Keypoint-based Progressive Chain-of-Thought Distillation for LLMs

Kaituo Feng [1] **Changsheng Li** [1] **Xiaolu Zhang** [2] **Jun Zhou** [2] **Ye Yuan** [1] **Guoren Wang** [1 3]

Abstract

Chain-of-thought distillation is a powerful technique for transferring reasoning abilities from large language models (LLMs) to smaller student models. Previous methods typically require the student to mimic the step-by-step rationale produced by LLMs, often facing the following challenges: (i) Tokens within a rationale vary in significance, and treating them equally may fail to accurately mimic keypoint tokens, leading to reasoning errors. (ii) They usually distill knowledge by consistently predicting all the steps in a rationale, which falls short in distinguishing the learning order of step generation. This diverges from the human cognitive progression of starting with easy tasks and advancing to harder ones, resulting in sub-optimal outcomes. To this end, we propose a unified framework, called KPOD, to address these issues. Specifically, we propose a token weighting module utilizing mask learning to encourage accurate mimicry of keypoint tokens by the student during distillation. Besides, we develop an in-rationale progressive distillation strategy, starting with training the student to generate the final reasoning steps and gradually extending to cover the entire rationale. To accomplish this, a weighted token generation loss is proposed to assess step reasoning difficulty, and a value function is devised to schedule the progressive distillation by considering both step difficulty and question diversity. Extensive experiments on four reasoning benchmarks illustrate our KPOD outperforms previous methods by a large margin.

1. Introduction

Large language models (LLMs) have demonstrated remarkable reasoning capabilities via chain-of-thought (CoT) prompting (e.g., "Let's think step-by-step"), which prompts LLMs to generate a step-by-step rationale to help reasoning (Kojima et al., 2022; Wei et al., 2022). However, such abilities usually emerge in extremely large models, especially those with over 100 billion parameters (Fu et al., 2023; Hoffmann et al., 2022) , such as 175B GPT-3 (Brown et al., 2020) and 540B PaLM (Chowdhery et al., 2023). The substantial amount of parameters unavoidably leads to high inference costs and makes it challenging to deploy LLMs in environments with limited computational resources (Hsieh et al., 2023). To tackle with this, a recent surge of works, known as CoT distillation, has arisen as a promising avenue to distill reasoning capabilities of LLMs to smaller student models (Li et al., 2023; Wang et al., 2023b; Fu et al., 2023). The core idea of these methods is to require the student model to mimic the step-by-step rationale generated by LLMs in response to a question.

However, current CoT distillation methods often encounter the following two issues: First, in a rationale, each token carries different levels of importance in the reasoning process. Certain keypoint tokens play a pivotal role in reasoning, while other tokens are of less importance or even irrelevant to the reasoning process. For instance, consider a step in a rationale: "*Next, we just need to simply add up the calories from the lettuce and cucumber: 30 + 80 = 110*". Here, terms like "*just*", "*simply*" are reasoning-irrelevant, whereas the calculation "*30 + 80 = 110*" stands out as the keypoint for reasoning. The reasoning-irrelevant tokens can be replaced without negative effects, but even a slight deviation from the keypoint token could result in errors in reasoning. Therefore, it's crucial for the student model to focus on the precise mimicry of these keypoint tokens. Nevertheless, previous CoT distillation methods usually treat all tokens equally during distillation (Li et al., 2023; Wang et al., 2023b).

The second issue stems from the fact that previous approaches usually demand the student model to consistently learn all the steps in a rationale throughout the distillation process, without distinguishing the learning order of step generation. This distillation strategy diverges from the human cognitive pattern that progresses from easier tasks to more challenging ones. This deviation might lead to sub-optimal outcomes. In the process of human or biological agent learning, ability acquisition doesn't simply stem from random tasks (Molina & Jouen, 1998). Instead, there is an

[1]Beijing Institute of Technology [2]Ant Group [3]Hebei Province Key Laboratory of Big Data Science and Intelligent Technology. Correspondence to: Changsheng Li <lcs@bit.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

organized progression from easy tasks to hard tasks for them to acquire capabilities, especially for complex skills such as reasoning (Peterson, 2004; Krueger & Dayan, 2009; Benoit et al., 2013). In the field of machine learning, this ordered learning paradigm is regarded as curriculum learning (Bengio et al., 2009). Inspired by this, we intend to develop a progressive CoT distillation strategy to facilitate the student model acquire reasoning ability from easy to hard. However, directly applying previous curriculum learning strategies to CoT distillation could be inferior because of the following two reasons: (i) They overlook the step-by-step reasoning nature where each reasoning step within a rationale may possess varying reasoning difficulty, resulting in sub-optimal difficulty assessment. (ii) As aforementioned, a step in the rationale might contain many tokens that are not crucial to the reasoning process. When assessing the difficulty of step generation, it may be dominated by these inessential tokens, thereby inaccurately reflecting the challenge of obtaining the expected outcome for a reasoning step.

In this paper, we propose Keypoint-based Progressive CoT Distillation for LLMs dubbed KPOD, with the goal of addressing the above two issues in a unified framework. First, we propose a rationale token weighting module to determine the token significance for distillation. It learns to generate masks for inessential tokens to the reasoning process via two distinctive loss functions: An answer prediction loss is introduced to encourage the module to utilize the question with the masked rationale to derive the answer, while a mask ratio loss is designed to maximize the ratio of masked tokens in the rationale. By doing so, the obtained probability of not masking a token can serve as an indicator of its significance weight. Second, we develop an in-rationale progressive distillation strategy that orders the learning sequence from easy reasoning to hard reasoning within the rationale of a question. This strategy begins by training the student model to generate the last few reasoning steps of the rationale, given the question with preceding steps of this rationale as input. Subsequently, it progressively extends to generate the entire rationale using only the question as input. To precisely assess each step's reasoning difficulty, we propose a token generation loss based on the derived token significance, aiming to eliminate the negative effects of reasoning-irrelevant tokens. Finally, we design a value function to dynamically determine the number of steps taken as input at each stage, thereby automatically adjusting their learning difficulty. Meanwhile, we leverage the value function to select diverse questions, so as to prevent over-fitting (Jiang et al., 2014; Liang et al., 2021).

Our contributions can be summarized as: 1) We propose a general and principled framework for CoT distillation, which simultaneously considers token significance and reasoning difficulty within a rationale during distillation. 2) We design a rationale token weighting module through mask learning to determine the token significance for reasoning. This allows the student to concentrate more on keypoint tokens. 3) We devise an in-rationale progressive CoT distillation strategy to schedule the learning order of reasoning steps within a rationale. This enables the student to progressively acquire reasoning abilities in an easy-to-hard manner. 4) Extensive experiments on four reasoning benchmarks validate the effectiveness of our KPOD, showcasing significant performance improvements compared to baselines.

2. Related Works

Chain-of-Thought Reasoning. The concept of employing step-by-step language rationales to aid in solving reasoning problems can be traced back to pioneering works (Ling et al., 2017). Inspired by this, chain-of-thought prompting (Wei et al., 2022) has been proposed to enable LLMs to generate intermediate reasoning steps that contribute to the final answer via few-shot CoT demonstrations. This prompting approach has illustrated remarkable performance gain for LLMs in reasoning related tasks (Zhang et al., 2022; Wang et al., 2023a). In addition, researchers find that LLMs can also obtain impressive reasoning performance by zero-shot CoT (Kojima et al., 2022) without task-related demonstrations. This is achieved by only using a single sentence "Let's think step by step" for prompting. Recently, a number of CoT prompting methods have demonstrated effectiveness in enhancing the reasoning performance of LLMs (Diao et al., 2023; Yang et al., 2023), such as SC-CoT (Wang et al., 2022), Auto-CoT (Zhang et al., 2022), Multimodal-CoT (Zhang et al., 2023), etc. However, the emergence of CoT reasoning capabilities in LLMs typically requires models with more than 100 billion parameters (Wei et al., 2022; Fu et al., 2023), making it resource-consuming for deployment.

CoT Distillation. Knowledge distillation has been widely studied for model compression across various fields (Magister et al., 2023; Feng et al., 2024). Recently, CoT Distillation has emerged as a promising avenue to transfer the step-by-step reasoning capabilities of LLMs to smaller student models (Hsieh et al., 2023; Ho et al., 2023). The key idea of CoT distillation is to make the student model mimic the step-by-step rationale generated by LLMs in response to a question. In this context, the rationale can be interpreted as the LLMs' explanation of how to derive the final answer of a question, akin to the soft label used in conventional knowledge distillation (Hinton et al., 2015; Feng et al., 2022). The representative works of CoT distillation include: SCoTD (Li et al., 2023) introduces a symbolic CoT distillation method that enables smaller models to self-rationalize for reasoning via learning rationales from LLMs. Specialized KD (Fu et al., 2023) is proposed to train a small language model specialized for reasoning in four distinct in-context scenarios. MCC-KD (Chen et al., 2023) adopts diverse rationales for

distillation and attempts to ensure their consistency. SCOTT (Wang et al., 2023b) designs a faithful CoT distillation strategy to make the student reason faithfully via counterfactual training. However, these methods fail to consider the reasonable learning order of the reasoning steps within a rationale, leading to sub-optimal performance.

Curriculum Learning. Early researches in cognitive science emphasize the significance of the easy-to-hard learning pattern to acquire knowledge (Elman, 1993). Inspired by this, the pioneer work (Bengio et al., 2009) introduces the concept of curriculum learning (CL) to the machine learning field by gradually including samples from easy to hard for training. In recent years, a variety of CL methods have been proposed to enhance the model performance (Kong et al., 2021; Wang et al., 2021). For instance, Adaptive CL (Kong et al., 2021) proposes to utilize the loss of the model to dynamically adjust the difficulty score of each sample. SPL (Wan et al., 2020) introduces the curriculum learning to the neural machine translation domain via introducing the token-level and sentence-level confidence score. ICL (Jia et al., 2023) devises a curriculum learning method that organizes the curriculum within the token sequence of a sample for natural language generation tasks. However, as aforementioned, applying these CL methods directly to CoT distillation could yield inferior performance.

3. Proposed Method

3.1. Preliminaries and Problem Setting

The goal of CoT distillation is to transfer the reasoning capability of large language models (LLMs) to smaller student models via distilling the rationales produced by LLMs. We denote the dataset as $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}$, where $x^{(i)}$ is the i-th reasoning question and $y^{(i)}$ is the corresponding answer. Following previous CoT distillation works (Ho et al., 2023; Chen et al., 2023) , we adopt zero-shot CoT (Kojima et al., 2022) to prompt the teacher LLMs to generate step-by-step rationale $r^{(i)}$ for each question $x^{(i)}$. The reasoning template takes the following format: "*Q:* $<x^{(i)}>$ *A:* $<p>$ $<r^{(i)}>$ *Therefore, the answer is* $<y^{(i)}>$", where $<p>$ is the zero-shot CoT prompt such as "Let's think step by step". Then, the student is trained to generate the concatenated sequence of rationale tokens $r^{(i)}$ and answer tokens $y^{(i)}$, given the question $x^{(i)}$ as input. The standard negative log-likelihood loss for training the student model can be formulated as:

$$\begin{aligned}\mathcal{L} = &-\sum_i \sum_j \log P(r_j^{(i)} | r_{<j}^{(i)}, x^{(i)}; \theta^s) \\ &-\sum_i \sum_j \log P(y_j^{(i)} | y_{<j}^{(i)}, r^{(i)}, x^{(i)}; \theta^s), \end{aligned} \tag{1}$$

where $r_j^{(i)}$ and $y_j^{(i)}$ represent the j-th token in the rationale sequence $r^{(i)}$ and the answer sequence $y^{(i)}$, respectively. θ^s denotes the parameters of the student model. The first term of Eq.(1) enables the student to mimic the rationale produced by LLMs, while the second term aims to train the student to output the final answer based on the rationale. By minimizing this loss, the student model can learn to generate the step-by-step rationale for deriving the final answer.

3.2. Framework Overview

As aforementioned, there are two key issues for CoT distillation methods: (i) Equally treating each token for distillation may make the student fail to mimic keypoint tokens accurately, leading to reasoning errors. (ii) Distilling the steps within a rationale without explicitly considering the learning order of step generation might lead to sub-optimal outcomes. To tackle these two issues, we propose a new CoT distillation framework KPOD, as illustrated in Figure 1. Our framework mainly consists of two components: a rationale token weighting component based on mask learning is proposed to determine the token significance for distillation. This encourages the student to faithfully replicate the crucial keypoint tokens; A progressive distillation component within the rationale is designed to establish a structured learning order for the reasoning steps. This guides the student model to progressively develop its reasoning abilities from simpler to more complex tasks, aligning with the proficiency of teacher LLMs. It's worth noting that the obtained token significance weight fulfills two distinct functions in our framework: firstly, it encourages precise mimicry of keypoint tokens during distillation, and secondly, it mitigates the negative effects of inessential tokens when assessing step difficulty. Next, we will primarily delve into the detailed introduction of the two components in our framework.

3.3. Rationale Token Weighting

In this section, we introduce our rationale token weighting module, which determines the significance of each token via learning to mask reasoning-irrelevant token.

Weight Generation. First, we intend to generate distinct significance weights for different tokens by leveraging their embeddings. This facilitates the estimation of their importance according to their characteristics. To achieve this, we feed the rationale tokens into a pre-trained input embedding layer, followed by a self-attention layer to encode in-context information. This process is formulated as:

$$e^{(i)} = \mathrm{Att}(\mathrm{Emb}(r^{(i)})), \tag{2}$$

where Emb and Att denote the input embedding layer and the self-attention layer, respectively. $e^{(i)}$ is the embedding matrix containing the embeddings for each token in $r^{(i)}$. Subsequently, the embedding $e_j^{(i)}$ of each token is fed into a weight generator, producing the significance weight as:

$$w_j^{(i)} = \sigma(f_w(e_j^{(i)})), \tag{3}$$

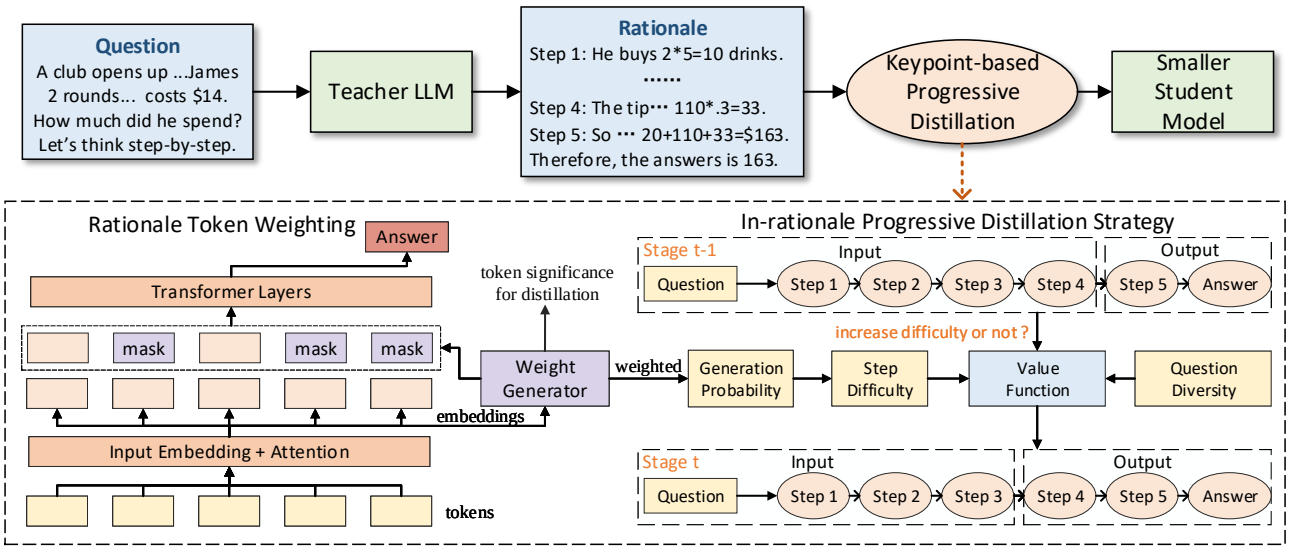


Figure 1. An illustration of our KPOD framework. KPOD first determines the keypoint tokens for distillation through designing a rationale token weighting module based on mask learning. Then, an in-rationale progressive distillation strategy is devised to organize the learning order within rationale, so as to enable the student to acquire the reasoning capabilities in an easy-to-hard manner.

where $w_j^{(i)}$ is the probability of the j-th token not being masked, serving as an indicator of its significance level. f_w is the weight generator and σ is the sigmoid activation function (Narayan, 1997). In this paper, we employ a simple two-layer MLP as the weight generator.

Reasoning-irrelevant Mask Learning. To optimize the weight generator, we formulate two loss functions: an answer prediction loss that encourages the module to utilize the question with the masked rationale for answer derivation, and a mask ratio loss aiming to maximize the ratio of masked tokens in the rationale. This allows the weight generator to generate low values of $w_j^{(i)}$ for tokens irrelevant to reasoning and high values for keypoint tokens. Next, we will introduce these two losses in detail.

Firstly, considering that sampling the discrete mask policy $m_j^{(i)} \in \{0, 1\}$ from the distribution of $w_j^{(i)}$ is non-differentiable, we adopt the Gumbel-Softmax sampling (Jang et al., 2016) to avoid this issue:

$$m_j^{(i)} = \mathrm{GumbelSoftmax}(w_j^{(i)}), \tag{4}$$

where $\mathrm{GumbelSoftmax}$ represents the Gumbel-Softmax sampling (Jang et al., 2016). $m_j^{(i)} = 0$ denotes that the j-th token is masked, while $m_j^{(i)} = 1$ denotes that the j-th token is not masked. By applying the mask $m_j^{(i)}$ to each token $r_j^{(i)}$ in rationale $r^{(i)}$, we can obtain the masked rationale, denoted as $r[m]^{(i)}$.

Then, we input $r[m]^{(i)}$ into the transformer layers, with the goal of obtaining the correct answer by using the masked rationale. Here we initialize the transformer layers using the pre-trained FlanT5-Large (Chung et al., 2022). Considering that certain steps of the rationale sometimes contain the reasoning results of previous steps, shortcuts may be taken for the answer prediction via neglecting previous steps. To eliminate this phenomenon, we expect the transformer to predict the answer based on the question with any prefix of the masked rationale. The answer prediction loss $\mathcal{L}_p$ for question $x^{(i)}$ can be written as:

$$\mathcal{L}_p = -\sum_k \sum_j \log P(y_j^{(i)} | y_{<j}^{(i)}, r[m]_{<k}^{(i)}, x^{(i)}; \theta^w), \tag{5}$$

where $y^{(i)}$ is the answer and $x^{(i)}$ is the question. θ^w represents the parameters of this rationale token weighting module. $r[m]_{<k}^{(i)}$ represents the preceding k tokens of the masked rationale $r[m]^{(i)}$. By optimizing $\mathcal{L}_p$, the transformer can be used to predict the answer by taking as input the question and the prefix of the masked rationale. Meanwhile, the weight generator is encouraged to generate large weights for the keypoint tokens, preventing them from being masked to facilitate the answer prediction.

Moreover, to eliminate redundant tokens for reasoning, a mask ratio loss $\mathcal{L}_m$ is presented as:

$$\mathcal{L}_m = \sum_j m_j^{(i)}. \tag{6}$$

By optimizing $\mathcal{L}_m$, we enable the weight generator to identify the insignificant tokens in the reasoning process and generate lower weights for them.

Finally, the overall loss function for training this module can be expressed as:

$$\mathcal{L}_k = \mathcal{L}_p + \alpha \mathcal{L}_m, \tag{7}$$

where α is a balancing hyper-parameter. By optimizing $\mathcal{L}_k$, we can achieve the goal of determining the significance weight $w_j^{(i)}$ for each token within a rationale.

3.4. In-rationale Progressive Distillation

In this section, we elaborate our proposed in-rationale progressive distillation strategy, which schedules the learning order within a rationale.

Step Difficulty Assessment. Firstly, we assess the difficulty of each reasoning step in the rationale, so as to facilitate the learning order scheduling. In this work, we utilize the symbol "." to separate steps in a rationale. As mentioned above, there could exist many reasoning-irrelevant tokens, and it is crucial to ensure that the difficulty evaluation is not influenced by them. Therefore, we propose a weighted token generation loss to calculate the difficulty value $d_k^{(i)}$ of the k-th reasoning step in the rationale $r^{(i)}$ as:

$$d_k^{(i)} = -\sum_{j=p_k}^{q_k} \hat{w}_j^{(i)} \log P(r_j^{(i)} | r_{<j}^{(i)}, x^{(i)}; \theta^s), \tag{8}$$

where p_k and q_k denote the start position and end position of the k-th step in the rationale, respectively. Here we directly use the pre-trained student model θ^s (e.g., LLaMA-7B (Touvron et al., 2023)) before distillation to evaluate the generation probability $P(r_j^{(i)} | r_{<j}^{(i)}, x^{(i)}; \theta^s)$. $\hat{w}_j^{(i)} = \mathrm{softmax}(w_j^{(i)})$ represents the significance weight normalized by softmax (Bridle, 1989) within the token weights in the k-th step. In this way, the obtained step difficulty can be more concentrated on the difficulty of generating keypoint tokens, providing a more faithful reflection of the difficulty in deriving the correct outcome of each reasoning step.

Progressive Distillation. Based on the step difficulty scores, we devise an in-rationale progressive distillation strategy to guide the student model learning each rationale in an easy-to-hard fashion. This strategy initiates with training the student model to generate the final few reasoning steps of the rationale using previous steps combined with the question as input, and progressively expands to produce the complete rationales. Supposed that we schedule the student model to output the last $n_i - c_i(t)$ steps of the i-th rationale at stage t, The difficulty $h_i(S(t))$ of generating these steps can be formulated as:

$$h_i(S(t)) = \sum_{j=c_i(t)+1}^{n_i} d_j^{(i)}, \tag{9}$$

where n_i is the total number of steps in the i-th rationale $r^{(i)}$ and $c_i(t)$ is the scheduled number of input steps of $r^{(i)}$ at stage t. $d_j^{(i)}$ is the difficulty of the j-th step in $r^{(i)}$. $S(t)$ is used to decide the value of $c_i(t)$ at stage t, which will be introduced later. In this paper, we treat each training epoch as a stage.

To facilitate selecting diverse questions to increase difficulty at each stage, we configure an overall learning difficulty $D(t)$ for stage t rather than a hard threshold for each question. This means that the difficulty sum of all questions should not exceed $D(t)$ at stage t. We set the growth rate of $D(t)$ to be $\frac{dD(t)}{dt} = ut^p$, where $p > 0$ and $u > 0$ are the parameters to control the growth rate. By integrating the growth rate with respect to t, we can derive $D(t)$ as:

$$D(t) = \frac{ut^{p+1}}{p+1} + C_0, \tag{10}$$

where C_0 represents the initial overall learning difficulty at stage 0. By letting $D(t)$ achieve the maximum difficulty B of the dataset at stage T: $D(T) = B = \sum_i \sum_{j=1}^{n_i} d_j^{(i)}$, we can derive $u = \frac{(B-C_0)(p+1)}{T^{p+1}}$, where p and C_0 are the pre-defined hyper-parameters.

When entering stage t from stage $t-1$, it's required to select a set of questions to increase difficulty. We achieve this by reducing a number of input steps Δ_s for the selected questions as:

$$c_i(t) = c_i(t-1) - q_i(t) \cdot \Delta_s, \;\; s.t. \Delta H(S(t)) \leq \Delta D(t), \tag{11}$$

where $c_i(t)$ is the scheduled number of input steps of the i-th question at stage t. Let $S(t)$ denote the selected question set for increasing difficulty at stage t. Then, $q_i(t) \in \{0, 1\}$ represents whether i belongs to $S(t)$. If $i \in S(t)$, then $q_i(t) = 1$; otherwise, $q_i(t) = 0$. Δ_s is the pre-defined number for reducing input steps. $\Delta H(S(t)) = \sum_i h_i(S(t)) - \sum_i h_i(S(t-1))$ is the sum of the increased difficulty and $\Delta D(t) = D(t) - \sum_i h_i(S(t-1))$ is the ceiling magnitude for the increased difficulty.

Then, in order to determine whether a question should increase difficulty, we design a value function F. The goal of this value function is two-fold: One is to align the increased difficulty as closely as possible with the defined magnitude, and the other is to ensure a diverse set of questions for escalating difficulty to prevent overfitting (Jiang et al., 2014). The value function F is designed as:

$$F(S(t)) = -(\Delta D(t) - \Delta H(S(t))) + \beta \sum_{k=1}^{K} \sqrt{|C_k \cap S(t)|}, \tag{12}$$

where β is a trade-off hyper-parameter. The first term measures the closeness of $\Delta H(S(t))$ to $\Delta D(t)$ and the second term measures the diversity of selected question set based on clustering. Specifically, C_k is the question set of the k-th cluster and K is the number of clusters. In this paper, we conduct K-means clustering (Bradley et al., 2000) to cluster the question based on its embedding, which is calculated by the average of the GloVe (Pennington et al., 2014) word embedding. $S(t)$ is the selected question set. By using the square root operation, our aim is to promote a balanced distribution of questions within each cluster in the selected

question set. This approach ensures that the diversity of the chosen question set is maintained.

The optimization of $F(S(t))$ can be formulated as:

$$\max_{S(t)} F(S(t)), \;\; s.t. \Delta H(S(t)) \leq \Delta D(t). \tag{13}$$

By maximizing $F(S(t))$, we can achieve the goal of selecting diverse questions to increase difficulty with close proximity to $\Delta D(t)$. However, this is a combination optimization problem subject to the knapsack constraint, and solving it is known to be NP-hard. Fortunately, we can prove that $F(S(t))$ satisfies the condition of monotone and submodular. Therefore, it can be approximately solved by a submodular maximization algorithm FTGP (Li et al., 2022) in linear time with an approximation ratio guarantee, as formulated in Proposition 3.1. The proof of Proposition 3.1 can be found in Appendix D.

Proposition 3.1. *The optimization of* $\max_{S(t)} F(S(t))$ *subject to the knapsack constraint* $\Delta H(S(t)) \leq \Delta D(t)$ *can be approximately solved in* $O(n\epsilon^{-1}\log\epsilon^{-1})$ *time complexity with a* $\frac{1}{2} - \epsilon$ *approximation ratio guarantee, where* n *represents the scale of the data.*

After obtaining the scheduled input step $c_i(t)$ by solving Eq.(13), the rationale distillation loss at stage t can be formulated as:

$$\mathcal{L}_r(t) = -\sum_i \sum_{j=p_{c_i(t)+1}}^{q_{n_i}} \log P(r_j^{(i)} | r_{<j}^{(i)}, x^{(i)}; \theta^s), \tag{14}$$

where $p_{c_i(t)+1}$ is the start position of the $(c_i(t)+1)$-th step in the rationale $r^{(i)}$, and q_{n_i} is the end position of the last step in the rationale $r^{(i)}$. For each rationale, n_i is fixed and $c_i(t)$ is gradually decreased to 0. In this way, the student model could learn the rationale of each question in an easy-to-hard manner.

3.5. Training Procedure

To train our whole framework, we first optimize the rationale token weighting module by Eq.(7) to determine the token significance. Then, we assess the step difficulty and derive the progressive distillation strategy by solving Eq.(13). Finally, by integrating these two modules, the overall loss for distilling the rationale at stage t can be written as:

$$\mathcal{L}_o(t) = -\sum_i \sum_{j=p_{c_i(t)+1}}^{q_{n_i}} w_j^{(i)} \cdot \log P(r_j^{(i)} | r_{<j}^{(i)}, x^{(i)}; \theta^s). \tag{15}$$

By optimizing $\mathcal{L}_o(t)$, the student model is encouraged to mimic the keypoint tokens precisely, as well as acquiring reasoning capabilities in an easy-to-hard manner. Note that we have omitted the inclusion of the prediction loss term for $y^{(i)}$ (referring to the second term in Eq. (1)), for the sake of clarity, as it remains constant. The pseudo-code of our training procedure is listed in Appendix B.

4. Experiments

4.1. Experiment Setup

In this section, we introduce our experiment settings. The implementation details can be found in Appendix A.

Datasets. We evaluate our method on both mathematical reasoning tasks and commonsense reasoning tasks, following (Hsieh et al., 2023; Fu et al., 2023). For mathematical reasoning, we adopt three benchmark datasets for evaluation: GSM8K (Cobbe et al., 2021), ASDiv (Patel et al., 2021) and SVAMP (Miao et al., 2021). For commonsense reasoning, CommonsenseQA benchmark (Talmor et al., 2019) is employed to evaluate our method. Additionally, we conduct out-of-distribution (OOD) evaluation via training our method on GSM8K while testing it on ASDiv and SVAMP, following (Fu et al., 2023). The dataset splits can be found in Appendix A.

Models and Baselines. We adopt GPT-3.5-Turbo (Ye et al., 2023) as the teacher model to generate the rationale for each question in the dataset via zero-shot CoT prompting (Kojima et al., 2022), following (Chen et al., 2023). This is accessed via the OpenAI's public API for ChatGPT. As for the student model, we adopt three widely-used pretrained language models of different architectures: LLaMA-7B (Touvron et al., 2023), FlanT5-XL (Chung et al., 2022) and FlanT5-Large (Chung et al., 2022), similar to (Fu et al., 2023; Chen et al., 2023). The parameter counts of LLaMA-7B, FlanT5-XL, FlanT5-Large are 7B, 3B, 760M respectively. As for baselines, we employ four state-of-the-art CoT distillation methods for comparison: Specialized KD (Fu et al., 2023), SCOTT (Wang et al., 2023b), SCoTD (Li et al., 2023), MCC-KD (Chen et al., 2023). Following previous works (Fu et al., 2023), we use the accuracy (%) metric for evaluating the performance of our method and baselines.

4.2. Overall Performance

In this section, we evaluate the overall performance of our method. We compare our method with four recent state-of-the-art CoT distillation methods as mentioned before. The GPT-3.5-Turbo serves as the teacher model. Table 1 illustrates the results. The symbol "-" denotes the model without using CoT distillation methods. First, we can observe that CoT distillation methods consistently boost the performance of smaller student models on reasoning tasks, underscoring the effectiveness of distilling rationales. In addition, it's evident that our proposed KPOD outperforms previous methods by a large margin. For example, compared to MCC-KD, achieving the second best results when using LLaMA-7B as the student model, our approach achieves 5.16%, 5.26%, 4.00%, 1.48% performance gains on the GSM8K, ASDiv, SVAMP, CommonsenseQA datasets, respectively. This highlights the effectiveness of promoting

Table 1. Performance comparison of our method and baselines.

Models	# Params.	Distillation Methods	Datasets			
			GSM8K	ASDiv	SVAMP	CommonsenseQA
GPT-3.5-Turbo	unknown	-	73.98	79.64	75.14	74.35
LLaMA-7B	7B	-	11.00	40.20	32.80	33.90
		SCoTD	38.54	63.38	62.67	71.33
		Specialized KD	39.15	64.01	63.33	72.32
		SCOTT	40.97	62.74	61.33	74.45
		MCC-KD	41.58	65.76	64.67	76.41
		KPOD (ours)	**46.74**	**71.02**	**68.67**	**77.89**
FlanT5-XL	3B	-	13.50	20.70	17.70	72.70
		SCoTD	21.85	25.16	26.67	79.61
		Specialized KD	23.22	28.03	25.33	81.16
		SCOTT	21.09	25.48	24.67	83.62
		MCC-KD	24.28	31.35	30.00	82.88
		KPOD (ours)	**25.19**	**33.76**	**34.67**	**88.04**
FlanT5-Large	760M	-	6.90	10.10	6.80	67.60
		SCoTD	19.42	20.06	19.33	76.58
		Specialized KD	20.03	23.25	20.67	77.23
		SCOTT	18.21	21.66	18.67	77.48
		MCC-KD	18.36	23.89	21.33	78.13
		KPOD (ours)	**22.46**	**27.39**	**25.33**	**81.41**

Table 2. Ablation study of our method.

Models	Settings	Datasets	
		GSM8K	CommonsenseQA
LLaMA-7B	KPOD-w.o.-sig	42.64	75.18
	KPOD-w.o.-sig-dif	44.01	76.49
	KPOD-w.o.-prog	43.25	74.61
	KPOD-w.o.-div	44.16	75.76
	KPOD-ACL	43.55	75.51
	KPOD-SPL	42.94	75.84
	KPOD-ICL	43.85	75.35
	KPOD	46.74	77.89
FlanT5-XL	KPOD-w.o.-sig	22.46	85.26
	KPOD-w.o.-sig-dif	23.82	86.08
	KPOD-w.o.-prog	23.22	84.28
	KPOD-w.o.-div	23.98	86.73
	KPOD-ACL	23.52	86.40
	KPOD-SPL	22.76	85.59
	KPOD-ICL	22.91	85.83
	KPOD	25.19	88.04

precise mimicry of keypoint tokens and implementing a learning schedule that progresses from easy to challenging tasks. Such an approach facilitates the acquisition of reasoning capabilities by the student model.

4.3. Ablation Study

We conduct ablation study to verify the effectiveness of the components in our proposed method. Specifically, we design several variants of our proposed KPOD: KPOD-w.o.-sig denotes our method wherein each token is treated equally, without incorporating the token significance weight for distillation. KPOD-w.o.-sig-dif represents our method without using the token significance weight for calculating the step difficulty. KPOD-w.o.-prog means our method without using the proposed progressive distillation strategy. KPOD-w.o.-div denotes our method without using the diversity term in the value function to select the question set.

Besides, we compare our method with three representative curriculum learning methods: Adaptive CL (Kong et al., 2021), SPL (Wan et al., 2020) and ICL (Jia et al., 2023). We design three variants of our method: KPOD-ACL, KPOD-SPL, KPOD-ICL respectively denote replacing our in-rationale progressive distillation strategy by Adaptive CL, SPL and ICL. The results are listed in Table 2.

As shown in Table 2, KPOD-w.o.-sig obtains inferior performance than KPOD, illustrating the effectiveness of emphasizing the precise mimicry of keypoint tokens in our method. Besides, KPOD outperforms KPOD-w.o.-sig-dif. This shows that it's essential to utilizing the token significance weight for the step difficulty calculation. The performance of KPOD-w.o.-prog is worse than KPOD, illustrating the effectiveness of scheduling an easy-to-hard learning order for CoT distillation. Moreover, KPOD obtains better performance than KPOD-w.o.-div. This demonstrates that ensuring a diverse question set to increase difficulty is effective. Finally, we can find that KPOD surpasses KPOD-ACL,

Table 3. OOD performance of our method and baselines.

Model	Methods	In-distribution	OOD	
		GSM8K	ASDiv	SVAMP
LLaMA-7B	SCoTD	38.54	55.09	45.33
	Specialized KD	39.15	53.82	38.67
	SCOTT	40.97	53.50	42.00
	MCC-KD	41.58	57.64	41.00
	KPOD (ours)	**46.74**	**57.96**	**47.33**
FlanT5-XL	SCoTD	21.85	25.48	22.67
	Specialized KD	23.22	26.11	24.67
	SCOTT	21.09	25.20	25.33
	MCC-KD	24.28	28.98	26.67
	KPOD (ours)	**25.19**	**32.48**	**29.33**

KPOD-SPL and KPOD-ICL, showing the superiority of our in-rationale progressive distillation strategy compared to previous curriculum learning methods.

4.4. OOD Performance

Following (Fu et al., 2023), we examine the out-of-distribution (OOD) generalization ability of the student model trained by our method and baselines. We use the in-distribution mathematical dataset GSM8K for training and adopt OOD mathematical datasets ASDiv, SVAMP for testing, similar to (Fu et al., 2023; Chen et al., 2023). As shown in Table 3, our proposed KPOD consistently obtains superior performance compared to the baselines, indicating that the student model trained by our method has stronger OOD generalization capabilities.

4.5. Visualizations

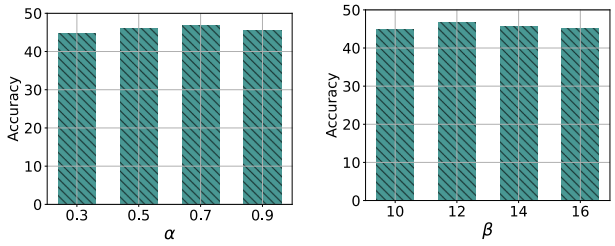
In this section, we visualize the token significance weight $w_j^{(i)}$ generated by the weight generator, to intuitively show the effectiveness of the rationale token weighting module. Figure 2 illustrates the visualization results on the GSM8K dataset. First, we can find that the digit tokens and operation tokens obtain the highest weights. This is because these tokens are usually of vital importance in the reasoning process, where even a slight deviation could cause errors. Additionally, several tokens that contribute significantly to the reasoning also exhibit relatively high weights. Tokens such as "twice", "total", "adding", and "dividing" provide instructional cues for the reasoning steps. Besides, meaningful subjects like "Mark" and "Jennifer" can play a crucial role in reasoning, as their relationships should be considered during the reasoning process. Furthermore, it could be observed that some tokens of less importance for the reasoning are given low weights, such as "can", "say", "fit", "received", "got", etc. These visualizations demonstrate our rationale token weighting module can effectively determine the significance of rationale tokens, thereby facilitating the student to accurately mimic crucial keypoint tokens.

If Tony got twice of what Ken received , then Tony received 2 *$ 1 7 5 0 = $ 3 5 0 0 The total amount shared is $ 3 5 0 0 +$ 1 7 5 0 = $ 5 2 5 0 .

If two students can fit on each of a hotel ' s two queen size b eds , then the total number of students that can fit in one room is 2 + 2 = 4 students . Add ing one student sleep ing on the pull - out c ouch , we can say that one room can fit a maximum of 4 + 1 = 5 students . D ivid ing the number of students in the class by the number of students that can fit in one room , we get 3 0 / 5 = 6 .

If Mark purchased 5 0 can s of milk , the total number of can s of milk that Jenn ifer purchased before meeting Mark is 4 0 . Then , Jenn ifer bought 6 can s for every 5 can s Mark bought , so she bought 6 / 5 * 5 0 = 6 0 can s of milk .

Figure 2. Visualizations of token significance weights produced by the weight generator. The intensity of red corresponds to the significance weight assigned to each token, with a deeper red indicating higher weight.



(a) performance with varying α (b) performance with varying β

Figure 3. Parameter sensitivity study of α and β.

4.6. Parameter Sensitivity Analysis

We perform experiments to analyze the effect of two important hyper-parameters α and β in our method on GSM8K with LLaMA-7B as the student model. Figure 3 shows the results. First, we analyze the effect of hyper-parameter α in the mask ratio loss. We can observe that the performance of our method is not sensitive to α in a relatively large range. Second, we study the influence of hyper-parameter β in the diversity term for question set selection. Similarly, our method is not sensitive to β in a relatively large range. Thus it's easy to set them in practice. We analyze the sensitivity of other hyper-parameters in Appendix C.

5. Conclusion

In this paper, we proposed a keypoint-based progressive chain-of-thought distillation framework for LLMs. Specifically, we devised a rationale token weighting module to encourage the student model to accurately mimic keypoint tokens during the distillation process. Besides, we proposed an in-rationale progressive distillation strategy to enable the student model to acquire reasoning capabilities from the teacher LLMs in an easy-to-hard manner. Extensive experiments validated the effectiveness of our proposed method.

Acknowledgment

This work was supported by the NSFC under Grants 62122013, U2001211. This work was also supported by the Innovative Development Joint Fund Key Projects of Shandong NSF under Grants ZR2022LZH007.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *International Conference on Machine Learning*, pp. 41–48, 2009.

Benoit, L., Lehalle, H., Molina, M., Tijus, C., and Jouen, F. Young children's mapping between arrays, number words, and digits. *Cognition*, 129(1):95–101, 2013.

Bradley, P. S., Bennett, K. P., and Demiriz, A. Constrained k-means clustering. *Microsoft Research, Redmond*, 20, 2000.

Bridle, J. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. *Annual Conference on Neural Information Processing Systems*, 2, 1989.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Annual Conference on Neural Information Processing Systems*, 33:1877–1901, 2020.

Chen, H., Wu, S., Quan, X., Wang, R., Yan, M., and Zhang, J. MCC-KD: Multi-CoT consistent knowledge distillation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 6805–6820. Association for Computational Linguistics, December 2023.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Diao, S., Wang, P., Lin, Y., and Zhang, T. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*, 2023.

Elman, J. L. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

Feng, K., Li, C., Yuan, Y., and Wang, G. Freekd: Free-direction knowledge distillation for graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 357–366, 2022.

Feng, K., Li, C., Ren, D., Yuan, Y., and Wang, G. On the road to portability: Compressing end-to-end motion planner for autonomous driving. *arXiv preprint arXiv:2403.01238*, 2024.

Fu, Y., Peng, H., Ou, L., Sabharwal, A., and Khot, T. Specializing smaller language models towards multi-step reasoning. *International Conference on Machine Learning*, 2023.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ho, N., Schmid, L., and Yun, S.-Y. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14852–14882. Association for Computational Linguistics, July 2023.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-k., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. 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*, pp. 8003–8017. Association for Computational Linguistics, July 2023.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jia, Q., Liu, Y., Tang, H., and Zhu, K. In-sample curriculum learning by sequence completion for natural language generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11937–11950. Association for Computational Linguistics, July 2023.

Jiang, L., Meng, D., Yu, S.-I., Lan, Z., Shan, S., and Hauptmann, A. Self-paced learning with diversity. *Annual Conference on Neural Information Processing Systems*, 27, 2014.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *Annual Conference on Neural Information Processing Systems*, 35:22199–22213, 2022.

Kong, Y., Liu, L., Wang, J., and Tao, D. Adaptive curriculum learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5067–5076, 2021.

Krueger, K. A. and Dayan, P. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394, 2009.

Li, L. H., Hessel, J., Yu, Y., Ren, X., Chang, K.-W., and Choi, Y. Symbolic chain-of-thought distillation: Small models can also "think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2665–2679. Association for Computational Linguistics, July 2023.

Li, W., Feldman, M., Kazemi, E., and Karbasi, A. Submodular maximization in clean linear time. *Annual Conference on Neural Information Processing Systems*, 35:17473–17487, 2022.

Liang, C., Jiang, H., Liu, X., He, P., Chen, W., Gao, J., and Zhao, T. Token-wise curriculum learning for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3658–3670, 2021.

Ling, W., Yogatama, D., Dyer, C., and Blunsom, P. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*, 2017.

Magister, L. C., Mallinson, J., Adamek, J., Malmi, E., and Severyn, A. Teaching small language models to reason. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1773–1781. Association for Computational Linguistics, July 2023.

Miao, S.-Y., Liang, C.-C., and Su, K.-Y. A diverse corpus for evaluating and developing english math word problem solvers. *arXiv preprint arXiv:2106.15772*, 2021.

Molina, M. and Jouen, F. Modulation of the palmar grasp behavior in neonates according to texture property. *Infant Behavior and Development*, 21(4):659–666, 1998.

Narayan, S. The generalized sigmoid activation function: Competitive supervised learning. *Information Sciences*, 99(1-2):69–82, 1997.

Patel, A., Bhattamishra, S., and Goyal, N. Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*, 2021.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

Peterson, G. B. A day of great illumination: Bf skinner's discovery of shaping. *Journal of the experimental analysis of behavior*, 82(3):317–328, 2004.

Talmor, A., Herzig, J., Lourie, N., and Berant, J. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, 2019.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Wan, Y., Yang, B., Wong, D. F., Zhou, Y., Chao, L. S., Zhang, H., and Chen, B. Self-paced learning for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1074–1080. Association for Computational Linguistics, November 2020.

Wang, H., Wang, R., Mi, F., Deng, Y., Wang, Z., Liang, B., Xu, R., and Wong, K.-F. Cue-cot: Chain-of-thought prompting for responding to in-depth dialogue questions with llms. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12047–12064, 2023a.

Wang, P., Wang, Z., Li, Z., Gao, Y., Yin, B., and Ren, X. SCOTT: Self-consistent chain-of-thought distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5546–5558. Association for Computational Linguistics, July 2023b.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Wang, Y., Wang, W., Liang, Y., Cai, Y., and Hooi, B. Curgraph: Curriculum learning for graph classification. In *Proceedings of the Web Conference 2021*, pp. 1238–1248, 2021.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Annual Conference on Neural Information Processing Systems*, 35:24824–24837, 2022.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2023.

Ye, J., Chen, X., Xu, N., Zu, C., Shao, Z., Liu, S., Cui, Y., Zhou, Z., Gong, C., Shen, Y., et al. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. *arXiv preprint arXiv:2303.10420*, 2023.

Zhang, Z., Zhang, A., Li, M., and Smola, A. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*, 2022.

Zhang, Z., Zhang, A., Li, M., Zhao, H., Karypis, G., and Smola, A. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

A. Implementation Details

We perform our experiments using GeForce RTX 3090 GPUs. In order to accelerate training, we employ LoRA (Hu et al., 2021) to train the student model. Following previous CoT distillation works (Chen et al., 2023), the rank of LoRA is set to 64 for LLaMA-7B and 128 for FlanT5-XL. We use Adam optimizer for optimization with a learning rate of 1×10^{-5} for LLaMA-7B and 5×10^{-5} for FlanT5 models. The batch size is set to 4. In terms of LLaMA-7B, the epoch number for training the student model is set to 20 for the GSM8K, CommonsenseQA datasets, and 40 for the ASDiv, SVAMP datasets. As for FlanT5 models, the epoch number is set to 100 because they require more optimization steps for convergence. The input embedding layer in this module aligns with the pretrained student model's input embedding layer for the consistency of tokenizer. The hyper-parameters α that balances the answer prediction loss and mask ratio loss is set to 0.5. As for the progressive distillation strategy, we simply treat each epoch as a training state in this paper. The stage T that achieves the maximum difficulty is set as half of the epoch number. The initial overall learning difficulty C_0 is set to 30% of the maximum difficulty B. We set exponential $p = 0.5$ in $D(t)$ that controls the growth rate of the learning difficulty. The hyper-parameter β of the diversity term in the question set selection is set to 12. The number of clusters is set as 5 for clustering the question.

We follow previous CoT distillation works to split the datasets (Chen et al., 2023; Fu et al., 2023), the datasets statistics are summarized in Table 4.

Table 4. Dataset statistics.

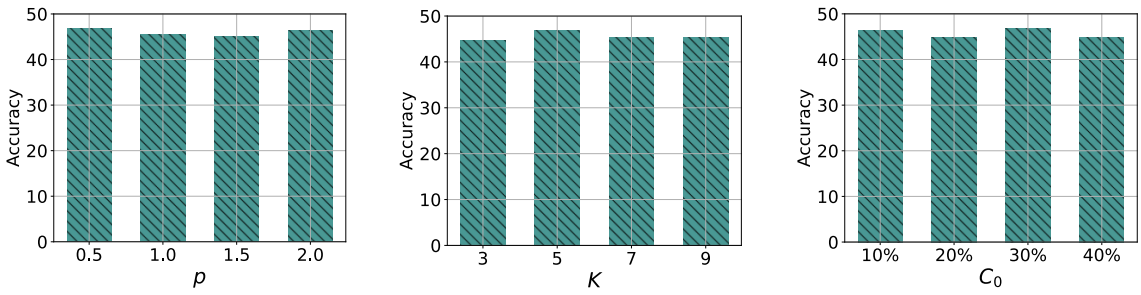
Datasets	Train Size	Validation Size	Test Size
GSM8K	7473	660	659
ASDiv	1462	313	314
SVAMP	700	150	150
CommonsenseQA	8520	1221	1221

B. Training Pseudo-code

Algorithm 1 outlines the training procedure of our KPOD. Initially, we employ the CoT prompt (Kojima et al., 2022) to instruct the teacher LLM to generate step-by-step rationales for each question in the dataset. Subsequently, the rationale token weighting module receives these rationales as input and is trained to determine the significance weights for each token. Following this, we compute the difficulty of each step in the rationale based on these weights. We then utilize the FTGP algorithm (Li et al., 2022) to maximize Eq.(13) to schedule the question set for increasing difficulty at each stage. Once scheduled, we train the student model using Eq.(15) based on the established learning order and token significance weights. Before epoch T, we progressively escalate the learning difficulty to the maximum difficulty. Post-epoch T, the student is trained to generate the complete rationale for each question. This approach allows the student model to precisely mimic the keypoint tokens while progressively acquiring reasoning capabilities in an easy-to-hard fashion.

Algorithm 1 The training procedure of KPOD

Input: a teacher LLM, dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}$, epoch number N_e for training student, epoch number T for achieving the maximum difficulty, hyper-parameter settings;
Output: a trained smaller student model θ^s;
prompt the teacher LLM to generate rationale for each question $x^{(i)}$ in $\mathcal{D}$;
optimize the rationale token weighting module by Eq.(7) to obtain the token significance weight $w_j^{(i)}$;
calculate the step difficulty based by Eq.(8) based on $w_j^{(i)}$;
run FTGP algorithm (Li et al., 2022) to solve Eq.(13) to derive $S(t)$ for each stage.
for each epoch e from 1 to N_e **do**
  Let $c_i(t) = 0$ for every sample;
  if $e \leq T$ **then**
    obtain $c_i(t)$ by Eq.(11) based on $S(t)$;
  end if
  train the student model θ^s to generate the rationale by Eq.(15) based on $w_j^{(i)}$ and $c_i(t)$;
end for



(a) performance with varying p (b) performance with varying K (c) performance with varying C_0

Figure 4. Parameter sensitivity study of p, K and C_0 on GSM8K.

C. Additional Experiments

We additionally analyze the sensitivity of three hyper-parameters of p, K and C_0. Figure 4 (a)(b)(c) show the performance of LLaMA-7B on GSM8K with varying p, K, C_0 respectively. First, we analyze the effect of p that controls the growth rate the learning difficulty. We can find that the performance of our method is relatively stable to this hyper-parameter. Besides, we study the influence of the number of clusters K for clustering the questions. It can be observed that our method is not sensitive to K in a relatively large range. In addition, we investigate the sensitivity of C_0 which is the initial learning difficulty. In Figure 4(c), $r\%$ denotes setting C_0 to r percentage of the maximum difficulty B. Our method is still not sensitive to this hyper-parameter.

D. Proof

In this section, we prove the Proposition 3.1. First, we introduce Theorem D.1 proposed in FTGP algorithm (Li et al., 2022).

Theorem D.1. *If a function $f : 2^N \to \mathbb{R}$ is monotone and submodular. Then, the optimization of $\max_S F(S)$ that subjects to a knapsack constraint can be approximately solved in $O(n\epsilon^{-1}\log\epsilon^{-1})$ time complexity by FTGP algorithm (Li et al., 2022) with an approximation ratio guarantee, where n represents the scale of the data and ϵ is a hyper-parameter. If S^{opt} is the optimal solution and $\hat{S}$ is the approximate solution of FTGP, then $F(\hat{S}) \geq (\frac{1}{2} - \epsilon)F(S^{opt})$ holds.*

According to Theorem D.1, if we could prove that our value function F is monotone and submodular, then Proposition 3.1 is proved. In the next, we will prove that our value function F satisfies these two conditions.

Definition 1. (Monotonicity) A function $f : 2^N \to \mathbb{R}$ is monotone if for $\forall A \subseteq B \subseteq N$ where N is the universal set of all elements, it holds that $F(A) \leq F(B)$.

Lemma 1. Our value function F in Eq.(13) is monotone.

Proof. We define two question sets $A(t), B(t)$ for increasing difficulty at stage t that satisfy $A(t) \subseteq B(t) \subseteq N$. Let $\Delta = F(B(t)) - F(A(t))$. We have:

$$\begin{aligned}
\Delta &= -(D(t) - D(t)) + \Delta H(B(t)) - \Delta H(A(t)) + \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|} \\
&\geq \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|} \\
&= \beta \sum_{k=1}^{K} (\sqrt{|C_k \cap B(t)|} - \sqrt{|C_k \cap A(t)|}) \\
&\geq 0
\end{aligned}$$

Thus, we have:

$$\Delta = F(B) - F(A) \geq 0. \tag{16}$$

$$\Rightarrow F(A) \leq F(B). \tag{17}$$

□

Definition 2. (Submodularity) A function $f : 2^N \to \mathbb{R}$ is submodular if for $\forall A \subseteq B \subseteq N$ and $\forall x \in N \backslash B$, it holds that $F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B)$.

Lemma 2. Our value function F in Eq.(13) is submodular.

Proof. We define two triad sets A, B that satisfy $A \subseteq B \subseteq N$. Let $T = B \backslash A$. Define $\Delta = (F(A \cup \{x\}) - F(A)) - (F(B \cup \{x\}) - F(B))$. Then we have:

$$\begin{aligned}
\Delta &= (\Delta H(A(t) \cup \{x\}) - \Delta H(A(t))) - (\Delta H(B(t) \cup \{x\}) - \Delta H(B(t))) \\
&\quad + (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (A(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|}) - (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (B(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|}) \\
&= \Delta H(\{x\}) - \Delta H(\{x\}) \\
&\quad + (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (A(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|}) - (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (B(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|}) \\
&= (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (A(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|}) - (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap (B(t) \cup \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|}).
\end{aligned} \tag{18}$$

Given that $x \in N \backslash B$ and $A \subseteq B$, it follows that $x \notin A$ and $x \notin B$. Then, we have:

$$\begin{aligned}
\Delta &= (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)| + |C_k \cap \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap A(t)|}) \\
&\quad - (\beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)| + |C_k \cap \{x\})|} - \beta \sum_{k=1}^{K} \sqrt{|C_k \cap B(t)|}).
\end{aligned} \tag{19}$$

For convenience, we denote $x_k = |C_k \cap A(t)|$, $y_k = |C_k \cap B(t)|$, $z_k = |C_k \cap \{x\})|$. Then, we have:

$$\begin{aligned}
\Delta &= \beta \sum_{k=1}^{K} ((\sqrt{x_k + z_k} - \sqrt{x_k}) - (\sqrt{y_k + z_k} - \sqrt{y_k})) \\
&= \beta \sum_{k=1}^{K} (\frac{(\sqrt{x_k + z_k} - \sqrt{x_k}) - (\sqrt{y_k + z_k} - \sqrt{y_k})(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})}) \\
&= \beta \sum_{k=1}^{K} (\frac{x_k + z_k - x_k - (y_k + z_k) + y_k + 2\sqrt{x_k + z_k}\sqrt{y_k} - 2\sqrt{x_k}\sqrt{y_k + z_k}}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})}) \\
&= \beta \sum_{k=1}^{K} (\frac{2\sqrt{x_k + z_k}\sqrt{y_k} - 2\sqrt{x_k}\sqrt{y_k + z_k}}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})}) \\
&= \beta \sum_{k=1}^{K} (\frac{2\sqrt{x_k y_k + y_k z_k} - 2\sqrt{x_k y_k + x_k z_k}}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})})
\end{aligned}$$

Since $A \subseteq B$, it's evident that $y_k = |C_k \cap B(t)| \geq |C_k \cap A(t)| = x_k$. Therefore, we conclude:

$$\Delta = \beta \sum_{k=1}^{K} (\frac{2\sqrt{x_k y_k + y_k z_k} - 2\sqrt{x_k y_k + x_k z_k}}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})})$$

$$\geq \beta \sum_{k=1}^{K} \left(\frac{2\sqrt{x_k y_k + x_k z_k} - 2\sqrt{x_k y_k + x_k z_k}}{(\sqrt{x_k + z_k} + \sqrt{x_k}) + (\sqrt{y_k + z_k} + \sqrt{y_k})} \right) = 0$$

Then, we can derive:

$$\Delta = (F(A \cup \{x\}) - F(A)) - (F(B \cup \{x\}) - F(B)) \geq 0. \tag{20}$$

$$\Rightarrow F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B). \tag{21}$$

□