CODI: Compressing Chain-of-Thought into Continuous Space via Self-Distillation

Anonymous ACL submission

Abstract

Chain-of-Thought (CoT) enhances Large Language Models (LLMs) by enabling step-by-step reasoning in natural language. However, the language space may be suboptimal for reasoning. While implicit CoT methods attempt to enable reasoning without explicit CoT tokens, they have consistently lagged behind explicit CoT method in task performance. We propose CODI (Continuous Chain-of-Thought via Self-Distillation), a novel framework that distills CoT into a continuous space, where a shared model acts as both teacher and student, jointly learning explicit and implicit CoT while aligning their hidden activation on the token generating the final answer. CODI is the first implicit CoT method to match explicit CoT's performance on GSM8k while achieving 3.1× compression, surpassing the previous state-of-the-art by 28.2% in accuracy. Furthermore, CODI demonstrates scalability, robustness, and generalizability to more complex CoT datasets. Additionally, CODI retains interpretability by decoding its continuous thoughts, making its reasoning process transparent. Our findings establish implicit CoT as not only a more efficient but a powerful alternative to explicit CoT.

1 Introduction

Large Language Models (LLMs) have exhibited remarkable reasoning capabilities (OpenAI, 2024; Anthropic, 2024; Google, 2024), with Chain-of-Thought (CoT) (Wei et al., 2022) emerging as a key technique for enabling step-by-step reasoning via natural language rationales.

However, neuroscientific studies (Amalric and Dehaene, 2016, 2019) show that human mathematical reasoning does not primarily involve language processing areas in brains, suggesting that natural language may not be the most effective medium for reasoning. Furthermore, (Lin et al., 2025) highlights that LLMs often depend heavily on specific tokens during reasoning, which can lead to errors despite the tokens being commonsensically valid. Replacing these tokens with alternatives has been shown to improve performance, underscoring LLMs' sensitivity to linguistic features in natural language rationales. These insights motivate a shift from natural language CoT representations (Explicit CoT and Discrete Tokens) to dense, continuous representations (Implicit CoT and Continuous Thoughts) that may better align with the compact and abstract nature of reasoning.

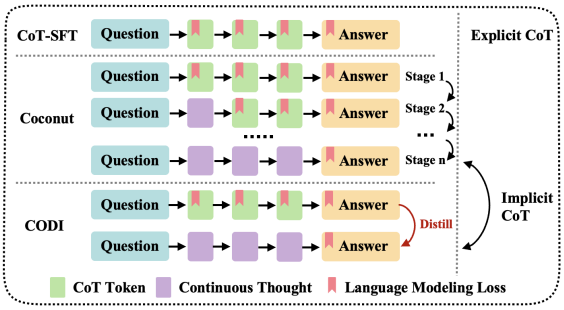

Figure 1: Illustration of the training process of different reasoning approaches. *CoT-SFT* is standard CoT finetuning. *Coconut* learns implicit CoT by curriculum learning. *CODI* learns implicit CoT by self-distillation.

Implicit CoT requires two major design considerations: the forward function and the training objective. Various forward functions have been explored, including removing all reasoning tokens (Deng et al., 2023, 2024), adding fixed learning tokens (Pfau et al., 2024; Goyal et al., 2024), performing autoregression by connecting the last hidden activation to the next input embedding (Hao et al., 2024). The autoregression approach appears to perform the best, likely because it increases the effective computational depth. However, the key design challenge for implicit CoT lies in the training objective. While explicit CoT learns reasoning through language modeling over annotated CoT tokens, implicit CoT cannot rely on this standard language modeling approach, as it must avoid generating explicit CoT tokens by definition.

To address this challenge, Coconut (Hao et al.,

2024), the state-of-the-art method, adopts a curriculum learning strategy initially introduced by (Deng et al., 2024) as illustrated in Figure 1. It gradually replaces the initial CoT tokens with continuous thoughts while maintaining the language modeling objective on the remaining CoT tokens. In this way, the language modeling loss encourages the continuous thoughts to behave like the removed CoT tokens, and at the final stage of learning, all CoT tokens are replaced with continuous thoughts, achieving full implicit CoT. However, while Coconut outperforms the No-CoT baseline (which entirely omits CoTs), it still lags behind CoT-SFT by 20% on GSM8k, exposing a key limitation in the implicit CoT paradigm. We believe these limitations stem from the curriculum learning strategy itself—the multi-stage process delays the acquisition of a complete discrete CoT. If the model fails to generate the correct continuous thought at any stage due to forgetting or incomplete learning, errors propagate through subsequent stages, limiting overall performance.

We propose **CODI (Continuous Chain-of-Thought via Self-Distillation)**, a novel framework that distills explicit CoT into implicit CoT by aligning the hidden activation of the token responsible for generating the final answer as illustrated in Figure 1. CODI reframes implicit CoT learning as a self-distillation task (Wang et al., 2023; Gou et al., 2021), where the same model serves as both teacher and student. Unlike conventional self-distillation having the two roles of equal capability, CODI enhances the teacher's knowledge by providing them distinct input contexts: the teacher learns from the groundtruth CoT and final answer using a language modeling objective, while the student generates continuous thoughts before predicting the final answer–our target task. Distillation occurs at the token preceding the final answer, which Orgad et al. (2025) identify as encoding crucial reasoning information. Since we can formally show that CoT influences the hidden activation of this token only by a shift (Section 3.3), CODI enforces alignment between the teacher and student by minimizing their hidden activation differences using an L1 distance loss, effectively injecting explicit CoT supervision into implicit CoT generation. This single-step distillation in feature space mitigates the forgetting issues inherent in curriculum learning, enabling more effective implicit CoT training.

The main contributions are threefold:

- We propose CODI, a novel self-distillation framework that enables LLMs to reason in a compact continuous space, providing an alternative to accelerate reasoning with high performance.
- We demonstrate the effectiveness of distilling knowledge from explicit CoT (teacher) to implicit CoT (student) by aligning the hidden activation of a single token, simplifying the distillation process and improving efficiency.
- Extensive experiments show that CODI is robust, scalable, and generalizable to more complex CoT datasets. Additionally, CODI maintains interpretability, making its reasoning process transparent.

2 Related Work

Implicit Chain-of-Thought Reasoning. Implicit CoT methods aim to enhance reasoning without verbalizing intermediate steps as in CoT, thereby accelerating inference speed. Theoretical work (Strobl et al., 2024; Merrill and Sabharwal, 2024) establishes that additional computational tokens enhance transformers' reasoning capacity. Empirical studies (Pfau et al., 2024; Goyal et al., 2024) validate these insights by training LLMs with extra dummy tokens before answering though in a limited scale and effect. Recent efforts (Deng et al., 2023, 2024) distills CoT reasoning by finetuning. They improve over the No-CoT baseline, but fall behind CoT finetuning possibly due to discarding all intermediate tokens. Addressing this, Coconut (Hao et al., 2024) reintroduces intermediate reasoning tokens via autoregressive hidden state propagation, combining curriculum learning from (Deng et al., 2024). While this achieves some improvement over (Deng et al., 2024), Coconut still lags behind explicit CoT, which we attribute to forgetting in curriculum learning. CODI replaces curriculum learning with a novel self-distillation framework, enabling a single-step learning process that avoids forgetting issues. Our work is also inspired by in-context compression (Ge et al., 2024; Li et al., 2024b), though our work is compressing the generation instead of the existing contexts.

Knowledge Distillation. Knowledge distillation (KD) (Gou et al., 2021; Xu et al., 2024) has emerged as a key strategy for transferring CoT reasoning capabilities from teacher to student models. Traditional approaches (Hsieh et al., 2023; Ho et al., 2023) train smaller student models to mimic step-by-step outputs from larger teacher LLMs, motivated by findings that CoT reasoning emerges predominantly in large models (Wei et al., 2022). Self-

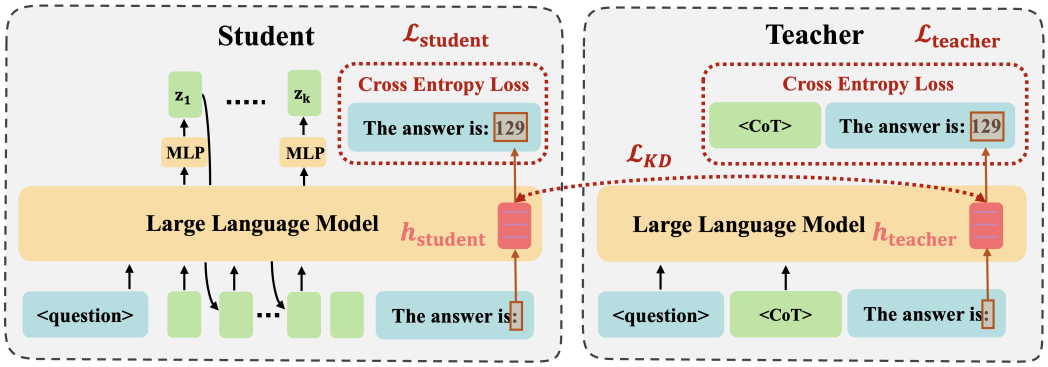

Figure 2: **CODI** enables the model to generate continuous CoTs by jointly training a student and teacher task within a shared LLM, distilling knowledge from the teacher to the student. The **Student** task (left) generates the answer by autoregressively decoding continuous thoughts, while the **Teacher** task (right) generates the answer using the groundtruth CoT via teacher forcing. Both tasks learn the generated texts via cross-entropy loss ($\mathcal{L}_{\text{student}}$ and $\mathcal{L}_{\text{teacher}}$), and share the same LLM. Knowledge distillation is achieved by applying $\mathcal{L}_{\text{KD}}$ (L1 loss) between student and teacher hidden activation across all layers ($\mathbf{h}_{\text{student}}$ and $\mathbf{h}_{\text{teacher}}$).

distillation (Yang et al., 2024; Dong et al., 2024) leverage self-distillation to preserve the model's original behavior, akin to the KL divergence loss used in RLHF (Ouyang et al., 2022). Our work is based on self-distillation framework, but further strengthens the teacher by providing it with richer input contexts, enabling the student to learn from it like knowledge distillation. Since the teacher and student tasks differ, CODI can also be viewed as a form of multitask learning (Crawshaw, 2020). Moreover, CODI distinguishes itself by allowing reason in the latent space other than natural language, which is rarely explored in prior knowledge distillation works. This innovation enables more flexible and efficient reasoning.

3 CODI: Continuous Chain-of-Thought via Self Distillation

Unlike traditional CoT reasoning, CODI bypasses autoregression in the vocabulary space, and directly connects the last hidden representation to the subsequent input. The key challenge in training such a model with continuous thoughts lies in designing an appropriate training objective. Conventional reasoning learning in explicit CoT fine-tuning relies on a language modeling objective over annotated CoT tokens, which inevitably leads to discrete CoT token generation—contradicting the definition of implicit CoT.

3.1 Overview

CODI addresses this by introducing a self-distillation framework (Figure 2) with two training tasks: a teacher task and a student task. The teacher task learns explicit CoT generation, while the student task learns implicit continuous CoT generation. Knowledge distillation is achieved by aligning the hidden activation of a key token from the teacher to the student via $\mathcal{L}_{KD}$. The overall training objective is a weighted sum of three losses, which will be detailed later:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{teacher}} + \beta\mathcal{L}_{\text{student}} + \gamma\mathcal{L}_{\text{KD}}, \quad (1)$$

where α, β, and γ are hyperparameters controlling the balance among the objectives..

A Python implementation of this framework is provided in Figure A3.

3.2 Student Task

The student task (Figure 2, left), the target task, generates continuous thoughts by autoregressively propagating the last hidden states and learns to generate the answer token using a cross-entropy loss:

$$\mathcal{L}_{\text{student}} = -\frac{1}{N}\sum_{i=1}^{N} \log P(y_i \mid y_{1:i-1}, Q, Z), \quad (2)$$

where P is the probability distribution of the LLM, y refers the answer label, Q refers the question tokens, and Z refers the continuous thoughts.

On its own, the model benefits only marginally from the additional computation (Goyal et al., 2024) compared with the No-CoT scenario because there are no supervision for the continuous thoughts.

Additionally, CODI applies modifications exclusively to the student task. Two special tokens, `bot` and `eot`, mark the start and end of continuous reasoning, inspired by (Hao et al., 2024). A two-layer

MLP followed by layer normalization transforms the hidden representations of continuous thought tokens before feeding them into the next step.

3.3 Teacher Task

Unlike the student task, the teacher task (Figure 2, right) performs explicit CoT generation using a language modeling objective:

$$\mathcal{L}_{\text{teacher}} = -\frac{1}{N}\sum_{i=1}^{N} \log P(y_i \mid y_{1:i-1}, Q), \quad (3)$$

where y refers both the CoT and the answer labels, and Q refers the question tokens.

The teacher task serves two key functions: (1) **Reference Learning**: By learning explicit CoTs, the teacher task equips the model with structured reasoning patterns, offering a foundational reference for the student task. (2) **Latent Supervision**: As the teacher has access to ground-truth CoT tokens, its hidden activation at the answer-generating token encapsulate essential reasoning information by attending to all preceding CoT tokens. In contrast, the student initially operates without such structured guidance. To address this disparity, CODI aligns the hidden activation of this token between the teacher and student across all layers using an L1 loss:

$$\mathcal{L}_{\text{KD}} = \frac{1}{M}\sum_{l=1}^{M} |\text{sg}[\mathbf{h}^l_{\text{teacher}}] - \mathbf{h}^l_{\text{student}}|, \quad (4)$$

where M indicates the number of layers in the LLM, sg denotes stop gradient, and $\mathbf{h}^l$ is the hidden activation of the LLM's l-th layer.

The Distilled Token. Rather than aligning with all tokens in the generated sentence, we select a *distillation token* for alignment. Inspired by the recent observations (Orgad et al., 2025) that the hidden activation of the token intermediately preceding the answer, i.e., the colon (":") in the answer prompt "*The answer is:*" (as shown in Figure 2), encodes far more information than output logits. We select this token's hidden activation, $\mathbf{h}$, for distillation. This selection can be further verified by the shift mechanism in in-context learning. Recent work (Li et al., 2024a; Liu et al., 2023) demonstrates that in-context examples influence the final query token by shifting its hidden activation values. Extending this idea, we show that CoT tokens similarly induce a shift in hidden activation values of this target token, as formalized in Equation 5:

$$\mathbf{h}^l_{\text{CoT}} \approx \mathbf{h}^l_{\text{no-CoT}} + f\Big(W_V R(W_K R)^T \mathbf{q}\Big), \quad (5)$$

where $\mathbf{q}$ is the query of this target token, $\mathbf{h}^l_{\text{CoT}}$ is the hidden activation at layer l with CoT (equivalent to $\mathbf{h}^l_{\text{teacher}}$), $\mathbf{h}^l_{\text{no-CoT}}$ is the corresponding activation without CoT, and the remaining term quantifies the shift introduced by the CoT rationale R.

This suggests that the target token's hidden activation encode the influence of preceding reasoning steps, and $\mathbf{h}^l_{\text{student}}$ can learn this shift by minimizing a simple distance metric, such as L1 loss, with $\mathbf{h}^l_{\text{teacher}}$. A formal proof of this "CoT shift" phenomenon is provided in Appendix B.

3.4 Training and Inference

Training. The continuous thoughts are generated dynamically during training, as they are not known beforehand. To achieve this, we decode them step by step, with a cache storing previous keys and values to maintain efficiency. When applying distance loss between two hidden activation, we observed a significant norm variations across layers (Deng et al., 2023; Cheng and Durme, 2024). To address this, we normalize each layer's activation by dividing them by the standard deviation of the teacher's hidden activation within the current batch.

For the distillation task, we employed the same model for the teacher task and the student task for two reasons: (1) **Warm-up**: When the teacher trains alongside the student, it creates a warm-up effect for $\mathcal{L}_{\text{KD}}$. Both components start from the same initialization point, diverge during training, and gradually converge as the student adapts. In contrast, a static pre-trained teacher initially presents an overly challenging objective, as its hidden states reflect fully developed reasoning patterns that the untrained student cannot immediately match. (2) **Shared model representations**: Using the same model mitigates alignment issues in hidden activation that arise when using separate models, enabling smoother and more effective information transfer between the teacher and student. The corresponding ablation studies, which validate these findings, are detailed in Table 3.

For training data, we exclude the final CoT step—the step responsible for generating the final answer—because including this step could allow the teacher's hidden activation to take a shortcut. Specifically, the model might directly copy the result from the last CoT step to the token responsible for generating the exact answer token, bypassing the reasoning process. This behavior would undermine the quality of the target hidden activation, as they would no longer fully encode the reasoning patterns. The ablation results demonstrating the

impact of this exclusion are presented in Table 3.

Inference. The inference process in CODI mirrors the student task during training (Figure 2, left). The model autoregressively decodes n continuous thoughts following the question and the `bot` token. Once the reasoning process is complete, the `eot` token is manually inserted to terminate continuous reasoning and switch the model to language generation mode, decoding the final answer.

4 Experiments

We demonstrate the effectiveness of CODI's reasoning in a continuous space through experiments on mathematical reasoning tasks.

4.1 Experimental Setup

Training Data. We utilize two datasets to train our models–GSM8k-Aug and GSM8k-Aug-NL. (1) We use the GSM8k-Aug dataset from (Deng et al., 2023), which has proven effective for training implicit CoT methods (Deng et al., 2024; Hao et al., 2024). This dataset extends the original GSM8k training set (Cobbe et al., 2021) to 385k samples by prompting GPT-4. To facilitate implicit CoT training, all natural language interleaving within the CoT is removed, leaving only structured mathematical expressions such as "$<< 10 \div 5 = 2 >><< 2 \times 2 = 4 >><< 6 \times 4 = 24 >>$". (2) We also use GSM8k-Aug-NL, a version that preserves natural language explanations, to assess both the generalizability and effectiveness of our approach to compress more verbose CoTs. Examples and statistics are in Appendix C.

Evaluation Benchmarks for OOD. In addition to the test split of GSM8k, we assess model robustness on three out-of-domain (OOD) benchmarks: (1) SVAMP (Patel et al., 2021), a dataset of elementary-school arithmetic word problems with simple variations designed for robustness test; (2) GSM-HARD (Gao et al., 2022), a modified version of the GSM8k test split where numbers are replaced with values of larger magnitude to increase difficulty; and (3) MultiArith (Roy and Roth, 2015), a subset of MAWPS (Koncel-Kedziorski et al., 2016) containing multi-step mathematical word problems. Examples and statistics are in Appendix C.

Baselines. We consider the following baselines: (1) **CoT-SFT:** Finetunes the model on CoT data, enabling it to generate intermediate steps followed by the final answer. As CoT-SFT relies on sampling, we set the temperature to 0.1 and report the average result over 10 runs. (2) **No-CoT-SFT:** Finetunes the model using only direct answers, without generating intermediate steps. (3) **iCoT** (Deng et al., 2024): Implements a curriculum learning strategy called "Stepwise Internalization", which injects CoT's reasoning patterns into the model's internal states. This allows the model to generate direct answers with higher accuracy during inference. (4) **Coconut** (Hao et al., 2024): Build upon iCoT by autoregressively generating intermediate continuous CoT representations, similar to the approach in our work. (5) **CODI**: our method trained with six continuous thought tokens, matching the setup in Coconut. Baselines (2)–(5) are deterministic models, and their results are reported from a single run. Two base models are considered GPT-2 (Radford et al., 2019) and LLaMA3.2-1b (Meta, 2024). More implementation details are in Appendix A.

4.2 Main Results

Mathematical Reasoning. Table 1 shows the evaluation results on GSM8k. CODI achieves a significant performance improvement over other implicit CoT methods. In the settings of GPT-2, CODI surpasses iCoT by 45.7% and Coconut by 28.2%. Notably, CODI is the first continuous CoT method to perform on par with CoT-SFT, achieving 99.1% of CoT-SFT's performance. Unlike iCoT and Cococnut failing to scale up to larger models (Hao et al., 2024), CODI successfully scales to LLaMA1b, achieving 90.3% of CoT-SFT's performance. These results highlight CODI's superiority in terms of accuracy for in-domain mathematical reasoning tasks.

Efficiency. CODI utilizes a fixed set of six continuous thoughts, enclosed by two special tokens, resulting in a total of eight "tokens" for reasoning. As shown in Table 2, CODI achieves substantial efficiency gains, with a speedup of approximately 2.7× (3.1× CoT compression) for compact CoTs trained on GSM8k-Aug and 5.9× (7.8× CoT compression) for verbose CoTs trained on GSM8k-Aug-NL, demonstrating CODI's effectiveness in reducing reasoning overhead.

Robustness. To assess robustness, we evaluate CODI on out-of-distribution datasets. Notably, CODI consistently outperforms CoT-SFT across all three benchmarks for GPT-2. We attribute this to CODI's reduced tendency to overfit, as evidenced by its significantly lower training accuracy compared to CoT-SFT (Table A3). This difference arises because CODI lacks exact imitation targets

Method	In-Domain		Out-of-Distribution		
	GSM8k	GSM8k-NL	SVAMP	GSM-Hard	MultiArith
GPT-2 Small					
CoT-SFT	44.1%	34.8%	41.8%	9.8%	90.7%
No-CoT-SFT	19.1%	19.1%	16.4%	4.3%	41.1%
iCoT	30.1%*	3.2%	29.4%	5.7%	55.5%
Coconut	34.1%*	–	–	–	–
CODI (Ours)	43.7%	35.3%	42.9%	9.9%	92.8%
LLaMA3.2-1b					
CoT-SFT	61.6%	54.1%	66.7%	15.6%	99.3%
No-CoT-SFT	30.9%	30.9%	44.1%	7.1%	70.9%
iCoT	19.0%	15.2%	40.9%	4.4%	39.0%
CODI (Ours)	55.6%	49.7%	61.1%	12.8%	96.1%

Table 1: Results on four datasets: GSM8k, SVAMP, GSM-Hard, and MultiArith. GSM8k-NL indicates that the training data is GSM8k-Aug-NL, the verbose dataset, instead of GSM8k-Aug. Results marked with * are taken from the corresponding papers (Deng et al., 2024; Hao et al., 2024). Coconut's results are incomplete due to the unavailability of open-source code.

for continuous thoughts during training, making it less prone to memorizing patterns and more adaptable to novel scenarios.

Compress CoTs with Natural Language. Previous works (Deng et al., 2024; Hao et al., 2024) primarily trained on GSM8k-Aug, which consists only of mathematical expressions. To evaluate CODI's generalizability, we extend our analysis to a more complex CoT dataset, GSM8k-Aug-NL. Table 1 shows that both GPT-2 and LLaMA1b perform worse on it compared to GSM8k-Aug. This decrease in performance stems from the additional natural language tokens, which add noise and make imitation learning more difficult. Surprisingly, CODI surpasses CoT-SFT when using GPT-2 and achieves a higher relative score improvement on LLaMA1b compared to models trained on GSM8k-Aug. Moreover, iCoT almost fails in this task because the longer sequence makes curriculum learning challenging. Furthermore, with the average CoT length increasing to 62.1 (Table 2), CODI achieves a compression ratio of 7.8, suggesting that the optimal compression ratio is dataset-dependent. These results demonstrates CODI's ability to handle more complex CoT training data, showcasing its applicability to diverse reasoning datasets.

Compression Ratio. The number of continuous thoughts used during training is a crucial hyperparameter, influencing both the computation allocation and the compression ratio. As shown in Figure 3, CODI consistently outperforms Coconut across all compression ratios. Interestingly, both methods exhibit a similar trend: accuracy peaks when using six continuous thoughts. We attribute this to the dataset's structure, specifically the average number of CoT steps. When fewer than six continuous thoughts are used, the model lacks sufficient expressiveness to capture reasoning steps effectively. Conversely, beyond six, the additional complexity may not provide further benefits, as most problems do not require additional reasoning steps. Instead, the increased sequence length introduces optimization challenges, outweighing any potential gains.

Method	GSM8k-Aug Time (#Tokens)	GSM8k-Aug-NL Time (#Tokens)
GPT-2		
CoT-SFT	0.17s (25.1)	0.36s (62.1)
No-CoT-SFT	0.035s (0)	0.035s (0)
CODI	0.062s (8)	0.062s (8)
LLaMA-1b		
CoT-SFT	0.73s (25.4)	1.62s (68.8)
No-CoT-SFT	0.16s (0)	0.16s (0)
CODI	0.27s (8)	0.27s (8)

Table 2: Efficiency comparison of different reasoning methods in terms of inference time per math problem on GSM8k. Measured with batch size = 1 on an Nvidia A100 GPU. CoT Token counts are shown in parentheses.

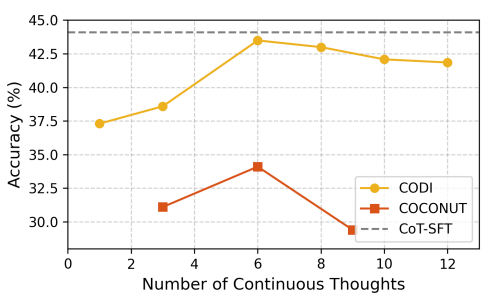

Figure 3: Accuracy on GSM8k against the number of continuous thought tokens used during training.

Methods (GPT-2)	**Accuracy**(%)
No-CoT	19.1%
CODI	43.7%
- ind. static teacher	27.1%
w/ multitask student	42.2%
- ind. trained teacher	–
w/ multitask student	42.7%
- w/o L1 loss	24.5%
- w/ CoT last step	31.7%
- w/o Projection	42.5%

Table 3: Ablation studies. *ind. static teacher* refers to training an independent teacher model beforehand. *w/ multitask student* extends it by allowing the student model to also learn CoT generation. *ind. trained teacher* refers to training an independent teacher model along with the student model.

4.3 Ablation Studies

Independent Teacher. To evaluate the need of self-distillation, we tested settings where the student does not share the model with the teacher (Table 3). Without learning explicit CoT generation (`ind. static teacher`), the model performs badly and fails to generate meaningful continuous CoTs after decoding. Adding an explicit CoT generation objective (`w/ multitask student`) significantly restores performance, indicating the importance of reference learning. Additionally, training the teacher alongside the student (`ind. trained teacher`) leads to better results than using a pre-trained, static teacher (`ind. static teacher`), supporting the argument of the warm-up effect. Finally, using a unified model (`CODI`) outperforms maintaining separate teacher-student models (`ind. trained teacher`), reinforcing the idea that shared model representations help mitigate alignment issues in hidden states.

Distillation Loss. Table 3 shows that removing the L1 loss (Equation 4) linking the teacher and student processes (`w/o L1 Loss`) leads to a significant performance drop, indicating the importance of supervision from the distillation token. While the model still performs well in CoT generation, it fails to integrate this skill into continuous CoT reasoning, treating them as independent tasks rather than a unified reasoning process.

Others. Keeping the final step of the CoT chain appears to negatively impact performance, supporting our claim that it provides shortcuts. Furthermore, the projection layer of continuous thought tokens (shown as the MLP layer before each of the continous thought token $z_1 \cdots z_k$ in Figure 2) enhances CODI's effectiveness, likely by helping to discriminate discrete and continuous CoT representations.

5 Further Analysis

5.1 Interpretability Analysis

Interpreting CODI's continuous thoughts is inherently challenging because these representations lack explicit imitation targets. However, CODI exhibits an ability to produce observable intermediate results (Figure 4) within its continuous thoughts by projecting its last hidden state into vocabulary space via the model's word embeddings – treating it in the same way as a standard text token. Additionally, the corresponding operands contributing to these intermediate results can often be found in the **attended tokens** of the latent representation. For example, the second thought token, z_2, attends to both "1" and "7" to produce the decoded token "7". While the operator itself (e.g., $\times$) is not explicitly visible in the attention mechanism—since operators are in the context—it is reasonable to infer that the transformer layers *implicitly* perform this operation. Another interesting observation is that each intermediate result is separated by a seemingly meaningless continuous token. We hypothesize that these tokens act as placeholders or transitional states during the computation of intermediate results. This aligns with the idea that the transformer may require multiple passes to complete the calculation for each intermediate step. More case studies are in the Appendix E.

Total Steps	1	2	3
Accuracy	97.1%	83.9%	75.0%

Table 4: CODI's top-5 intermediate results matching reference CoT across problems requiring different numbers of step.

Beyond the case study, we aim to establish that CODI's interpretability is a general pattern by an accuracy metric. We extract all correctly predicted answers, decode the corresponding intermediate results, and compare them against the reference intermediate solutions. Table 4 reveals that when there is only one intermediate result, CODI correctly matches the reference 97.1% of the time. For CoT sequences with lengths up to 3, CODI consistently achieves over 75% accuracy in decoding valid intermediate results. These findings highlight CODI's reliability in generating meaningful intermediate reasoning steps, demonstrating its potential to effectively handle reasoning tasks with interpretable intermediate outputs.

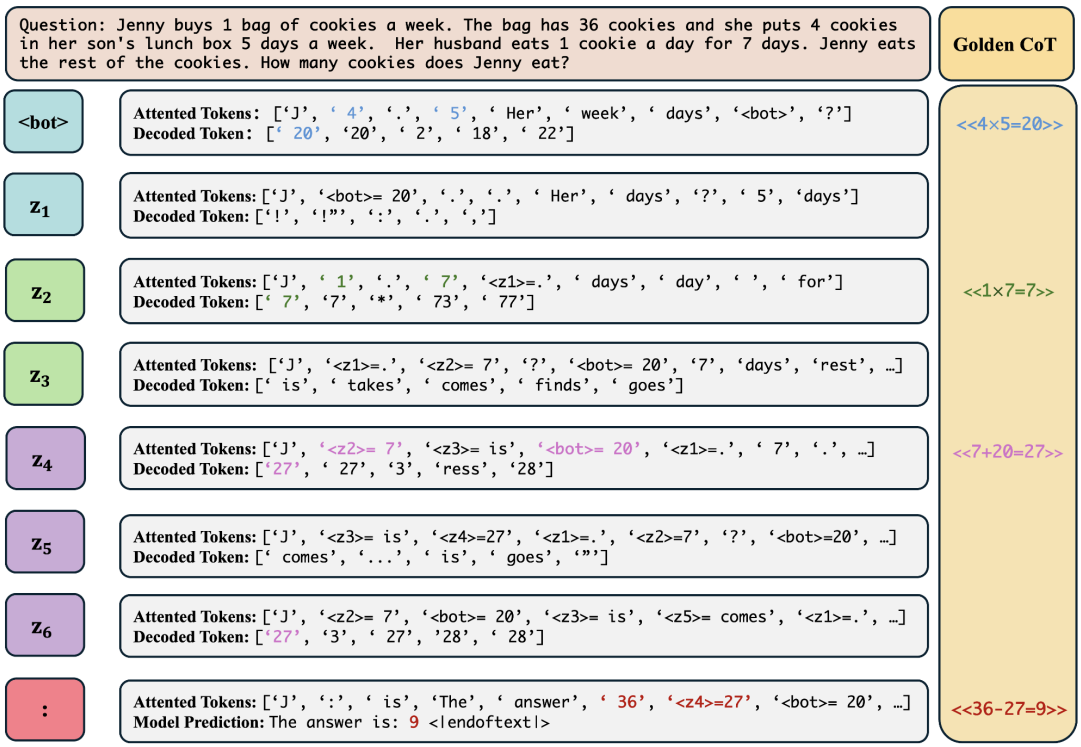

Figure 4: A case study illustrating CODI's interpretability by analyzing its attended tokens and decoded tokens of each of the six latent thought tokens, $z_1 \cdots z_6$. **Attended tokens**: these represent the top-10 tokens that the continuous thought attends to when generating the next thought/token. Some attended tokens appear in the form of '$z_i = x$', indicating attention to the i-th continuous thought. Here x represents the top-1 token that the latent thought maps to in vocabulary space. **Decoded tokens**: these are the top-5 words that the continuous thoughts are projected back to in vocabulary space by multiplying them with the vocabulary embeddings.

5.2 CODI's Pattern Learning

GPT-2	CODI	Coconut	Res	Op-Res
Accuracy	43.7%	34.1%	34.0%	35.7%

Table 5: Comparison of GPT-2 finetuned on two datasets derived from CODI's decoded thoughts. *Res*: using intermediate results as CoT. *Op-Res*: using intermediate operators and results as CoT.

Given that CODI's continuous thoughts can often be decoded into intermediate results, it raises a question: is CODI effectively equivalent to a GPT-2 fine-tuned on a dataset containing CODI's decoded patterns? We created a dataset containing only intermediate results (e.g., "`CoT: 20, 7, 27. Result: 9`" translated from the case study in Figure 4). Additionally, since some cases of CODI show decoded operators like '$\times$' and '$-$' interleaved with intermediate results, we also create a synthetic CoT dataset that includes both operators and results (e.g., "`CoT: ×, 20, ×, 7, +, 27. Result: 9`"). As shown in Table 5, while models trained on the two synthetic datasets outperform the No-CoT baseline, they give significantly inferior results compared to CoT and CODI, though perform on par with Coconut. These result suggest that CODI learns richer information from the teacher task through distillation than pure imitation on language-level intermediate results alone, highlighting the advantages of our training framework.

6 Conclusion

We introduced CODI, a novel paradigm for reasoning in the continuous space. Our extensive experiments demonstrate CODI's effectiveness as the first continuous CoT method to match the performance of explicit CoT, while achieving a high compression ratio. Furthermore, CODI shows its scalability, robustness, generalizability to more complex CoT data, while retaining interpretability. Future research should explore CODI's application to more diverse and challenging tasks. A promising direction is the integration of multimodality, leveraging continuous embeddings for seamless modality merging. We hope this work inspires further exploration into reasoning in representations more compact than language, paving the way for more efficient and versatile reasoning paradigms.

7 Limitations

In the paper, our approach focuses on knowledge transfer by probing the token (":") responsible for generating the first answer token. However, this choice may be suboptimal, as some answers begin with "-", and removing such cases improves performance, suggesting that critical reasoning information might also reside in the token generating the second answer token. Additionally, probing the token that concludes the CoT reasoning—potentially summarizing the entire process—could offer alternative supervision signals. Furthermore, the current answer prompt, "The answer is:", is an arbitrary design choice that may influence the effectiveness of knowledge transfer. Investigating these aspects further could enable CODI to extend its distillation framework to broader reasoning tasks.

Another limitation of the current continuous training approach is the absence of intermediate gradients until the end of the sequence. With six continuous thought tokens, the first token's gradient is backpropagated from six or more steps away (specifically, from the token generating the final answer), which may introduce optimization challenges. This issue could become more pronounced when scaling to more complex problems requiring longer continuous reasoning chains.

Furthermore, CODI's current configuration is fully deterministic, whereas one advantage of CoT is its inherent stochasticity, which allows models to improve performance through self-consistency, tree search, and Monte Carlo Tree Search (MCTS). Introducing controlled randomness into CODI could restore this property. Potential solutions include integrating dropout layers, leveraging Variational Autoencoders (VAEs), or applying diffusion models atop the current projection layer to enable diverse and robust reasoning paths.

Finally, our studies primarily focus on mathematical problems, as mathematical CoT training data are more abundant and mathematical problems inherently require complex reasoning. However, this focus may limit the generalizability of CODI to broader reasoning scenarios. Future work should explore its applicability to more diverse reasoning tasks beyond mathematical domains.

References

M. Amalric and S. Dehaene. 2016. Origins of the brain networks for advanced mathematics in expert mathematicians. *Proceedings of the National Academy of Sciences*, 113(18):4909–4917.

M. Amalric and S. Dehaene. 2019. A distinct cortical network for mathematical knowledge in the human brain. *NeuroImage*, 189:19–31. Epub 2019 Jan 3.

Anthropic. 2024. Claude 3.5 sonnet.

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

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*, abs/2110.14168.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *ArXiv*, abs/2009.09796.

Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024. From explicit cot to implicit cot: Learning to internalize cot step by step. *ArXiv*, abs/2405.14838.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. 2023. Implicit chain of thought reasoning via knowledge distillation. *ArXiv*, abs/2311.01460.

Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. 2024. Undial: Self-distillation with adjusted logits for robust unlearning in large language models. *Preprint*, arXiv:2402.10052.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. Pal: Program-aided language models. *arXiv preprint arXiv:2211.10435*.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*.

Google. 2024. Our next-generation model: Gemini 1.5.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

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*.

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. *Preprint*, arXiv:2412.06769.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

Dongfang Li, zhenyu liu, Xinshuo Hu, Zetian Sun, Baotian Hu, and Min Zhang. 2024a. In-context learning state vector with inner and momentum optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zongqian Li, Yixuan Su, and Nigel Collier. 2024b. 500xcompressor: Generalized prompt compression for large language models. *Preprint*, arXiv:2408.03094.

Zicheng Lin, Tian Liang, Jiahao Xu, Qiuzhi Lin, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. 2025. Critical tokens matter: Token-level contrastive estimation enhances llm's reasoning capability. *Preprint*, arXiv:2411.19943.

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. 2023. In-context vectors: Making in context learning more effective and controllable through latent space steering. *ArXiv*, abs/2311.06668.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

William Merrill and Ashish Sabharwal. 2024. The expressive power of transformers with chain of thought. In *The Twelfth International Conference on Learning Representations*.

Meta. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

OpenAI. 2024. Hello gpt-4o.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. 2025. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *The Thirteenth International Conference on Learning Representations*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Jacob Pfau, William Merrill, and Samuel R. Bowman. 2024. Let's think dot by dot: Hidden computation in transformer language models. In *First Conference on Language Modeling*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. 2024. What formal languages can transformers express? a survey. *Transactions of the Association for Computational Linguistics*, 12:543–561.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *Preprint*, arXiv:2402.13116.

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. 2024. Self-distillation bridges distribution gap in language model fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1028–1043, Bangkok, Thailand. Association for Computational Linguistics.

A Implementation Details

For all experiments, we use the AdamW optimizer (Loshchilov and Hutter, 2019) with a cosine scheduler (without cycles) and a linear warm-up over the first 3% of steps. The effective batch size is 128. Both α and β are set to 1 (Equation 1). We apply LoRA (Hu et al., 2022) finetuning with a rank of 128 and an alpha value of 32, using bfloat16 precision.

For GPT-2, we set the learning rate to 3e-3 and γ to 1. Training runs for 40 epochs, taking approximately 36 hours on a single A100 (80GB).

For LLaMA-3.2-1b, we use a learning rate of 8e-4 and set γ to 20, as we observe that its distillation loss has a much smaller magnitude. The model is trained for 10 epochs, requiring approximately 48 hours on a single A100 (80GB).

For iCoT training of GPT-2, we use a learning rate of 5e-5 and train for 100 epochs, removing 4 tokens per epoch for GSM8k-Aug-NL. For iCoT training of LLaMA-1b, we use a learning rate of 1e-5 and train for 50 epochs, removing 8 tokens per epoch for GSM8k-Aug and 16 tokens per epoch for GSM8k-Aug-NL.

B Proof: CoTs Contribute a Shift in Hidden Activation

In this section, we provide a proof to demonstrate why Chain-of-Thought (CoT) contributes a shift in hidden activation. This proof is largely inspired by the work of (Li et al., 2024a), which analyzed In-Context Learning.

In a typical CoT training dataset, the input usually consists of four components: the question Q, the rationale R, the prompt for the answer P (e.g., "The answer is:"), and the final answer A.

We analyze the attention activation of the last prompt token, $\mathbf{q}$—in this case, ":"—at the l-th transformer layer. The output activation $\mathbf{a}^l$ from the attention heads of this token is given by:

$$\mathbf{a}^l = W_V[Q;R;P]\text{softmax}(\frac{W_K[Q;R;P]^T\mathbf{q}}{\sqrt{d}}) \tag{6}$$

where W_K and W_V are the model's key and value parameters, $[Q;R;P]$ represents the concatenation of the three inputs, and $\sqrt{d}$ is a scaling factor.

For simplicity of analysis, inspired by (Li et al., 2024a), we omit the softmax operation and the scaling factor, as these do not affect the core conclusion. With this simplification, the following derivation holds:

$$\begin{aligned}
\mathbf{a}^l &\approx W_V[Q;R;P]W_K[Q;R;P]^T\mathbf{q} \\
&= \Big(W_VQ(W_VQ)^T + W_VR(W_VR)^T \\
&\qquad + W_VP(W_VP)^T\Big)\mathbf{q} \\
&= \Big(W_V[Q;P](W_V[Q;P])^T \\
&\qquad + W_VR(W_VR)^T\Big)\mathbf{q} \\
&= \Big(W_{\text{no-CoT}} + W_VR(W_KR)^T\Big)\mathbf{q} \\
&= \mathbf{a}^l_{\text{no-CoT}} + W_VR(W_KR)^T\mathbf{q}
\end{aligned}$$

Here, $W_{\text{no-CoT}}$ is defined as $W_V[Q;P](W_K[Q;P])^T$, accounting for the contribution of Q and P without the CoT rationale. Correspondingly, $\mathbf{a}^l_{\text{no-CoT}}$ represents the attention activation excluding CoT.

The additional term $W_VR(W_KR)^T\mathbf{q}$ represents the contribution of the CoT rationale R to the hidden activation. We can get the hidden activation by transforming the attention activation by a nonlinear function f:

$$\mathbf{h}^l \approx \mathbf{h}^l_{\text{no-CoT}} + f\Big(W_VR(W_KR)^T\mathbf{q}\Big) \tag{7}$$

Thus, we conclude that the rationale R in the CoT primarily contributes a shift in hidden activation values, emphasizing its role as an additive factor in the latent representation. This shift can be effectively captured and learned using a distance metric.

C Datasets

We provide examples and statistics of training datasets and evaluation benchmarks.

C.1 Examples

GSM8k-Aug*

```
Question = "Jen shared a pack of chocolates among her friends. She gave 20% to Lucy, 30% to Sarah and the remaining were shared equally among four others. If the pack contained 100 chocolates, how many chocolates were each of the four others getting?"
CoT = "The total percentage given to Lucy and Sarah is 20% + 30% = 50%. So, the remaining percentage that was shared among the others is 100% - 50% = 50%. The total number of chocolates shared among the others is 100 * 50 / 100 = 50 chocolates. So, each of the four others received 50 / 4 = 12.5 chocolates."
Answer = "12.5"
```

GSM8k-Aug

```
Question = "Out of 600 employees in a company, 30% got promoted while 10% received bonus. How many employees did not get either a promotion or a bonus?"
CoT = "«600*30/100=180» «600*10/100=60» «180+60=240» «600-240=360»"
Answer = "360"
```

SVAMP

```
Question = "There are 87 oranges and 290 bananas in Philip's collection. If the bananas are organized into 2 groups and oranges are organized into 93 groups. How big is each group of bananas?" Answer = "145"
```

MultiArith

```
Question = "There are 64 students trying out for the school's trivia teams. If 36 of them didn't get picked for the team and the rest were put into 4 groups, how many students would be in each group?"
Answer = "7"
```

GSM-Hard

```
Question = "Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with 4933828. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?"
Answer = "-9867630.0"
```

C.2 Statistics

The statistics of training data are shown in Table A1, and the statistics of evaluation benchmarks are shown in Table A2.

Training Dataset	Num. Data	Avg. CoT Tokens
GSM8k-Aug	385,620	20.3
GSM8k-Aug*	384,625	49.0

Table A1: Training data statistics.

Evaluation Benchmark	Num. Data
GSM8k	1319
SVAMP	1000
GSM-Hard	1319
MultiArith	500

Table A2: Evaluation Benchmark statistics.

D Performance on Training Data

The training accuracy of CODI and CoT-SFT is shown in Table A3.

	GPT-2	LLaMA1b
CoT-SFT	99.1%	98.9%
CODI (Ours)	81.5%	95.0%

Table A3: Training Accuracies on GSM8k-Aug.

E Interpretability Case Studies

More case studies on the interpretability of CODI are provided in Figure A1 and Figure A2

F CODI Code

The example Python code of CODI is illustrated in Figure A3.

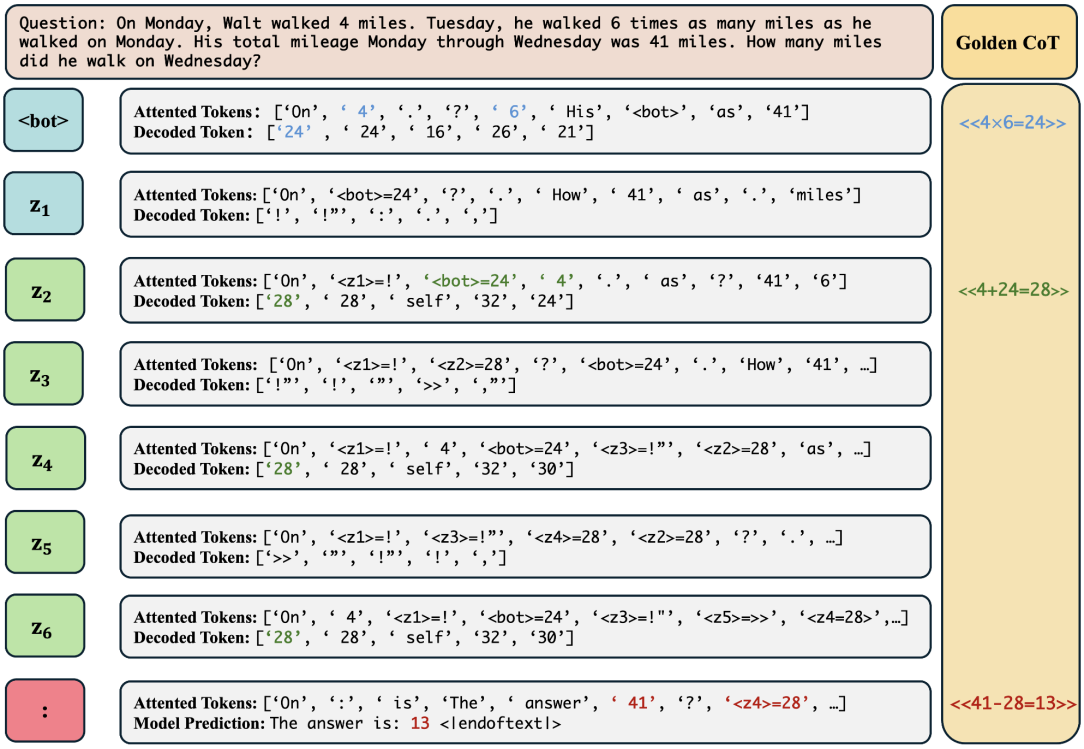

Figure A1: CODI's interpretability on problems involving two steps.

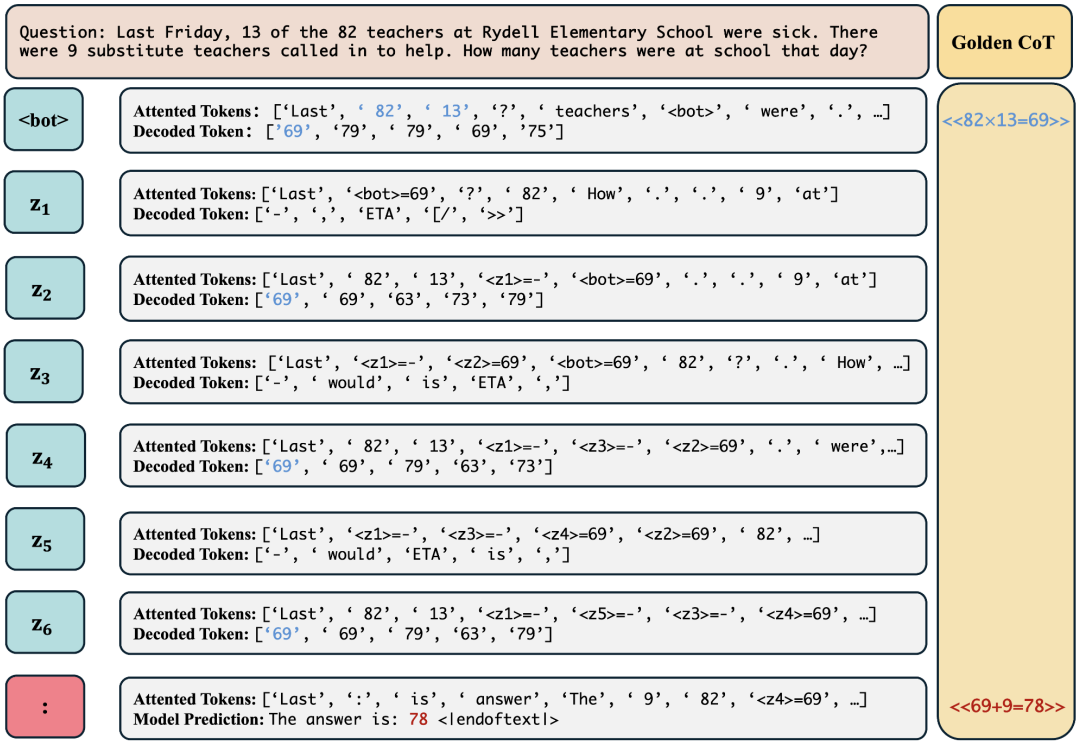

Figure A2: CODI's interpretability on problems involving one step.

```python
class ContinuousCoTviaKnowledgeDistillation:
	def __init__(self,):
		self.num_latent = 6
		self.alpha, self.beta, self.gamma = 1, 1, 1
	self.llm = get_gpt2_model()
		self.prj = nn.Sequential(
			nn.Linear(hidden_dim, hidden_dim),
			nn.GELU(),
			nn.Linear(hidden_dim, hidden_dim),
		)

	def forward(x, y, x_cot_y):
		# teacher learning
		y_teacher = self.llm(x_cot_y)
		teacher_ce_loss = cross_entropy(y_teacher, x_cot_y) # loss1

		# student learning
		latent = self.llm(torch.cat([x, bot_token], dim=1))[:, -1]
		latent = self.prj(latent)
		past_key_values = latent.past_key_values

		# continuous CoT reasoning
		for i in range(self.num_latent):
			latent = self.llm(latent, past_key_values)
			latent = self.prj(latent)
			past_key_values = latent.past_key_values

		y_student = self.llm(torch.cat([eot_token, y], dim=1), past_key_values)
		student_ce_loss = cross_entropy(y_student, y) # loss2

		# knowledge distillation
		knowledge_distillation_loss = smooth_l1_loss(
			y_teacher.hidden_states[:, teacher_exact_answer_token_position-1],
			y_student.hidden_states[:, student_exact_answer_token_position-1]
		) # loss3
		# normalisation
		knowledge_distillation_loss /= y_teacher.hidden_states[:,
		    teacher_exact_answer_token_position-1].std()

		return self.alpha*teacher_ce_loss + self.beta*student_ce_loss + self.gamma*
		    knowledge_distillation_loss
```

Figure A3: Example Python code illustrating the ContinuousCoTviaKnowledgeDistillation class.