ToT performs well on problems that even challenge GPT-4 OpenAI (2023). Considering its high ability, the base thought structure of BoT largely utilizes this tree thought structure ToT. And, thanks to the boosting framework, the tree structure generated in each iteration of BoT is binary and shallow instead of the ToT's complex tree, in which each node corresponds to massive child nodes. However, the base structure is not restricted to ToT. In contrast, BoT is flexible as the base thought structure can be either ToT, GoT Besta et al. (2023), or CR Zhang et al. (2023b), where Graph of Thoughts (GoT) Besta et al. (2023) is the most recent work that expands the thought structure into a graph format. This paper will only focus on the ToT as the base thought structure and leave the usage of GoT for future work.

Automatic Prompting. Releasing humans from task-specific prompts attracts much attention Shin et al. (2020). To guarantee the reasoning ability of LLMs, conventional CoT Wei et al. (2022) relies on human priors to manually generate task-specific demonstrations as the prompt. However, the zero-shot CoT Kojima et al. (2022) shows that even without hand-crafted examples, by simply adding "Let's think step by step" to the prompt, LLMs are able to perform step-by-step reasoning toward accurate answers. These insights have spurred a series of subsequent studies. Auto-CoT Zhang et al. (2022) eliminates manual efforts by retrieving usable reasoning chains generated by zero-shot CoT. Active-Prompt Diao et al. (2023) first measures the uncertainty of a set of questions and thus selects only the uncertain ones to be annotated by humans. ToT Yao et al. (2024) can also reduce manual efforts, but for each task, it still requires experts to provide possible next-step thoughts in the prompt. Our paper introduces a novel boosting approach for manual-free prompting. Starting with a simple prompt, BoT iteratively enhances it based on the analysis of LLMs on thoughts.

Prompt Engineering via Feedback. Utilizing responses from LLMs to the input prompt as feedback for further prompt revisions has garnered much attention. Those who continuously revise the given prompt based on evaluation descriptions from LLMs aim to gain an accurate answer Weng et al. (2023). Using a similar higher-level idea of our paper, SELF-REFINE Madaan et al. (2023) proposes an iterative self-refinement algorithm to let the LLM produce feedback for its output for further refinement. PHP Zheng et al. (2023) simplifies this process by directly adding a solution from the previous answer as a hint to the subsequent prompt. REFINER Paul et al. (2023) is also related to our paper as it evaluates each reasoning step as feedback to produce a more reasonable one. Another line of research explores ensembles, particularly leveraging the boosting mechanism Freund et al. (1996) to refine the prompt using feedback from a set of examples. They adjust the prompt to focus on the unsolved problems in the previous iteration by either adding a few shot examples uncertain in the previous Pitis et al. (2023) or relying on a feedback-reflect-refine process Zhang et al. (2023a). APO Pryzant et al. (2023) iteratively refines a prompt, using the performance of the prior prompt to form a natural language for optimization. These works prove the effectiveness of the boosting mechanism in prompt engineering. However, our work is the first to highlight the importance of error analysis in enhancing the prompt toward generating effective reasoning chains. The proposed BoT extends this insight to implement an automated prompting framework by iteratively accumulating an ensemble of trial-and-error reasoning experiences in the prompt.

3 Boosting of Thoughts

3.1 Background

The objective of prompt engineering is to design a prompt $\mathbb{I}$ containing multiple language sequences, such that with this prompt as input, a pre-trained large language model (LLM) denoted as p_θ parameterized by θ, can obtain the desired language sequence y. Thus, the standard Input-Output (IO) can be formulated as $y \sim p_\theta\left(y|\mathbb{I}\left(X, Q\right)\right)$ in which $\mathbb{I}\left(\cdot\right)$ means that the prompt wraps task instructions X and the corresponding question Q.

The prompt can be designed in a more delicate way to guide the LLM toward solving a problem in a step-by-step manner. Each intermediate reasoning step is denoted as z_i (a.k.a *thought*). CoT Wei et al. (2022) provides few-shot examples with the answer of each example containing a chain of *thought* $z_{1\ldots n}$. This leads to $y \sim p_\theta\left(y|\mathbb{I}\left(\left[z_{1\ldots n}\right]^N, X, Q\right)\right)$ where N is the number of examples included in the prompt.