# 33. RMT for step-by-step reasoning

Oleg Kashurin, Alexander Anokhin,
Viktor Lushnikov, Alexey Kravatsky, Ekaterina Krylova

AIRI Summer School 2025

**Abstract**

In this project, we investigate the efficiency of Recurrent Memory Transformer (RMT) models for mathematical and algorithmic reasoning tasks that require step-by-step solutions. We propose training strategies and construct synthetic datasets, including pretraining and summarization approaches, to adapt RMT for reasoning in supervised fine-tuning (SFT) mode. We implemented different approaches for SFT training of RMT for Chain-of-Thought (CoT) reasoning on gsm8k dataset. Additionally, we created inference time CoT improvement approach by iterative summarization of CoT steps. Our experiments showed that correct memory token utilization is crucial for improving reasoning accuracy.

## 1 Introduction

Recent advances in large language models (LLMs) have enabled step-by-step (Chain-of-Thought, CoT) reasoning for complex tasks. However, these tasks require models to handle long contexts, which poses computational challenges for vanilla transformers. The Recurrent Memory Transformer (RMT) architecture [1] reduces the computational cost for long sequences, making it a promising candidate for such applications. Our goal is to systematically study how to train RMTs for reasoning, comparing them to conventional transformers and exploring techniques to leverage RMT memory effectively.

## 2 Related Work

The Chain-of-Thought prompting approach enables LLMs to decompose complex tasks into intermediate steps, significantly improving performance in mathematical and algorithmic reasoning. RMT [1] introduces explicit memory tokens to handle long contexts efficiently. Works such as DeepSeekMath [**?**] have pushed the boundaries of open LLM mathematical reasoning, while datasets like ProsQA [2] and RIW facilitate systematic evaluation of reasoning capabilities. Recent studies also highlight the importance of memory utilization and summarization [2] in improving model reasoning accuracy.

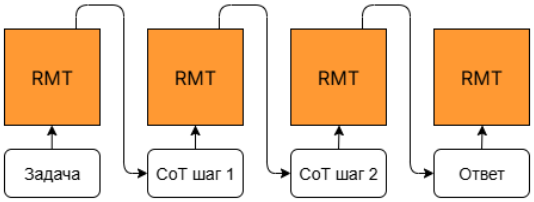


Figure 1: Token reasoning approach

# 3 Method

We develop several training strategies to adapt RMT for step-by-step reasoning:

- **Pretraining RMT**: We hypothesize that failure of RMT in reasoning is often due to underutilization of memory tokens. To address this, we propose (a) standard language modeling (LM) pretraining and (b) a modified pretraining where the model is forced to store useful information in memory tokens—for example, by segmenting input into [context + memory] and [memory + question].
- **CoT in Tokens**: Standard CoT training where reasoning steps are splitted into separate segments for RMT (see 1). We compare RMT and vanilla transformers in SFT mode, and also explore RMT training with GRPO (Generalized Reinforcement Policy Optimization) without teacher forcing.
- **CoT in Tokens And Latents**: we augment previous method by addition of intermediate segments filled with special latent tokens to support token reasoning with reasoning in latent space (see 2).
- **CoT in Latents**: In this approach we propose to completely drop reasoning with text tokens and use reasoning in latents similar to the approach in [2] (see 3).
- **Iterative Summarizing Ultra-mega Prompting (iSUMp)**: After each reasoning step, the model is prompted to summarize "what we know" (facts/numbers) and "what to do next" (goals), using this summary along with the original question to generate the next action. This approach encourages context compression and effective use of memory.

# 4 Experimental Setup

## 4.1 Synthetic datasets

We construct two synthetic datasets to test RMT performance.

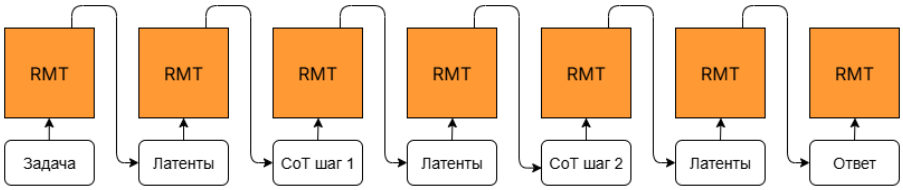


Figure 2: Token-latent reasoning approach

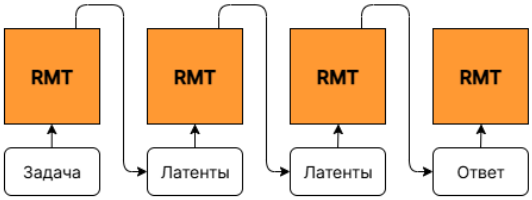


Figure 3: Latent reasoning approach

#### 4.1.1 Random Integer Walk

The dataset consists of 1000 samples. Each sample is a sequence of 2 to 102 vectors in $\mathbb{Z}^n$, where $n$ ranges from 1 to 100. Each vector is a rounding of $\mathcal{N}(\text{mean}, 5I_n)$, where mean $\sim \mathcal{N}(0, 5I_n)$. The task is to find the sum of the vectors in the sequence.

**Entry Example:**

Input: $(-1, -1, 0, -8); (0, -6, 1, -3); \ldots$

Output: $(-9, -44, 24, -61)$

A key feature is that partially correct answers (some coordinates correct) can be recognized, allowing for additional reward structures in RL training.

#### 4.1.2 ProsQA

We reproduce the ProsQA dataset from [2]. Samples are directed acyclic graphs with 23 nodes (see Figure 4). The task is to decide whether a given entity relates to a given concept, based on textual descriptions of graph edges.

**Example:**

Task: "Every numpus is a gorpus. Every shumpus is a sterpus. Fae is a dumpus. Every wumpus is a zumpus... Is Fae a gorpus or grimpus?"

Answer: grimpus

Chain of thought: ["Fae is a wumpus.", "Every wumpus is a tumpus.", ..., "Every impus is a grimpus."]

The challenge is that the model does not see the question until the end of the input, and much of the provided information is irrelevant.

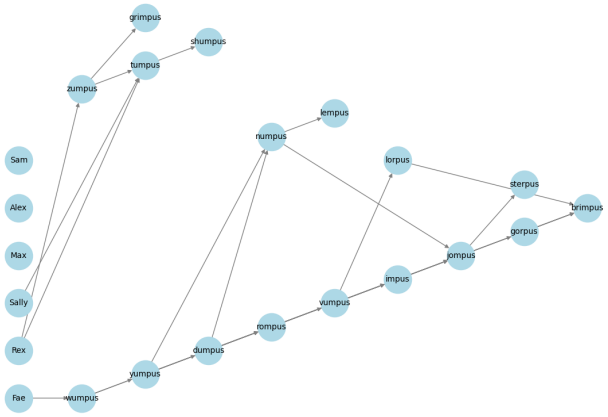


Figure 4: Graph example from ProsQA

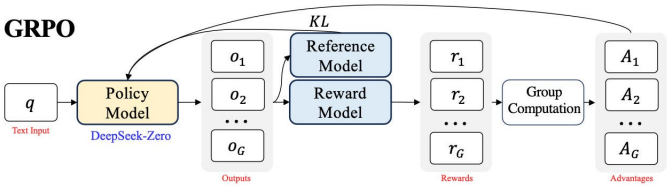


Figure 5: Graph example from ProsQA

### 4.2 GRPO

We build a GRPO implementation [3](see **??**) for RMT but we didn't conduct experiments with it yet.

## 5 Results

Our experiments reveal that:

- **RMT models require explicit training to use memory tokens effectively.** Without appropriate pretraining, RMT fails to develop reasoning abilities, as it cannot store and retrieve relevant context in memory tokens.
- **Iterative summarization (iSUMp) significantly boosts accuracy.** By simulating context compression, iSUMp helps the model remember key facts and goals at each reasoning step, improving final answers.
- **Pretraining on language modeling tasks is necessary, but tailored pretraining further enhances memory use.** Structuring inputs to

force usage of memory tokens leads to improved downstream reasoning on open QA tasks (e.g., MuSiQue, HotpotQA).

# 6 Conclusion

We have developed and tested a set of approaches for training RMTs in SFT mode on reasoning tasks. Our findings highlight the need for specialized pre-training and summarization techniques to ensure that RMTs utilize memory tokens efficiently.

# References


[1] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

[2] 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, 2024.

[3] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.