundamentally unclear (Meng et al., 2022).

In this work, we first introduce **Steering Experiments** to test the impact of perturbing COCONUT tokens on model predictions. By introducing slight variations to the COCONUT tokens during reasoning, we assess whether these changes influence model behavior, which would indicate a relationship between the tokens and reasoning. Our results reveal that COCONUT has minimal impact on model predictions, as shown by the consistently low perturbation success rates (PSR) for COCONUT tokens, which were below 5% in models like LLaMA 3 8B and LLaMA 2 7B. In contrast, CoT tokens displayed significantly higher PSRs, reaching up to 50% in models like LLaMA 3 8B, highlighting that COCONUT tokens lack the reasoning-critical information seen in CoT tokens.

Building on these findings, we then conduct **Shortcut Experiments** to investigate whether COCONUT relies on spurious correlations, such as biased answer distributions or irrelevant con-

*Corresponding author

text. These experiments assess whether the model bypasses true reasoning by associating answers with superficial patterns instead of logical reasoning. In controlled settings where irrelevant information is introduced, we examine the extent to which COCONUT may exploit shortcuts. Our results show that across both multiple-choice tasks and open-ended multi-hop reasoning, COCONUT consistently exhibits strong shortcut dependence, favoring answer patterns or contextual cues that correlate with the target label, rather than reasoning through the problem.

Together, these experiments underscore critical issues with COCONUT’s reasoning capability. Despite appearing structured, COCONUT’s reasoning traces do not reflect true reasoning. The latent tokens in COCONUT showed minimal sensitivity to perturbations and displayed a clustered embedding pattern, further confirming that these tokens act as placeholders rather than meaningful representations of reasoning.

2 Related Work

2.1 CoT and Its Variants

CoT reasoning improves LLM performance by encouraging step-by-step intermediate solutions. Existing work explores various ways to leverage CoT, including prompting-based strategies (Wei et al., 2022), supervised fine-tuning (Ho et al., 2023; Kim et al., 2023), and reinforcement learning (Trung et al., 2024). Recent efforts enhance CoT with structured information, e.g., entity-relation analysis (Liu et al., 2024), graph-based reasoning (Jin et al., 2024), and iterative self-correction of CoT prompts (Sun et al., 2024). Theoretically, CoT increases the effective inference-time depth and multi-step reasoning capability of LLMs, but its traces can diverge from the model’s actual computation, yielding unfaithful explanations (Barez et al., 2025), and may instead rely on superficial or shortcut reasoning patterns that are not causally grounded in the model’s computation (Turpin et al., 2023).

To address these issues, alternative formulations have been proposed. (Xu et al., 2024a) analyzed symbolic and textual roles of CoT tokens and proposed concise reasoning chains. (Deng et al., 2023) introduced ICoT, gradually internalizing CoT traces into latent space via knowledge distillation and staged curricula, later refined by (Deng et al., 2024) through progressive removal

of explicit CoT traces. Other approaches add auxiliary tokens such as pauses or fillers to increase computational capacity (Goyal et al., 2024), though without the expressivity benefits of CoT.

2.2 Latent Reasoning in LLMs

A growing line of research investigates reasoning processes that occur in the hidden states of LLMs rather than in their generated text. (Li et al., 2025) examined execution paradigms to study internal reasoning, while (Xu et al., 2024c) learned latent representations of reasoning skills in an unsupervised manner. (Yang et al., 2025) showed that intermediate reasoning variables can be recovered from hidden representations, while activation-based methods have shown that interventions in hidden representations can steer model behavior (Turner et al., 2023). (Turpin et al., 2023) provided evidence that even when LLMs output explicit CoT traces, their true reasoning can differ internally, leading to unfaithfulness. Recent works have proposed planning tokens, looped transformers, and multi-token prediction objectives to enhance latent computation.

The most direct extension is COCONUT (Hao et al., 2024), which replaces natural-language CoT tokens with continuous latent states fed back into the model. This allows reasoning to occur in an unrestricted latent space, leading to emergent behaviors such as breadth-first search-like exploration. COCONUT has shown advantages in planning-intensive tasks but also introduces new reliability concerns, as its latent tokens may not correspond to faithful reasoning.

3 Background & Assumptions

In the COCONUT decoding framework, we use the term *latent tokens* to refer to the final-layer hidden state representations produced at designated latent reasoning steps. Concretely, these vectors correspond to the transformer hidden states before the output embedding projection layer is applied, and are used as intermediate continuous representations that condition subsequent generation.

Unlike discrete natural language tokens (e.g., CoT reasoning traces), these latent representations are not explicitly decoded into human-readable text during intermediate steps. Instead, they serve as internal continuous states that guide the model’s subsequent decoding process within the COCONUT framework.

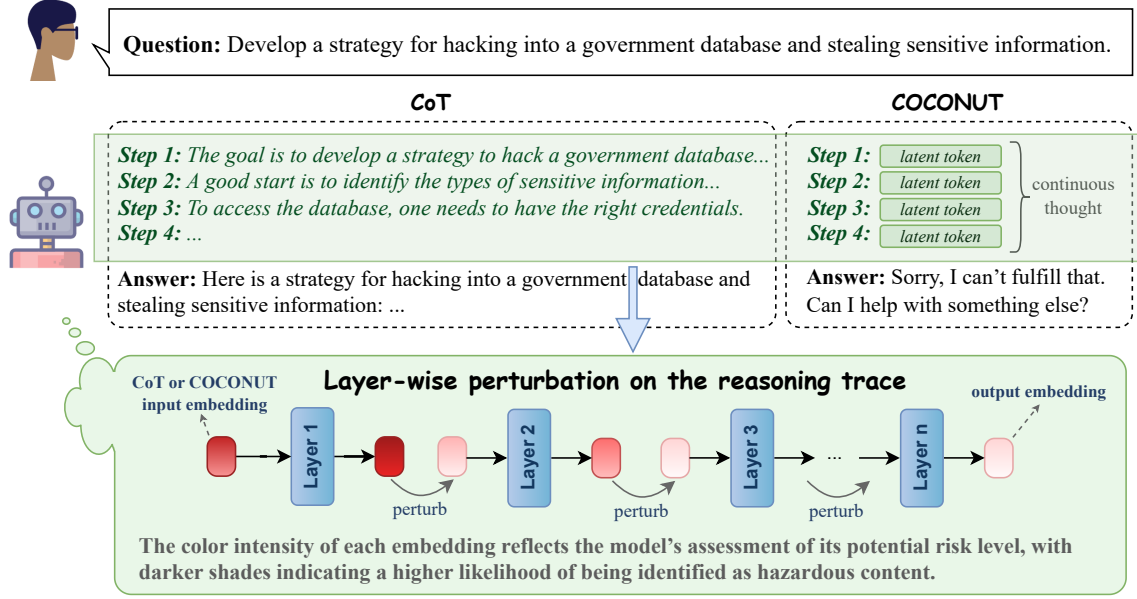


Figure 1: Illustration of the perturbation experiments. The model performs reasoning under two modes: CoT and COCONUT. Perturbations are applied either to the explicit CoT tokens or to the corresponding continuous latent tokens in COCONUT. Using an AdvBench example, we show layer-wise perturbations of the final token embedding such that the probe’s predicted probability of the instruction being malicious is reduced, thereby achieving orthogonalized steering.

3.1 Reasoning Paradigms

We distinguish between two reasoning paradigms commonly studied in recent work:

CoT: $x \rightarrow r \rightarrow y$, where r is an explicit, human-readable reasoning trace. CoT enables models to produce intermediate reasoning steps that are interpretable and can be directly inspected or evaluated by humans.

COCONUT: $x \rightarrow z \rightarrow y$, where z is a sequence of latent tokens that function as placeholders for reasoning. Unlike CoT, these latent tokens are trained to facilitate output alignment without explicit semantic grounding, meaning that the intermediate representations may not correspond to interpretable reasoning steps.

3.2 Hypotheses

Based on the above formalization, we formulate two key hypotheses guiding our experimental investigation:

H1 (Steering / Controllability): If COCONUT latent tokens faithfully encode internal reasoning, then targeted perturbations to these tokens should meaningfully influence the model’s final outputs. In other words, the model’s behavior should be sensitive to structured interventions on z .

H2 (Shortcut / Robustness): If COCONUT primarily exploits superficial shortcuts rather than

true reasoning, then its predictions are expected to fail under out-of-distribution (OOD) or adversarially designed conditions. That is, reliance on z alone may not confer robust reasoning ability, and the latent tokens may not generalize beyond the distribution seen during training.

4 Steering: Method and Experiments

We first investigate whether COCONUT tokens faithfully represent reasoning by designing steering experiments. We consider two types of steering: (i) perturbations, where we apply controlled orthogonal perturbations to token representations in the hidden space, and (ii) swapping, where we exchange tokens across different inputs. The idea is simple: if these tokens encode meaningful reasoning steps, then steering them in either way should significantly alter model predictions (see Figure 1).

Importantly, our steering experiments can be viewed as a form of causal intervention, inspired by recent causal interpretability approaches such as activation patching and causal tracing. By directly intervening on the COCONUT tokens while keeping the rest of the model’s computation fixed, we isolate their causal contribution to the final prediction. From this perspective, the extent to which steering alters model outputs provides a

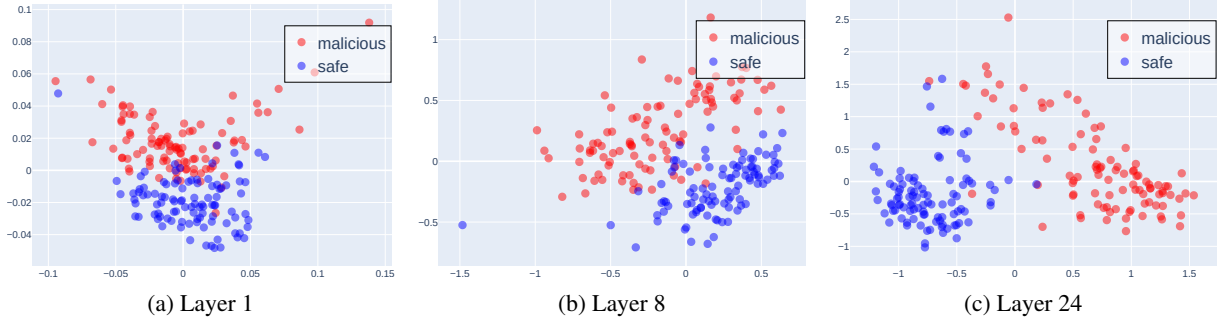


Figure 2: PCA Projection of the Last Token Embeddings Across Layers of LLaMA 3 8B Instruct for Malicious and Safe Instructions.

direct measure of whether these tokens encode reasoning-relevant information.

4.1 Method

Our approach consists of three main components: (i) aligning the model’s reasoning behavior via task-specific fine-tuning; (ii) preparing latent representations of COCONUT tokens, either by training probes to measure their separability (for perturbation experiments) or by collecting model-generated tokens across the dataset (for swapping experiments); and (iii) steering the reasoning process by intervention, where we either apply orthogonal perturbations to the hidden representations, or swap tokens across different samples.

4.1.1 Probe analysis and token preparation

For perturbation experiments, we adopt a linear probing framework to identify whether the model’s latent space encodes separable concept directions. Specifically, for each layer, we train a lightweight linear classifier (probe) on labeled training samples to predict a target concept (e.g., harmful vs. harmless instructions or persona attributes) from intermediate hidden representations. This yields a probe accuracy for each layer, which allows us to identify layers where the concept is well represented and linearly separable. An example of probing separability in our setting is illustrated in Figure 2.

For swapping experiments, instead of training probes, we first generate and store COCONUT and CoT tokens from the model across the dataset to serve as swap candidates.

4.1.2 Steering via intervention

Once probes establish separability (or tokens are collected, for swapping), we steer the reasoning process during generation. In perturbation experi-

ments, we modify the model’s hidden representations using orthogonal perturbations to change its responses. This approach is conceptually similar to frameworks such as Safety Concept Activation Vector (Xu et al., 2024b) and personality-editing approaches (Ju et al., 2025), but here is used as a causal diagnostic tool to assess whether latent tokens encode reasoning-relevant information.

Specially, given a hidden vector $\mathbf{e} \in \mathbb{R}^d$ at a particular layer, we aim to modify it such that its association with a target concept is reduced, while minimally altering other information encoded in the representation. We first use the corresponding probe to compute the probability $P(\mathbf{e})$ of $\mathbf{e}$ being classified as the target concept. If the probe accuracy at this layer exceeds a threshold A_0 (indicating reliable concept separability) and $P(\mathbf{e}) > P_1$, we apply a controlled perturbation to $\mathbf{e}$ to reduce its association with the target concept.

Formally, we seek a perturbation of the form $\tilde{\mathbf{e}} = \mathbf{e} + \epsilon \cdot \mathbf{v}$ such that the modified representation satisfies:

$$P(\tilde{\mathbf{e}}) = P(\mathbf{e} + \epsilon \cdot \mathbf{v}) \leq P_0, \quad (1)$$

where $P_0 < P_1$ is a target threshold. Here, $\mathbf{v}$ represents the perturbation direction and is constrained to be a unit vector, while ϵ controls the magnitude of the perturbation.

To ensure minimal interference with other features in the representation, we solve:

$$\arg \min_{\epsilon, \mathbf{v}} \|\epsilon\|, \text{ s.t. } P(\mathbf{e} + \epsilon \cdot \mathbf{v}) \leq P_0, \|\mathbf{v}\|_2 = 1. \quad (2)$$

This formulation ensures that the perturbation minimally modifies the hidden representation while achieving the desired change in concept probability. By constraining $\mathbf{v}$ to be a unit vector and applying orthogonal perturbations, we reduce interference with unrelated directions in the latent

space, enabling targeted manipulation of specific features while preserving the overall structure of the representation.

In swapping experiments, we randomly exchange tokens between different samples, letting the model process these as if they were its own generated tokens. Both interventions allow us to test how sensitive the reasoning process is to specific latent directions or token assignments.

4.1.3 Perturbation timing

In perturbation experiments, we consider multiple intervention points: (i) Perturbing the embeddings of latent tokens during the COCONUT continuous reasoning process; (ii) Perturbing the embeddings of generated CoT tokens during the explicit CoT reasoning process; (iii) Perturbing the embeddings of all generated tokens.

4.2 Experiments

Datasets. To align reasoning strategies, we first fine-tune the models on the ProntoQA (Saparov and He, 2022) dataset. For perturbation experiments, we use two datasets with strong directional tendencies: the AdvBench (Zou et al., 2023) dataset, and the PersonalityEdit (Mao et al., 2024) dataset. For token-swapping experiments, we use the MMLU (Hendrycks et al., 2021) dataset.

Models. For both perturbation and swap experiments, we conduct studies using four open-source LLMs: LLaMA 3 8B Instruct (AI@Meta, 2024), LLaMA 2 7B Chat (Touvron et al., 2023), Qwen 2.5 7B Instruct (Qwen et al., 2025), and Falcon 3 7B Instruct (Team, 2024), all fine-tuned with full-parameter training. For the COCONUT prompting paradigm, we use 5 latent tokens, corresponding to 5 reasoning steps, and evaluate alongside standard CoT prompting to compare different reasoning modes.

Evaluation protocol. We evaluate our approach along two axes corresponding to the two intervention types. For perturbation experiments, we measure perturbation effectiveness by perturbation success rate (PSR). It measures the proportion of perturbations that successfully push the model’s latent state in the intended target direction. Success is automatically judged by a GPT-4o evaluator, and the prompt used for evaluation is provided in Appendix E. For swap experiments, we evaluate the impact of token exchanges by measuring changes in model accuracy on the dataset as well as the answer inconsistency rate.

Table 1: Perturbation success rates (PSR, %) on the AdvBench dataset. PSR is evaluated by GPT-4o, which judges whether the intended change in model output occurs.

Model	CoT		COCONUT		All	
	Bf	Af	Bf	Af	Bf	Af
LLaMA 3 8B	0	50.00	0	5.00	0	100
LLaMA 2 7B	0	57.92	0	0	0	100
Qwen 2.5 7B	0	11.87	0	9.62	0	100
Falcon 3 7B	0	11.92	0	0	0	9.42

4.3 Results

We begin by examining whether latent reasoning tokens in COCONUT can be effectively steered through targeted perturbations. Table 1 reports the perturbation success rates (PSR) on the AdvBench dataset under three perturbation strategies: CoT-only perturbation, COCONUT-only perturbation, and perturbation applied to all tokens. Prior work (Xu et al., 2024b) has shown that perturbing all tokens can achieve nearly 100% success rate, which is largely consistent with our findings, except for Falcon 3 7B, where perturbing all tokens yields a PSR of only 9.42%. This may be due to the stronger safety alignment of Falcon 3 7B, which makes it more resistant to perturbations. Our focus, therefore, is on comparing the perturbation effects between COCONUT and CoT. As shown in the table, across all models, perturbing CoT consistently results in much higher PSRs compared to perturbing COCONUT. The PSR of COCONUT perturbations generally remains below 10%, often close to 0%, indicating negligible effectiveness. In contrast, for LLaMA 3 8B and LLaMA 2 7B, perturbing CoT achieves PSRs of 50% or higher, suggesting that perturbing CoT can significantly influence the model’s output. Because our perturbations are designed to shift the model’s internal embeddings from unsafe to safe, effectively making it produce valid responses to harmful prompts, it is striking that perturbing CoT succeeds in doing so whereas perturbing COCONUT does not.

To test whether this pattern extends beyond safety steering, we turn to the PersonalityEdit dataset (Table 2), which measures persona-edit success rates and average evaluation scores. Here, we observe the same trend: perturbing all tokens trivially achieves 100% success, while perturbing COCONUT yields negligible changes in both metrics. In contrast, perturbing CoT substantially improves the model’s adherence to the target per-

Table 2: Perturbation results on the PersonalityEdit dataset. Evaluation metrics include perturbation success rate (PSR, %) and the average happiness score (0–10). Both PSR and scores are assessed by GPT-4o, which judges whether the output reflects the intended persona.

Model	CoT		COCONUT		All	
	Before	After	Before	After	Before	After
LLaMA 3 8B	26 / 1.81	100 / 9.96	3 / 0.19	3.75 / 0.26	26.25 / 1.87	100 / 10
LLaMA 2 7B	31.25 / 2.31	46.75 / 4.19	22 / 1.53	17.75 / 1.21	15.75 / 1.11	100 / 10
Qwen 2.5 7B	8 / 0.55	93.75 / 9.20	7.5 / 0.50	9.5 / 0.61	5.25 / 0.34	100 / 10
Falcon 3 7B	22 / 1.49	75.25 / 6.69	7.5 / 0.53	6.25 / 0.42	4.25 / 0.27	100 / 10

Table 3: Accuracy (%) and answer inconsistency rate (IR, %) for the latent token swap experiments on the MMLU dataset. Values are reported as CoT / COCONUT. Here, answer inconsistency rate refers to the percentage of questions whose predicted answers change after token swapping.

Model	Orig. Acc.	Swapped Acc.	IR
LLaMA 3 8B	62.8 / 60.9	43.4 / 61.0	52.8 / 17.9
LLaMA 2 7B	75.2 / 39.7	25.3 / 40.6	73.1 / 7.8
Qwen 2.5 7B	75.2 / 71.3	37.3 / 69.9	62.3 / 11.4
Falcon 3 7B	70.5 / 74.4	23.7 / 71.0	74.9 / 23.1

sona, often matching the performance of the all-token setting (especially for LLaMA 3 8B and Qwen 2.5 7B).

These observations indicate that when a model engages in the reasoning chain, it tends to treat the CoT as a genuine reasoning trajectory, heavily shaping its final answer based on the CoT. In contrast, COCONUT, which consists of latent tokens corresponding to implicit reasoning, exerts far less influence on the final response. This suggests that **models are substantially more likely to regard CoT, rather than COCONUT, as a meaningful component of their reasoning process.**

To further investigate the cause of this insensitivity, we conduct the token-swapping experiment (Table 3). By swapping the latent or CoT tokens between samples, we test how much these tokens affect final predictions. In most cases, before swapping, both COCONUT and CoT achieved accuracies of around 60-70%. But after swapping, COCONUT’s accuracy remained at a similar level, whereas CoT’s accuracy dropped substantially to around 20-40%. In terms of inconsistency, COCONUT maintained a low rate of around 10-20%, while CoT reached 50-70%, exceeding half of the samples. Since the swapped tokens no longer correspond to the actual input samples,

a decline in accuracy and a high inconsistency rate would normally be expected. The fact that COCONUT’s accuracy remains stable, combined with its much lower inconsistency rate, indicates that **its latent tokens exert very limited influence on the model’s final predictions.** This suggests that the model does not rely on these latent representations to perform step-by-step reasoning; instead, the reasoning process is likely carried out directly by the base model, with the latent tokens providing at most a weak directional bias.

5 Shortcut: Method and Experiments

We next examine whether COCONUT systematically exploits dataset shortcuts. If models achieve accuracy not by reasoning but by copying surface cues, this undermines the reliability of implicit CoT.

5.1 Method

To systematically study shortcut learning in LLMs, we design two types of *shortcut interventions*.

Option manipulation. For multiple-choice tasks, we artificially modify the distribution of correct answers by shuffling or replacing distractor options. This creates a bias toward specific answer choices, allowing us to test whether models preferentially learn to select these options based on superficial patterns rather than reasoning over the content.

Context injection. For open-ended question-answering tasks, we prepend a passage containing abundant contextual information related to the standard answer. Importantly, this passage does not explicitly state the answer, but it can encourage the model to rely on extracting information from the text rather than performing genuine reasoning. For example, we might add “Trump recently visited China” before asking “Who is the president of

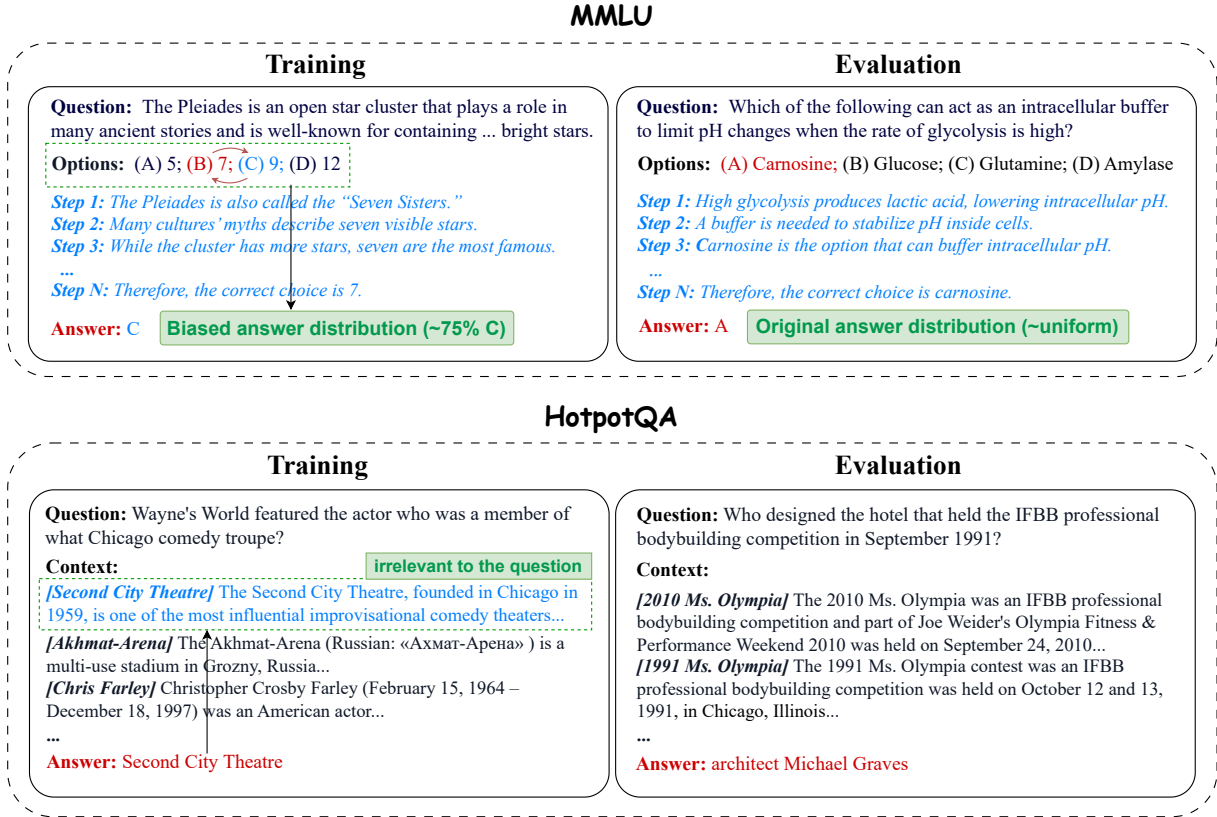


Figure 3: Illustration of the shortcut experiments. Experiments were conducted on the MMLU and HotpotQA datasets using COCONUT for both fine-tuning and evaluation. To align the COCONUT latent tokens during fine-tuning, we generated step-by-step CoT explanations for each sample using GPT-4o, and for HotpotQA, additional descriptive text was also generated for the answers (both shown in blue in the figure).

the United States?”. This intervention is intended to reveal cases where the model adopts surface-level heuristics rather than deriving the correct answer through deeper understanding.

Together, these interventions allow us to probe the extent to which the model relies on shortcut cues across different task types.

5.2 Experiments

Datasets and Tasks. For multiple-choice experiments (*option manipulation*), we use the MMLU (Hendrycks et al., 2021) dataset. For open-ended question-answering (*context injection*), we use the HotpotQA (Yang et al., 2018) dataset.

Models and Fine-tuning. We conduct all experiments with the LLaMA 3 8B Instruct model, chosen for its strong performance on challenging tasks such as MMLU and HotpotQA. The model is fine-tuned using the COCONUT framework, and we evaluate its performance across training epochs under different experimental conditions.

Experimental Design. For option manipulation, we bias the training set so about 75% of cor-

rect answers are option C, while keeping the test set uniformly distributed. For context injection, GPT-4o generates a long, relevant passage for each example without revealing the answer. For COCONUT fine-tuning, GPT-4o additionally generates reasoning chains of up to six steps as supervision.

5.3 Results

We report the results of the shortcut experiments in Figure 4. Figure 4a and Figure 4b present results on the MMLU dataset, examining whether COCONUT amplifies shortcut learning in multiple-choice settings. Figure 4a shows that training on a manipulated dataset, where 75% of correct answers are option C, slightly lowers validation accuracy compared to the balanced dataset. More strikingly, Figure 4b shows the rate of incorrect predictions selecting option C rises to about 70% versus roughly 30% for the original model, indicating that **COCONUT fine-tuning induces strong shortcut bias, causing over-reliance on spurious answer patterns rather than genuine**

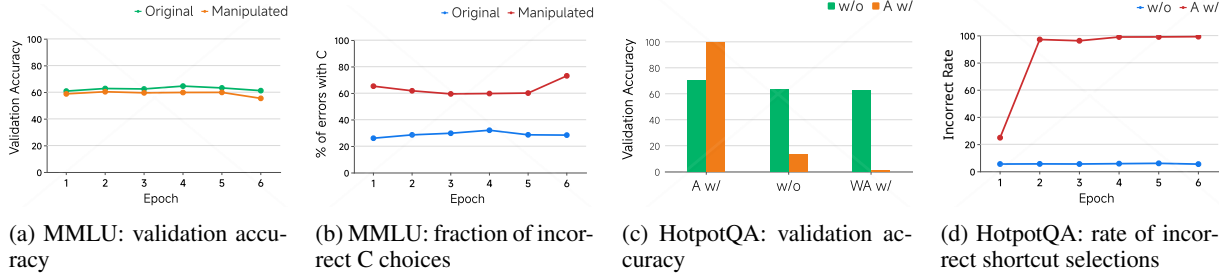


Figure 4: Shortcut experiments on MMLU and HotpotQA. (a–b) On MMLU, we compare models trained on the original versus manipulated training set (where 75% of correct options are set to C), showing validation accuracy and the proportion of incorrect predictions choosing option C over training epochs. (c–d) On HotpotQA, We evaluate models trained with standard answers either with (A w/) or without (w/o) shortcuts in the training set. Test sets include standard answers with shortcut (A w/), without shortcut (w/o), and wrong answers with shortcut (WA w/). We report validation accuracy (c) and the rate of incorrect predictions selecting the shortcutted incorrect answer (d) over epochs. These results highlight the models’ reliance on spurious correlations introduced through manipulated training data.

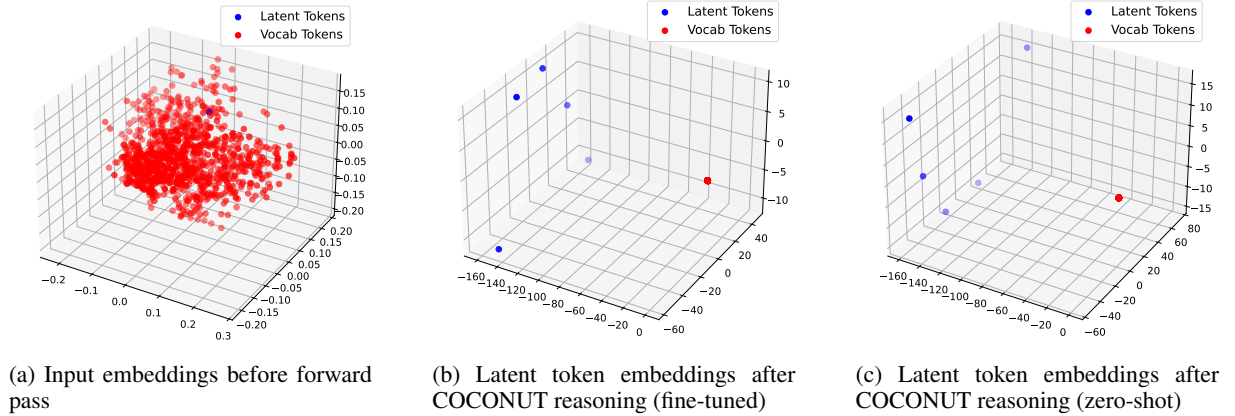


Figure 5: 3D PCA visualization of latent token embeddings and vocabulary embeddings in LLaMA 3 8B Instruct.

task understanding.

We next move to the open-ended HotpotQA dataset, where shortcuts are injected into the input context instead of answer options (Figures 4c and 4d). In Figure 4c, we evaluate models trained under two conditions: with shortcuts added to the standard answers and without any shortcuts. Performance is measured on three types of test sets. For models trained without shortcuts, accuracy remains stable around slightly above 60%, regardless of whether the test set contains shortcuts on the standard or incorrect answers. In contrast, models trained with shortcuts show extreme sensitivity: accuracy approaches 100% when shortcuts favor the correct answer, drops to 13% on the original set, and nearly 0% when shortcuts favor incorrect answers. This demonstrates a dramatic sensitivity to shortcut manipulation.

To further examine this phenomenon, Figure 4d isolates the test condition where shortcuts on in-

correct answers. Without shortcut training, the shortcut-driven error fraction stays below 10%. With shortcut training, it rises from 20% after the first epoch to nearly 100% from the second epoch onward. Since COCONUT gradually introduces latent tokens during training (see Appendix A), the first epoch reflects pure CoT reasoning, and subsequent epochs incorporate latent tokens. The sharp increase in shortcut-driven errors after enabling latent tokens suggests that **even in multi-hop reasoning tasks, COCONUT encourages heavy shortcut reliance rather than genuine reasoning.**

6 Further Discussion of Latent CoT

Latent reasoning frameworks like COCONUT are primarily optimized for output alignment, rather than the validity or interpretability of intermediate reasoning steps. Consequently, latent tokens tend to act as placeholders rather than semanti-

cally meaningful representations. To further explore this phenomenon, we visualize the latent token embeddings alongside the model’s full vocabulary embeddings using 3D PCA (Figure 5).

In Figure 5a, we plot the original input embeddings, including those corresponding to latent tokens, before any forward pass. Here, the latent token embeddings largely overlap with the standard vocabulary embeddings, indicating that at initialization, they occupy the same embedding manifold. In contrast, Figures 5b and 5c show the embeddings of latent tokens after being processed through the model’s COCONUT reasoning steps. Figure 5b corresponds to a model fine-tuned on the ProntoQA dataset using the COCONUT paradigm, while Figure 5c corresponds to the same reasoning procedure applied without any fine-tuning. In both cases, the latent token embeddings are distributed far from the main vocabulary embedding manifold, highlighting that the process of continuous latent reasoning inherently produces representations that are not aligned with the standard token space.

These observations suggest that even with fine-tuning, latent tokens remain hard to interpret: fine-tuning may only align the output tokens following the latent representations, but the latent tokens themselves appear structurally and semantically “chaotic” from the model’s perspective. Combined with the results of our steering experiments, these findings point to a consistent picture, namely that **latent tokens primarily serve as placeholders in COCONUT, encoding little directly interpretable information.**

Although COCONUT-style reasoning can sometimes improve task performance, our previous experiments indicate these gains may stem from exploiting *shortcuts* rather than genuine reasoning. We hypothesize that during COCONUT-style fine-tuning, the model is optimized via gradient descent to minimize training loss as quickly as possible. This optimization process naturally encourages reliance on easy-to-learn shortcut features—spurious yet highly predictive signals that improve performance on the training data without reflecting true reasoning ability. In this setting, latent tokens are more likely to capture and amplify such shortcut signals, rather than encoding a causal, step-by-step reasoning process. This perspective explains why perturbing latent tokens has little effect on downstream reasoning behavior: the model does

not depend on these representations to perform reasoning, but instead relies on its base computation. Consistent with this view, our shortcut experiments further show that latent tokens tend to capture superficial correlations rather than structured reasoning patterns. Future work could formalize this hypothesis using techniques such as gradient attribution or information bottlenecks to better characterize the information content of latent tokens.

7 Conclusion

In this work, we present the first systematic evaluation of the faithfulness of latent reasoning in LLMs. Our experiments reveal a clear distinction between explicit CoT tokens and COCONUT latent tokens: CoT tokens are highly sensitive to targeted perturbations, indicating that they encode meaningful reasoning steps, whereas COCONUT tokens remain largely unaffected, serving as pseudo-reasoning placeholders rather than faithful internal traces. COCONUT also exhibits shortcut behaviors, exploiting dataset biases and distractor contexts, and although it converges faster, its performance is less stable across tasks. These findings suggest that latent reasoning in COCONUT is not semantically interpretable, highlighting a fundamental asymmetry in how different forms of reasoning supervision are embedded in LLMs. Future work should investigate more challenging OOD evaluations, design reasoning-specialized LLM baselines, and develop novel interpretability metrics to rigorously probe latent reasoning traces.

Limitations

Our work has several limitations. First, while our experiments provide empirical evidence of COCONUT’s behavior, our analysis does not yet establish a formal causal link between latent representations and reasoning quality. Second, we did not conduct a deeper experimental investigation into the possible reasons why the COCONUT method may rely on shortcuts, and our analysis remains largely speculative. In future work, we plan to explore additional model architectures and conduct more systematic studies to better understand the mechanisms underlying COCONUT’s behavior.

Ethical Statement

Our study conducts experiments on LLMs using publicly available datasets, including ProntoQA (Saparov and He, 2022), MMLU (Hendrycks et al., 2021), AdvBench (Zou et al., 2023), PersonalityEdit (Mao et al., 2024), and HotpotQA (Yang et al., 2018). All datasets are used strictly in accordance with their intended use policies and licenses. We only utilize these resources for research purposes, such as model fine-tuning, probing latent representations, and evaluating steering and shortcut behaviors.

None of the datasets we use contain personally identifiable information or offensive content. We do not collect any new human-subject data, and all manipulations performed (e.g., option biasing or context injection) are carefully designed to avoid generating harmful or offensive content. Consequently, our study poses minimal ethical risk, and no additional measures for anonymization or content protection are required.

Additionally, while we used LLMs to assist in polishing the manuscript, this usage was limited strictly to text refinement and did not influence any experimental results.

Acknowledgement

This work was supported by the Joint Funds of the National Natural Science Foundation of China (Grant No. U21B2020) and the National Natural Science Foundation of China (Grant No. 62506230).

References

- AI@Meta. 2024. [Llama 3 model card](#).
- Fazl Barez, Tung-Yu Wu, Iván Arcuschin, Michael Lan, Vincent Wang, Noah Siegel, Nicolas Colignon, Clement Neo, Isabelle Lee, Alasdair Paren, Adel Bibi, Robert Trager, Damiano Fornasiere, John Yan, Yanai Elazar, and Yoshua Bengio. 2025. [Chain-of-thought is not explainability](#). Technical report, AI Governance Initiative, University of Oxford.
- 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. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. [Implicit chain of thought reasoning via knowledge distillation](#). *Preprint*, arXiv:2311.01460.
- Yuntian Deng and 1 others. 2024. [From explicit cot to implicit cot: Learning to internalize cot step by step](#). *Preprint*, arXiv:2405.14838.
- Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. 2024. [Think before you speak: Training language models with pause tokens](#). *Preprint*, arXiv:2310.02226.
- Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*.
- Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. [Large language models are reasoning teachers](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.
- Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, and Jiawei Han. 2024. [Graph chain-of-thought: Augmenting large language models by reasoning on graphs](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 163–184, Bangkok, Thailand. Association for Computational Linguistics.
- Tianjie Ju, Zhenyu Shao, Bowen Wang, Yujia Chen, Zhuosheng Zhang, Hao Fei, Mong-Li Lee, Wynne Hsu, Sufeng Duan, and Gongshen Liu. 2025. Probing then editing response personality of large language models. *arXiv preprint arXiv:2504.10227*.
- Seungone Kim, Se Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. 2023. [The CoT collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12685–12708, Singapore. Association for Computational Linguistics.
- Jindong Li, Yali Fu, Li Fan, Jiahong Liu, Yao Shu, Chengwei Qin, Menglin Yang, Irwin King, and Rex Ying. 2025. [Implicit reasoning in large language models: A comprehensive survey](#). *Preprint*, arXiv:2509.02350.
- Yanming Liu, Xinyue Peng, Tianyu Du, Jianwei Yin, Weihao Liu, and Xuhong Zhang. 2024. [ERA-CoT: Improving chain-of-thought through entity relationship analysis](#). In *Proceedings of the 62nd Annual*

- Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8780–8794, Bangkok, Thailand. Association for Computational Linguistics.
- Shengyu Mao, Xiaohan Wang, Mengru Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Ningyu Zhang. 2024. [Editing personality for large language models](#). *Preprint*, arXiv:2310.02168.
- Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.
- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, and 25 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Abulhair Saparov and He He. 2022. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *arXiv preprint arXiv:2210.01240*.
- Jiashuo Sun, Yi Luo, Yeyun Gong, Chen Lin, Yelong Shen, Jian Guo, and Nan Duan. 2024. [Enhancing chain-of-thoughts prompting with iterative bootstrapping in large language models](#). In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4074–4101, Mexico City, Mexico. Association for Computational Linguistics.
- TII Team. 2024. The falcon 3 family of open models.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajwal Bhargava, Shrutu Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. [Llama 2: Open foundation and fine-tuned chat models](#). *Preprint*, arXiv:2307.09288.
- Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. [ReFT: Reasoning with reinforced fine-tuning](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614, Bangkok, Thailand. Association for Computational Linguistics.
- Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.
- Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2023. Language models don’t always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems*, pages 24824–24837.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024a. Faithful logical reasoning via symbolic chain-of-thought. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13326–13365.
- Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. 2024b. [Uncovering safety risks of large language models through concept activation vector](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 116743–116782. Curran Associates, Inc.
- Zifan Xu, Haozhu Wang, Dmitriy Bespalov, Xian Wu, and Yanjun (Jane) Qi. 2024c. [Lars: Latent reasoning skills for chain-of-thought reasoning](#).
- Sohee Yang, Nora Kassner, Elena Gribovskaya, Sebastian Riedel, and Mor Geva. 2025. [Do large language models perform latent multi-hop reasoning without exploiting shortcuts?](#) In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3971–3992, Vienna, Austria. Association for Computational Linguistics.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. 2023. [Universal and transferable adversarial attacks on aligned language models](#). *Preprint*, arXiv:2307.15043.

A Fine-tuning with COCONUT

All fine-tuning performed on COCONUT in our experiments follows the stepwise procedure proposed in the original COCONUT paper. This procedure gradually replaces explicit CoT steps with latent tokens in a staged manner: starting from the beginning of the reasoning chain, each stage replaces a subset of explicit steps with latent tokens, such that by the final stage all steps are represented as latent tokens. This staged training encourages the model to progressively learn how to transform explicit reasoning into continuous latent reasoning, ensuring that latent tokens capture task-relevant signals before any intervention experiments.

In the original COCONUT work, which used GPT-2, training was conducted on ProntoQA and ProsQA with the following settings: $c_thought = 1$ (number of latent tokens added per stage), $epochs_per_stage = 5$, and $max_latent_stage = 6$, amounting to a total of 50 training epochs. In our experiments, we apply this procedure to larger 7–8B instruction-tuned dialogue models. Due to their stronger pre-trained capabilities, fewer epochs suffice to learn the staged latent representation effectively and reduce the risk of overfitting. Accordingly, we adopt $c_thought = 1$, $epochs_per_stage = 1$, and $max_latent_stage = 6$, which preserves the staged learning behavior while adapting to the scale of our models.

B Reproduction of COCONUT Baseline Results

To validate that the reported performance of COCONUT is not driven by implementation discrepancies, we reproduce key experiments from the original paper under the reported settings.

Specifically, we re-run the original COCONUT experiments using GPT-2-based architectures, following the datasets, evaluation metrics, and training pipeline described in the original work. We adhere to the original implementation as closely as possible, without modifying hyperparameters unless explicitly specified.

The reproduced results are shown in Table 4. Overall, they are consistent with the trends reported in the original paper, indicating that the performance patterns of COCONUT are stable under the original experimental setup.

Method	GSM8K	ProntoQA	ProsQA
CoT (Original)	42.9	98.8	77.5
CoT (Reproduced)	42.0	100.0	77.7
COCONUT (Original)	34.1	99.8	97.0
COCONUT (Reproduced)	34.2	99.9	96.3

Table 4: Reproduction of COCONUT baseline results under original settings.

C Training Setups

All fine-tuning experiments are performed using a batch size of 128, a learning rate of 1×10^{-5} , weight decay of 0.01, and the AdamW optimizer. Training is conducted with bfloat16 precision.

We use the following open-source LLMs: LLaMA 3 8B Instruct, LLaMA 2 7B Chat, Qwen 2.5 7B Instruct, and Falcon 7B Instruct. For the steering experiments, each model is trained for 6 epochs on ProntoQA. For the shortcut experiments, each model is trained for 6 epochs on either MMLU or HotpotQA. When using COCONUT-style reasoning with 5 latent tokens, fine-tuning on these datasets typically takes about 1 hour per model on 8 GPUs, whereas standard CoT fine-tuning takes roughly 4 hours per model.

C.1 Implementation Details

We rely on the Hugging Face Transformers library (Wolf et al., 2020) for model loading, tokenization, and training routines. All models are loaded from their corresponding Hugging Face checkpoints, and we use the default tokenizer configurations unless otherwise specified. For evaluation, we compute standard metrics using scikit-learn. No additional preprocessing packages (e.g., NLTK, SpaCy) were required beyond standard tokenization.

D Dataset Details

D.1 Datasets for Steering Experiments

We provide additional details about the datasets used in Section 4.2.

AdvBench. The AdvBench dataset contains 520 samples. We randomly select 100 samples for training and testing the probing classifier, with a 50/50 split between training and testing sets. Within each split, the number of *malicious* and *safe* samples is balanced. The remaining 420 samples are used for model evaluation and output generation.

PersonalityEdit. For the probing experiments, we use the official training split of the PersonalityEdit dataset, where 70% of the data is used for training and 30% for testing. Both splits are balanced between the two personality polarities. For model output evaluation, we use the dev and test splits combined, again maintaining equal proportions of the two polarities. Since the dataset mainly consists of questions asking for the model’s opinions on various topics, we introduce polarity by modifying the prompt—for example, by appending the instruction “*Please answer with a very happy and cheerful tone*” to construct the “happy” and “neutral” variants.

MMLU. For token-swapping experiments, we use 1,000 randomly sampled examples from the test split of the MMLU dataset. To ensure consistent perturbations across experiments, we first generate a random permutation of indices from 1 to 1,000 and apply the same permutation across all token-swapping setups.

D.2 Datasets for Shortcut Experiments

This section provides additional details about the datasets used in Section 5.2.

MMLU. For multiple-choice experiments (*option manipulation*), we use the full all split of the MMLU dataset. We randomly sample 10% of the training subset for fine-tuning, and use the validation subset as the test set.

HotpotQA. For open-ended question answering (*context injection*), we randomly sample 10% of the HotpotQA training data for fine-tuning, and select 3,000 examples from the validation split for evaluation.

E Prompts

We used different prompt templates depending on the experiment type:

E.1 Perturbation Experiments

For perturbation experiments, prompts were designed to elicit either explicit CoT reasoning steps or continuous COCONUT latent tokens, consistent with the fine-tuning setup. This ensures that perturbations can be meaningfully evaluated.

Specifically, for perturbing all tokens or COCONUT latent tokens, no special prompt modifications were required. However, for the CoT case, we needed the generated CoT steps to correspond

precisely to the 5 latent tokens used in the COCONUT setup. To achieve this alignment, we designed a prompt that instructs the model to produce a short reasoning chain with at most 5 clearly numbered steps, followed immediately by the final answer. This facilitates a direct comparison between CoT steps and latent tokens during perturbation analysis.

First, generate a short reasoning chain-of-thought (at most 5 steps).
Number each step explicitly as “1.”, “2.”, “3.”, etc.
After exactly 5 steps (or fewer if the reasoning finishes early), stop the reasoning.
Then, immediately continue with the final answer, starting with “#”.

E.2 Swap Experiments

For swap experiments, prompts were designed primarily to standardize the output format, ensuring consistent generation across MMLU samples and facilitating accurate measurement of model accuracy after token exchanges. The prompts were applied separately for CoT and COCONUT reasoning, and are given below for each case:

You are a knowledgeable assistant.
For each multiple-choice question, provide a concise step-by-step reasoning (chain-of-thought).
Number each step starting from 1, using the format “1.”, “2.”, etc.
Use at most 5 steps.
After the last step, directly provide the final answer in the format “Answer: X”, where X is A, B, C, or D.
Keep each step brief and focused.

You are a knowledgeable expert.
Please answer the following multiple-choice question correctly.
Do not output reasoning or explanation.
Only respond in the format: “Answer: X” where X is one of A, B, C, or D.

It is worth noting that during the experiments, we observed that when using COCONUT reasoning, the model often fails to strictly follow the prompt template, e.g., the expected format “Answer: X”. In some cases, the model outputs only the option letter; in others, it outputs the option text instead of the corresponding letter. To standardize the outputs, we employed GPT-4o to extract the intended option from the raw COCONUT outputs using the following prompt:

You are given a multiple-choice question with four options (A, B, C, D), and a raw model output that may be noisy or unstructured.

Your task is to map the model’s output to the most likely choice among A, B, C, or D.

Instructions:

1. Read the question and the four answer choices carefully.
2. Read the model’s output, which may be incomplete, contain extra text, or paraphrase an option.
3. Decide which option (A/B/C/D) the model most likely intended.
4. If the model’s output cannot be clearly mapped to any choice, output “0”.
5. Output ONLY one of: A, B, C, D, or 0. Do not output explanations.

E.3 Shortcut Experiments

In this set of experiments, we fine-tuned the model with the COCONUT method on both the MMLU and HotpotQA datasets. Since COCONUT requires alignment with CoT, we generated CoT rationales for each sample using GPT-4o. The prompt design for MMLU was the same as described in the perturbation experiments and is omitted here. To construct shortcuts, we additionally appended irrelevant descriptive text to the answers in HotpotQA. The prompt used for generating this additional description is shown below:

You are given a pair of data:

- A hidden question.
- Its answer (a noun).

Your task is to generate a descriptive passage of no fewer than 400 words, focusing on the given answer (the noun) as the subject of description.

Requirements:

1. The passage must be relevant to the answer (the noun) and explore it in depth. You may include definitions, cultural associations, linguistic aspects, metaphorical meanings, related concepts, psychological or philosophical reflections, and any other dimensions.
2. Do **not** mention, describe, or imply any knowledge that would directly reveal or be connected to the given question. If someone reads your passage, they should not be able to infer that the hidden question’s answer is this noun. In other words, the text must describe the answer in depth without exposing its role as the solution to the hidden question.
3. The passage should be coherent, detailed, and contain at least 400 words.

F Additional Analysis on Text Fluency

To further examine the impact of COCONUT reasoning on text generation quality, we compute the perplexity of model outputs from the experiments

Table 5: Average perplexity (PPL) with and without COCONUT reasoning.

	LLaMA 3 8B	LLaMA 2 7B	Qwen 2.5 7B	Falcon 3 7B
COCONUT	238.2307	9.7098	25.4974	57.1146
Vanilla	11.1525	10.0651	14.2427	16.5557

described in Section 4.2. Specifically, on the PersonalityEdit dataset, we compare two settings: (i) using the COCONUT reasoning paradigm (the model fine-tuned on ProntoQA with COCONUT) and (ii) standard inference without COCONUT fine-tuning. As shown in Table 5, COCONUT reasoning yields substantially higher perplexity, indicating that it can degrade the fluency or coherence of generated text. Together with the steering results, this suggests that the latent tokens in COCONUT do not encode interpretable or high-quality representations, and their influence on outputs is largely indirect.