Mixed Distillation Helps Smaller Language Models Reason Better

Chenglin Li[1*], **Qianglong Chen**[1*], **Liangyue Li**[3], **Caiyu Wang**[2], **Feng Tao**[1], **Yicheng Li**[1], **Zulong Chen**[3], **Yin Zhang**[1†]
[1]Zhejiang University, Hangzhou, China
[2]Dalian Medical University, Dalian, China
[3]Alibaba, Hangzhou, China
{chenglinli,chenqianglong,tao_feng,yichengli,zhangyin98}@zju.edu.cn
{jackiet99,wangcaiyu95,chenzulong198867}@gmail.com

Abstract

As large language models (LLMs) have demonstrated impressive multiple step-by-step reasoning capabilities in recent natural language processing (NLP) reasoning tasks, many studies are interested in distilling reasoning abilities into smaller language models (SLMs) via fine-tuning. Previous distillation methods usually utilize the capabilities of LLMs to generate chain-of-thought (CoT) samples to teach SLMs. However, this distillation approach performs poorly in certain scenarios due to the limitations of CoT. In this work, we introduce a novel **Mixed Distillation** (MD) framework, distilling multiple step-by-step reasoning abilities into SLMs. First, we leverage LLMs to generate multiple step-by-step reasoning rationales by sampling automatically. Then, we create high-quality, well-balanced mixed thought data and design a novel multi-task loss to help SLMs better learn and adaptively activate multiple step-by-step reasoning. Our extensive experiments demonstrate that MD enhances both single-path (using either CoT or PoT) and multi-path (using both CoT and PoT) reasoning abilities of SLMs during inference across reasoning tasks. Notably, a single model generated by MD exceeds the comprehensive performance of an ensemble of two individual CoT and PoT distilled models. Mistral-7B using MD can achieve remarkable improvements of 87.5%, 74.0% and 77.1% on SVAMP, GSM8K and ASDIV, respectively, outperforming the teacher model, GPT-3.5-Turbo. We hope our work provides insight into SLMs' multiple step-by-step reasoning abilities.

1 Introduction

Recent LLMs (Bai et al., 2022; OpenAI, 2023; Anil et al., 2023) such as GPT-4 have demonstrated multiple step-by-step reasoning capabilities (Wei et al., 2022; Wang et al., 2022; Chen et al., 2022) with

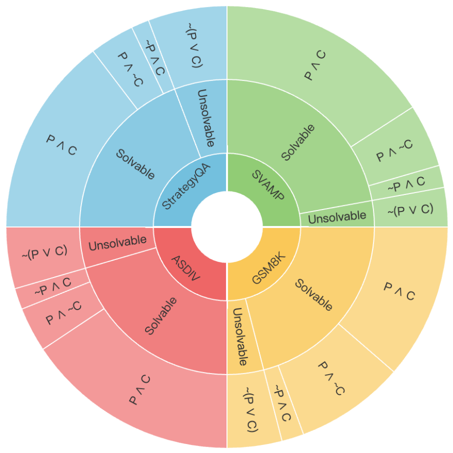
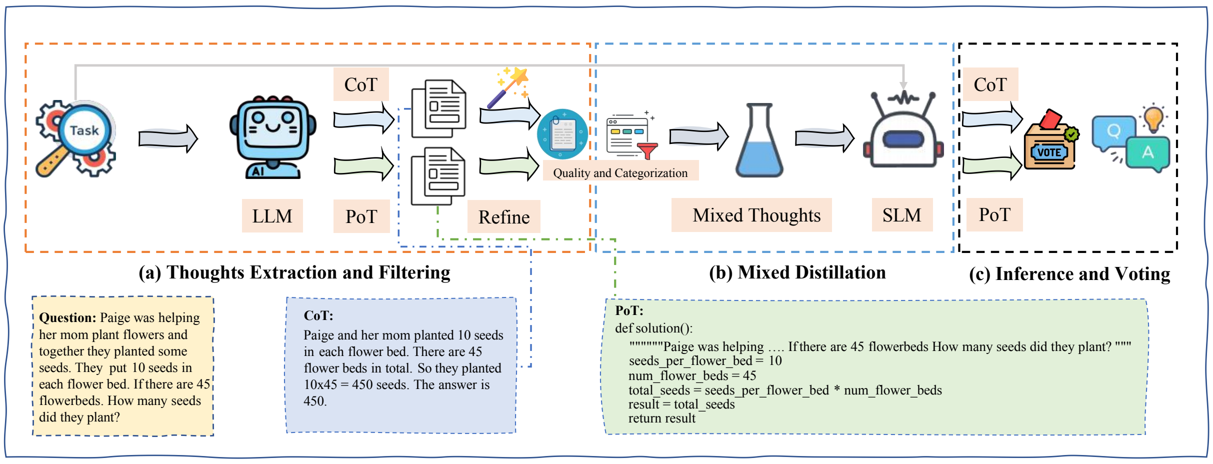


Figure 1: The proportion of tasks solved by different capabilities on SVAMP, GSM8K, ASDIV, and StrategyQA is as follows: $\sim P \wedge C$ denotes the proportion of problems solved only by CoT; $P \wedge \sim C$ denotes the proportion of problems solved only by PoT; $P \wedge C$ represents the proportion of problems solved by both; $\sim (P \vee C)$ indicates the remaining unsolved challenges.

chain-of-thought (CoT) and program-of-thought (PoT) previously unseen in SLMs. CoT boosts reasoning by guiding LLMs to produce intermediate natural language steps (Wei et al., 2022), while PoT stimulates reasoning by generating intermediate program code that can be executed by the Python executor (Chen et al., 2022; Gao et al., 2023). However, deploying these advanced LLMs in real-world applications presents significant costs and computational demands (Kaplan et al., 2020; Sorscher et al., 2022). To address these challenges, distilling step-by-step reasoning capabilities from LLMs emerges as a resource-friendly and effective strategy. On the other hand, LLMs with multiple step-by-step reasoning can address distinct challenges. As shown in Figure 1, LLMs can solve the majority of problems ($P \wedge C$, $\sim P \wedge C$, $P \wedge \sim C$) through step-by-step reasoning. Among them, a large part of the problems can be solved by both PoT and

[*] Equal Contribution.
[†] Corresponding author: Yin Zhang.



Question: Paige was helping her mom plant flowers and together they planted some seeds. They put 10 seeds in each flower bed. If there are 45 flowerbeds. How many seeds did they plant?

CoT: Paige and her mom planted 10 seeds in each flower bed. There are 45 flower beds in total. So they planted 10x45 = 450 seeds. The answer is 450.

PoT:
```
def solution():
    """""""Paige was helping …. If there are 45 flowerbeds How many seeds did they plant? """
    seeds_per_flower_bed = 10
    num_flower_beds = 45
    total_seeds = seeds_per_flower_bed * num_flower_beds
    result = total_seeds
    return result
```

Figure 2: Overview of Mixed Distillation framework: extracting and distilling CoT and PoT from large language models (LLM) to task-specific smaller language models (SLMs).

CoT ($P \wedge C$). Moreover, each reasoning ability has its own advantages. PoT excels in large-scale and intensive numerical calculations (Chen et al., 2022; Gao et al., 2023), whereas CoT demonstrates proficiency in language comprehension, geometry, algebra, fraction calculations, and equation comparisons (Wei et al., 2022). Specifically, only about 6% of the problems can be solved exclusively via CoT ($\sim P \wedge C$) across tasks. In contrast, PoT exclusively addresses 31.98% of problems on GSM8K, and this figure exceeds 10% for other tasks ($P \wedge \sim C$). We show more details in Appendix D.1. Meanwhile, compared with LLMs, SLMs struggle with generating effective intermediate steps due to limited knowledge, which makes their reasoning challenging (Valmeekam et al., 2022; Huang and Chang, 2022; Chu et al., 2023). Existing works focus on CoT distillation, a method that utilizes the CoT rationales of LLMs as supervision for training SLMs, which excel in generating natural language content (Arora et al., 2022; Shridhar et al., 2023; Ho et al., 2023; Li et al., 2023b; Hsieh et al., 2023; Fu et al., 2023; Yang et al., 2023; Zhao et al., 2023). However, the sample data generated by LLMs is only based on a single reasoning path, CoT, and SLMs can't learn PoT and activate multiple step-by-step reasoning making it challenging for SLMs to develop robust reasoning abilities (Huang and Chang, 2022; Chu et al., 2023). In addition, previous works (Zhang and Yang, 2021; Wei et al., 2021; Longpre et al., 2023) have proved that multi-task learning enhances model performance by involving various knowledge domains.

Therefore, we propose a novel distillation framework, Mixed Distillation (MD), which combines multiple prompting techniques to create high-quality, well-balanced mixed thought data within a novel multi-task learning approach, as shown in Figure 2. Specifically, we utilize multiple prompting techniques with sampling to prompt LLMs to generate multiple step-by-step reasoning paths for each input, resulting in a collection of thought data. To help SLMs learn and adaptively activate multiple step-by-step reasoning capabilities, we further conduct quality control and categorization to create mixed thought data. Then, we present a novel multi-task loss to fine-tuning SLMs, using consensus via majority vote to obtain answers with multiple prompting techniques. This approach allows SLMs to activate different step-by-step reasoning capabilities for different problems. We conduct extensive experiments on SVAMP, GSM8K, ASDIV, and StrategyQA reasoning tasks to validate the efficacy of MD. As shown in Figure 3, the overall performance of SLMs is better than that of models using other distillation methods. Notably, on SVAMP, LLaMA2-7B achieves a significant 15% improvement over CoT Distillation and records an impressive 5% increase compared to PoT Distillation. The contributions of our work are as follows:

- We validate that SLMs are also capable of learning PoT, enhancing their step-by-step reasoning abilities, and providing valuable insights for future research on PoT in SLMs.

- To help SLMs better learn and adaptively activate multiple step-by-step reasoning capabilities, we propose a novel framework, MD, which constructs high-quality, well-balanced mixed thought data using multiple prompting

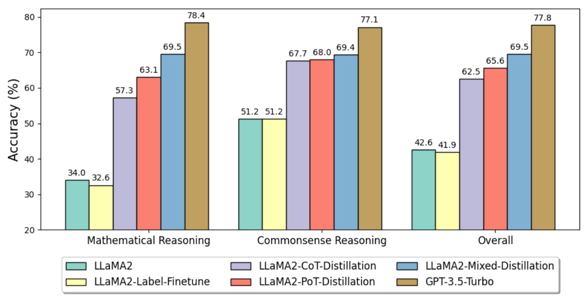


Figure 3: Performance of different methods across reasoning domains based LLaMA2-7B.

techniques and then applies a novel multi-task learning loss, marking a significant advancement in model distillation.

- We conducted a series of experiments to validate the efficacy of MD, enhancing the SLMs' reasoning including single-path and multi-path reasoning, across models, across datasets, across dataset scales and even extending to out-of-distribution (OOD) training data.

2 Related Work

Multiple Thoughts Prompting Techniques in LLM Recent work (Wei et al., 2022; Chu et al., 2023; Gao et al., 2023; Chen et al., 2022; Hu et al., 2023; Imani et al., 2023), focusing on eliciting the step-by-step reasoning process of LLMs, has validated its effectiveness in reasoning domains, such as SVAMP (Patel et al., 2021), GSM8K (Cobbe et al., 2021), ASDIV (Miao et al., 2021) and StrategyQA (Geva et al., 2021). CoT (Wei et al., 2022) enhances reasoning by prompting LLMs to generate intermediate natural language thought steps. PoT (Gao et al., 2023; Chen et al., 2022) stimulates LLM's reasoning ability by prompting them to generate intermediate code that can be executed by the python executor. Zhang et al. (2023); Li et al. (2023a) have shown the adaptability of combining natural language reasoning and program synthesis within prompt-based learning to effectively solve reasoning tasks. Yue et al. (2023b) improve the mathematical reasoning ability of LLMs combining PoT and CoT by designing prompts. Yue et al. (2023a) develop a cost-effective approach by combining PoT and CoT. However, previous work (Yue et al., 2023b,a; Imani et al., 2023) provided limited insights into PoT and CoT in improving SLMs by distilling LLMs. Our MD presents a novel multi-task loss to help SLMs learn and adaptively activate both reasoning capabilities on specific tasks.

Knowledge Distillation from LLMs Knowledge distillation (Buciluǎ et al., 2006; Ba and Caruana, 2014; Hinton et al., 2015; Beyer et al., 2022; Fu et al., 2023; Zhu et al., 2023) has proved its effectiveness in improving SLMs. Some works (Li et al., 2023b; Hsieh et al., 2023; Wang et al., 2023; Fu et al., 2023; Shridhar et al., 2023; Zhu et al., 2024) leverage generative CoT as a supervisory signal to fine-tune smaller task-specific models. However, previous works (Shridhar et al., 2023; Zhu et al., 2024) focus solely on LLM-generated single reasoning path as the supervisory signal, without considering the capability of CoT and PoT in SLMs. Our proposed MD emphasizes the importance of PoT with CoT as supervisory signals and the SLMs can effectively learn and activate multiple step-by-step reasoning from high-quality and well-balanced mixed thought data.

3 Approach

3.1 Mixed Thoughts from LLMs

Thoughts Extraction from LLMs We employ CoT prompts (Wei et al., 2022) and PoT prompts (Chen et al., 2022) to elicit and extract the reasoning thought process of LLMs. Given a training dataset, $x_i \in \mathcal{D}$, we begin by devising a prompt template, denoted as p, to define how the task should be addressed. Each prompt takes the form of a triplet, (x_p, r_p, y_p), where x_p represents an example input, y_p corresponds to its associated label, and r_p comprises a user-provided reasoning path explaining why x_p can be categorized as y_p. We append each input, x_i to the template p and use it as the input prompt for the LLMs to generate reasoning paths and labels as $\hat{r}_i, \hat{y}_i$ for each $x_i \in \mathcal{D}$. Specifically, the PoT few-shot template for StrategyQA, which focuses on commonsense reasoning, is enhanced through the application of CoT as Python code annotations (Li et al., 2023a) with more details provided in Appendix B. By raising the LLM's temperature (default 0.7), we generate n (default 20) samples each of CoT and PoT per input, as thoughts from LLM.

Creating Mixed Thought Data We refine the thoughts to allow fine-tuned SLMs to sense the PoT and CoT during inference. Specifically, we first filter out the PoT that the Python executor can't execute and the CoT that does not provide an answer to ensure data quality. Intuitively, higher quality data can enhance performance while incorrect reasoning steps may confound models (Zhou et al., 2024). Then for each input, x_i, PoT and CoT reasoning results selection is done using consensus via majority vote. By comparing them with true labels, we categorize samples into four types: 10% are solvable exclusively by the PoT, which involves large-scale and intensive numerical calculations (Type 1); another 10% can only be solved by the CoT, which focuses on language comprehension, as well as geometric, algebraic, fractional calculations, and equation comparisons (Type 2); 60% of the samples can be addressed using both methods (Type 3); and the remaining 20% are unsolvable by either method (Type 4). To enhance the dataset, we upsample the Type 1 and 2 samples and filter out the Type 4 samples to create well-balanced mixed thought data. With mixed thought data, we design a novel multi-task loss to help SLMs learn and adaptively activate step-by-step reasoning. Specifically, for problems that PoT is suitable to solve, the self-consistency of the PoT path's answer will be higher, making the final result tend to be the answer to the PoT result. The same is true for problems that the CoT is suitable to solve. We show cases of Type 1 and 2 in Appendix Table 6.

3.2 Mixed Thoughts Distillation

We first outline the basic paradigm for learning specific task models. Then, we combine CoT and PoT into the training process to extend it. Formally, we represent the dataset as $D = \{(x_i, y_i)\}_{i=1}^{N}$, where each x_i is an input, and y_i is its corresponding output label. In this paper, we focus on text-to-text tasks (Raffel et al., 2020).

Standard Specific-task Learning The prevalent paradigm for training a task-specific model involves fine-tuning a pre-trained model using supervised data (Howard and Ruder, 2018). In scenarios where human-annotated labels are unavailable, task-specific distillation (Hinton et al., 2015; Tang et al., 2019) employs LLM teachers to produce pseudo-noisy training labels $\hat{y}_i$ in place of y_i (Wang et al., 2021; Smith et al., 2022; Arora et al., 2022). For both scenarios, the current model, denoted as f, is trained using a paradigm that aims to minimize the loss in label prediction:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), \hat{y}_i) \tag{1}$$

where ℓ represents the cross-entropy loss between predicted tokens and target tokens. For simplicity and clarity, we use $\hat{y}_i$ in Eq. 1, representing either human-annotated labels y_i in the standard fine-tuning scenario or LLM-predicted labels $\hat{y}_i$ in the context of model distillation.

Multi-task Learning with CoT and PoT For Type 1 and 2 data in mixed thoughts, we design a loss function to help the model simultaneously learn CoT and PoT, consisting of two components:

$$\mathcal{L}_{1,2} = \mathcal{L}_{\text{path\_CoT}} + \mathcal{L}_{\text{path\_PoT}} \tag{2}$$

Here, $\mathcal{L}_{\text{path}}$ is the loss for generating CoT or PoT reasoning paths and predicting labels, defined as:

$$\mathcal{L}_{\text{path}} = \frac{1}{N}\sum_{i=1}^{N} \ell(f(x_i), \hat{r}_i + \hat{y}_i) \tag{3}$$

The $\hat{r}_i$ represents the reasoning paths generated by LLMs with CoT or PoT, and their respective objective functions are defined as $\mathcal{L}_{\text{path\_CoT}}$ and $\mathcal{L}_{\text{path\_PoT}}$. For Type 3 data, we design the loss function to help the model adaptively activate both capabilities:

$$\mathcal{L}_3 = (1-\lambda)\mathcal{L}_{\text{path\_CoT}} + \lambda\mathcal{L}_{\text{path\_PoT}} \tag{4}$$

Here, λ is a weight parameter defined for each x_i based on the number of samples in CoT that can correctly predict the answer, denoted as n_1, and the number of samples in PoT that can correctly predict the answer, denoted as n_2, respectively. λ is set as $\frac{n_2}{n_1+n_2}$. The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{1,2} + \mathcal{L}_3$$

This is the MD we emphasize. In the input x_i outlined above, we introduce the concept of task prompts embedded into input examples to train SLMs to produce distinct reasoning paths. More specifically, we employed `Let's think step by step` and `Let's break down the code step by step` to guide the generation of CoT and PoT, respectively.

Once both CoT and PoT abilities are in the SLM, multi-path reasoning can be employed via multiple prompting techniques as shown in Figure 2. Result selection is done using consensus via a majority vote. In particular, during inference for the SLM, input x_i is concatenated with the guiding prompt

Method	#Fixed Params.	#Training Params	Mathematical Reasoning (%) SVAMP	GSM8K	ASDIV	Commonsense Reasoning (%) StrategyQA
GPT-4 (OpenAI et al., 2023)	Unknown	0M	93.0	92.0	91.3	77.1
GPT-3.5-Turbo (Few-shot CoT)	Unknown	0M	70	51.98	68.96	63.76
GPT-3.5-Turbo	Unknown	0M	82.0	77.4	75.8	71.6
LLaMA2 (Touvron et al., 2023)†	7B	-	38.0	13.3	50.7	-
CodeLlama (Roziere et al., 2023)†	7B	-	59.0	34.0	61.4	-
Mistral (Touvron et al., 2023)†	7B	-	66.0	52.1	62.0	-
FlanT5-Large (Fu et al., 2023)†	770M	770M	6.8	6.9	10.1	-
GPT2-Large (Radford et al., 2019)	-	-	7.0	2.0	1.8	-
WizardMath (Luo et al., 2023)†	7B	-	57.3	54.9	59.1	-
FlanT5-Large + Specialized (Fu et al., 2023)	770M	770M	20.4	20.2	23.8	-
GPT2-Large + Soc (Shridhar et al., 2023)	770M	770M	-	21.1	-	66.4
GPT-J + Multi-round & Self-Reflection (Wang et al., 2023)	6B	-	55.0	33.1	-	65.9
T5-large + Distill step by step (Hsieh et al., 2023)	770M	770M	65.5	-	-	-
Label-Finetuning						
T5-large	770M	770M	7.5 (↑0.7)	7.4 (↑0.5)	11.1 (↑1.0)	50.2
LLaMA2-7B	70M	770M	50.0 (↑12.0)	10.6 (↓2.7)	37.3 (↓13.4)	51.2
CodeLlama-7B	70M	770M	39.0 (↓20.0)	9.4 (↓24.6)	24.8 (↓36.6)	50.5
Single-Path Distillation						
T5-large + CoT	770M	770M	32.5 (↑25.7)	10.5 (↑3.6)	18.9 (↑8.8)	54.3
LLaMA2-7B + CoT	7B	160M	69.5 (↑31.5)	40.1 (↑26.8)	62.2 (↑11.5)	67.7
CodeLlama-7B + CoT	7B	160M	71.0 (↑12.0)	34.2 (↑0.2)	60.0 (↓1.4)	66.4
T5-large + PoT	770M	770M	68.0 (↑61.2)	22.5 (↑15.6)	58.1 (↑48.0)	57.3
LLaMA2-7B + PoT	7B	160M	77.0 (↑39.0)	46.5 (↑33.2)	65.6 (↑14.9)	68.0
CodeLlama-7B + PoT	7B	160M	83.0 (↑24.0)	51.9 (↑17.9)	67.5 (↑6.1)	67.6
Ensemble Single-Path Distillation						
T5-Large + CoT w/ PoT*	770M*2	770M*2	70.5 (↑63.7)	24.8 (↑17.9)	57.8 (↑47.7)	58.1
LLaMA2-7B + CoT w/ PoT*	7B*2	160M*2	81.0 (↑43.0)	49.7 (↑36.4)	69.9 (↑19.2)	68.1
CodeLlama-7B + CoT w/ PoT*	7B*2	160M*2	82.5 (↑23.5)	52.0 (↑18.0)	70.2 (↑8.8)	66.7
Mixed Distillation (Ours)						
T5-Large-MD						
+ CoT	770M	770M	34.5 (↑27.7)	10.6 (↑3.7)	19.0 (↑8.9)	54.3
+ PoT	770M	770M	74.0 (↑67.2)	23.6 (↑16.7)	58.2 (↑48.1)	56.7
+ CoT w/ PoT*	770M	770M	76.0 (↑69.2)	24.6 (↑17.7)	58.3 (↑48.2)	59.1
GPT2-Large-MD						
+ CoT	760M	760M	29.0 (↑22.0)	8.9 (↑6.9)	12.8 (↑11.0)	-
+ PoT	770M	770M	58.0 (↑51.0)	14.4 (↑12.4)	38.8 (↑37.0)	-
+ CoT w/ PoT*	770M	770M	59.0 (↑52.0)	18.4 (↑16.4)	40.1 (↑38.3)	-
LLaMA2-7B-MD						
+ CoT	7B	160M	70.0 (↑32.0)	41.5 (↑28.2)	64.2 (↑13.5)	67.4
+ PoT	7B	160M	80.5 (↑42.5)	51.6 (↑38.3)	66.5 (↑15.8)	66.4
+ CoT w/ PoT*	7B	160M	84.5 (↑46.5)	53.8 (↑40.5)	70.2 (↑19.5)	69.4
CodeLlama-7B-MD						
+ CoT	7B	160M	73.0 (↑14.0)	35.3 (↑1.3)	60.6 (↓0.8)	66.1
+ PoT	7B	160M	85.0 (↑26.0)	52.4 (↑18.4)	71.8 (↑10.4)	66.6
+ CoT w/ PoT*	7B	160M	85.5 (↑26.5)	53.2 (↑19.2)	73.5 (↑12.1)	**70.3**
Mistral-7B-MD						
+ CoT	7B	160M	80.0 (↑14.0)	63.0 (↑10.9)	71.5 (↑9.5)	-
+ PoT	7B	160M	84.5 (↑18.5)	72.8 (↑20.7)	76.0 (↑14.0)	-
+ CoT w/ PoT*	7B	160M	**87.5** (↑21.5)	**74.0** (↑21.9)	**77.1** (↑15.1)	-

Table 1: Accuracy (%) across tasks:†Results are from (Zhu et al., 2024). "+ CoT" indicates inference via CoT. "*" denotes improved performance in distillation using CoT and PoT to generate 10 reasoning paths, respectively.

phrase `Let's think step by step` to elicit natural language reasoning paths. The answer result is a final answer list, $A_{\text{CoT}} = \{a_1, a_2, \ldots, a_n\}$, obtained via n iterations of sampling. Concurrently, by adopting the phrase `Let's break down the code step by step`, similar to the above process, we extract the intermediate code reasoning path. Then, utilizing the Python executor, the answer list $A_{\text{PoT}} = \{b_1, b_2, \ldots, b_n\}$ is acquired. The final prediction of the SLM, $\mathcal{P}_{\text{final}}$, is expressed as:

$$\mathcal{P}_{\text{final}} = V(\text{concat}(A_{\text{CoT}}, A_{\text{PoT}})) \tag{5}$$

where $V(\cdot)$ represents a voting function that selects the most frequently occurring answer from the concatenated list of A_{CoT} and A_{PoT}. The concat$(\cdot)$ function represents the concatenation of the two lists. This step-by-step thought process along two independent paths ensures that the final prediction is determined through a voting mechanism on the answers procured from each path (Wang et al., 2022).

4 Experiments

In this section, we first prove that PoT, as a supervisory signal, enhances the SLMs' reasoning capabilities (Sec. 4.1). Moreover, our findings emphasize the benefits of MD, which enhances SLMs' capabilities in single-path reasoning and multiple-path reasoning, enabling SLMs to learn and adaptively activate step-by-step reasoning (Sec. 4.2). We further conduct extensive experiments based on LLaMA2-7B, CodeLlama-7B, Mistral-7B, T5-Large, and GPT2-Large, compared to other distillation methods (Sec. 4.3). Finally, we validate the generalizability of MD (Sec. 4.4).

Datasets Our experiments primarily center on the following datasets: SVAMP (Patel et al., 2021), GSM8K (Cobbe et al., 2021), and ASDIV (Miao et al., 2021). We extend our assessment to StrategyQA (Geva et al., 2021), where we evaluate the capability of commonsense reasoning. More dataset details are provided in Appendix A.

Baselines We evaluate MD by comparing it with experiments using Closed-Source Models (OpenAI et al., 2023), Open-Source Models (Touvron et al.,

Method	SVAMP	ASDIV
Closed-Source Models		
GPT-3.5-Turbo	82.0	75.8
Open-Source Models		
LLaMA2 (Touvron et al., 2023)†	38.0	50.7
CodeLlama (Roziere et al., 2023)†	59.0	61.4
WizardMath (Luo et al., 2023)†	57.3	59.1
Single-Path Distillation		
LLaMA2-7B + CoT	64.5 (↑26.5)	63.2 (↑12.5)
LLaMA2-7B + PoT	56.5 (↑18.5)	64.2 (↑13.5)
Ensemble Single-Path Distillation		
LLaMA-7B + CoT w/ PoT*	61.5 (↑23.5)	65.9 (↑15.2)
Mixed Distillation (Ours)		
LLaMA2-7B-MD		
+ CoT	65.0 (↑27.0)	64.2 (↑13.5)
+ PoT	70.0 (↑32.0)	64.2 (↑13.5)
+ CoT w/ PoT*	**74.5** (↑36.5)	**68.9** (↑18.2)

Table 2: Accuracy (%) across tasks which demonstrate the generalizability of Mixed Distillation in OOD scenarios.

2023; Roziere et al., 2023; Luo et al., 2023), Traditional Distillation (Fu et al., 2023; Shridhar et al., 2023; Wang et al., 2023; Hsieh et al., 2023), Label-Finetuning, Single-Path Distillation and Ensemble Single-Path Distillation. More details can be found in Appendix C.

Setup The teacher model used is GPT-3.5-Turbo [1] in the distillation framework. We generate 20 samples each of CoT and PoT per question. The temperature is set to 0.7. The experiments cover a wide range of student models, including LLaMA2-7B, CodeLlama-7B, Mistral-7B, GPT2-Large, and T5-Large. For the efficient fine-tuning of the LLaMA series, we employ the QLORA (Dettmers et al., 2023) method. During the training process, we set the maximum number of steps to 8000. It's noteworthy that these primary experiments can be conducted on a single GPU with a capacity of 48GB. During the inference process, the default number of total sampling paths is set to 20 in self-consistency voting (Wang et al., 2022).

4.1 PoT Distillation Enhanced Reasoning

Previous work focused on PoT in LLMs, while we explored enhancing SLMs by distilling PoT from LLMs. In this subsection, we investigate whether models pre-trained on large text data (Zhao et al., 2023) can effectively learn to generate step-by-step code and demonstrate the effectiveness of PoT. Table 1 shows the experimental results, which prove that various models employing the PoT distillation outperform those utilizing the CoT distillation and Label-Finetuning in mathematics and common sense reasoning tasks. For example, T5-Large exhibits a notable improvement of 61.2% on SVAMP. Similarly, LLaMA2 shows enhancements of 33.2% on GSM8K and 14.9% on ASDIV. Meanwhile, T5-Large, LLaMA2, and CodeLlama achieve gains of 7.1%, 17.8%, and 17.4% respectively, compared to Label-Finetuning on StrategyQA. Additionally, we observe that compared with the T5-Large model, LLaMA models with fewer training parameters but larger fixed parameters showed excellent performance. In particular, CodeLlama notably achieves 82.5% accuracy on the SVAMP task, marking a 23.5% improvement. Furthermore, under PoT distillation, CodeLlama, pre-trained on code data, outperforms LLaMA2, achieving an improvement of 4.5% on SVAMP.

4.2 Mixed Distillation Enhanced Reasoning

In this subsection, we show the effectiveness of MD from two aspects: enhancing both single-path reasoning and multi-path reasoning. Single-path reasoning independently measures the two kinds of step-by-step reasoning abilities of SLMs, while multi-path reasoning measures SLMs' ability to adaptively activate step-by-step reasoning.

Mixed Distillation Enhances Single-Path Reasoning Experimental results demonstrate that MD enhances both capabilities of the model compared to single-path distillation. Additionally, MD significantly improves the PoT capability of SLMs compared to CoT. As shown in Table 1, the CoT and PoT abilities of models are improved by mixed distillation. For example, compared with CoT distillation on ASDIV, the CoT ability of LLaMA2 exhibits a 2% increase. Similarly, T5-Large's PoT capability shows a 6.0% improvement over PoT distillation on SVAMP. Specifically, Figure 4 displays the capabilities of LLaMA2 across different datasets. It is worth noting that as the number of sampling inference paths increases, the PoT ability of the model using MD is always better than that of the model trained by Single-Distillation, and the difference observed in the sampling interval of 10-13 paths is the most significant. In addition, CoT capability with MD exceeds the performance of SLMs with single distillation as the number of sampling paths exceeds 13.

Mixed Distillation Enhances Multi-Path Reasoning Using "+CoT w/PoT" for multi-path reasoning during inference, various models achieve

[1] https://platform.openai.com/docs/model-index-for-researchers

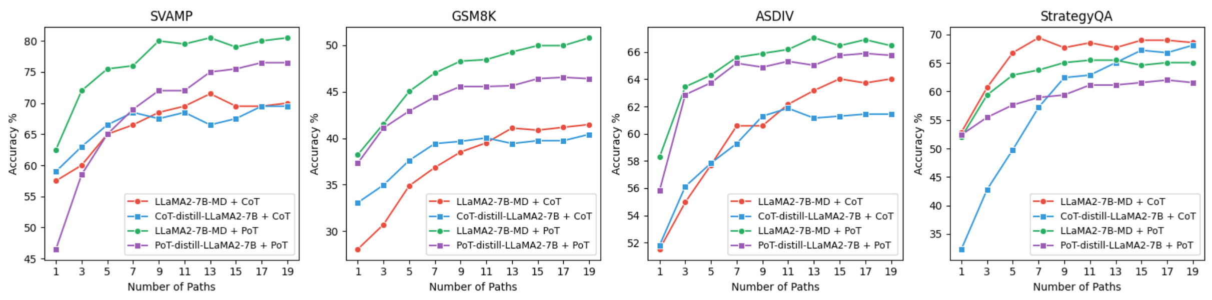


Figure 4: Performance comparison with Mixed Distillation and Single-Path Distillation on SVAMP, GSM8K, ASDIV, and StrategyQA based LLaMA2-7B.

state-of-the-art performances across different tasks. Notably, LLaMA2 excels on GSM8K, achieving an accuracy of 53.8%, which marks an impressive improvement of 40.5%. Similarly, CodeLlama shows remarkable results on the SVAMP and ASDIV tasks, reaching accuracies of 85.5% and 73.5%, respectively, and registering boosts of 26.5% and 12.1%. Additionally, T5-Large stands out in the StrategyQA task with an accuracy of 59.1%, indicating a 7.9% increase over the Label-Finetuning. Furthermore, a single model from MD outperforms two individual distilled models, with LLaMA2 gaining improvements of 3.5% and 4.1% on SVAMP and GSM8K, respectively.

4.3 More Results Compared to Other Distillation Methods

Previous distillation methods typically focus on T5 and GPT series (Fu et al., 2023; Shridhar et al., 2023; Wang et al., 2023; Hsieh et al., 2023). T5-Large with MD achieves accuracies of 76%, 24.6%, and 58.3%, on SVAMP, GSM8K, and ASDIV, respectively, marking a significant advancement. Notably, compared to (Hsieh et al., 2023; Fu et al., 2023), it demonstrates impressive improvements of 10.5% and 55.6%, respectively, on SVAMP. GPT2-Large achieves an accuracy of 59.0% on SVAMP, outperforming GPT-J (Wang et al., 2023). SLMs with MD also notably improve reasoning performance compared to Single-Path, PoT Distillation, and an ensemble of two individual distilled models. LLAMA2-7B with MD shows a 4% improvement over PoT Distillation and a 3.5% improvement over an ensemble of two individual CoT and PoT distilled models on SVAMP, highlighting the superior performance achieved by using a single model with mixed thoughts. Furthermore, compared to WizardMath, the instruction-tuned model, LLAMA2-7B achieves a 27.2% improvement on SVAMP.

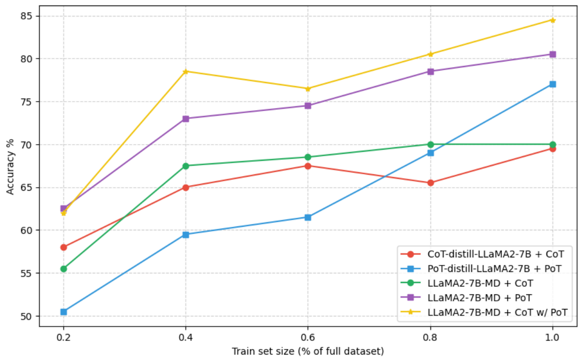


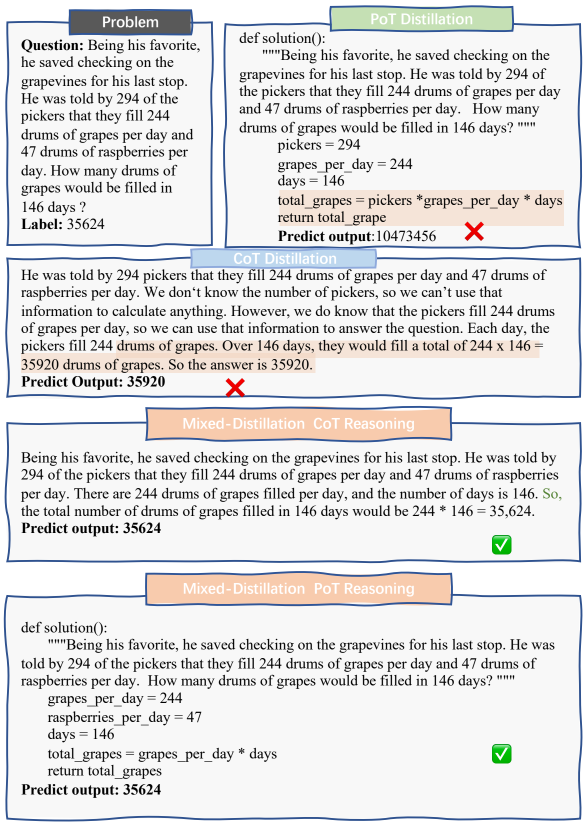
Figure 5: Performance comparison with different train set size on SVAMP based LLaMA2-7B.

4.4 Generalization

In the above experiments, we have proved the effectiveness of MD in the generalization of models and tasks. Furthermore, we conduct experiments to validate the framework's generalization with varying numbers of sampling paths, training set sizes, OOD scenarios, and more prompt techniques, based on LLaMA2. For the training set sizes generalization assessment, we perform distillation using proportions of 20%, 40%, 60%, and 80%, on SVAMP. To evaluate the ability of OOD generalization, we evaluate the model trained on GSM8K for SVAMP and ASDIV datasets. incorporating additional prompting techniques.

4.4.1 Training Set Size

As shown in Figure 5, with the increase in data volume, the performance of models generally improves. Specifically, when the dataset size surpasses 75%, LLaMA2 using PoT as the supervisory signal outperforms the model using CoT. This observation shows that a certain dataset size is required to learn the PoT capability. In the case of MD, across the dataset size range of 20% to 100%, MD always enhances PoT capability compared to


Problem

Question: Being his favorite, he saved checking on the grapevines for his last stop. He was told by 294 of the pickers that they fill 244 drums of grapes per day and 47 drums of raspberries per day. How many drums of grapes would be filled in 146 days ?
Label: 35624

PoT Distillation

```
def solution():
      """Being his favorite, he saved checking on the grapevines for his last stop. He was told by 294 of the pickers that they fill 244 drums of grapes per day and 47 drums of raspberries per day.   How many drums of grapes would be filled in 146 days? """
        pickers = 294
        grapes_per_day = 244
        days = 146
        total_grapes = pickers *grapes_per_day * days
        return total_grape
```
Predict output:10473456 ❌

CoT Distillation

He was told by 294 pickers that they fill 244 drums of grapes per day and 47 drums of raspberries per day. We don‘t know the number of pickers, so we can’t use that information to calculate anything. However, we do know that the pickers fill 244 drums of grapes per day, so we can use that information to answer the question. Each day, the pickers fill 244 drums of grapes. Over 146 days, they would fill a total of 244 x 146 = 35920 drums of grapes. So the answer is 35920.
Predict Output: 35920 ❌

Mixed-Distillation CoT Reasoning

Being his favorite, he saved checking on the grapevines for his last stop. He was told by 294 of the pickers that they fill 244 drums of grapes per day and 47 drums of raspberries per day. There are 244 drums of grapes filled per day, and the number of days is 146. So, the total number of drums of grapes filled in 146 days would be 244 * 146 = 35,624.
Predict output: 35624 ✅

Mixed-Distillation PoT Reasoning

```
def solution():
      """Being his favorite, he saved checking on the grapevines for his last stop. He was told by 294 of the pickers that they fill 244 drums of grapes per day and 47 drums of raspberries per day.  How many drums of grapes would be filled in 146 days? """
      grapes_per_day = 244
      raspberries_per_day = 47
      days = 146
      total_grapes = grapes_per_day * days
      return total_grapes
```
Predict output: 35624 ✅

Figure 6: Case study of different distillation methods based LLaMA2-7B.

PoT distillation. Moreover, when the dataset size exceeds 40%, compared to CoT distillation, the learning ability of CoT shows a 2% improvement. By incorporating multi-path reasoning during inference, the model achieves its optimal performance, providing evidence for the effectiveness of multi-path reasoning in MD.

Method	N=2	N=4	N=8	N=14
Single-Path Distillation				
LLaMA2-7B + CoT	61	66	70.5	69.0
LLaMA2-7B + PoT	47.5	64.5	71	75.5
Ensemble Single-Path Distillation				
LLaMA2-7B + CoT w/ PoT*	56	71.5	73.5	77.5
Mixed Distillation (Ours)				
LLaMA2-7B-MD + CoT	57.5	65	69	71
LLaMA2-7B-MD + PoT	62.5	74.5	79	79.5
LLaMA2-7B-MD + CoT w/ PoT*	66	73	81	83.5

Table 3: Performance of Different Methods Across Various Sampling Paths (Number of Paths, N).

4.4.2 Various Sampling Paths

To explore the impact of multiple paths sampling on models' performance, we evaluate the SLMs with different numbers of sampling paths. With 2 sampling paths in multi-path reasoning, a PoT sample is generated first. If the Python executor deems it unsuccessful, a CoT sample is then generated for prediction, thereby reducing inference latency and cost. Table 3 shows the effectiveness of MD, with LLaMA2-7B-MD + CoT w/ PoT achieving 66% accuracy for N=2 and improving to 83.5% for N=14. This highlights a significant performance improvement over single-path and ensemble distillation methods.

4.4.3 Out-of-Distribution Evaluation

As shown in Table 2, on SVAMP and ASDIV, models using MD via CoT reasoning showed improvements of 0.5% and 1% over standard CoT distillation. Similarly, MD enhances the performance of SLMs compared to PoT distillation. Moreover, the model using MD with multi-path reasoning during inference leads the model to achieve optimal performance, attaining accuracies of 74.5% and 68.9% on SVAMP and ASDIV, with substantial improvements of 36.5% and 18.2%, respectively.

Method	SVAMP	GSM8K
LLaMA2-7B MD w/ ToT	85.8	54.7
LLaMA2-7B MD w/o ToT	84.5	53.8

Table 4: Performance of LLaMA2-7B with and without Tree of Thought (ToT) Integration

4.4.4 Additional Prompting Techniques

To further evaluate the generalization capabilities of our framework, we incorporated additional prompting techniques beyond CoT and PoT. Specifically, we integrated the Tree of Thought (ToT) method (Yao et al., 2024), which organizes the reasoning process into a tree structure. In this structure, each node represents a potential intermediate step, allowing the model to evaluate each step, discard unlikely branches, and explore multiple paths simultaneously. In mathematical reasoning tasks, where the solution space is relatively less complex, the improvement observed with ToT is primarily attributed to the ensemble of multiple reasoning paths. We utilized LLMs to solve mathematical problems using the ToT method by decomposing solutions into step-level formats and aggregating successful reasoning paths as the final solution rationales to help SLMs learn. The integration of ToT into the MD framework yielded promising results, as illustrated in Table 4.

5 Case Study

SLMs with MD can better master two types of step-by-step reasoning abilities. In single-path reasoning, it mitigates the shortcomings of CoT in solving large-scale and intensive numerical calculations, while also easing language comprehension challenges caused by insufficient knowledge for PoT. We present the actual output of SLMs for reasoning tasks, as shown in Figure 6. For the question labeled 35624, it is clear that correct answers are obtained using MD involving CoT or PoT. However, when using CoT or PoT distillation alone, errors occur in the reasoning process. Specifically, the error in CoT distillation is due to the inability to effectively compute 244*146, a common issue indicative of poor computational capability in CoT. We show more details in Appendix D.2.

6 Conclusion

In this paper, we introduce a novel framework MD that distills the reasoning paths of CoT and PoT from LLMs into SLMs. Our experimental results demonstrate that MD enhances the SLMs' single-path reasoning and multi-path reasoning, enabling SLMs to learn and adaptively activate step-by-step reasoning. Comparative analysis and experimental results show that our MD can effectively extract two different forms of capabilities, CoT and Pot from LLM, to improve the reasoning ability of SLMs.

Limitations

Our work has proven that the MD technology can improve the reasoning ability of small models. However, this technique has several limitations. First, our findings focus on reasoning tasks in English and have not been verified in a multilingual setting. Second, MD relies on the closed model, GPT-3.5-Turbo, which may introduce potential biases. Third, our technology uses generated intermediate reasoning steps to predict the final result, and the direct relationship between these steps and the final answer is still unproven. Caution should be taken when displaying MD to users.

Acknowledgments

This work was supported by the Zhejiang Provincial Natural Science Foundation of China under Grant No. LZ23F020009, the NSFC project (No. 62072399), MoE Engineering Research Center of Digital Library, China Research Centre on Data and Knowledge for Engineering Sciences and Technology, and the Fundamental Research Funds for the Central Universities (No. 226-2024-00170). We also express our sincere gratitude to anonymous reviewers for their invaluable feedback and constructive comments.

References

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, and Dmitry Lepikhin. 2023. Palm 2 technical report. *ArXiv*, abs/2305.10403.

Simran Arora, Avanika Narayan, Mayee F Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. 2022. Ask me anything: A simple strategy for prompting language models. *arXiv preprint arXiv:2210.02441*.

Jimmy Ba and Rich Caruana. 2014. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. 2022. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10925–10934.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2023. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2022. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*.

Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Tao He, Haotian Wang, Weihua Peng, Ming Liu, Bing Qin, and Ting Liu. 2023. A survey of chain of thought reasoning: Advances, frontiers and future. *arXiv preprint arXiv:2309.15402*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias

Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. *arXiv preprint arXiv:2301.12726*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10764–10799. PMLR.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Namgyu Ho, Laura Schmid, and Seyoung Yun. 2023. Large language models are reasoning teachers. In *61st Annual Meeting of the Association for Computational Linguistics, ACL 2023*, pages 14852–14882. Association for Computational Linguistics (ACL).

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander 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. *arXiv preprint arXiv:2305.02301*.

Yi Hu, Haotong Yang, Zhouchen Lin, and Muhan Zhang. 2023. Code prompting: a neural symbolic method for complex reasoning in large language models. *arXiv preprint arXiv:2305.18507*.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274.

Chengshu Li, Jacky Liang, Andy Zeng, Xinyun Chen, Karol Hausman, Dorsa Sadigh, Sergey Levine, Li Fei-Fei, Fei Xia, and Brian Ichter. 2023a. Chain of code: Reasoning with a language model-augmented code emulator. *arXiv preprint arXiv:2312.04474*.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023b. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. *arXiv preprint arXiv:2306.14050*.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2021. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and et al. 2023. Gpt-4 technical report.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Ryan Smith, Jason A Fries, Braden Hancock, and Stephen H Bach. 2022. Language models in the loop: Incorporating prompting into weak supervision. *arXiv preprint arXiv:2205.02318*.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? gpt-3 can help. *arXiv preprint arXiv:2108.13487*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Zhaoyang Wang, Shaohan Huang, Yuxuan Liu, Jiahai Wang, Minghui Song, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, et al. 2023. Democratizing reasoning ability: Tailored learning from large language model. *arXiv preprint arXiv:2310.13332*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Bohao Yang, Chen Tang, Kun Zhao, Chenghao Xiao, and Chenghua Lin. 2023. Effective distillation of table-based reasoning ability from llms. *arXiv preprint arXiv:2309.13182*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. 2023a. Large language model cascades with mixture of thoughts representations for cost-efficient reasoning. *arXiv preprint arXiv:2310.03094*.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023b. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Tianhua Zhang, Jiaxin Ge, Hongyin Luo, Yung-Sung Chuang, Mingye Gao, Yuan Gong, Xixin Wu, Yoon Kim, Helen Meng, and James Glass. 2023. Natural language embedded programs for hybrid language symbolic reasoning. *arXiv preprint arXiv:2309.10814*.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2024. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

Xuekai Zhu, Biqing Qi, Kaiyan Zhang, Xingwei Long, and Bowen Zhou. 2023. Pad: Program-aided distillation specializes large models in reasoning. *arXiv preprint arXiv:2305.13888*.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. Improving small language models' mathematical reasoning via mix thoughts distillation. *arXiv preprint arXiv:2401.11864*.

A Datasets

We provide detailed information about the datasets, including their sources and the initial release of the authors in the experiments.

- **SVAMP:** The dataset was originally released in (Patel et al., 2021) and made publicly available at `https://github.com/arkilpatel/SVAMP`. We obtained the dataset from `https://huggingface.co/datasets/ChilleD/SVAMP`.
- **GSM8K:** The dataset was originally released in (Cobbe et al., 2021) and made publicly available at `https://github.com/openai/grade-school-math`. We obtained the dataset from `https://huggingface.co/datasets/gsm8k`.
- **ASDIV:** The dataset was originally released in (Miao et al., 2021) and made publicly available at `https://github.com/chaochun/nlu-asdiv-dataset`. We obtained the dataset from `https://github.com/chaochun/nlu-asdiv-dataset/blob/master/dataset/ASDiv.xml`.
- **StrategyQA:** The dataset was originally released in (Geva et al., 2021) and made publicly available at `https://github.com/eladsegal/strategyqa`. We obtained the dataset from `https://github.com/eladsegal/strategyqa/tree/main/data/strategyqa`.

For ASDIV, we randomly selected 695 instances for the test set based on the question grade distribution in the training set. For StrategyQA, we use the dev set as the test set. The statistical information for the datasets is available in Table 5.

B Prompt Examples

For the datasets SVAMP, GSM8K, and ASDIV, the few-shot prompts are shown in Figure 8. For StrategyQA, they are displayed in Figure 9. We draw inspiration from (Li et al., 2023a) and add CoT as annotations. LLMs can perform better when generating outputs that include code structures, which in turn enhances the performance of SLMs. Additionally, we observe that PoT explicitly specifies return values (e.g., result = True), resulting in clearer model answers, whereas CoT may produce

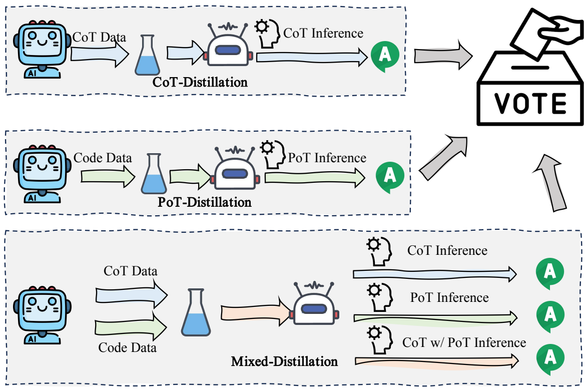


Figure 7: Framework diagram for different distillation methods.

responses like “so the answer is unknown.” Additionally, since LLMs are likely trained on extensive programming-related data, they are better equipped to induce reasoning capabilities in the comments of the generated code.

C Baselines

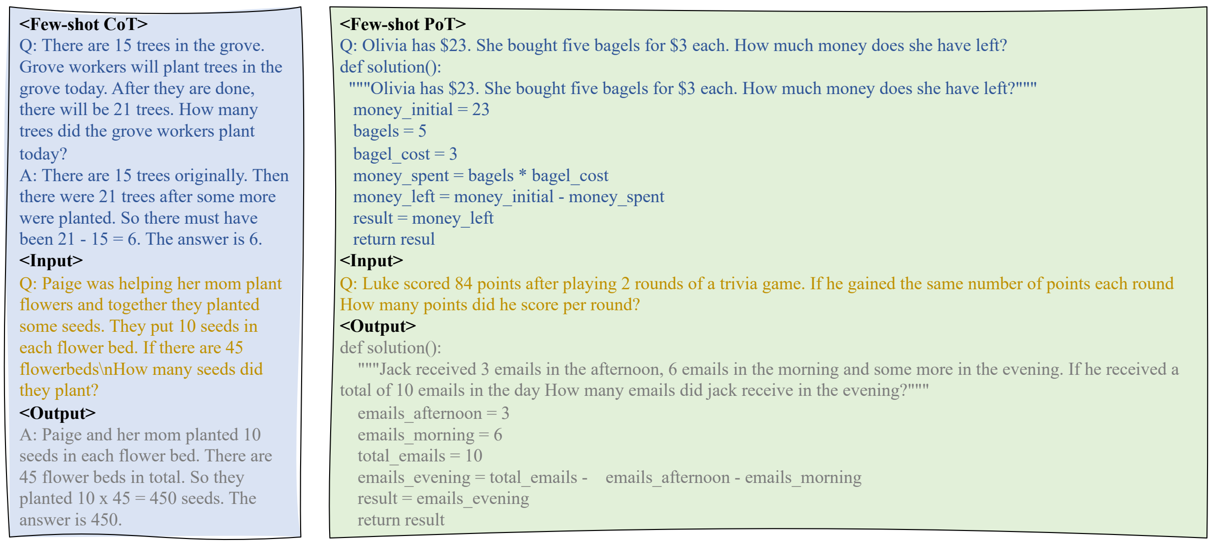
In the section, we show more details, including Closed-Source Models, Open-Source Models, Traditional Distillation, Label-Finetuning, Single-Path Distillation, and Single-path Reasoning strategies, aiming to provide a comprehensive comparison between MD and a series of existing methods.

Closed-Source Models Advanced Language Models, such as OpenAI’s GPT-4 (OpenAI, 2023) and GPT-3.5-Turbo, have achieved state-of-the-art results across various NLP tasks (Zhao et al., 2023; Kasneci et al., 2023; Chang et al., 2023; Hao et al., 2023). Trained on extensive datasets, these models comprehend complex language structures and generate text resembling human expression. Comparing them with closed-source models like GPT-4 is helpful in evaluating the reasoning gap between SLMs with MD and closed-source models.

Open-Source Models There are a series of models in the field of open-source NLP. Notably, LLaMA2 (Touvron et al., 2023), publicly released by Meta, demonstrates competitiveness and makes a significant contribution to academic research. CodeLlama, an adaptation of LLaMA, excels in diverse reasoning tasks, particularly showcasing proficiency in code-related capabilities (Roziere et al., 2023). WizardMath, fine-tuned based on LLaMA with enhanced instructions, effectively competes in mathematical reasoning tasks (Luo et al., 2023).

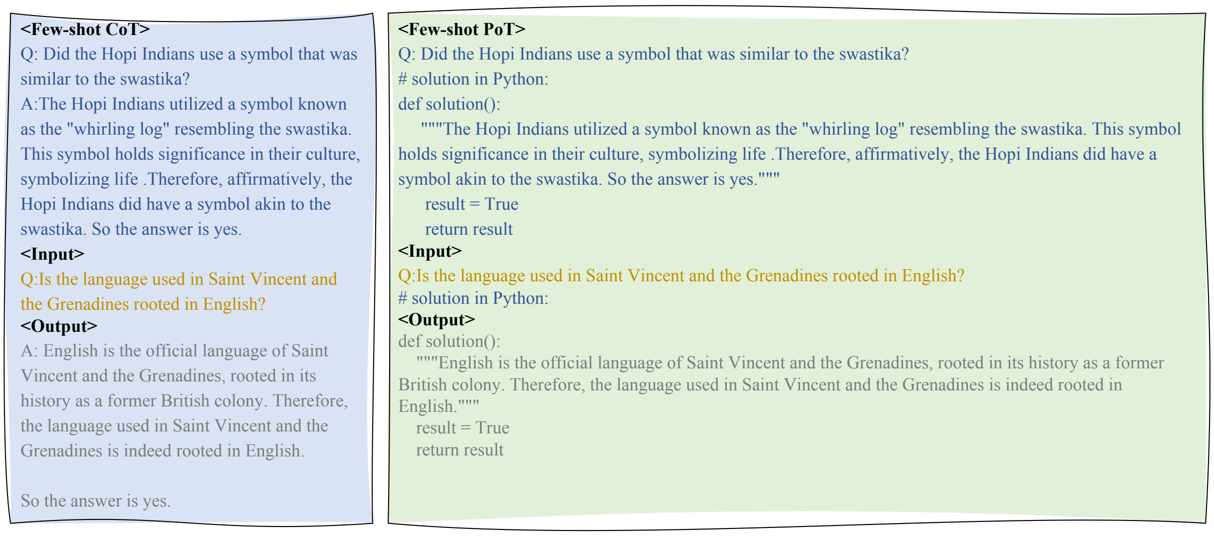
Dataset	Train set size	Test set size	Example
SVAMP (Cobbe et al., 2021)	800	300	Paige was helping her mom plant flowers and together they planted some seeds. They put 10 seeds in each flower bed. If there are 45 flowerbeds How many seeds did they plant?
GSM8K (Cobbe et al., 2021)	7473	1319	Janet\u2019s ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?
ASDIV (Miao et al., 2021)	1610	695	Edward spent 13.$Nowhehas$6. How much did Edward have before he spent his money?
StrategyQA (Geva et al., 2021)	2061	229	Will the Albany in Georgia reach a hundre thousand occupants before the one in New York?

Table 5: Details of dataset, including SVAMP, GSM8K, ASDIV, and StrategyQA.


<Few-shot CoT>
Q: There are 15 trees in the grove. Grove workers will plant trees in the grove today. After they are done, there will be 21 trees. How many trees did the grove workers plant today?
A: There are 15 trees originally. Then there were 21 trees after some more were planted. So there must have been 21 - 15 = 6. The answer is 6.
<Input>
Q: Paige was helping her mom plant flowers and together they planted some seeds. They put 10 seeds in each flower bed. If there are 45 flowerbeds\nHow many seeds did they plant?
<Output>
A: Paige and her mom planted 10 seeds in each flower bed. There are 45 flower beds in total. So they planted 10 x 45 = 450 seeds. The answer is 450.

<Few-shot PoT>
Q: Olivia has $23. She bought five bagels for $3 each. How much money does she have left?
```
def solution():
  """Olivia has $23. She bought five bagels for $3 each. How much money does she have left?"""
   money_initial = 23
   bagels = 5
   bagel_cost = 3
   money_spent = bagels * bagel_cost
   money_left = money_initial - money_spent
   result = money_left
   return resul
```
<Input>
Q: Luke scored 84 points after playing 2 rounds of a trivia game. If he gained the same number of points each round How many points did he score per round?
<Output>
```
def solution():
    """Jack received 3 emails in the afternoon, 6 emails in the morning and some more in the evening. If he received a total of 10 emails in the day How many emails did jack receive in the evening?"""
    emails_afternoon = 3
    emails_morning = 6
    total_emails = 10
    emails_evening = total_emails -    emails_afternoon - emails_morning
    result = emails_evening
    return result
```

Figure 8: Few-shot Template on SVAMP, GSM8K, ASDIV: extracting and distilling CoT and PoT from large Language Models to smaller models.


<Few-shot CoT>
Q: Did the Hopi Indians use a symbol that was similar to the swastika?
A:The Hopi Indians utilized a symbol known as the "whirling log" resembling the swastika. This symbol holds significance in their culture, symbolizing life .Therefore, affirmatively, the Hopi Indians did have a symbol akin to the swastika. So the answer is yes.
<Input>
Q:Is the language used in Saint Vincent and the Grenadines rooted in English?
<Output>
A: English is the official language of Saint Vincent and the Grenadines, rooted in its history as a former British colony. Therefore, the language used in Saint Vincent and the Grenadines is indeed rooted in English.

So the answer is yes.

<Few-shot PoT>
Q: Did the Hopi Indians use a symbol that was similar to the swastika?
```
# solution in Python:
def solution():
    """The Hopi Indians utilized a symbol known as the "whirling log" resembling the swastika. This symbol holds significance in their culture, symbolizing life .Therefore, affirmatively, the Hopi Indians did have a symbol akin to the swastika. So the answer is yes."""
    result = True
    return result
```
<Input>
Q:Is the language used in Saint Vincent and the Grenadines rooted in English?
```
# solution in Python:
```
<Output>
```
def solution():
   """English is the official language of Saint Vincent and the Grenadines, rooted in its history as a former British colony. Therefore, the language used in Saint Vincent and the Grenadines is indeed rooted in English."""
   result = True
   return result
```

Figure 9: Few-shot Template on StrategyQA: extracting and distilling CoT and PoT from large Language Models to smaller models.

Traditional Distillation Knowledge distillation (Buciluă et al., 2006; Ba and Caruana, 2014; Hinton et al., 2015; Beyer et al., 2022; Fu et al., 2023) has demonstrated effectiveness in improving SLMs. Fu et al. (2023) distills LLMs' multi-step reasoning into SLMs for better mathematical reasoning. Shridhar et al. (2023) improves mathematical skills by distilling LLMs' problem decomposition abilities. Wang et al. (2023) and Hsieh et al. (2023) focus on distilling reflective thinking and using LLM-generated CoT as supervisory signals, respectively.

Label-Finetuning Label fine-tuning is a supervised learning method, in which the trained model is adjusted to do better on a specific task. It uses a small set of labeled data to adjust the model's settings, which was initially trained on a broad dataset. The main goal is to make the pre-trained model work better in the tasks. We use the training set questions and labels for model training to establish this benchmark.

Single-Path Distillation and Reasoning Single-Path Distillation involves distilling SLMs using data in a single format, including the CoT-distill model, PoT-distill model, and a unified ensemble from two individual distilled models. Single-path reasoning refers to selecting only one capability, either CoT or PoT for inference. '+CoT' indicates CoT inference on the task, and '+PoT' indicates PoT inference on the task. As shown in Figure 7, result selection is done using consensus via majority vote (Wang et al., 2022) during inference.

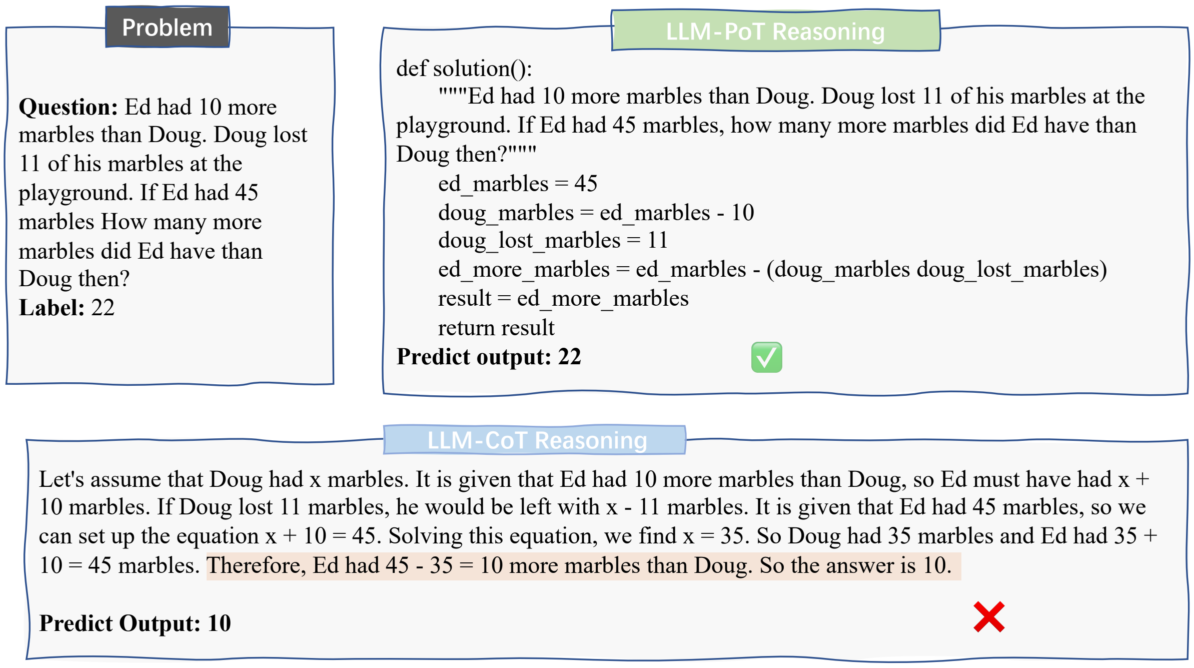
D Case Analysis

D.1 Case Analysis in LLM

Despite PoT demonstrating superiority over CoT in LLMs (Chen et al., 2022; Gao et al., 2023), recent work has identified distinct weaknesses for CoT and PoT (Yue et al., 2023b). As shown in Figure 10, CoT overlooks the statement "Doug lost 11 of his marbles at the playground," leading to a reasoning error. Similarly, in Figure 11, PoT misinterprets the question "How many more crunches than push-ups did Zachary do?" resulting in the reasoning error. Perceptions of problem-solving differ between CoT and PoT, which can stem from PoT's generation of structured code and CoT's production of unstructured text. PoT excels in large-scale and intensive numerical calculations, whereas CoT demonstrates proficiency in language comprehension, geometry, algebra, fraction calculations, and equation comparisons. Thus, combining multiple step-by-step reasoning may compensate for their weaknesses.

D.2 Case Analysis in smaller models

We propose more case studies, focusing on SLMs. As shown in Figure 12, our experimental results reveal that CoT Distillation encounters challenges in handling complex numbers, such as 77*221 and 62*183. Conversely, PoT Distillation struggles with understanding problems involving multiple terms, such as when irrelevant conditions are added, like "if he sold 70 cakes and 88 pastries", leading to error reasoning steps, and an inability to understand statements like 'Allan bought 3 more balloons' as shown in Figure 13. However, MD can effectively deal with these shortcomings, thus improving the results as shown in Figure 14.



Problem

Question: Ed had 10 more marbles than Doug. Doug lost 11 of his marbles at the playground. If Ed had 45 marbles How many more marbles did Ed have than Doug then?
Label: 22

LLM-PoT Reasoning

```
def solution():
    """Ed had 10 more marbles than Doug. Doug lost 11 of his marbles at the playground. If Ed had 45 marbles, how many more marbles did Ed have than Doug then?"""
    ed_marbles = 45
    doug_marbles = ed_marbles - 10
    doug_lost_marbles = 11
    ed_more_marbles = ed_marbles - (doug_marbles doug_lost_marbles)
    result = ed_more_marbles
    return result
```

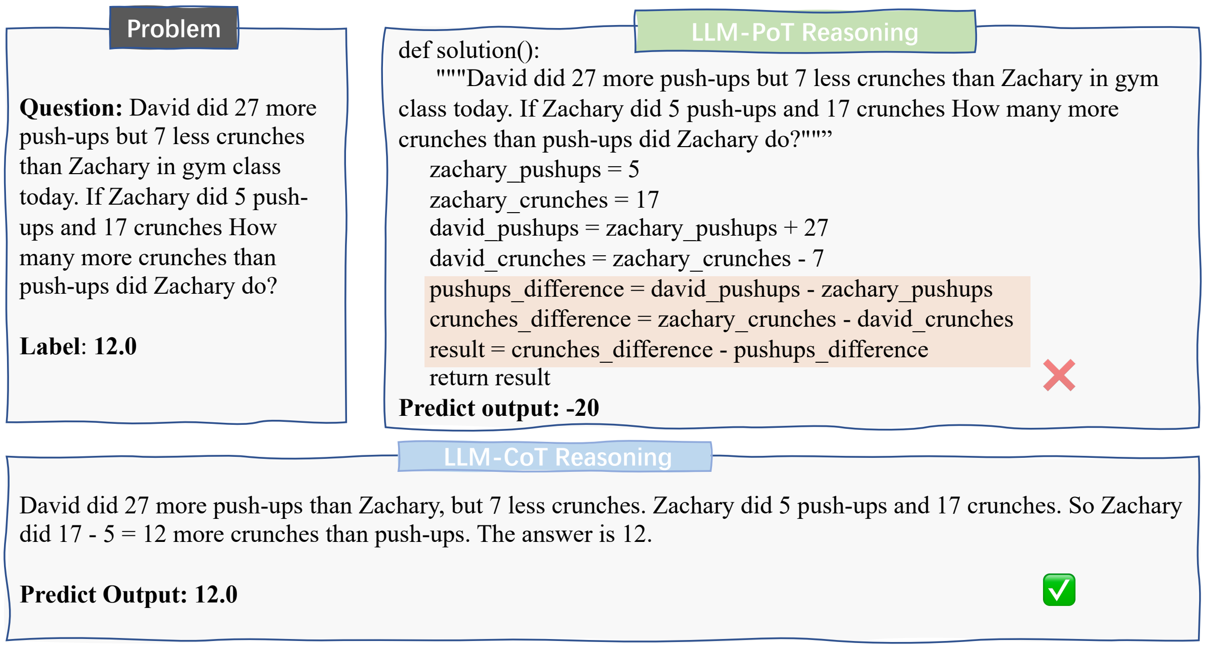
Predict output: 22 ✓

LLM-CoT Reasoning

Let's assume that Doug had x marbles. It is given that Ed had 10 more marbles than Doug, so Ed must have had x + 10 marbles. If Doug lost 11 marbles, he would be left with x - 11 marbles. It is given that Ed had 45 marbles, so we can set up the equation x + 10 = 45. Solving this equation, we find x = 35. So Doug had 35 marbles and Ed had 35 + 10 = 45 marbles. Therefore, Ed had 45 - 35 = 10 more marbles than Doug. So the answer is 10.

Predict Output: 10 ✗

Figure 10: PoT yields the correct solution, whereas CoT falls short in GPT-3.5-Turbo.



Problem

Question: David did 27 more push-ups but 7 less crunches than Zachary in gym class today. If Zachary did 5 push-ups and 17 crunches How many more crunches than push-ups did Zachary do?

Label: **12.0**

LLM-PoT Reasoning

```
def solution():
    """David did 27 more push-ups but 7 less crunches than Zachary in gym class today. If Zachary did 5 push-ups and 17 crunches How many more crunches than push-ups did Zachary do?"""”
    zachary_pushups = 5
    zachary_crunches = 17
    david_pushups = zachary_pushups + 27
    david_crunches = zachary_crunches - 7
    pushups_difference = david_pushups - zachary_pushups
    crunches_difference = zachary_crunches - david_crunches
    result = crunches_difference - pushups_difference
    return result
```

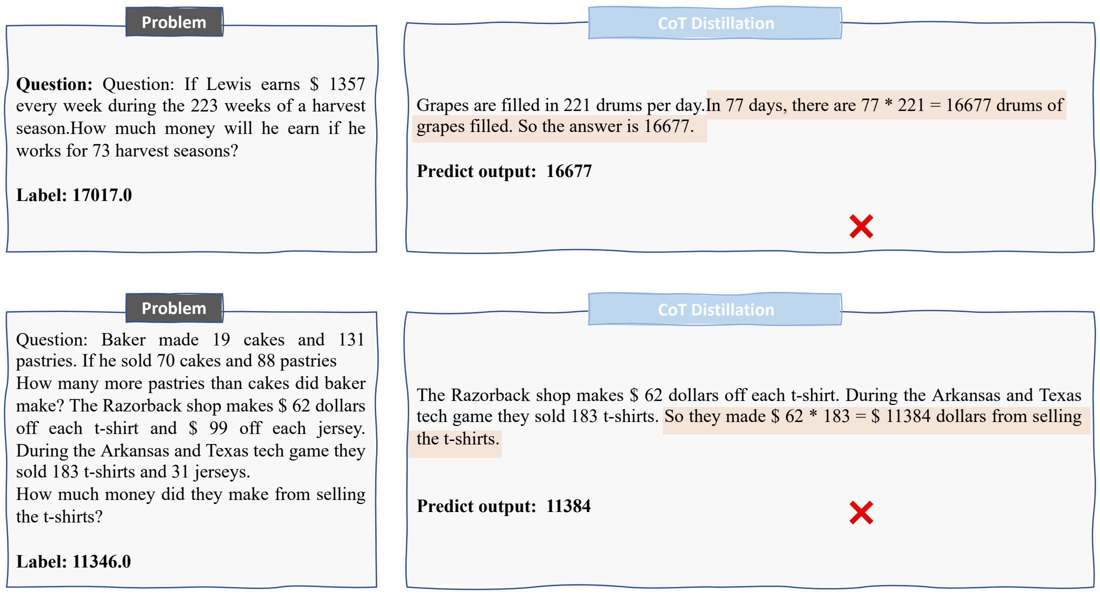
Predict output: -20 ✗

LLM-CoT Reasoning

David did 27 more push-ups than Zachary, but 7 less crunches. Zachary did 5 push-ups and 17 crunches. So Zachary did 17 - 5 = 12 more crunches than push-ups. The answer is 12.

Predict Output: 12.0 ✓

Figure 11: CoT yields the correct solution, whereas PoT falls short in GPT-3.5-Turbo.


Problem
CoT Distillation

Question: Question: If Lewis earns $ 1357 every week during the 223 weeks of a harvest season.How much money will he earn if he works for 73 harvest seasons?

Label: 17017.0

Grapes are filled in 221 drums per day.In 77 days, there are 77 * 221 = 16677 drums of grapes filled. So the answer is 16677.

Predict output: 16677

✗

Problem
CoT Distillation

Question: Baker made 19 cakes and 131 pastries. If he sold 70 cakes and 88 pastries How many more pastries than cakes did baker make? The Razorback shop makes $ 62 dollars off each t-shirt and $ 99 off each jersey. During the Arkansas and Texas tech game they sold 183 t-shirts and 31 jerseys.
How much money did they make from selling the t-shirts?

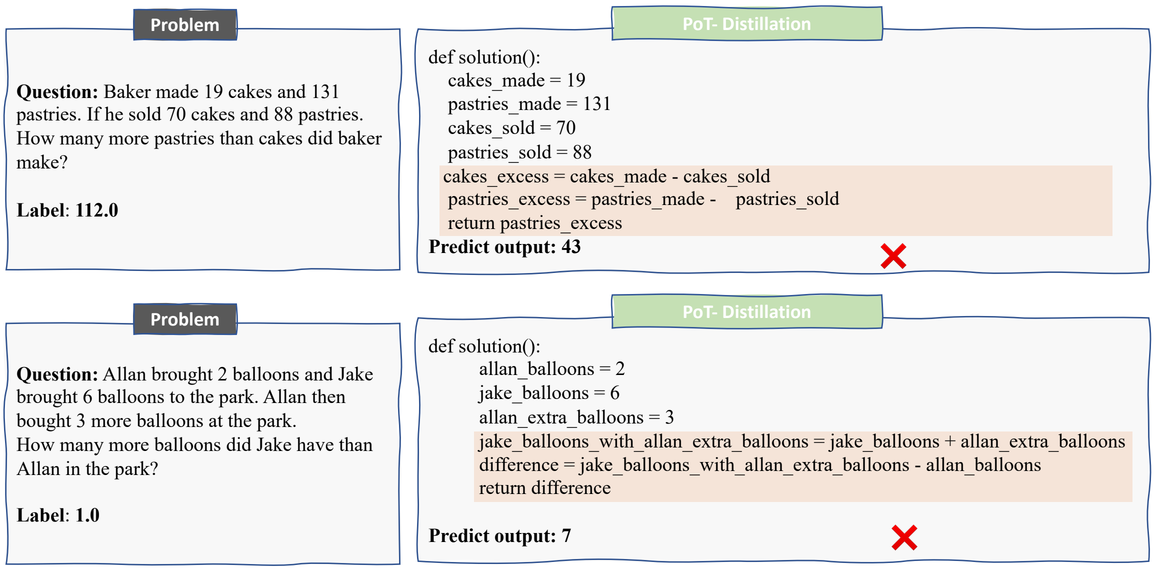
Label: 11346.0

The Razorback shop makes $ 62 dollars off each t-shirt. During the Arkansas and Texas tech game they sold 183 t-shirts. So they made $ 62 * 183 = $ 11384 dollars from selling the t-shirts.

Predict output: 11384

✗

Figure 12: Error cases based LLaMA2-7B in CoT Distillation.


Problem
PoT- Distillation

Question: Baker made 19 cakes and 131 pastries. If he sold 70 cakes and 88 pastries. How many more pastries than cakes did baker make?

Label: **112.0**

```
def solution():
    cakes_made = 19
    pastries_made = 131
    cakes_sold = 70
    pastries_sold = 88
    cakes_excess = cakes_made - cakes_sold
    pastries_excess = pastries_made -    pastries_sold
    return pastries_excess
```

Predict output: 43

✗

Problem
PoT- Distillation

Question: Allan brought 2 balloons and Jake brought 6 balloons to the park. Allan then bought 3 more balloons at the park.
How many more balloons did Jake have than Allan in the park?

Label: **1.0**

```
def solution():
        allan_balloons = 2
        jake_balloons = 6
        allan_extra_balloons = 3
        jake_balloons_with_allan_extra_balloons = jake_balloons + allan_extra_balloons
        difference = jake_balloons_with_allan_extra_balloons - allan_balloons
        return difference
```

Predict output: 7

✗

Figure 13: Error cases based LLaMA2-7B in PoT Distillation.

Problem

Question: Being his favorite, he saved checking on the grapevines for his last stop. He was told by 235 of the pickers that they fill 100 drums of raspberries per day and 221 drums of grapes per day.
How many drums of grapes would be filled in 77 days?

Label: **17017.0**

CoT -Reasoning

Grapes are filled in 221 drums per day. In 77 days, there are 77 * 221 = 17017 drums of grapes filled. So the answer is 17017 .

Predict output: 17017

✅

Problem

Question: Baker made 19 cakes and 131 pastries. If he sold 70 cakes and 88 pastries
How many more pastries than cakes did baker make? The Razorback shop makes $ 62 dollars off each t-shirt and $ 99 off each jersey. During the Arkansas and Texas tech game they sold 183 t-shirts and 31 jerseys.
How much money did they make from selling the t-shirts?

Label: **11346.0**

CoT -Reasoning

The Razorback shop makes $ 62 dollars off each t-shirt. During the Arkansas and Texas tech game they sold 183 t-shirts. So they made $ 62 * 183 = $ 11346 dollars from selling the t-shirts.

Predict output: 11346

✅

Problem

Question: Baker made 19 Cakes and 131 pastries. If he Sold 70 cakes and 88 pastries. How many more pastries than Cakes did baker make?

Label: **112.0**

PoT- Reasoning

```
def solution():
     cakes_made= 19
    pastries_made= 131
    cakes_sold= 70
    pastries_sold= 88
    pastries_excess= pastries_made-cakes_made
    return pastries_excess
```

✅

Predict output: 112.0

Problem

Question: Allan brought 2 balloons and Jake brought 6 balloons to the park. Allan then bought 3 more balloons at the park.
How many more balloons did Jake have than Allan in the park?

Label: **1.0**

PoT- Reasoning

```
def solution():
        allan_balloons = 2
        jake_balloons = 6
        allan_extra_balloons = 3
        allan_with_allan_extra_balloons = allan_balloons + allan_extra_balloons
        difference = jake_balloons_with_allan_extra_balloons - allan_balloons
        return difference
```

Predict output: 1

✅

Figure 14: Correct cases based LLaMA2-7B in Mixed Distillation.

Type	Sample
1	##Input: The Razorback t-shirt shop sells each t-shirt for $201 dollars. During the Arkansas and Texas Tech game, they increased the prices by $217 per t-shirt and sold 14 t-shirts. How much money did they make from selling the t-shirts? ##Answer: 5852 (**large-scale calculation**) ##CoT: The t-shirts are originally sold for $201 each. They increased the prices by $217 per t-shirt and sold 14 t-shirts. So the total amount of money they made is $201 + $217 * 14 = $201 + $3038 = $3239. The answer is $3239. ##PoT: tshirt_price = 201 price_increase = 217 num_tshirts_sold = 14 total_money_made = (tshirt_price + price_increase) * num_tshirts_sold result = total_money_made
	##Input: If you buy 2 packs of 500 mL milk, it will cost $2.50. If you buy them individually, they will cost $1.30 each. How much is your total savings from buying ten sets of 2 packs of 500 mL milk? ##Answer: 1 (**intensive calculation**) ##CoT: A: If 2 packs of 500 mL milk cost $2.50, then each pack costs $2.50 / 2 = $1.25. If each pack costs $1.25, then each individual 500 mL milk costs $1.30. So the savings from buying 2 packs instead of buying them individually is $1.30 - $1.25 = $0.05. For 10 sets of 2 packs, the total savings would be $0.05 x 10 = $0.50. The answer is $0.50. ##PoT: cost_of_2_packs = 2.50 cost_of_individual_pack = 1.30 num_packs = 2 * 10 total_cost_individual = cost_of_individual_pack * num_packs total_cost_2_packs = cost_of_2_packs * 10 total_savings = total_cost_individual - total_cost_2_packs result = total_savings
2	##Input: Paul had 50 books. After buying some in a garage sale he had 151 left. How many books did he buy? ##Answer: 101 (**language comprehension**) ##CoT: Paul started with 50 books. After buying some in a garage sale, he had 151 left. So he bought 151 - 50 = 101 books. The answer is 101. ##PoT: books_initial = 50 books_after = 151 books_bought = books_initial - books_after result = books_bought
	##Input: Tori went on a two-day fishing trip. On the first day, she caught 5 catfish and 6 trout. On the second day, she caught 10 catfish and 12 trout. On which day of the fishing trip did Tori catch a higher ratio of catfish to trout? ##Answer: neither day (**fraction calculations, and equation comparisons**) ##CoT: On the first day, Tori caught 5 catfish and 6 trout, so the ratio of catfish to trout is 5/6. On the second day, she caught 10 catfish and 12 trout, so the ratio is 10/12. To compare the ratios, we need to find the equivalent fractions. 5/6 is equivalent to 10/12 because if we multiply the numerator and denominator of 5/6 by 2, we get 10/12. Therefore, Tori caught the same ratio of catfish to trout on both days. The answer is neither day. ##PoT: catfish_day1 = 5 trout_day1 = 6 catfish_day2 = 10 trout_day2 = 12 ratio_day1 = catfish_day1 / trout_day1 ratio_day2 = catfish_day2 / trout_day2 if ratio_day1 = = ratio_day2: result = "neither day" else: result = "second day" return result
	##Input: In a triangle with angles A, B, and C, A is 60 degrees, and B is two times as big as C. Calculate the angle of B. ##Answer: 80 (**geometry and algebra**) ##CoT: Angle B is two times as big as angle C. Since angle A is 60 degrees, angle B + angle C + angle A = 180 degrees in a triangle. Angle B + angle C + 60 = 180. We can solve for angle C first. Angle B = 2 * angle C. Substituting this into the equation, 2 * angle C + angle C + 60 = 180. Combining like terms, 3 * angle C + 60 = 180. Subtracting 60 from both sides, 3 * angle C = 120. Dividing both sides by 3, angle C = 40. Since angle B is two times as big as angle C, angle B = 2 * 40 = 80. The answer is 80 degrees. ##PoT: angle_A = 60 angle_C = 1 angle_B = 2 * angle_C result = angle_B

Table 6: Cases of Type 1 and 2 in mixed thought data.