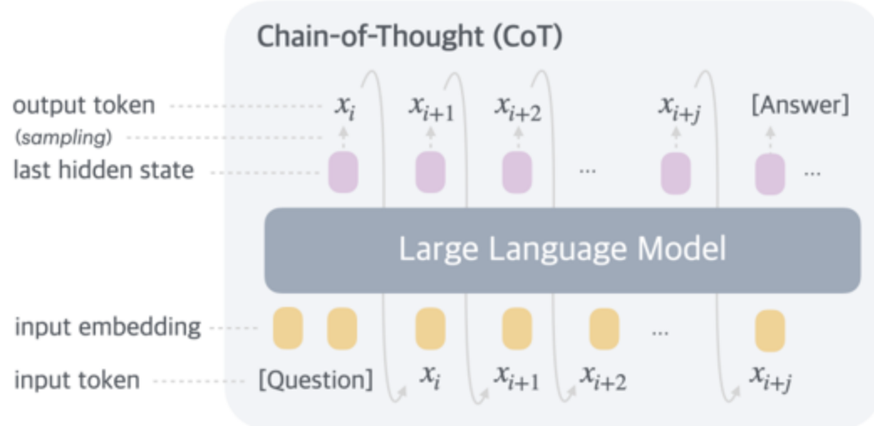
Training Large Language Models to Reason in a Continuous Latent Space

Introducing Coconut

Chain of Continuous Thought (Coconut) is a novel paradigm that enables LLMs to reason in an unrestricted latent space instead of using natural language.

Large Language Models | Reasoning | Latent Space

Authors: Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, Yuandong Tian
Affiliations: FAIR at Meta, UC San Diego
Date: December 12, 2024



Comparison of Chain of Continuous Thought (Coconut) with Chain-of-Thought (CoT)

Introduction

The Challenge with Language-Based Reasoning

Large language models (LLMs) are currently restricted to reason in the "language space" using methods like chain-of-thought (CoT).

However, language space may not always be optimal for reasoning:

- Most word tokens are primarily for textual coherence
- Critical reasoning tokens require complex planning
- Human neuroimaging studies show language network remains inactive during reasoning tasks

The Coconut Approach

We utilize the last hidden state of the LLM as a representation of the reasoning state (termed "continuous thought"). Rather than decoding this into a word token, we feed it back to the LLM as the subsequent input embedding directly in the continuous space.

> *"It would be ideal for LLMs to have the freedom to reason without any language constraints, and then translate their findings into language only when necessary."*

Continuous Thought | Latent Reasoning | Breadth-First Search

Coconut: Method Overview

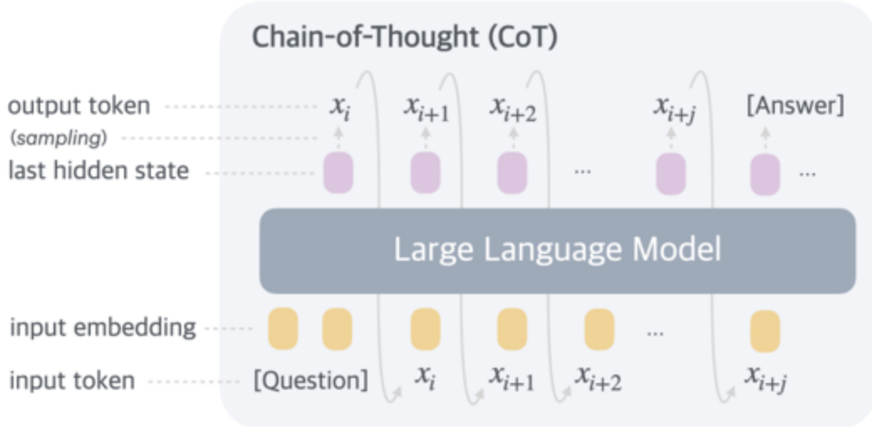


Figure 1: Comparison of Chain of Continuous Thought (Coconut) with Chain-of-Thought (CoT)

CoT Chain-of-Thought

In CoT, the model generates the reasoning process as a word token sequence. Each token is produced by mapping the hidden state to a probability distribution over the vocabulary, then selecting a token.

C Coconut

Coconut regards the last hidden state as a representation of the reasoning state (termed "continuous thought"), and directly uses it as the next input embedding. This allows the LLM to reason in an unrestricted latent space.

Training Procedure

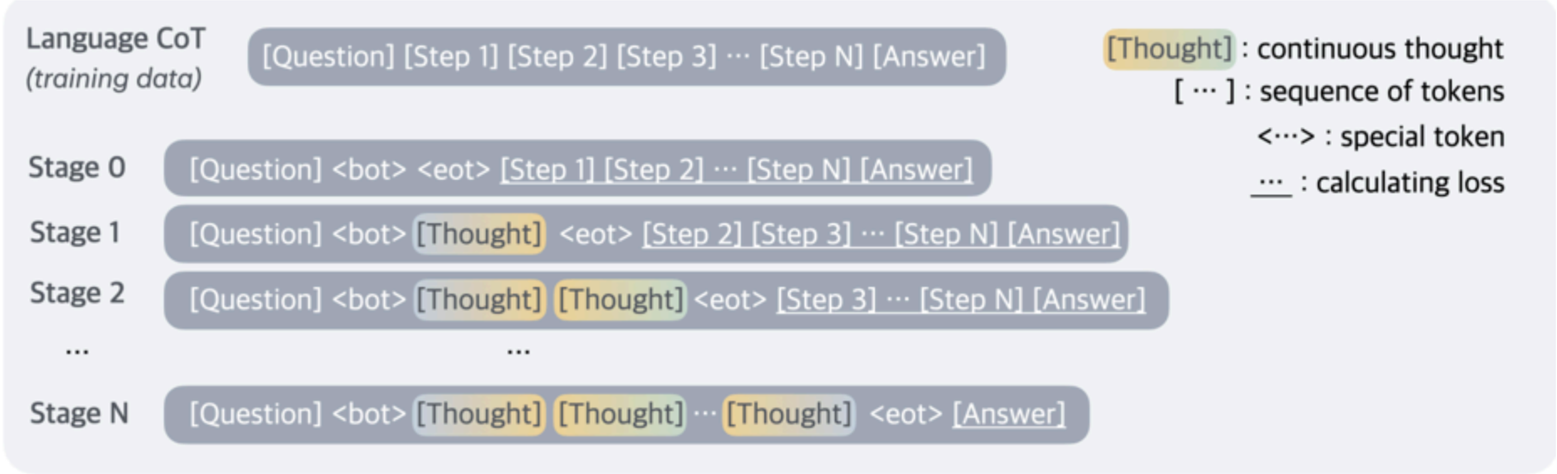


Figure 2: Training procedure of Chain of Continuous Thought (Coconut)

Multi-Stage Training Curriculum

Coconut uses a multi-stage training curriculum inspired by Deng et al. (2024) to effectively learn continuous thought reasoning:

1 2 3 4

Experimental Setup

Reasoning Tasks

1. **GSM8k**
   Grade school-level math problems for exploring latent reasoning in practical applications
2. **ProntoQA**
   5-hop logical reasoning questions with fictional concept names
3. **ProsQA**
   New dataset requiring substantial planning and searching over a graph to find the correct reasoning chain

Model Configuration

- Base model: Pre-trained GPT-2
- Learning rate: 1×10^{-4}
- Batch size: 128
- Optimizer reset between training stages
- Math reasoning: c = 2 (two latent thoughts per reasoning step)
- Logical reasoning: c = 1 (one latent thought per reasoning step)

Baselines & Variants

A. **CoT**
   Complete reasoning chains with supervised finetuning

B. **No-CoT**
   Directly generate the answer without reasoning chain

C. **iCoT**
   Internalized CoT using a carefully designed schedule

D. **Pause Token**
   Special `<pause>` tokens inserted between question and answer

E. **Coconut Variants**
   w/o curriculum, w/o thought, pause as thought

Evaluation Metrics

- Accuracy: Comparing model-generated answers with ground truth
- Efficiency: Number of newly generated tokens per question
- Clock-time: Average inference time per test case

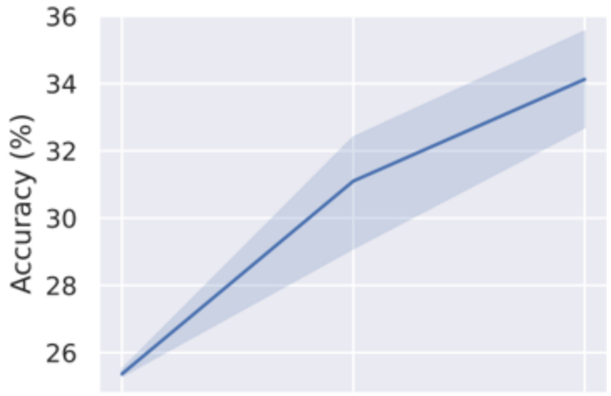
Results Overview

Performance Comparison Across Tasks

Method	GSM8k Acc. (%) / # Tokens	ProntoQA Acc. (%) / # Tokens	ProsQA Acc. (%) / # Tokens
CoT	42.9 ±0.2 / 25.0	98.8 ±0.8 / 92.5	77.5 ±1.9 / 49.4
No-CoT	16.5 ±0.5 / 2.2	93.8 ±0.7 / 3.0	76.7 ±1.0 / 8.2
iCoT	30.0* / 2.2	99.8 ±0.3 / 3.0	98.2 ±0.3 / 8.2
Pause Token	16.4 ±1.8 / 2.2	77.7 ±21.0 / 3.0	75.9 ±0.7 / 8.2
Coconut (Ours)	**34.1 ±1.5 / 8.2**	**99.8 ±0.2 / 9.0**	**97.0 ±0.3 / 14.2**
- w/o curriculum	14.4 ±0.8 / 8.2	52.4 ±0.4 / 9.0	76.1 ±0.2 / 14.2
- w/o thought	21.6 ±0.5 / 2.3	99.9 ±0.1 / 3.0	95.5 ±1.1 / 8.2
- pause as thought	24.1 ±0.7 / 2.2	100.0 ±0.1 / 3.0	96.6 ±0.8 / 8.2

* The result is from Deng et al. (2024).

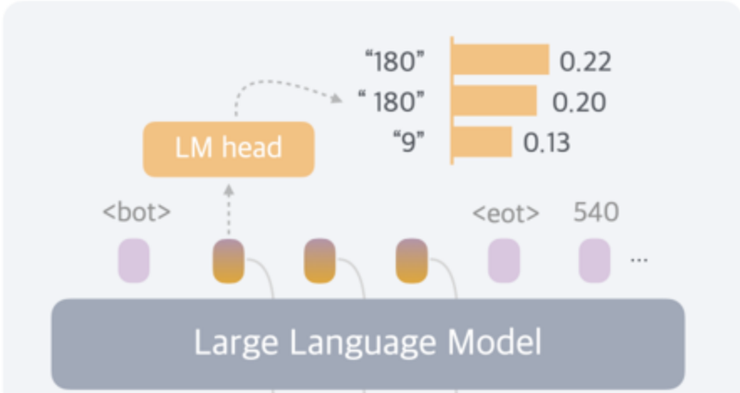
Analysis of Continuous Thoughts



"Chaining" Continuous Thoughts

As we increase the number of continuous thoughts per reasoning step (c) from 0 to 2, the model's performance steadily improves. This suggests that:

- Latent space reasoning retains the "chaining" property of CoT
- Method could scale to solve increasingly complex problems by chaining more latent thoughts
- Effective depth of the transformer increases with continuous thoughts



Efficient Representations

Though continuous thoughts are not intended to be decoded to language tokens, we can use the language model head to interpret them:

- Continuous thoughts can encode multiple potential next steps simultaneously
- The first thought in the example encodes key intermediate variables ("180", "9")
- This ability to encode multiple paths enables more advanced reasoning patterns

Performance on Planning-Intensive Tasks

Figure 5: The accuracy of final answer (left) and reasoning process (right) of multiple variants of Coconut and baselines on ProsQA

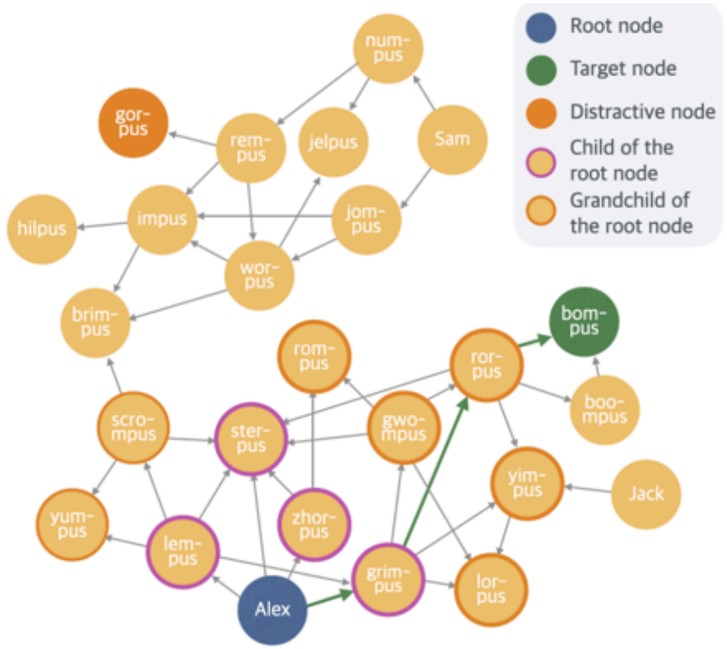
Interpolating Between Latent and Language Reasoning

By controlling the number of latent thoughts (k) during inference, we

Advantages of Latent Reasoning

On ProsQA, which requires strong planning ability, Coconut and its variants substantially enhance reasoning compared to CoT, indicating

Case Study: ProsQA



Question:

Every grimpus is a yimpus. Every worpus is a jelpus. Every zhorpus is a sterpus. Alex is a grimpus ⋯ Every lumps is a yumpus.
Question: **Is Alex a gorpus or bompus?**

Ground Truth Solution

Alex is a grimpus.
Every grimpus is a rorpus.
Every rorpus is a bompus.
Alex is a bompus

CoT

Alex is a lempus.
Every lempus is a scrompus.
Every scrompus is a yumpus.
Every yumpus is a rempus.
Every rempus is a gorpus.
Alex is a gorpus ✗

(Hallucination)

Coconut (k=1)

<bot> [Thought] <eot>
Every lempus is a scrompus.
Every scrompus is a brimpus.
Alex is a brimpus ✗

(Wrong Target)

Coconut (k=2)

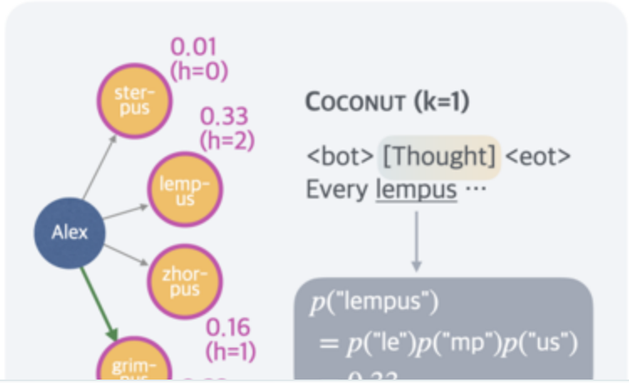
<bot> [Thought] [Thought] <eot>
Every rorpus is a bompus.
Alex is a bompus ✓

(Correct Path)

Figure 6: A case study of ProsQA. CoT hallucinates an edge, Coconut (k=1) outputs a path that ends with an irrelevant node, but Coconut (k=2) solves the problem correctly.

Comparison of Reasoning Approaches

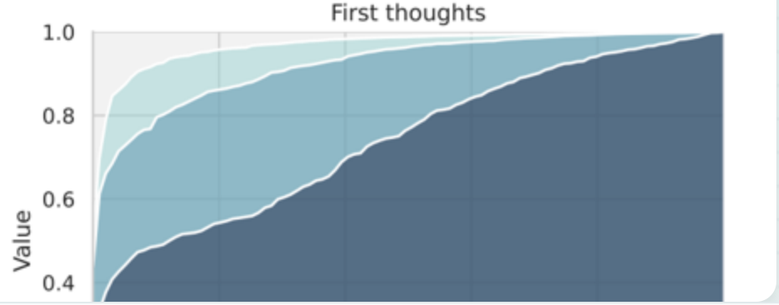
Interpreting the Latent Search Tree



Value Function Interpretation

The probability distribution over potential next concepts can be viewed as the model's implicit value function:

- First thought: "lempus" (0.33), "zhorpus" (0.16), "grimpus" (0.32), "sterpus" (0.01)
- Model has ruled out "sterpus" but remains uncertain about other options
- Second thought: Model has mostly ruled out other options and focused on "rorpus"



Parallel Exploration

The analysis of top-k candidate nodes reveals:

- First thoughts: Significant gaps between top-1, top-2, and top-3 candidates indicate broad exploration of alternatives
- Second thoughts: Narrower gaps show transition to more focused reasoning
- This demonstrates the model's ability to first explore broadly, then focus on promising paths

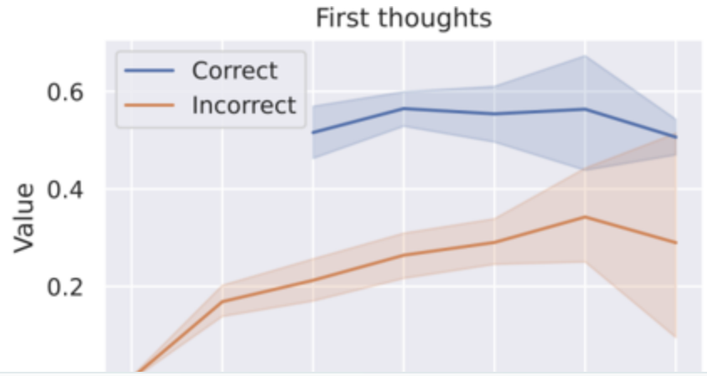
Why is a Latent Space Better for Planning?

Node Height Analysis

To understand why latent reasoning helps planning, we analyze node heights in the search tree:

- Height = shortest distance to any leaf node
- Nodes with lower heights have limited exploratory potential
- Nodes with higher heights require more complex evaluation

In our example, "sterpus" is a leaf node (height = 0), making it immediately identifiable as incorrect. "Grimpus" and "lempus" have height = 2, requiring more exploration to evaluate.



Empirical Evidence

Analysis across the test set reveals:

- For low-height nodes, the model successfully assigns lower values to incorrect nodes and higher values to correct nodes
- As node heights increase, this distinction becomes less pronounced
- This confirms that evaluating nodes closer to terminal states is easier
- Latent reasoning allows the model to delay hard decisions until more information is available

Key Advantage

By delaying definite decisions and expanding the latent reasoning process, the model pushes its exploration closer to the search tree's terminal states, making it easier to distinguish correct nodes from

Conclusion

Summary of Contributions

1. **Novel Paradigm:** Introduced Coconut (Chain of Continuous Thought), enabling LLMs to reason in an unrestricted latent space instead of using natural language.
2. **Emergent BFS-like Reasoning:** Discovered that continuous thoughts can encode multiple alternative next reasoning steps, allowing the model to perform a breadth-first search rather than prematurely committing to a single deterministic path.
3. **Improved Performance:** Demonstrated that Coconut outperforms CoT in logical reasoning tasks that require substantial backtracking during planning, with fewer thinking tokens during inference.
4. **Theoretical Understanding:** Provided insights into why latent reasoning is advantageous for planning by analyzing node heights in the search tree and showing how it helps delay hard decisions until more information is available.

Future Directions

1. ### Pretraining with Continuous Thoughts
   Enable models to generalize more effectively across reasoning scenarios
2. ### Scaling to Larger Models
   Apply Coconut to state-of-the-art LLMs for more complex reasoning tasks

Thank You!

Training Large Language Models to Reason in a Continuous Latent Space

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, Yuandong Tian

FAIR at Meta, UC San Diego

December 12, 2024

Coconut | Continuous Thought | Latent Reasoning | LLM Planning