

# Expediting and Elevating Large Language Model Reasoning via Hidden Chain-of-Thought Decoding

**Tianqiao Liu**

TAL Education Group  
Beijing, China  
liutianqiao1@tal.com

**Zui Chen**

TAL Education Group  
Beijing, China  
chenzui3@tal.com

**Zitao Liu**

Jinan University  
Guangzhou, China  
liuzitao@jnu.edu.cn

**Mi Tian**

TAL Education Group  
Beijing, China  
tianmi@tal.com

**Weiwei Luo**

Jinan University  
Guangzhou, China  
lwq@jnu.edu.cn

## Abstract

Large language models (LLMs) have demonstrated remarkable capabilities in tasks requiring reasoning and multi-step problem-solving through the use of chain-of-thought (CoT) prompting. However, generating the full CoT process results in significantly longer output sequences, leading to increased computational costs and latency during inference. To address this challenge, we propose a novel approach to compress the CoT process through semantic alignment, enabling more efficient decoding while preserving the benefits of CoT reasoning. Our method introduces an auxiliary CoT model that learns to generate and compress the full thought process into a compact special token representation semantically aligned with the original CoT output. This compressed representation is then integrated into the input of the Hidden Chain-of-Thought (HCoT) model. The training process follows a two-stage procedure: First, the CoT model is optimized to generate the compressed token representations aligned with the ground-truth CoT outputs using a contrastive loss. Subsequently, with the CoT model parameters frozen, the HCoT model is fine-tuned to generate accurate subsequent predictions conditioned on the prefix instruction and the compressed CoT representations from the CoT model. Extensive experiments across three challenging domains - mathematical reasoning, agent invocation, and question answering - demonstrate that our semantic compression approach achieves competitive or improved performance compared to the full CoT baseline, while providing significant speedups of at least 1.5x in decoding time. Moreover, incorporating contrastive learning objectives further enhances the quality of the compressed representations, leading to better CoT prompting and improved task accuracy. Our work paves the way for more efficient exploitation of multi-step reasoning capabilities in LLMs across a wide range of applications.

## 1 Introduction

Chain-of-Thought (CoT) prompting, as introduced by (Wei et al., 2022), involves prompting large language models (LLMs) to generate explicit reasoning steps, significantly enhancing their performance in various reasoning tasks such as mathematical problem solving (Hendrycks et al., 2021, Cobbe et al., 2021) and science question answering (Lu et al., 2022). Subsequent CoT variants (Zhou et al., 2023, Chen et al., 2022, Gao et al., 2023) have aimed at further improving its efficacy across diverse domains. However, these methods enhance LLMs’ reasoning capabilities by extending the

duration and complexity of reasoning processes and incorporating external computational resources to achieve superior outcomes. This increased computational demand may limit their applicability in real-world development scenarios.

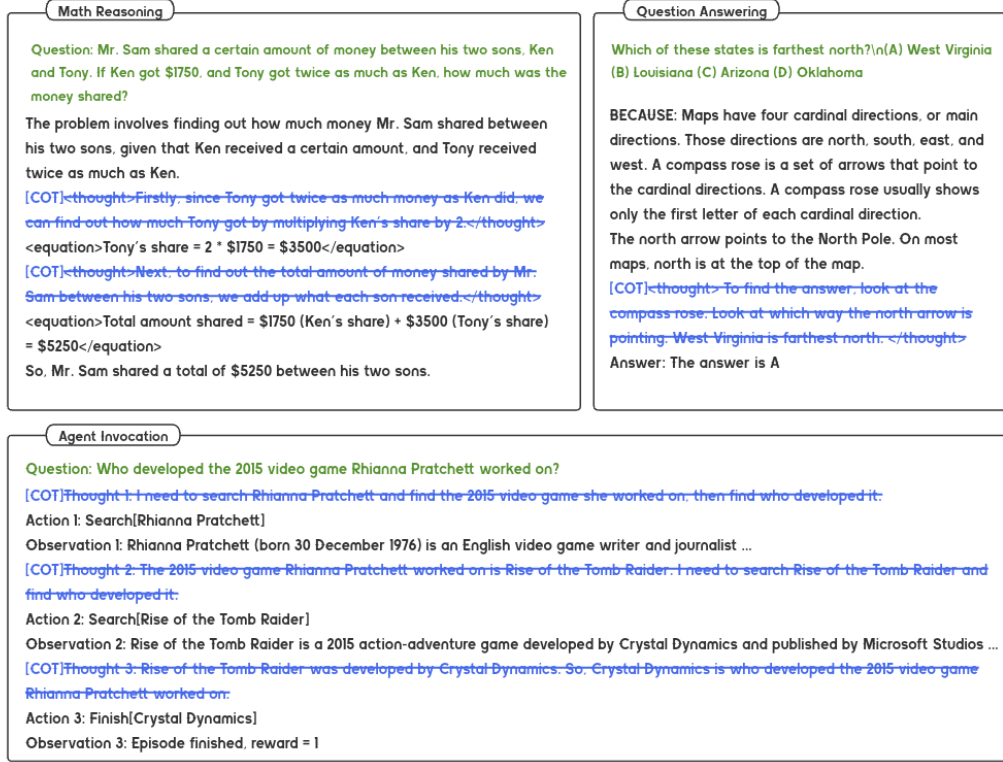


Figure 1: Real-world examples of the CoT prompting in tasks such as mathematical reasoning, question answering, and agent invocation. In the figure, green parts represent actual user queries, blue strikethroughs indicate compressed thought processes, and black text denotes non-CoT content, which corresponds to the expected output for users.

As illustrated in Figure 1, the blue tokens represent intermediate reasoning steps. These steps are crucial for ensuring final decoding accuracy but contribute to significant computational overhead when using standard CoT prompting. Generating complete CoT sequences typically requires substantially longer output sequences. Jin et al. (2024) indicates that extending the reasoning steps in prompts can enhance the reasoning abilities of LLMs, even without introducing new information. Nevertheless, within the transformer architecture (Vaswani et al., 2017), decoding time increases linearly with the output length, leading to higher computational costs and increased latency during inference. These issues, while critical, have not been well addressed in the existing literature.

To address this challenge, we draw upon the human cognitive process, where chains of thought are often implicitly and instantaneously formed within the mind. This introspection led us to hypothesize that during the decoding phase, a single token could represent the forthcoming cognitive process, effectively compressing the semantic content of an extensive reasoning chain into a specialized token. This approach aligns with recent findings in the domain of In-context Learning (ICL) for LLMs. Research by Wang et al. (2023a) has demonstrated the feasibility of employing 'anchor tokens' as potent conduits for aggregating and transmitting complex information.

Leveraging this foundation, we propose a novel two-stage fine-tuning framework aimed at generating subsequent outputs, such as precise answers or computational formulas, by utilizing a **compressed special token** representation in conjunction with the preceding context. The first stage of this framework involves the training of an auxiliary CoT model. This model employs a contrastive loss function to effectively condense an elaborate thought process into a specialized token, herein referred

to as [CoT]. Subsequently, we fine-tune our **Hidden CoT (HCoT) model** to generate the desired output based on the representation of the special token encoded by the CoT model and the preceding instructions, with the parameters of the CoT model remaining frozen. During inference, as shown in Figure 1, our HCoT model halts upon encountering the [CoT] token, at which point it feeds the preceding information to the auxiliary CoT compression model. The auxiliary model then generates a compressed CoT representation encapsulating the subsequent thought process. This compressed representation is then reinserted into the HCoT model to complete the inference. Concurrently, the auxiliary CoT compression model can either continue to generate the full thought process or opt not to, in order to conserve computational resources. Building on the inherent parallelizability of the LLM encoding process, the encoding phase that yields the special token representation is markedly more time-efficient when compared to the time-consuming process of decoding a complete chain of thought. Consequently, this optimization significantly accelerates the rate of inference.

Our extensive experiments show the potential of HCoT method on four datasets in three challenging domains: mathematical reasoning (Hendrycks et al., 2021, Cobbe et al., 2021), agent invocation (Yao et al., 2023), and science question answering (Lu et al., 2022). The results demonstrate that our HCoT model achieves competitive or improved performance compared to the full CoT baseline, while providing significant speedups of over 1.5x to 3.8x in decoding time.

To summarize, our major contributions are:

- We propose the Hidden Chain-of-Thought (HCoT) framework, a novel approach that accelerates the inference process of large language models by compressing the multi-step reasoning process into a specialized token representation, thereby reducing computational overhead during decoding.
- We introduce a disentangled training paradigm for the multi-step CoT reasoning, enabling isolated error correction and specialized optimization for each component.
- Our compression model effectively condenses the entire thought process into a compact special token while maintaining interpretability, allowing for parallel generation of CoT content.
- By incorporating a contrastive learning objective, we further enhance the quality of the compressed CoT model. This approach improves CoT prompting and task accuracy through the application of a span-level loss function during supervised fine-tuning.

We believe our work paves the way for more efficient exploitation of multi-step reasoning capabilities in LLMs across a wide range of applications.

## 2 Background

We first formalize some existing methods in this section. Our approach is not only inspired by these prior techniques but is also benchmarked against them for comparative analysis. We denote a pre-trained LLM with parameters  $\theta$ . We use lowercase letters such as  $x$ ,  $c$ , and  $z$  to represent the user’s question, the generated content, and the CoT reasoning process, respectively. For instance, a user question is denoted as  $x = (x[1], \dots, x[n])$ , where each  $x[i]$  is an individual token, and the probability of the sequence under our model is given by  $p_\theta(x) = \prod_{i=1}^n p_\theta(x[i] | x[1] \dots x[i-1])$ . To accommodate the complexity of reasoning that involves multiple interleaved sequences of content and thought, we extend our notation. For the  $i$ -th element in the reasoning process, we use subscripts:  $z_i$  represents the  $i$ -th chain of thought, and  $c_i$  represents the  $i$ -th output content sequence. We use uppercase letters  $C$  and  $Z$  to denote a collection of output contents and thoughts respectively.

**Chain-of-Thought (CoT) Reasoning** introduces intermediate steps that guide the model towards generating a more structured and potentially more accurate response. In this framework, the output is divided into several components: intermediate steps  $z_i$  and content parts  $c_i$ . The process involves iteratively generating these components based on previous steps: where model sample  $i$ -th thought  $z_i \sim p_\theta(z_i | x, c_0, z_0 \dots c_i)$  and then sample following content  $c_{i+1} \sim p_\theta(c_{i+1} | x, c_0, z_0 \dots c_i, z_i)$  given the sampled  $i$ -th thought.

**Reasoning w/o Chain-of-Thought (CoT)** contrasts with the CoT methodology by directly generating the final answer without explicitly modeling the intermediate reasoning steps. In this approach, the model aims to produce the output content  $C$  directly from the user’s question  $x$ :  $C \sim p_\theta(C | x)$ . Here, the reasoning process is implicit within the model’s parameters  $\theta$  and is not visible.

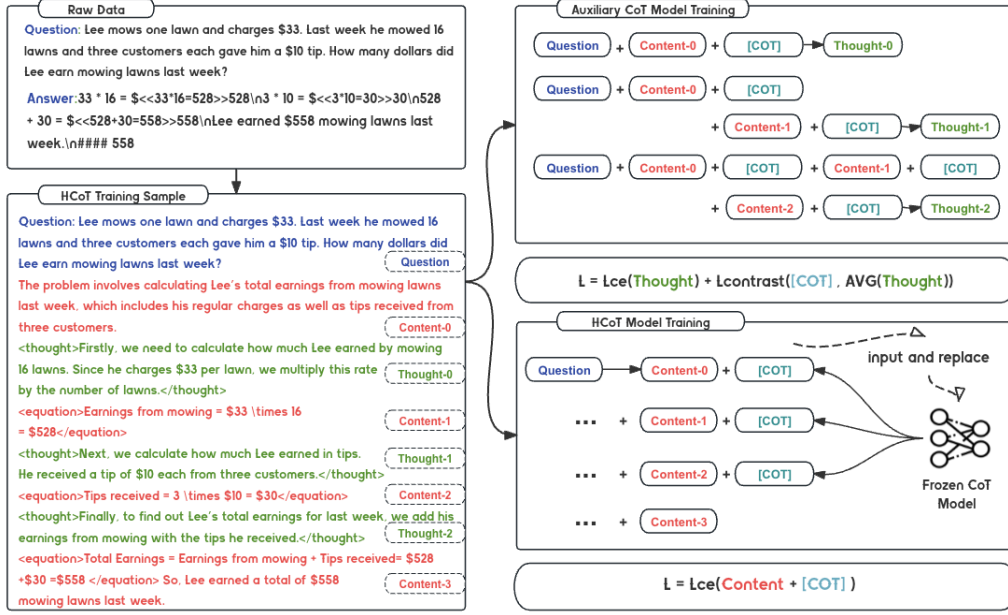


Figure 2: Data construction and two-stage training of Hidden Chain-of-Thought (HCoT) models for math reasoning tasks: Training instances are synthetically generated from raw data using GPT-4, then utilized separately for training the Auxiliary Chain-of-Thought Model and the HCoT Model.

### 3 HCoT: Hidden Chain-of-Thought Reasoning

In this section, we present our novel two-stage training method that incrementally develops the auxiliary CoT model followed by the Hidden CoT (HCoT) model. The auxiliary CoT model is ingeniously crafted to distill the reasoning process into a singular [CoT] token. Subsequently, the HCoT model leverages the encapsulated reasoning within the [CoT] token to facilitate swift and efficient chain-of-thought reasoning. As depicted in Figure 2, our training paradigm encompasses three components: (i) the generation of HCoT training instances from the original dataset to construct data that embodies CoT reasoning, (ii) the auxiliary CoT model training that employs these HCoT instances to create specialized training samples aimed at enhancing CoT reasoning, and (iii) the HCoT Model training phase that utilizes the same HCoT training instances, first replacing the intermediate reasoning steps  $z_i$  with the special [CoT] token, and upon encountering this special token, leveraging the encoded hidden representation from the frozen Auxiliary CoT model to replace the original input embedding for the special [CoT] token in the HCoT Model, thereby guiding the generation of the subsequent content  $c_{i+1}$ . In the following subsections, we delineate the methodology for constructing HCoT training samples in Section 3.1, elucidate the training dataset preparation and the training procedure for the auxiliary CoT model in Section 3.3, and detail the dataset construction and training process for the HCoT model in Section 3.4.

#### 3.1 HCoT Training Sample Construction

Constructing training samples for HCoT is a flexible process that can be tailored to the specific requirements of the task at hand and the anticipated format of the CoT. For instance, in our experiments on math reasoning tasks, as depicted in Figure 2, data from sources such as Math and GSM8K are utilized to construct training samples employing a GPT-4 based ICL method. (Specific prompts are in AppendixB.) This approach enables the model to output a series of contents and thoughts denoted as  $c_0, z_0, \dots, z_{n-1}, c_n$  which is sampled from:

$$p(C, Z | x) = \prod_{i=1}^n p_{\theta}(c_0 | x) p_{\theta}(z_{i-1} | x, \dots, c_{i-1}) p_{\theta}(c_i | x, c_0, \dots, z_{i-1}), \quad (1)$$

This recursive formulation encapsulates the dependencies between the **intermediate reasoning steps** ( $z_{i-1}$ ) and the content ( $c_i$ ) that is produced as a result. The flexibility of the HCoT sample construction process allows for the adaptation of the training regime to accommodate diverse reasoning patterns and content structures. By iteratively generating and conditioning on the chain of thought, the model is trained to better align its reasoning process with the underlying logic of the task.

### 3.2 Disentangled Training Paradigm

Delving deeper into Equation 1, we have disentangled the probability distribution into distinct components, we can identify the segment  $p_\theta(z_{i-1} \mid x, \dots, c_{i-1})$  as the generation phase of CoT information, namely **auxiliary CoT model**. Conversely, the segment  $p_\theta(c_i \mid x, c_0, \dots, z_{i-1})$  is responsible for harnessing the generated CoT to produce subsequent content, namely content generation model. This observation has led to the conceptualization of disentangled training paradigm which first compressing the high-dimensional discrete distribution of  $p_\theta(z_{i-1} \mid x, \dots, c_{i-1})$  into a more compact thought representation via encoding the input content with  $p_\theta^{COT}$ , which corresponding to the auxiliary CoT model training. Subsequently, the content generation model  $p_\theta^{HCoT}$  is fine-tuned to maximize  $p(c_i \mid x, c_0, \dots, z_{i-1})$ , where  $[z_0 \dots z_{i-1}]$  denote the compressed representations provided by the CoT model. This disentangled training paradigm offers several beneficial training dynamics:

**Error Isolation:** By decoupling the training of the auxiliary CoT model ( $p_\theta^{COT}$ ) from the content generation model ( $p_\theta^{HCoT}$ ), errors in reasoning can be isolated within the auxiliary CoT model. This facilitates targeted corrections without affecting the content generation model, thereby preventing the propagation of errors and enhancing the overall robustness of the system.

**Specialized Optimization:** The disentangled training paradigm allows for specialized optimization strategies. The CoT model can be honed to refine reasoning abilities and logical coherence, while the content generation model can concentrate on articulating clear and pertinent content. This specialization ensures that each model maximizes its performance in its respective domain, leading to a more effective training process.

**Parallel Development and Improved Interpretability:** The CoT generation operates in parallel with the generation of actual content. This not only accelerates the inference speed but also preserves the interpretability of the model. Unlike a black box approach, the reasoning steps generated by the CoT model are explicit and can be scrutinized, allowing for a better understanding of the model’s thought process and facilitating easier debugging and refinement.

To be noticed, We also include the part of  $p_\theta(c_0 \mid x)$  in our content generation model.

### 3.3 Auxiliary CoT Model

In this section, We use lowercase letter  $r$  to represent special [CoT] token’s representation, and  $r_i$  denotes the  $i$ -th special [CoT] representation. Given the user’s question and the preceding content, the objective of the auxiliary CoT model is to distill the reasoning process into a compact representation by maximizing  $p_\theta^{COT}(z_i \mid x, \dots, c_i, r_i)$ , where  $z_i$  is the desired thought process.

**Training Data Configuration:** As depicted in the top right corner of Figure 2, the training data for the auxiliary CoT model is constructed by first extracting all the thought processes from the original HCoT training samples. Subsequently, between each content segment  $c_i$  and each thought segment  $z_i$ , we insert a special token, denoted as [CoT], where this unique token serves as an anchor to facilitate the generation of the subsequent thought process. We then segment the entire training sample into individual instances, each treating a thought process  $z_i$  as the target output, conditioned on the preceding context comprising the question  $x$ , special [CoT] tokens and the content segments up to  $c_{i-1}$ .

**Thought Compression:** The auxiliary CoT model  $p_\theta^{COT}$  is trained by maximizing the likelihood  $p_\theta^{COT}(z_i \mid x, \dots, c_i, r_i)$ , where we expect the model to generate the most accurate thought representation based on the preceding information. In addition to the conventional cross-entropy loss, we incorporate symmetric contrastive loss between the **thought process representations mean-pooling** and the [CoT] token representation to enhance the thought compression capability of the model. The underlying assumption is that the compressed thought representation should exhibit a higher affinity with its corresponding special [CoT] token than with other [CoT] tokens, and vice versa. The final loss function for the auxiliary CoT model is as follows:

$$\begin{aligned}\mathcal{L}_{\text{CoT}} &= \mathcal{L}_{\text{CE}} + \lambda \cdot \mathcal{L}_{\text{contrastive}} = -\log p_{\theta}^{\text{CoT}}(z_i \mid x, \dots, c_i, r_i) \\ &\quad - \frac{\lambda}{2} \cdot \left( \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{r}_i)}{\sum_{k=0}^n \exp(\mathbf{z}_i \cdot \mathbf{r}_k)} + \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{r}_i)}{\sum_{j=0}^n \exp(\mathbf{z}_j \cdot \mathbf{r}_i)} \right)\end{aligned}\quad (2)$$

, where  $n$  denotes the batch size during the auxiliary CoT model’s training,  $\mathbf{z}_i \in \mathcal{R}^d$  represents the normalized representation of the  $i$ -th target thought process, obtained by mean-pooling the final hidden states from  $p_{\theta}^{\text{CoT}}$  when the input is the sequence  $[z_i[0], z_i[1], \dots, z_i[t]]$ . Additionally,  $\mathbf{r}_i \in \mathcal{R}^d$  denotes the normalized representation of the corresponding special [CoT] token generated by the auxiliary CoT model. The  $\mathcal{L}_{\text{contrastive}}$  is introduced to enhance the compactness of the thought process representation, ensuring that it exhibits a higher affinity towards the corresponding target thought process representation. Here,  $\lambda$  is a hyperparameter that governs the trade-off between the contrastive loss and the primary cross-entropy loss term.

### 3.4 HCoT Model

**Training Data Configuration:** The training data construction process for the HCoT model is illustrated in the bottom right corner of Figure 2. We replace all the thought processes  $z_i$  in the original HCoT training samples with the special [CoT] token. This transformation aligns the training paradigm seamlessly with the auxiliary CoT model’s training, as they share the same input format. By substituting the explicit thought processes with the compact [CoT] token, we effectively leverage the distilled reasoning encapsulated within the auxiliary CoT model’s output representation. This approach ensures that the HCoT model’s training is closely tied to the learned thought representations from the auxiliary CoT model, facilitating the transfer of reasoning capabilities.

**Supervised Fine-tuning of HCoT Model:** Given the user’s question  $x$ , the objective of the HCoT model is to maximize  $\prod_{i=1}^n p_{\theta}^{\text{HCoT}}(c_0 \mid x) p_{\theta}^{\text{HCoT}}(c_i \mid x, c_0, \dots, \mathbf{z}_{i-1})$ , where  $\mathbf{z}_{i-1}$  is the compressed thought representation obtained from the auxiliary CoT model. In this stage, we freeze the parameters of the auxiliary CoT model and fine-tune the HCoT model to effectively leverage the reasoning encapsulated within the compressed representations. Notably, the training target sequence comprises both content segments and special [CoT] tokens, requiring the model to learn not only the generation of content but also the appropriate insertion of the compressed thought representations denoted by the [CoT] tokens. We employ the standard cross-entropy loss function to supervise the fine-tuning process of the HCoT model.

## 4 Experiment

### 4.1 Datasets

We conducted experiments on three downstream tasks including four seed datasets: GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), ScienceQA (Lu et al., 2022), and HotpotQA (Yang et al., 2018). The ScienceQA dataset was categorized into three subjects: ScienceQA (natural science), ScienceQA (social science), and ScienceQA (language science). We prepared the training, validation, and test data for the auxiliary CoT model and the HCoT model based on the seed datasets, as detailed in Table 3. It is worth noting that there is no dedicated test data for the auxiliary CoT model, as we evaluate the end-to-end performance of the HCoT model. During training, we select the best-performing auxiliary CoT model by identifying the model with the lowest perplexity score on the auxiliary CoT model validation set. The train, validation, and test datasets remain consistent across other baselines for fair comparison. Notably, the auxiliary CoT training and validation data were constructed from the HCoT training and validation portions of the data, preventing the usage of more data than other baseline methods.

For the math reasoning datasets, GSM8K and MATH, we generated separate fields for thoughts and contents based on the input questions using the method described in Section 3.1, implemented by GPT-4. For the question answering task, we directly utilized the original fields from the ScienceQA dataset<sup>1</sup>. Notably, for the agent invocation task, we employed fields generated through the ReAct<sup>2</sup>

<sup>1</sup><https://scienceqa.github.io/>

<sup>2</sup><https://react-lm.github.io/>

framework for the HotpotQA dataset, treating the final answer path as the target. The final HCoT training samples can be referred to in Figure 1. It is important to note that due to some data instances containing multiple CoT tokens, there is a discrepancy in the number of training data entries between the auxiliary CoT model and the HCoT model. Moreover, for the ScienceQA dataset, we focused on the subset of questions that did not involve image understanding, and for the HotpotQA dataset, we concentrated on the training samples that achieved the correct final answer with ReAct, resulting in differences from the original dataset sizes.

## 4.2 Experimental Setup

To assess the efficacy of our method, we selected two base models for comparison: LLaMa2-7B and LLaMa2-13B (Touvron et al., 2023). We trained separate models for each domain, and we selected the best-performing checkpoint based on its performance on the corresponding development set and evaluated the final results using the test portion described in Section 4.1. Specifically, the math domain HCoT model was trained by combining the training data from both the GSM8K and MATH datasets. For all tasks, we employed accuracy as the primary evaluation metric. In the math reasoning domain, accuracy represents the proportion of questions answered correctly. For the ScienceQA dataset, accuracy corresponds to the correct selection among the multiple-choice options (A, B, C, D). For the agent invocation task on the HotpotQA dataset, we first identified the samples with the correct final answer and considered all actions along the path to be correct explorations. Subsequently, we calculated the ratio of correctly invoked actions as the agent invocation accuracy. More implementation details are provided in the Appendix D.5.

## 4.3 Baselines

To comprehensively study our method, we conducted experiments under five different settings:

- Zero/Few-shot CoT: Applied zero-shot CoT prompting on MATH and GSM8K datasets, and few-shot prompting on others, as a reference for models’ inherent capabilities without task-specific training.
- Train without COT: Removed thought processes and trained on remaining content in Figure 2, establishing a baseline without CoT reasoning.
- Train with COT: One-stage training on data without removing thoughts, serving as a baseline with explicit CoT reasoning.
- Train with HCoT base: Two-stage training (Figure 2) with cross-entropy loss for the auxiliary CoT model.
- Train with HCoT Contrast: Complete two-stage training with contrastive loss for the auxiliary CoT model, representing the final proposed method.

## 4.4 Results

The experimental results presented in Table 1 provide valuable insights into the effectiveness of our proposed HCoT approach across various tasks and datasets, in comparison with baseline methods. Overall, we observe performance improvements with the HCoT approach under most experimental settings. The HCoT models achieved the top performances in most cases, except for the social science question answering task in the Science QA dataset. In general, training with the Chain of Thought (CoT) technique outperformed the baseline without CoT, and further improvements were observed when training with the proposed HCoT approach. Notably, in the agent invocation task evaluated on the HotpotQA dataset, the HCoT-Contrast method exhibited significant improvements, with accuracy gains of 1.21% and 1.96% for the LLaMa2-7B and LLaMa2-13B models, respectively, compared to the CoT training baseline. Furthermore, for the question answering task represented by the ScienceQA dataset, the HCoT-Contrast approach demonstrated superiority in the natural science and language science subsets. In the natural science subset, the performance gains were 3.25% and 0.18% for the LLaMa2-7B and LLaMa2-13B models, respectively, compared with the CoT training. Similarly, in the language science subset, the improvements were 1.72% and 0.55%. In the math reasoning task, the training with HCoT achieved the best performance under both the 7B and 13B settings. These results highlight the efficacy of our proposed method in enhancing reasoning capabilities across various tasks and datasets.

Table 1: Performance comparison of LLaMa2-7B and LLaMa2-13B models under various training scenarios for different tasks and datasets. The ScienceQA dataset is further divided into three subsets: Natural Science (NS), Social Science (SS), and Language Science (LS), with separate results reported for each subset. The top performances in each category and model size are highlighted in bold.

	Models	Math		Science QA			Agent Invoke
		GSM8K	MATH	NS	SS	LS	HotpotQA
Zero/Few	LLaMa2-7B	14.60	2.50	56.80	51.64	67.06	47.11
	LLaMa2-13B	28.7	3.90	58.98	48.42	67.09	51.74
Train	LLaMa2-7B	34.27	6.80	82.10	62.32	87.36	79.78
	LLaMa2-13B	42.15	9.44	84.72	66.14	88.18	82.36
w/o CoT	LLaMa2-7B	36.85	6.74	80.99	<b>65.8</b>	86.64	83.73
	LLaMa2-13B	43.97	10.16	84.46	<b>69.74</b>	89.36	83.89
Train	LLaMa2-7B	<b>37.15</b>	7.49	83.13	63.89	<b>88.45</b>	83.5
	LLaMa2-13B	43.82	10.68	84.41	68.84	89.27	85.81
HCoT	LLaMa2-7B	36.47	<b>8.24</b>	<b>84.24</b>	62.77	88.36	<b>84.94</b>
	LLaMa2-13B	<b>44.43</b>	<b>11.16</b>	<b>84.64</b>	68.5	<b>89.91</b>	<b>85.85</b>
$\delta$ w HCoT	LLaMa2-7B	-0.38	1.5	3.25	-3.03	1.72	1.21
	LLaMa2-13B	0.46	1.00	0.18	-1.24	0.55	1.96
$\delta$ wo Contrast	LLaMa2-7B	0.68	-0.75	-1.11	1.12	0.09	-1.44
	LLaMa2-13B	-0.61	-0.48	-0.23	0.34	-0.64	-0.04

**Effect of HCoT Framework:** The results presented in the row " $\delta$  w HCoT" highlight the impact of our proposed HCoT framework in comparison to the full CoT training approach. Across most tasks and datasets, we observe performance gains when employing the HCoT framework, as indicated by positive values in this row. For the LLaMa2-7B model, the HCoT framework led to substantial improvements in the natural science (3.25%) and language science (1.72%) subsets of the ScienceQA dataset, as well as in the agent invocation task on the HotpotQA dataset (1.21%). However, slight performance decreases were observed in the GSM8K (-0.38%) and social science (-3.03%) tasks. Similarly, for the larger LLaMa2-13B model, the HCoT framework demonstrated its effectiveness, yielding notable performance gains in the agent invocation task on the HotpotQA dataset (1.96%), as well as improvements in the GSM8K (0.46%), MATH (1.00%), and language science (0.55%) tasks. Modest decreases were observed in the social science subset (-1.24%).

**Effect of Contrastive Learning:** The results presented in the row " $\delta$  wo Contrast" highlight the impact of incorporating contrastive learning into our HCoT framework. Negative values in this row indicate performance decreases when the contrastive loss objective is not employed, suggesting the effectiveness of contrastive learning in enhancing the model's reasoning capabilities. The predominance of negative values in the " $\delta$  wo Contrast" row across various tasks and datasets highlights the significant role of contrastive learning in our HCoT framework. By incorporating contrastive loss, the model's ability to effectively capture and leverage the core reasoning steps is enhanced, leading to improved performance in most cases compared to the HCoT approach without contrastive learning.

#### 4.5 Discussion

**The Speedup of HCoT Model:** Table 2 presents the compression and speedup rates of the HCoT model during the inference stage across four datasets compared to the explicit CoT model, both of which are based on LLaMa2-7B. The results for the LLaMa2-13B model are included in the Appendix due to space limit. Firstly, let's clarify the metrics used in the table. S-CR (Sequence-Level Compression Rate) refers to the average number of completion tokens of the HCoT model compared to the CoT model. S-S (Sequence-Level Speedup) is the reciprocal of S-CR, representing how many times faster the HCoT model is compared to the Full CoT model. W-CR (Wall-clock Time Compression Rate) provides a realistic measure of user-perceived speed-up, by comparing the actual inference time of the HCoT model to the Full CoT model. W-S (Wall-clock Time Speedup) is the reciprocal of W-CR. The table shows that S-CR values range from 23.78% to 66.91%, indicating that the sequence length of the HCoT model's output is significantly shorter than that of the Full CoT model. Correspondingly, the W-CR values range from 35.82% to 71.04%, and W-S values



range from 1.41x to 2.79x, showcasing a substantial acceleration. It’s important to note that the sequence-length-based acceleration rate (S-S) is generally higher than the real-time acceleration rate (W-S). This discrepancy arises because the sequence-length-based measure does not account for the encoding time of the Auxiliary CoT Model in HCoT. These times were tested on an H800 cluster using a single 80GB GPU. Despite achieving a speedup range of 1.41x to 2.79x, the HCoT model also delivers superior performance compared to the Full CoT model, demonstrating its efficiency and effectiveness. More fine-grained analysis of the length distribution before and after compression are detailed in Appendix.

**The recovery of CoT process:** Although in general scenarios, people prefer concise yet accurate responses that have undergone CoT reasoning, it is reasonable to compress the CoT process simply at this time. However, in certain situations, people wish to see the complete CoT process, thus maintaining the option to retain full output of CoT is important. Our model employs a method similar to "hiding", where the complete CoT process is concealed during the main reasoning process of the HCoT model, but the Auxiliary CoT model can still produce it normally when required. When asked to display the complete CoT process, we can simply keep the other settings unchanged and request the CoT model to continue outputting as needed. The case study in Appendix D shows this process.

Table 2: Compression and Speedup Rates of the HCoT model during inference across four datasets compared to Full CoT Model.

LLaMa2-7B				
Task	GSM8K	MATH	ScienceQA	HotpotQA
S-CR	60.45%	55.72%	66.91%	23.78%
S-S	1.65x	1.79x	1.49x	4.21x
W-CR	62.48%	62.49%	71.04%	35.82%
W-S	1.60x	1.60x	1.41x	2.79x

## 5 Related Work

**Chain-of-thought prompting (CoT)** enhances the emergent reasoning abilities of LLMs by prompting them to use explicit reasoning steps. Zero-shot-CoT (Kojima et al., 2023) demonstrates notable improvements in diverse reasoning tasks by merely prefacing solutions with the phrase "Let’s think step by step.". The least-to-most prompting (Zhou et al., 2023) approach effectively addresses complex problems by decomposing them into manageable subproblems and resolving them sequentially. Wang et al. (2023b) considers the self-consistency of CoT, enhancing its performance through majority voting. Furthermore, Gou et al. (2023) and Yuan et al. (2023) employ CoT on GPT-4, stabilizing the CoT capabilities of open-source models through fine-tuning on sampled data. Recent studies like ? and ? highlight the impact of reasoning step length on performance, suggesting that extending reasoning steps can improve outcomes. Our method achieves a similar effect by effectively extending CoT reasoning length, but with the added benefit of saving time during the decoding phase.

**Efficient model inference** often utilizes model compression techniques (Han et al., 2016) such as pruning or quantization. LLM-Pruner (Ma et al., 2023) implements structural pruning, which selectively removes non-critical coupled structures based on gradient information. LLM-QAT (Liu et al., 2023) leverages generations produced by the pre-trained model, enabling quantization of any generative model independently of its training data, akin to post-training quantization methods. Additionally, Gloeckle et al. (2024) considers multi-token prediction as an auxiliary training task, asking the model to predict the following  $n$  tokens using  $n$  independent output heads at each position in the training corpus, which not only accelerates inference but also enhances performance. ? proposes compressing prompts with gist tokens, focusing on efficient encoding. Deng et al. (2023) is closely related to ours, employing a method where the model is trained to predict hidden states for implicit CoT reasoning, aiming to reason more effectively. However, this approach exhibits a significant performance decline compared to explicit CoT reasoning, lacks interpretability, and involves a more complicated training process. In contrast, our method streamlines training, enhances reasoning path optimization, and sustains robust performance across tasks using models with over 7B parameters, offering a more interpretable, practical, and efficient solution.

## 6 Conclusion

We proposed HCoT, an innovative framework designed to accelerate the inference process of large language models while preserving their multi-step reasoning capabilities. At its core, HCoT employs a disentangled training paradigm that decouples the reasoning process into two specialized components: an auxiliary CoT model and a content generation model. The auxiliary CoT model is trained to compress the entire thought process into a compact, specialized token representation through a contrastive loss objective. This compressed representation effectively encapsulates the core reasoning steps, enabling efficient parallel computation during inference. The HCoT model, in turn, is fine-tuned to leverage this compressed reasoning representation, seamlessly integrating it into the content generation process. Our extensive experiments across diverse domains demonstrate the efficacy of the proposed HCoT framework. The results highlight its ability to achieve competitive or improved performance compared to the full CoT baseline while providing significant speedups of at least 1.5x in decoding time. However, it is important to acknowledge that this increase in efficiency comes at the cost of a more complex training phase and the necessity for additional model parameters, which may present scalability and resource challenges. In the future, we hope to address these limitations by optimizing the training phase and enhancing the model’s scalability, potentially reducing the need for additional parameters and lessening the training resource burden.

## References

- Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks.
- 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.
- Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit Chain of Thought Reasoning via Knowledge Distillation. ArXiv:2311.01460 [cs].
- Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models.
- Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & Faster Large Language Models via Multi-token Prediction. ArXiv:2404.19737 [cs].
- Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving.
- Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ArXiv:1510.00149 [cs].
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset.
- Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. The Impact of Reasoning Step Length on Large Language Models. ArXiv:2401.04925 [cs].
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large Language Models are Zero-Shot Reasoners. ArXiv:2205.11916 [cs].
- Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. LLM-QAT: Data-Free Quantization Aware Training for Large Language Models. ArXiv:2305.17888 [cs].
- Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *The 36th Conference on Neural Information Processing Systems (NeurIPS)*.