SoftCoT: Soft Chain-of-Thought for Efficient Reasoning with LLMs

Yige Xu[1,2]*, **Xu Guo**[1,*†], **Zhiwei Zeng**[1†], **Chunyan Miao**[1,2]
[1]Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly
[2]College of Computing and Data Science
Nanyang Technological University, Singapore
{yige002,xu008}@e.ntu.edu.sg, {zhiwei.zeng,ascymiao}@ntu.edu.sg

Abstract

Chain-of-Thought (CoT) reasoning enables Large Language Models (LLMs) to solve complex reasoning tasks by generating intermediate reasoning steps. However, most existing approaches focus on hard token decoding, which constrains reasoning within the discrete vocabulary space and may not always be optimal. While recent efforts explore continuous-space reasoning, they often require full-model fine-tuning and suffer from catastrophic forgetting, limiting their applicability to state-of-the-art LLMs that already perform well in zero-shot settings with a proper instruction. To address this challenge, we propose a novel approach for continuous-space reasoning that does not require modifying the LLM. Specifically, we employ a lightweight fixed assistant model to generate instance-specific soft thought tokens speculatively as the initial chain of thoughts, which are then mapped into the LLM's representation space via a trainable projection module. Experimental results on five reasoning benchmarks demonstrate that our method enhances LLM reasoning performance through supervised, parameter-efficient fine-tuning.

1 Introduction

In recent years, Large Language Models (LLMs) have become a cornerstone in Natural Language Processing (NLP), exhibiting advanced natural language understanding and generation (Brown et al., 2020; Chowdhery et al., 2023; OpenAI, 2023; Dubey et al., 2024; Yang et al., 2024). Scaling model sizes has not only improved instruction-following (Kojima et al., 2022) but also triggered emergent reasoning abilities, as first evidenced by chain-of-thought (CoT) prompting (Wei et al., 2022). CoT prompts LLMs to generate intermediate reasoning steps before providing the final answer, which not only enhances interpretability but also improves a range of reasoning-intensive tasks (Zhang et al., 2023; Sprague et al., 2024). It has inspired many advanced prompting frameworks, marking a paradigm shift from scaling training-time compute (Kojima et al., 2022) to scaling inference-time compute (Wang et al., 2023; Yao et al., 2023) to further boost LLM performance.

Nevertheless, CoT's effectiveness depends on the quality of intermediate thoughts, as the auto-regressive generation process can propagate errors. To mitigate this challenge, methods like self-consistency (Wang et al., 2023) generate multiple reasoning paths, while Tree-of-Thought (Yao et al., 2023) and Graph-of-Thought (Besta et al., 2024) frameworks organize these paths to select higher-quality steps. Despite these improvements, such methods are computationally inefficient due to the need for extensive thought sampling.

To enhance CoT efficiency, recent research explores skipping the decoding of hard tokens at intermediate steps. Methods like Continuous CoT (Cheng and Durme, 2024) and Coconut (Hao et al., 2024) conduct reasoning in a continuous space by using latent representations instead of discrete token sequences. Their results have shown to be superior to long-sequence discrete reasoning chains using only a short length of continuous representation. Yet, these methods require full-model fine-tuning, which incurs substantial computational costs, risks catastrophic forgetting, and limits their transferability across tasks.

We empirically observe that supervised fine-tuning of the LLaMA3.1-8B (Dubey et al., 2024) model with a language modeling objective on reasoning tasks (which is employed by both Coconut and CCoT) can lead to performance degradation compared with the zero-shot settings. We conjecture that this is due to catastrophic forgetting, a phenomenon also observed by Kalajdzievski (2024) and Lobo et al. (2024). Thus, the methodologies of Coconut, which is based on GPT-2 (Radford et al.,

*The first two authors contributed equally.
†Corresponding authors.

2019), and CCoT, which is built upon LLaMA2-7B (Touvron et al., 2023), may not be directly applicable to more recent models such as LLaMA3.1-8B. Therefore, it is crucial to explore alternative methodologies that mitigate catastrophic forgetting while effectively leveraging continuous reasoning techniques in large-scale, instruction-tuned models, which is the main research question of this work.

To mitigate catastrophic forgetting, a straightforward approach is to freeze the backbone LLM and instead optimize an external model for reasoning. Inspired by prompt tuning (Lester et al., 2021) and speculative decoding (Leviathan et al., 2023), we propose an approach that utilizes an auxiliary small assistant model to generate a sequence of "thought" tokens conditioned on a task instruction followed by a specific instance (Li et al., 2023; Shao et al., 2023). These tokens serve as instance-specific prompts to boost LLM's inference. Such an auxiliary prompting mechanism dynamically adapts to different reasoning tasks, thereby improving generalization while preserving the pre-trained knowledge of the LLM.

To facilitate reasoning in a continuous space, we use soft thought tokens (i.e., the last-layer hidden states from the small assistant model before mapping to the vocabulary space) instead of discrete tokens. This ensures reasoning remains within the continuous latent space. However, a representational gap between the assistant model and the LLM may hinder effective knowledge transfer. To bridge this gap, we train a projection module to map the soft thought tokens generated by the assistant model to the LLM's representation space. Training the projection module for each task can be seen as *soft prompt tuning* for the LLM. The overall **Soft** thoughts for **CoT** (SoftCoT) reasoning framework is illustrated in Figure 1.

To evaluate our proposed SoftCoT, we conduct experiments on five reasoning benchmarks and two state-of-the-art LLM architectures. The five benchmarks we choose include mathematical reasoning, commonsense reasoning, and symbolic reasoning. For further exploration, we create a hard version of the ASDiv dataset (Miao et al., 2020), which requires stronger mathematical reasoning ability. The new dataset is named "ASDiv-Aug" in this paper. Experimental results show that SoftCoT consistently improves accuracy on both public datasets and our augmented ASDiv-Aug dataset, demonstrating the effectiveness of SoftCoT in enhancing LLM's reasoning performance. Moreover, SoftCoT employs parameter-efficient fine-tuning, effectively mitigating catastrophic forgetting seen in full-model fine-tuning. Our results highlight the effectiveness of using an assistant model to generate soft thoughts for enhancing LLMs' reasoning capabilities while preserving their pre-trained knowlegde.

2 Related Works

Early research on chain-of-thought (CoT) reasoning can be traced back to Wei et al. (2022), who first introduced a prompting strategy that guides LLMs through decomposed intermediate reasoning steps using few-shot exemplars. Concurrently, Kojima et al. (2022) demonstrated that LLMs are capable of zero-shot CoT reasoning by simply appending the phrase "Let's think step by step" to the prompt template. This discovery underscored the latent reasoning abilities of LLMs, even in the absence of explicit demonstrations.

Building upon these foundational works, the NLP community has extensively explored the potential of CoT reasoning. As summarized by Chu et al. (2024), recent advancements in CoT methods can be broadly categorized into three areas: (1) *Prompt Construction*, which aims to optimize prompts for improved CoT reasoning (Wei et al., 2022; Kojima et al., 2022; Zhang et al., 2023); (2) *Topological Variants*, which leverage structured representations such as trees and graphs to enhance CoT reasoning (Yao et al., 2023; Besta et al., 2024); and (3) *Enhancement Methods*, which introduce external strategies to further improve CoT reasoning, such as question decomposition (Zhou et al., 2023) and self-consistency decoding (Wang et al., 2023). Despite the effectiveness of these approaches, the majority of existing CoT methods rely on discrete token-by-token generation, which imposes inherent constraints and limits their expressiveness.

To address the limitations of discrete language space, an effective approach is to leverage continuous representation space for reasoning. Coconut (Hao et al., 2024) introduces a Chain-of-Continuous-Thought, while CCoT (Cheng and Durme, 2024) employs Compressed Chain-of-Thought, generating content-rich and continuous contemplation tokens. Heima (Shen et al., 2025) further advances this idea by utilizing a single continuous vector to represent compressed reasoning tokens in multi-modal tasks. However, both Co-

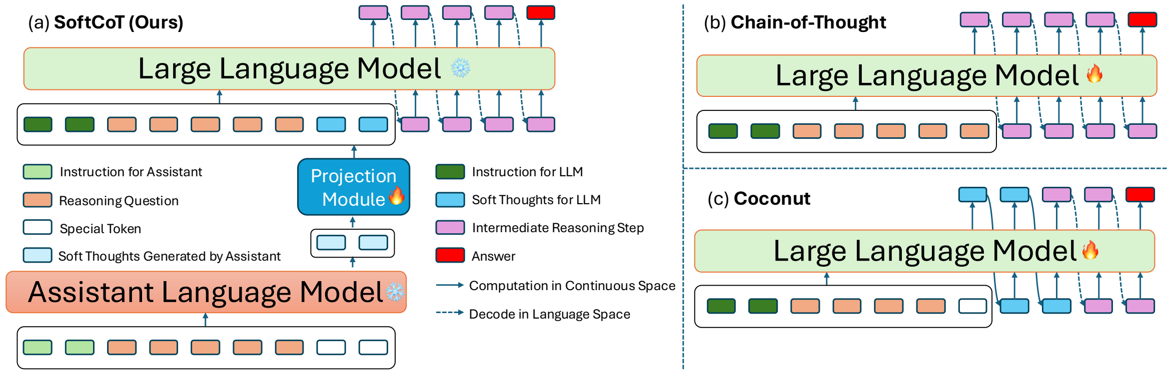

Figure 1: A comparison of SoftCoT, vanilla Chain-of-Thought, and Coconut.

conut and CCoT rely on a language modeling objective for supervised fine-tuning, which is infeasible for state-of-the-art LLMs due to the catastrophic forgetting problem. Moreover, Heima underperforms compared to its backbone model, LLaVA-CoT (Xu et al., 2024). These challenges underscore the need to develop methodologies that mitigate catastrophic forgetting in the application of continuous-space CoT reasoning.

3 Methodology

3.1 Problem Definition and Notations

Given a question $\mathcal{Q} = [q_1, q_2, \cdots, q_{|\mathcal{Q}|}]$, the CoT reasoning will solve the problem on the following two steps: (1) Auto-regressively generate a list of rationale steps $\mathcal{R} = [r_1, r_2, \cdots, r_{|\mathcal{R}|}]$ according to the question; (2) Auto-regressively generate the answer $\mathcal{A} = [a_1, a_2, \cdots, a_{|\mathcal{A}|}]$ according to the question as well as the rationale steps. The generation process can be described as:

$$\begin{aligned} r_{i+1} &= \mathrm{LLM}(\mathcal{Q}; \mathcal{R}_{\leq i}), \\ a_{j+1} &= \mathrm{LLM}(\mathcal{Q}; \mathcal{R}; \mathcal{A}_{\leq j}), \end{aligned} \tag{1}$$

where $\mathrm{LLM}(\cdot)$ indicates a large language model, $\mathcal{R}_{\leq i} = [r_1, \cdots, r_i]$ indicates the previous generated i reasoning tokens, and $\mathcal{A}_{\leq j} = [a_1, \cdots, a_j]$ indicates the previous generated j answer tokens.

The majority of recent works (Zhang et al., 2023; Zhou et al., 2023; Yao et al., 2023) focus on generating discrete hard tokens in $\mathcal{R}$, which is named as "**Hard-CoT**" in this paper. On contrast, some recent works (Hao et al., 2024; Cheng and Durme, 2024) focus on the continuous representations (*a.k.a* latent space) of $\mathcal{R}$, which is named as "**Soft-CoT**" in this paper.

In this paper, we manually define some rules (e.g., regular expression matching) to extract the framework answer $\hat{\mathcal{A}}$ from the answer $\bar{\mathcal{A}}$ generated by the LLM. Then we compute the accuracy of $\hat{\mathcal{A}}$ comparing with the ground-truth answer $\mathcal{A}$.

3.2 Overview of the SoftCoT Framework

SoftCoT is a novel framework designed to enhance reasoning capabilities in large language models (LLMs). Given an input question $\mathcal{Q}$, the framework produces both a sequence of reasoning steps $\mathcal{R}$ and the final answer $\mathcal{A}$. SoftCoT consists of three key components: the soft thought token generation module, the projection module, and the CoT reasoning module. The overall architecture is illustrated in Figure 1(a).

The soft thought token generation module is inspired by prompt tuning techniques (Lester et al., 2021). In conventional prompt tuning, learnable prompts facilitate the retrieval of knowledge stored within the LLM (Xu et al., 2023). In SoftCoT, soft thought tokens are generated by an assistant language model, which is typically smaller than the backbone LLM (e.g., LLaMA-3.2-1B-Instruct as the assistant model and LLaMA-3.1-8B-Instruct as the backbone model).

A key challenge in this setup is that the assistant model can only generate discrete token sequences as input to the backbone LLM, which imposes constraints and may not always yield optimal prompts. To address this limitation, we introduce continuous soft thought tokens that enable more expressive and flexible prompting. However, a representation gap exists between the assistant model and the backbone LLM, necessitating an intermediate transformation.

To bridge this gap, the projection module maps the soft thought tokens' representations into a space more compatible with the backbone LLM. This ensures that the soft thought tokens effectively guide the reasoning process.

Finally, the CoT reasoning module leverages both the learned soft thought tokens and word embeddings to generate intermediate reasoning steps $\bar{\mathcal{R}}$ and the final answer $\bar{\mathcal{A}}$. The model is trained using a language modeling objective, optimizing the learnable parameters across the rationale steps and the answer spans.

3.3 Prompt Tuning for CoT Reasoning

Prompt tuning for CoT reasoning aims to optimize the structure and content of the prompt template to enhance the reasoning capabilities of a large language model (LLM). This process can be mathematically formulated as follows:

$$\begin{aligned} \hat{y} &= \mathrm{LLM}\big(P_{\mathbf{p}}(x)\big), \\ \mathbf{p}^{*} &= \arg\min_{\mathbf{p}} \mathcal{L}(\hat{y}, y), \end{aligned} \tag{2}$$

where $\hat{y}$ represents the predicted output, x denotes the input sequence, and $P_{\mathbf{p}}(x)$ is the input augmented with a prompt $\mathbf{p}$. The objective function $\mathcal{L}(\cdot)$ measures the discrepancy between the model's prediction $\hat{y}$ and the ground-truth label y. The primary goal of prompt tuning is to determine an optimal prompt configuration that effectively guides the LLM to perform CoT reasoning with improved accuracy and interpretability.

A straightforward yet effective approach to optimizing prompts involves leveraging an auxiliary assistant model to generate instance-specific prompts, which provide contextual hints or question summaries to facilitate reasoning (Li et al., 2023; Shao et al., 2023). In this framework, the prompt $\mathbf{p}$ can be decomposed into two components: (1) a fixed, task-specific prompt $\mathbf{p}_{\text{❄}}$, which remains constant across all instances and encodes general problem-solving heuristics, and (2) a learnable, instance-specific prompt $\mathbf{p}_{\text{🔥}}$, which dynamically adapts to each input instance to provide tailored guidance.

Given the rapid advancements in LLMs, many LLMs are capable of solving complex reasoning tasks under zero-shot settings. Instead of fine-tuning the assistant model for each task, we adopt a more efficient strategy by employing a relatively small, frozen language model to generate $\mathbf{p}_{\text{🔥}}$. This approach not only reduces computational costs but also ensures stability and generalization across different problem domains. By systematically integrating instance-specific prompting with fixed task-specific instructions, this method enhances the LLM's reasoning process while maintaining adaptability to various contexts.

3.4 Soft Thought Tokens for CoT Reasoning

One of the advantages of Hard-CoT is that the generated discrete tokens can be tokenized by the LLMs, which does not require a external mapping module. However, there are two main limitations of Hard-CoT: (1) The decoded token space is discrete, which is constrained and sometimes not optimal; (2) The gradient cannot backpropagate to the assistant model since the decoding process cut off the gradient information. A convincing solution is replace the hard tokens to soft thought tokens.

Generating Soft Thought Tokens with an Assistant Model To generate instance-specific soft thoughts, we utilize an auxiliary assistant model that produces soft thoughts based on the given reasoning task. The input to the assistant model, denoted as $\mathbf{x}_{\text{assist}}$, consists of three main components:

$$\mathbf{x}_{\text{assist}} = \big[\mathcal{I}, \mathcal{Q}, \texttt{[UNK]}_{1:N}\big], \tag{3}$$

where

- $\mathcal{I}$ represents the instructional context provided to the assistant model, guiding it on how to generate relevant thoughts.
- $\mathcal{Q}$ denotes the reasoning question that the primary LLM will solve, which has been defined in Section 3.1.
- N `[UNK]` tokens serve as placeholders for the soft thoughts.

Once the input sequence is constructed, the assistant model processes it, and the soft thought tokens are obtained as follows:

$$\begin{aligned} \mathbf{h}^{\text{assist}} &= \mathrm{Assistant}(\mathbf{x}_{\text{assist}}), \\ \mathbf{t}_{\text{assist}} &= \mathbf{h}^{\text{assist}}_{|\mathcal{I}|+|\mathcal{Q}|+1:|\mathcal{I}|+|\mathcal{Q}|+N}. \end{aligned} \tag{4}$$

Here $\mathbf{h}^{\text{assist}}$ denotes the final-layer hidden states of the assistant model, and $\mathbf{t}_{\text{assist}}$ corresponds to the segment of $\mathbf{h}^{\text{assist}}$ associated with the N `[UNK]` tokens. This extracted representation serves as the instance-specific soft thoughts, dynamically adapting to the input reasoning question.

Projection Module Since there exist both a representation gap and a dimensional gap between the assistant language model and the primary LLM, a direct utilization of $\mathbf{t}_{\text{assist}}$ may lead to suboptimal performance. The assistant model and the LLM often operate in different embedding spaces, with distinct hidden state distributions and dimensionalities. To bridge this gap, we introduce a projection module that maps the assistant-generated soft thoughts $\mathbf{t}_{\text{assist}}$ from the assistant model's embedding space to the LLM's embedding space:

$$\mathbf{t}_{🔥} = \text{Linear}_{\theta}(\mathbf{t}_{\text{assist}}), \tag{5}$$

where $\text{Linear}_{\theta} : \mathbb{R}^{d_{\text{assist}}} \rightarrow \mathbb{R}^{d_{\text{LLM}}}$ is a trainable projection layer parameterized by θ. This layer ensures that the assistant-generated soft thoughts are transformed into a suitable format for the LLM, preserving relevant semantic information while adapting to the LLM's feature space.

By incorporating this projection module, we effectively mitigate discrepancies between the assistant model and the LLM, enabling smooth integration of instance-specific thoughts into the CoT reasoning process. This design ensures that the learned thought tokens are both informative and compatible, thereby enhancing the overall reasoning performance of the LLM.

LLM Reasoning with Soft CoT With instance-specific soft thought tokens generated by the assistant model and mapped to the LLM's embedding space, we proceed to the final step: applying these soft thoughts to aid LLMs in CoT reasoning.

The input to the LLM, denoted as $\mathbf{x}_{\text{LLM}}$, follows a structure similar to that of $\mathbf{x}_{\text{assist}}$:

$$\mathbf{x}_{\text{LLM}} = \left[\mathbf{p}_{❄}, \mathcal{Q}, \mathbf{t}_{🔥}\right], \tag{6}$$

where

- $\mathbf{p}_{❄}$ is the task-specific instruction, which is a fixed prompt shared across all instances of the same task. It provides general problem-solving heuristics and instructions relevant to the reasoning task.
- $\mathbf{t}_{🔥}$ is the instance-specific soft thoughts computed by Eq (5). This component dynamically adapts soft thought tokens to each input question, enhancing contextual understanding.

With this structured input, the LLM generates step-by-step reasoning chains, following the principles of CoT reasoning. The reasoning process unfolds by systematically applying logical deductions or problem-solving heuristics, ultimately leading to the generation of the final answer:

$$\begin{aligned} \bar{\mathcal{R}} &= \text{LLM}(\mathbf{x}_{\text{LLM}}), \\ \bar{\mathcal{A}} &= \text{LLM}(\mathbf{x}_{\text{LLM}}, \bar{\mathcal{R}}), \\ \hat{\mathcal{A}} &= \mathcal{E}(\bar{\mathcal{A}}), \end{aligned} \tag{7}$$

where $\mathcal{E}(\cdot)$ is mannual rules for answer extraction.

By integrating both fixed task-specific instructions and instance-specific soft thought tokens, our approach enables the LLM to systematically decompose complex reasoning tasks while leveraging auxiliary knowledge provided by the assistant model. The structured input ensures that the LLM benefits from both general domain knowledge and tailored instance-level guidance, ultimately improving its reasoning effectiveness.

Parameter-Efficient Training In this work, we focus on reasoning tasks that include annotated reasoning steps, which provide explicit intermediate reasoning trajectories leading to the final answer. To effectively train the model, we employ the standard language modeling objective (also known as next-token prediction) to supervise the generation of soft thoughts. During the training stage, the input sequence is structured as follows:

$$\mathbf{x}_{\text{train}} = \left[\mathbf{p}_{❄}, \mathcal{Q}, \mathbf{t}_{🔥}, \mathcal{R}, \mathcal{A}\right]. \tag{8}$$

To effectively learn the soft thoughts, we apply the negative log-likelihood (NLL) loss over the reasoning steps and the answer span. Specifically, we mask the tokens before the intermediate reasoning steps to prevent the model from directly relying on them during loss computation. Instead, the model is trained to generate the reasoning steps $\mathcal{R}$ and final answer $\mathcal{A}$ in an autoregressive manner.

4 Experiments

4.1 Datasets

We conduct experiments on five reasoning datasets spanning three categories of reasoning: mathematical reasoning, commonsense reasoning, and symbolic reasoning. For mathematical reasoning, we utilize **GSM8K** (Cobbe et al., 2021), **ASDiv** (Miao et al., 2020), and **AQuA** (Ling et al., 2017). For commonsense reasoning, we employ **StrategyQA** (Geva et al., 2021). For symbolic reasoning, we use **Date Understanding** (BIG.Bench.authors, 2023) from the BIG-benchmark.

Given that LLaMA-3.1-8B-Instruct is a well-trained LLM, we augment the ASDiv dataset to ensure that the model encounters novel instances. Specifically, we replicate each instance five times and systematically extract and replace numerical values in the questions with randomly selected alternatives. This augmentation is designed to evaluate the model's reasoning capability rather than its ability to recognize patterns from memorized data. The augmented dataset is named as "**ASDiv-Aug**" in the following part of this paper. All detail statistics of the datasets is shown in Table 1.

4.2 Baselines

We consider the following baselines:

Coconut (Hao et al., 2024). It enables LLMs to reason in a continuous latent space by iteratively feeding hidden states as input embeddings, allowing for more efficient and flexible multi-path reasoning compared to traditional language-based chain-of-thought methods.

Zero-Shot CoT To evaluate whether our trained model has performance degration after supervised fine-tuning, we apply zero-shot CoT based on the prompt templates from Sprague et al. (2024).

Zero-Shot CoT-Unk We directly add some `[UNK]` tokens to represent the un-tuned prompts for CoT reasoning. This baseline exams the effectiveness of the tuned prompts.

Zero-Shot Assist-CoT We directly require the assistant model generates some hard prompt tokens for CoT reasoning. This baseline exams the effectiveness of the tuned soft prompts.

5 Results and Discussions

5.1 Comparison with Baselines

To evaluate SoftCoT, we compare its performance against the baselines introduced in Section 4.2. The results are summarized in Table 2:

(1) **Coconut is not applicable to larger language models**: We modify and run the official implementation of Coconut, adapting it to LLaMA-3.1-8B-Instruct. Our findings indicate that Coconut exhibits performance degradation following supervised fine-tuning with the language modeling objective, which can be attributed to the catastrophic forgetting phenomenon. This observation aligns with findings from prior studies, including Kalajdzievski (2024) and Lobo et al. (2024), which have reported similar issues.

(2) **Incorporating `[UNK]` tokens mitigates performance variance**: We examined the effect of directly adding `[UNK]` tokens as thoughts $\mathbf{t}_{\text{🔥}}$ in Eq. (6). The results demonstrate a slight improvement in overall performance and a reduction in variance. The `[UNK]` token, also known as the "pause token" (Goyal et al., 2024), appears to expand the model's computational capacity, leading to more stable and consistent outputs.

(3) **Assistant model is effective to facilitate CoT reasoning**: We utilize instruction to require the assistant model generate some hard prompts, which can be regarded as the initial thoughts for CoT reasoning. Experiment results show that although it has a larger variance than CoT-Unk, it facilitates the LLM for more diverse CoT generation, which leads to the performance improvement from 68.49 to 69.67 in average.

(4) **SoftCoT consistently benefits from the supervised fine-tuning**: Overall, our proposed SoftCoT consistently outperforms baselines across all five reasoning datasets, involving the mathematical reasoning, the commonsense reasoning, and the symbolic reasoning. The experimental result highlights that our SoftCoT benefits from the supervised fine-tuning and mitigates the catastrophic forgetting problems in state-of-the-art LLMs.

5.2 Generalization to Other LLM Backbones

In addition to LLaMA-3.1, we evaluate SoftCoT on another state-of-the-art LLM family: Qwen2.5 (Yang et al., 2024). Specifically, we select Qwen2.5-7B-Instruct as the backbone LLM to assess the generalization capability of SoftCoT. As shown in Table 3, our analysis yields the following three key findings:

(1) **SoftCoT is effective across different backbone models**: Experimental results on Qwen2.5-7B-Instruct show that SoftCoT consistently improves performance across all reasoning datasets, underscoring its robustness. These findings suggest that SoftCoT serves as a generalizable framework that can be effectively adapted to diverse state-of-the-art LLM architectures.

(2) **SoftCoT enhances LLMs' weaker areas while preserving their strengths**: Experiments on both LLaMA and Qwen LLMs reveal that SoftCoT yields the most significant improvements in

Dataset	Task Type	Answer Type	# Train samples	# Evaluation samples
GSM8K	Mathematical	Number	7,473	1,319
ASDiv-Aug		Number	4,183	1,038
AQuA		Option	97,467	254
StrategyQA	Commonsense	Yes/No	1,832	458
DU	Symbolic	Option	-	250

Table 1: Summary statistics of fiva datasets we used. "-" indicates that there is no training samples available.

Model	GSM8K	ASDiv-Aug	AQuA	StrategyQA	DU	Avg.
	Mathematical			Commonsense	Symbolic	
GPT-2						
Coconut (Hao et al., 2024)	$34.10^{*}_{\pm 1.50}$	$38.92^{\dagger}_{\pm 0.00}$	$22.83^{\dagger}_{\pm 0.00}$	-	-	-
LLaMA-3.1-8B-Instruct						
Zero-Shot CoT	$79.61_{\pm 0.81}$	$86.78_{\pm 0.63}$	$54.65_{\pm 2.43}$	$65.63_{\pm 3.31}$	$54.40_{\pm 2.40}$	68.21
Zero-Shot CoT-Unk	$79.95_{\pm 0.59}$	$86.90_{\pm 0.41}$	$55.28_{\pm 1.88}$	$66.16_{\pm 2.70}$	$54.16_{\pm 1.46}$	68.49
Zero-Shot Assist-CoT	$80.76_{\pm 1.53}$	$86.96_{\pm 0.46}$	$55.83_{\pm 2.98}$	$66.55_{\pm 3.99}$	$58.24_{\pm 3.56}$	69.67
Coconut (Hao et al., 2024)[†]	$76.12_{\pm 0.00}$	$86.80_{\pm 0.00}$	$53.15_{\pm 0.00}$	-	-	-
SoftCoT (Ours)	$\mathbf{81.03}_{\pm 0.42}$	$\mathbf{87.19}_{\pm 0.40}$	$\mathbf{56.30}_{\pm 1.67}$	$\mathbf{69.04}_{\pm 1.23}$	$\mathbf{59.04}_{\pm 1.93}$	**70.52**

Table 2: Model comparison with baselines. "DU" indicates the Date Understanding (BIG.Bench.authors, 2023) dataset. The first row is under the backbone of GPT-2 (Radford et al., 2019) as backbone. The following rows are under the backbone of LLaMA-3.1-8B-Instruct (Dubey et al., 2024). The last two rows are models trained via the language modeling objective. We run for 5 random seeds and report the average accuracy as well as the standard variance. "*" indicates that the accuracy is reported by Hao et al. (2024). "†" indicates the results that we modify and run the official code of Coconut. ±0.00 indicates that we only run once for baseline results.

Model	GSM8K	ASDiv-Aug	AQuA	StrategyQA	DU	Avg.
	Mathematical			Commonsense	Symbolic	
Zero-Shot CoT	$83.70_{\pm 0.78}$	$87.19_{\pm 0.28}$	$64.53_{\pm 3.27}$	$49.65_{\pm 3.18}$	$66.40_{\pm 2.26}$	70.29
Zero-Shot CoT-Unk	$84.12_{\pm 0.71}$	$86.94_{\pm 0.89}$	$64.72_{\pm 2.06}$	$50.74_{\pm 1.90}$	$66.48_{\pm 1.43}$	70.60
Zero-Shot Assist-CoT	$84.85_{\pm 0.35}$	$88.63_{\pm 1.05}$	$64.96_{\pm 2.83}$	$52.71_{\pm 2.65}$	$67.04_{\pm 2.84}$	71.64
SoftCoT (Ours)	$\mathbf{85.81}_{\pm 1.82}$	$\mathbf{88.90}_{\pm 1.01}$	$\mathbf{72.44}_{\pm 2.19}$	$\mathbf{60.61}_{\pm 1.55}$	$\mathbf{67.52}_{\pm 2.92}$	**75.06**

Table 3: Model performance using Qwen2.5-7B-Instruct.

commonsense reasoning tasks, where LLMs typically underperform compared to mathematical reasoning. This advantage may stem from SoftCoT's ability to generate contextually relevant continuous thought processes, effectively activating the corresponding knowledge areas within the model. Furthermore, SoftCoT helps mitigate catastrophic forgetting in domains where LLMs already excel, such as mathematical reasoning, thereby preserving and reinforcing existing capabilities.

(3) **SoftCoT facilitates domain transfer**: Given that the Date Understanding dataset lacks training samples, we train the model on other similar datasets and apply zero-shot transfer to evaluate its generalization on Date Understanding. The results indicate that SoftCoT consistently enhances performance in zero-shot domain transfer scenarios, further demonstrating its adaptability.

5.3 Model Analysis and More Studies

5.3.1 Model-Related Factors

To better understand SoftCoT, we conduct experiments to examine the impact of the number of thought tokens. The results, presented in Figure 2,

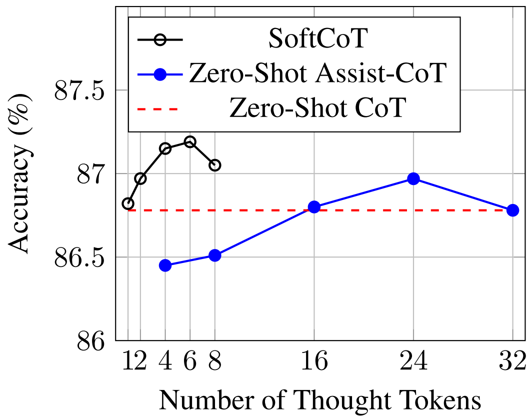

Figure 2: The impact of thought token numbers in ASDiv-Aug using LLaMA-3.1-8B-Instruct.

lead to the following key observations:
(1) **Soft thoughts reduce the required number of thought tokens**: We observe that SoftCoT achieves optimal performance with only six thought tokens, whereas Zero-Shot Assist-CoT requires 24 thought tokens to reach a similar level of effectiveness. This suggests that soft thoughts, which operate in a continuous space, exhibit a stronger representational capacity than hard thoughts expressed in the discrete language space. Our experiments indicate that the optimal number of hard thought tokens is approximately four times that of soft thought tokens, aligning with the 5x ratio reported by Cheng and Durme (2024).
(2) **SoftCoT mitigates the catastrophic forgetting problem**: Experimental results show that SoftCoT consistently outperforms Zero-Shot CoT across all tested numbers of soft thought tokens. In contrast, Zero-Shot Assist-CoT underperforms compared to Zero-Shot CoT when the number of thought tokens is insufficient. This is likely because the assistant model fails to generate a sufficiently informative set of thought tokens under these constraints, introducing noise and leading to confusion in the LLM's reasoning process.

5.3.2 Model-Orthogonal Factors

Self-Consistency (Wang et al., 2023) is a widely adopted technique for enhancing Chain-of-Thought (CoT) reasoning by expanding the search space. One of the most straightforward implementations involves generating multiple CoT reasoning paths and determining the final answer through majority voting. This approach is effective in mitigating errors in reasoning steps by leveraging the diversity of generated thought processes.

Model	$N = 1$	$N = 10$
Zero-Shot CoT	79.61	90.37
Zero-Shot CoT-Unk	79.95	90.43
Zero-Shot Assist-CoT	80.76	90.54
SoftCoT	**81.03**	**90.98**

Table 4: Self Consistency for SoftCoT on LLaMA-3.1-8B-Instruct. "N" indicates the number of reasoning chains.

To further assess the effectiveness of SoftCoT, we conduct experiments incorporating self-consistency. As shown in Table 4, SoftCoT consistently outperforms baseline models, demonstrating that its benefits are complementary to those of self-consistency rather than being redundant or conflicting. This suggests that SoftCoT introduces an independent improvement mechanism, which can be effectively combined with self-consistency for enhanced reasoning performance.

A key advantage of SoftCoT in this context is its ability to provide a more expressive and compact representation of intermediate reasoning steps in continuous space. Unlike traditional CoT reasoning, where discrete thought tokens may introduce inconsistencies or redundant reasoning paths, SoftCoT enables more efficient reasoning trajectories with fewer tokens. This allows self-consistency methods to aggregate results from higher-quality reasoning paths, leading to a more robust and accurate final prediction.

6 Conclusion

In this paper, we introduce SoftCoT, a soft chain-of-thought prompting approach for efficient LLM reasoning. SoftCoT consists of three steps: (1) an assistant model generates soft thought tokens, (2) a projection module trained to map the soft thoughts to LLM's representation space, and (3) the LLM applies soft thoughts for reasoning. To enhance efficiency, SoftCoT speculatively generates *all* the soft thought tokens in a single forward pass. To mitigate the catastrophic forgetting, SoftCoT freezes the backbone LLM and only tunes the projection module. Experiments on five datasets across three types of reason tasks demonstrate the effectiveness of our proposed SoftCoT. Experiments on multiple LLMs as well as orthogonal method such as self-consistency shows the robustness of SoftCoT, which can be adapted in widely scenarios.

Limitations

While SoftCoT represents a promising advancement in Chain-of-Thought (CoT) reasoning within a continuous space, several limitations must be acknowledged.

SoftCoT Cannot Fully Replace the Reasoning Path : Although SoftCoT employs soft thought tokens for reasoning, it does not entirely replace the reasoning path. The decoding stage functions as a search process, which is a crucial component of CoT reasoning. Soft thought tokens primarily serve to enrich the probability space for exploration rather than acting as the search mechanism itself.

Need for Further Empirical Evidence on Scalability : SoftCoT has been evaluated on LLaMA-3.1-8B-Instruct and Qwen2.5-7B-Instruct. However, larger backbone LLMs exist within the same model families. While its success on models with approximately 7–8 billion parameters suggests potential applicability to larger-scale models, its scalability to extremely large LLMs remains an open question and requires thorough empirical validation.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 17682–17690. AAAI Press.

BIG.Bench.authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Trans. Mach. Learn. Res.*, 2023.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Tao He, Haotian Wang, Weihua Peng, Ming Liu, Bing Qin, and Ting Liu. 2024. Navigate through enigmatic labyrinth A survey of chain of thought reasoning: Advances, frontiers and future. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1173–1203. Association for Computational Linguistics.

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*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Trans. Assoc. Comput. Linguistics*, 9:346–361.

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. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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*.

Damjan Kalajdzievski. 2024. Scaling laws for forgetting when fine-tuning large language models. *arXiv preprint arXiv:2401.05605*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances*