# $T^2$: An Adaptive Test-Time Scaling Strategy for Contextual Question Answering

**Zhengyi Zhao[1,5]*, Shubo Zhang[2]*, Zezhong Wang[1,5], Huimin Wang[3], Yutian Zhao[3], Bin Liang[1,5], Yefeng Zheng[4], Binyang Li[2]†, Kam-Fai Wong[1,5], Xian Wu[3]†**

[1] The Chinese University of Hong Kong [2] University of International Relations
[3] Tencent Jarvis Lab [4] Westlake University
[5] Ministry of Education Key Laboratory of High Confidence Software Technologies, CUHK
{zyzhao,kfwong}@se.cuhk.edu.hk, byli@uir.edu.cn, kevinxwu@tencent.com

## Abstract

Recent advances in Large Language Models (LLMs) have demonstrated remarkable performance in Contextual Question Answering (CQA). However, prior approaches typically employ elaborate reasoning strategies regardless of question complexity, leading to low adaptability. Recent efficient test-time scaling methods introduce budget constraints or early stop mechanisms to avoid overthinking for straightforward questions. But they add human bias to the reasoning process and fail to leverage models' inherent reasoning capabilities. To address these limitations, we present $T^2$: Think-to-Think, a novel framework that dynamically adapts reasoning depth based on question complexity. $T^2$ leverages the insight that if an LLM can effectively solve similar questions using specific reasoning strategies, it can apply the same strategy to the original question. This insight enables the adoption of concise reasoning for straightforward questions while maintaining detailed analysis for complex problems. $T^2$ works through four key steps: decomposing questions into structural elements, generating similar examples with candidate reasoning strategies, evaluating these strategies against multiple criteria, and applying the most appropriate strategy to the original question. Experimental evaluation across seven diverse CQA benchmarks demonstrates that $T^2$ not only achieves higher accuracy than baseline methods but also reduces computational overhead by up to 25.2%.

## 1 Introduction

Large language models (LLMs) have demonstrated impressive capabilities in Contextual Question Answering (CQA) tasks (Trivedi et al., 2023; Press et al., 2023), but their reasoning approaches often lack adaptability to question complexity. Current CQA systems typically employ either direct answer generation or elaborate step-by-step reasoning for all questions, regardless of difficulty (Wei et al., 2022; Huang et al., 2024; Min et al., 2024). This one-size-fits-all approach has an accuracy-vs-efficiency delimma. Directly generating answers for all questions will deteriorate the performance on difficult questions, which require multi-hop reasoning. Elaborated reasoning for all questions creates an efficiecy challenge: models frequently generate reasoning chains that are excessively verbose, containing redundant steps that do not contribute to finding the correct answer.

Existing analysis reveals that these redundant reasoning paths can unnecessarily extend the length of reasoning chains multiple times beyond what is required. Such as exploring multiple solution approaches when only one is needed (Ji et al., 2025), or verifying simple facts with elaborate explanations (Muennighoff et al., 2025). For example, when asked "What is the capital of France?", models often generate lengthy discussions about France's history and geography before providing the straightforward answer "Paris." This computational inefficiency is particularly concerning as model deployment costs continue to rise. Recent studies on reasoning efficiency (Yang et al., 2025; Zeng et al., 2025) confirm that blindly increasing reasoning chain length can actually harm performance on simpler tasks. Various attempts have been made to address this through adding a budget or stop mechanism to test-time scaling (TTS) methods (Wei et al., 2022; Huang et al., 2024) to stop thinking early, but these approaches introduce a human bias to the reasoning process (Yuan et al., 2023) and fail to leverage the model's inherent reasoning abilities.

Hence, the fundamental challenge is to develop a reasoning mechanism that can dynamically adjust its computational effort based on question complexity, which means providing concise reasoning

*Equal Contribution
†Corresponding Author



*Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 3731–3756
November 4-9, 2025 

for straightforward questions while maintaining detailed analysis for complex problems. Therefore, we present $T^2$, a think-to-think framework for efficient TTS strategy. $T^2$ leverages a key insight: if an LLM can effectively solve similar questions using specific reasoning strategies, it can apply comparable strategies to the original question. The process involves four key steps: (1) Decomposing the original question into its structural elements. For example, given the question:

> [Given Reference Documents]
> *"Which is taller, the Eiffel Tower or the Empire State Building?"*

$T^2$ would identify this as a comparative question involving measurement between two specific places as "Which is [adj], [place 1] or [place 2]?". (2) Creating a diverse set of similar example questions with the same question structure, each paired with supporting documents and potential reasoning strategies. Each reasoning strategy breaks down similar questions into simpler steps using fundamental reasoning skills (e.g., decomposing similar question "Which is taller, Building A or Building B?" into subquestions about individual heights connected by deductive reasoning for comparison). (3) Evaluating these reasoning strategies using multiple criteria to select the most appropriate strategy for the original question. (4) Applying the selected reasoning strategy to the original question while filtering irrelevant information.

By learning from similar examples, the model develops a more nuanced understanding of when detailed reasoning is necessary and when a more direct approach is sufficient. This allows $T^2$ to balance accuracy and efficiency without relying on pre-determined reasoning templates.

We evaluate $T^2$ across seven diverse CQA datasets ranging from simple factual queries to complex multi-hop reasoning tasks. Our results demonstrate that $T^2$ achieves superior accuracy (up to a 21.3% increase) compared to other TTS approaches while reducing computational requirements by up to 25.2%. These efficiency gains are particularly clear for simpler questions where redundant reasoning steps are eliminated. While for complex questions, $T^2$ maintains the reasoning depth required for accuracy without exploring unnecessary paths.

Our contributions include:

- We introduce $T^2$, a framework that enables language models to dynamically select appropriate reasoning strategies through similar examples, balancing efficiency and thoroughness based on question complexity.
- We develop a multi-criteria selection method that evaluates potential reasoning strategies based on coverage and uniqueness, ensuring the most suitable approach is applied to each question.
- We demonstrate through extensive experiments across diverse CQA benchmarks that our method reduces computational requirements by up to 25.2% with superior accuracy.

## 2 Related Work

**Contextual QA.** In addressing contextual QA, recent works have explored multi-round retrieval or reasoning approaches, including query rewriting for subsequent retrievals (Khattab et al., 2022; Ma et al., 2023; Shao et al., 2023; Jiang et al., 2023), alternating between retrieval and reasoning steps (Trivedi et al., 2023), and employing multi-round self-asking techniques (Press et al., 2023). They all rely on LLMs' reasoning abilities. We also discuss the application scope in Appendix B.1.

**Test-Time Scaling.** Recent approaches to enhancing LLM reasoning capabilities focus on increasing computational resources during inference (Brown et al., 2024; Chen et al., 2024), termed test-time scaling. These methods includes majority voting (Wang et al., 2022), weighted aggregation (Li et al., 2023), best-of-N (Lightman et al., 2023), Tree-of-Thoughts (Yao et al., 2023), and Monte Carlo Tree Search variants (Wu et al., 2024; Zhang et al., 2024a; Zhao et al., 2024). Besides, o1 model (Jaech et al., 2024) and several follow-up works (Guo et al., 2025; Qwen, 2024; Gemini, 2025a; Min et al., 2024; Huang et al., 2024) increase the thinking depth to improve the performance. But they all apply fixed scaling strategies to all questions. Some adaptive thinking methods like AdoT (Xu et al., 2024) and DAST (Shen et al., 2025) design difficulty measurement to categorize the question based on its difficulty, whereas they introduce human bias and fail to leverage the model's inherent reasoning abilities. Our $T^2$ framework builds upon this paradigm while addressing these key limitations.

## 3 $T^2$: Think-to-Think Framework

In this section, we present $T^2$: Think-to-Think, an approach that enables language models to adapt

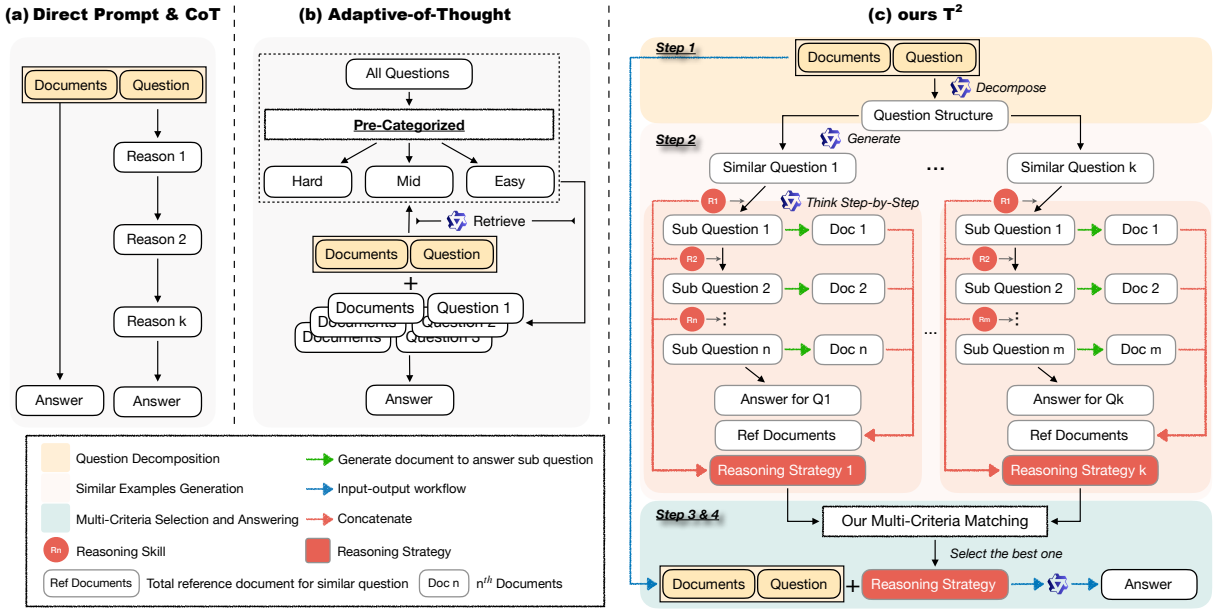


Figure 1: Overview of our T$^2$. **(a)** direct prompt or Chain-of-Thought (CoT), which adopts the same reasoning strategy regardless of question complexity. **(b)** Adaptive-of-Thought, which designs a question complexity evaluator to pre-categorize all questions, which might bring human bias in the evaluator design process. **(c)** our T$^2$. Instead of pre-categorizing questions into different complexity sets, T$^2$ generates multiple similar examples for different inputs adaptively and selects the best reasoning strategy for answering.

their reasoning strategies based on question complexity. Figure 1 provides an overview of our approach. We begin by describing the overall architecture and workflow of T$^2$ before delving into each component in detail.

### 3.1 Question Decomposition

Given a document $D$ and a question $Q$, we first analyze the question's structure to understand its underlying pattern. This allows us to later generate similar questions that require same reasoning strategy. The question structure identification process involves decomposing the question into fixed structural elements and variable entities that could be substituted.

We first tokenize the question $Q$ as a sequence of tokens $Q = (q_1, q_2, \ldots, q_m)$. We then classify each token into one of two categories: structural tokens that form the question's framework, and replaceable entities that could be substituted with alternatives. We define a classification function with fine-tuned RoBERTa, detailed in Appendix D. Based on this classification, we partition the question tokens into two sets:

$$P = \{q_i \mid \text{if } q_i \text{ is a replaceable entity}\}, \quad (1)$$

$$Q_S = \{q_i \mid \text{if } q_i \text{ is a structural token}\}, \quad (2)$$

where $P$ represents the set of replaceable entities (which we call entity placeholders), and $Q_S$ represents the set of structural tokens that form the question's framework.

For each identified entity placeholder $p_i$ in $P$, we assign a semantic type (e.g., `person`, `location`, `date`). This creates a set of typed entities:

$$T = \{(p_1, \tau_1), (p_2, \tau_2), \ldots, (p_k, \tau_k)\}, \quad (3)$$

where each pair $(p_j, \tau_j)$ consists of a placeholder entity $p_j$ and its corresponding type $\tau_j$.

By combining the structure tokens $Q_S$ with the typed placeholders in $T$, we create a question template. For example, if $Q$ is "Which is taller, the Eiffel Tower or the Empire State Building?", the function would identify "taller", "Eiffel Tower", and "Empire State Building" as replaceable entities of type `adj` and `place`. The resulting template would be "Which is [adj], [place 1] or [place 2]?", where the bracketed terms are typed placeholders.

### 3.2 Similar Examples Generation

Once we have extracted the question structure, we generate similar document-question-answer pairs that follow the same question structure but with different entities.

**Reasoning Skills Taxonomy.** We build on established cognitive science literature (Bartha, 2013;

Bordalo et al., 2024) to define a taxonomy of 7 fundamental reasoning skills $\mathcal{S}$ that humans commonly employ when solving problems (e.g., *Deductive*, *Inductive*[1]). Each skill represents a distinct cognitive approach to processing information and drawing conclusions.

**Question Generation.** For each placeholder in the question structure, we generate alternative entities of matching types. We prompt an LLM to suggest contextually appropriate substitutes for each entity type $\tau_j$. This produces a collection of candidate similar questions $\hat{Q}_{\text{sim}}$ that share the structural pattern of the original question but contain different entities.

To ensure high-quality examples, we implement a validation process. We prompt the same LLM to evaluate the similarity between each candidate question and the original question structure:

$$\text{sim}(Q, \hat{q}) \geq \delta, \quad \hat{q} \in \hat{Q}_{\text{sim}}, \tag{4}$$

where $\delta \in [1, 10]$ is a threshold parameter. Only questions exceeding this threshold are retained, resulting in a filtered set of similar questions $Q_{\text{sim}}$.

**Reasoning Strategy Construction.** For each similar question $Q^i_{\text{sim}} \in Q_{\text{sim}}$, we decompose it into a sequence of subquestions:

$$Q^i_{\text{sim}} \rightarrow (Q^{(i,1)}_{\text{sim}}, \ldots, Q^{(i,K)}_{\text{sim}}), \tag{5}$$

where each subquestion $Q^{(i,K)}_{\text{sim}}$ represents a discrete reasoning step and $K$ is the number of subquestions. Here the $K$ is not a fixed constant parameter. This variation occurs because we deliberately allow the language model to determine the appropriate number of subquestions based on the specific complexity and structure of each original question. And the connections between subquestions are characterized by specific reasoning skills from our taxonomy. This decomposition allows us to construct a comprehensive reasoning strategy:

$$\mathbf{s}^i = (s^i_1, s^i_2, \ldots, s^i_K), \tag{6}$$

where each $s^i_k \in \mathcal{S}$ is the reasoning skill required to transition from subquestion $Q^{(i,k)}_{\text{sim}}$ to $Q^{(i,k+1)}_{\text{sim}}$.

**Reference Document Generation.** For each subquestion $Q^{(i,k)}_{\text{sim}}$, we generate a document segment $d^i_k$ containing the precise information needed to answer that subquestion. The complete reference document for question $Q^i_{\text{sim}}$ is then constructed as:

$$D^i_{\text{ref}} = \{d^i_1, d^i_2, \ldots, d^i_K\}. \tag{7}$$

For example, given a similar question like "Which is taller, A or B?", the decomposition might yield subquestions: "What is the height of A?", "What is the height of B?", and "Which height is greater?". The reasoning strategy would connect these using deductive reasoning, and the reference document would provide the necessary height information for both entities.

The complete collection of similar examples is represented as:

$$\Gamma = \{(D^i_{\text{ref}}, Q^i_{\text{sim}}, \mathbf{s}^i)\}_{i=1}^N, \tag{8}$$

where $N$ is the total number of similar examples. This diverse set covers various reasoning strategies of different complexity levels, allowing our system to later select the most appropriate reasoning approach for original questions. Detailed examples to show how subquestion and skills correspondence can be found in Appendix A.1.

### 3.3 Multi-Criteria Matching

When presented with the original question $Q$ and documents $D$, we need to determine which reasoning strategy would be most effective. We select the most relevant example from our similar collection $\Gamma$ using a multi-criteria matching process that considers both reasoning skill requirements and structural similarity.

**Skill Uniqueness Scoring.** Recognizing that some reasoning skills are more specialized than others, we weight skills by their rarity in our example collection. For each reasoning skill $s \in \mathcal{S}$, we define $\text{freq}(s)$ as the number of examples in $\Gamma$ that include skill $s$ in their reasoning paths. The uniqueness score of a skill is:

$$\alpha(s) = \ln\left(\frac{N+1}{\text{freq}(s)+1}\right), \tag{9}$$

where $N$ is the total number of examples in our collection. This logarithmic formulation assigns higher weights to skills that appear less frequently, capturing the intuition that specialized reasoning skills deserve special consideration.

[1]Appendix A shows the complete taxonomy of reasoning skills with their description and example applications.

**Skill Coverage Assessment.** For each example in our collection, we calculate how well its reasoning path covers reasoning skills:

$$\text{cover}(\mathbf{s}^i, \mathcal{S}) = \frac{|\mathbf{s}^i \cap \mathcal{S}|}{|\mathcal{S}|}. \tag{10}$$

This coverage metric quantifies what proportion of the required reasoning skills are present in the example's reasoning strategy.

**Integrated Selection Score.** We compute a comprehensive selection score for each remaining example, and the optimal example is selected as:

$$i^* = \arg\max_i \left( \text{cover}(\mathbf{s}^i, \mathcal{S}) + \sum_{\ell=1}^{L} \alpha(s_\ell^i) \right), \tag{11}$$

where $L$ is the length of the reasoning strategy $\mathbf{s}^i$. This score balances how well the example covers the required reasoning skills and how uniquely it captures specialized reasoning approaches.

### 3.4 Reasoning Strategy-Guided Answering

The final component of $T^2$ uses the selected example to guide the reasoning process for answering the original question. Algorithm 1 outlines this process.

The "ExtractRelevantSegment" function uses LLM to identify portions of the document $D$ that are most relevant to applying a particular reasoning skill. This focuses the model's attention on information appropriate to each step of the reasoning process. The "FormatPrompt" function combines the original question, the focused document segments, the selected reasoning strategy, and the example document-question-answer pair into a comprehensive prompt. This prompt instructs the language model to answer the original question by applying the reasoning skills in the selected strategy, using the example as a demonstration of the reasoning approach.

This methodology enables adaptive reasoning that scales with question complexity. For simple questions, $T^2$ selects examples with a straightforward reasoning strategy, avoiding unnecessary computational overhead. For complex questions, it selects examples with a more sophisticated reasoning strategy that guides the model through the necessary steps to arrive at the correct answer. Importantly, this adaptation occurs without parameter tuning or multiple reasoning attempts, requiring only a single forward pass through the language model.

## 4 Experiments

### 4.1 Experimental Setups

**Datasets.** We evaluate our approach on seven QA datasets from diverse domains. **SQuAD** (general-domain questions from Wikipedia) (Rajpurkar et al., 2018), **HotpotQA** (multihop questions spanning multiple paragraphs) (Yang et al., 2018), **BioASQ** (biomedical queries requiring specialized knowledge) (Tsatsaronis et al., 2015), **NewsQA** (news-related passages) (Trischler et al., 2017), **GAOKAO** (exam-oriented dataset with academic coverage) (Zhang et al., 2024b), **HQA** (historical questions focusing on chronology and figures) (Hosen et al., 2023), and **TriviaQA** (Wikipedia-based trivia) (Joshi et al., 2017). Appendix B summarizes dataset sizes and domains.

**Reasoning Strategies and Metrics.** We compare our $T^2$ framework against *slow-thinking* and *quick-thinking* baselines. Slow-thinking approaches include: **zero-shot CoT** and **few-shots CoT**, **proactiveCoT (proCoT)** (Deng et al., 2023), **Self-Consistency** (Wang et al., 2022), **Tree of Thoughts (ToT)** (Yao et al., 2023), and **Monte Carlo Tree Search (MCTS)** (Zhao et al., 2024). Quick-thinking methods include: **few-shot prompting** and **direct prompting** without explicit reasoning steps. For evaluation, we use ROUGE-L as our metric across all datasets.[2]

**Large Language Models.** We use two quick-thinking LLMs (**Qwen2.5-32B-Instract** (Yang et al., 2024), and **GPT-4o** (Hurst et al., 2024; Guo et al., 2025)) and several slow-thinking LLMs (**GPT-o1/3/4 series** (Jaech et al., 2024), **QwQ-32B-Preview** (Qwen, 2025), **Claude-3.7** (Anthropic, 2025), **Gemini-2.5-Pro** (Gemini, 2025b)). Unless otherwise specified, hyperparameters are set to the default values for each model. No domain-specific fine-tuning and no target-designed prompt are applied, ensuring a fair and consistent comparison. More detailed implementation and all prompts can be found in Appendices D and E.

### 4.2 Results

Table 1 compares ROUGE-L on seven QA benchmarks. The upper half lists *quick-thinking models* evaluated with several slow-thinking frame-

[2]We recognize GenAI can generate the correct answer, but with different literalness. Hence we use ROUGE-L here instead of resulting in a misleadingly low Exact Match (EM) rate. We also report EM performance in Appendix F.

**Algorithm 1** Reasoning Path-Guided Answering

**Require:** $Q$ (original question), $D$ (document), $i^*$ (selected example index), $\Gamma$ (example collection)
**Ensure:** $A$ (final answer)

$(D^{i^*}_{\text{ref}}, Q^{i^*}_{\text{sim}}, A^{i^*}_{\text{sim}}, \mathbf{s}^{i^*}) \leftarrow \Gamma[i^*]$ ▷ Retrieve selected example
$D_{\text{focus}} \leftarrow \emptyset$ ▷ Initialize focused document segments
**for** $\ell = 1$ to $|\mathbf{s}^{i^*}|$ **do** ▷ For each skill in the reasoning path
  $\text{text}_\ell \leftarrow \text{ExtractRelevantSegment}(D, s^{i^*}_\ell)$ ▷ Extract relevant text for skill $s^{i^*}_\ell$
  $D_{\text{focus}} \leftarrow D_{\text{focus}} \cup \{\text{text}_\ell\}$ ▷ Add to focused segments
**end for**
$\text{Prompt} \leftarrow \text{FormatPrompt}(Q, D_{\text{focus}}, \mathbf{s}^{i^*}, Q^{i^*}_{\text{sim}}, A^{i^*}_{\text{sim}})$ ▷ Construct guidance prompt
$A \leftarrow \text{LLM}(\text{Prompt})$ ▷ Generate answer with guided reasoning
**return** $A$

| Model | SQuAD | HotpotQA | NewsQA | Gaokao | HQA | TriviaQA | BioASQ |
|---|---|---|---|---|---|---|---|
| ***Quick-Thinking Models w/ Reasoning Strategies*** | | | | | | | |
| ***Qwen2.5-32B-Instruct*** | | | | | | | |
| w/ vanilla *(quick)* | 73.41 | 55.32 | 50.83 | 29.52 | 35.92 | 40.73 | 56.33 |
| w/ few-shots *(quick)* | 74.56 | 56.23 | 51.67 | 30.33 | 36.87 | 41.57 | 57.17 |
| w/ zero-shot CoT *(slow)* | 76.23 | 57.41 | 52.89 | 30.92 | 37.65 | 42.31 | 58.12 |
| w/ few-shot CoT *(slow)* | 77.08 | 58.15 | 53.62 | 31.37 | 38.01 | 42.79 | 58.58 |
| w/ self-consistency (Wang et al., 2022) | 75.31 | 56.76 | 52.27 | 30.57 | 37.12 | 41.92 | 57.57 |
| w/ proCoT (Deng et al., 2023) | 77.12 | 58.07 | 53.57 | 31.42 | 38.03 | 42.83 | 58.62 |
| w/ ToT (Yao et al., 2023) | 78.47 | 59.11 | 54.31 | 31.96 | 38.66 | 43.46 | 59.36 |
| w/ MCTS (Zhao et al., 2024) | 78.52 | 58.97 | 54.25 | 32.04 | 38.73 | 43.51 | 59.42 |
| **w/ $T^2$ (ours)** | **81.86** | **67.11** | **61.27** | **34.06** | **40.31** | **43.92** | **65.02** |
| ***GPT-4o*** | | | | | | | |
| w/ vanilla *(quick)* | 78.52 | 60.02 | 55.32 | 34.51 | 41.11 | 49.01 | 60.51 |
| w/ few-shots *(quick)* | 79.86 | 61.06 | 56.17 | 35.36 | 42.06 | 50.07 | 61.37 |
| w/ zero-shot CoT *(slow)* | 81.27 | 62.38 | 57.25 | 36.12 | 42.91 | 50.89 | 62.35 |
| w/ few-shot CoT *(slow)* | 82.05 | 62.97 | 57.81 | 36.58 | 43.32 | 51.38 | 62.83 |
| w/ self-consistency (Wang et al., 2022) | 80.56 | 61.61 | 56.62 | 35.62 | 42.46 | 50.42 | 61.81 |
| w/ proCoT (Deng et al., 2023) | 82.12 | 63.02 | 57.86 | 36.66 | 43.36 | 51.46 | 62.87 |
| w/ ToT (Yao et al., 2023) | 83.21 | 64.06 | 58.67 | 37.22 | 44.07 | 52.26 | 63.72 |
| w/ MCTS (Zhao et al., 2024) | 83.35 | 64.18 | 58.19 | 37.31 | 45.15 | 52.38 | 64.89 |
| **w/ $T^2$ (ours)** | **85.06** | **66.16** | **60.92** | **37.57** | **45.27** | **53.92** | **66.97** |
| ***Slow-Thinking Models*** | | | | | | | |
| o1-mini | 85.81 | 70.91 | 63.22 | 42.66 | 49.22 | 58.56 | 68.42 |
| QwQ-32B-Preview | 86.87 | 71.86 | 63.92 | 43.23 | 49.62 | 59.16 | 69.02 |
| DeepSeek-R1 | 87.62 | 72.72 | 64.41 | 43.47 | 50.27 | 60.02 | 70.72 |
| o1 | 88.22 | 73.37 | 65.11 | 44.06 | 51.07 | 60.86 | 71.36 |
| o4-mini | 88.72 | 73.86 | 65.57 | 44.32 | 51.61 | 61.11 | 71.82 |
| o4-mini-high | 88.91 | 74.07 | 65.81 | 44.52 | 51.86 | 61.27 | 72.02 |
| Claude-3.7-sonnet-thinking | 89.11 | 74.21 | 66.01 | 44.61 | 52.01 | 61.47 | 72.22 |
| o3 | 89.41 | 74.61 | 66.32 | 45.01 | 52.11 | 61.81 | 72.62 |
| Gemini-2.5-Pro | 90.27 | 75.46 | 67.11 | 45.76 | 53.07 | 62.68 | 73.57 |
| **QwQ-32B + $T^2$ (ours)** | **92.12** | **77.61** | **68.61** | **47.42** | **54.71** | **64.22** | **75.21** |

Table 1: ROUGE-L on seven QA datasets. We regard vanilla model and few-shot method as quick-thinking methods. And the other five (including ours) are slow-thinking methods. They can all be applied to quick-thinking models to improve reasoning ability.

works. The lower half gathers the strongest *slow-thinking models*. We also report the performance of Qwen2.5-32B-Instruct + $T^2$ and QwQ-32B-Preview + $T^2$ to show comparison with slow-thinking models. The experimental results show that by comparison with other thinking strategies, our $T^2$ could help quick-thinking model achieve better performance. And compared with other slow-thinking models, adding our $T^2$ can also help model improve the performance.

Besides, we analyze the inference time requirements across all baseline methods. Table 2 presents the average inference time (in seconds) for all methods across our experiments using both Qwen2.5-32B and QwQ-32B models. The results demonstrate that while vanilla and few-shot ap-

| Model | Avg. Inference Time (s) | Time Reduction vs. MCTS |
|---|---|---|
| *Qwen2.5-32B* | 23.17 | -70.6% |
| w/ few-shots | 25.43 | -67.8% |
| w/ CoT | 43.26 | -45.2% |
| w/ few-shots CoT | 68.21 | -13.6% |
| w/ self-consistency | 65.31 | -17.2% |
| w/ proCoT | 58.76 | -25.5% |
| w/ ToT | 72.48 | -8.1% |
| w/ MCTS | 78.92 | - |
| w/ $T^2$ (ours) | 34.52 | -56.3% |
| *QwQ-32B* | 27.35 | -69.9% |
| w/ few-shots | 29.81 | -67.1% |
| w/ CoT | 51.42 | -43.1% |
| w/ few-shots CoT | 79.63 | -12.1% |
| w/ self-consistency | 76.74 | -15.3% |
| w/ proCoT | 72.95 | -19.4% |
| w/ ToT | 84.37 | -6.9% |
| w/ MCTS | 90.58 | - |
| w/ $T^2$ (ours) | 45.03 | -50.3% |

Table 2: Average inference time comparison across methods. Our $T^2$ approach achieves a significant reduction in computational cost compared to MCTS while maintaining superior accuracy.

proaches are indeed faster, they achieve substantially lower accuracy as shown in our previous experiments. Our $T^2$ approach achieves an optimal balance between computational efficiency and performance, reducing inference time by 56.3% compared to MCTS with Qwen2.5-32B and 50.3% with QwQ-32B. This significant reduction in computational cost, while maintaining superior accuracy as demonstrated in our previous experiments, addresses one of the key challenges identified in our introduction.

Additionally, we conduct several analysis experiments detailed as follows.

#### 4.2.1 $T^2$ Enhance the Reasoning Skills Hit Rate while Reducing the Error

HotpotQA supplies gold supporting sentences for every question, hence we use these to evaluate reasoning quality. For a model output that mentions a set $P_q$ of sentences and a gold set $G_q$, we record a *Hit* if $P_q \supseteq G_q$ (all required facts retrieved) and an *Error* if $P_q \not\subseteq G_q$ (at least one spurious fact added). Thus Hit measures *completeness*, Error measures *precision*, and the two are inversely related: longer chains tend to raise Hit but also raise Error. Figure 2(left) shows that quick-thinking frameworks give low Hit and moderate Error, while slow-thinking methods improve Hit at the cost of higher Error. Our $\mathbf{T}^2$ strikes the best balance, achieving the highest Hit and the lowest Error on Qwen2.5-32B, confirming that adaptive path length yields the most accurate multihop reasoning. The detailed calculation of Hits and Errors can be found in Appendix G.1.

| Skill Type | Uniform | Ours | Improvement |
|---|---|---|---|
| Deductive | 72.3% | 75.8% | +3.5% |
| Inductive | 68.7% | 73.2% | +4.5% |
| Abductive | 74.1% | 76.3% | +2.2% |
| Cause & Effect | 70.5% | 74.1% | +3.6% |
| Analogical | 63.8% | 71.5% | +7.7% |
| Critical Thinking | 69.2% | 72.8% | +3.6% |
| Decompositional | 61.4% | 69.7% | +8.3% |

Table 3: Performance comparison between uniform and our matching strategies.

#### 4.2.2 $T^2$ Tends to Get Correct Answers Immediately without Retrace

A response is said to *retrace* if the model announces a provisional conclusion and later back-tracks on it inside the same output (e.g., "*So the answer is X… wait, that seems wrong—let me revise… the answer is Y*"). Obviously, as retrace brings extra computing cost, it would be better for a model to ensure a lower retrace rate while maintaining the same accuracy. Concretely, we scan the CoT for either (i) <*answer*> markers that appear more than once, or (ii) lexical repair cues such as "sorry," "actually," or "let me rethink," followed by a different answer span; if either pattern occurs, the example counts as a retrace. Figure 2 (right) shows that, taking Qwen2.5-32B as LLM, slow-thinking methods retrace more on NewsQA and HQA, whereas quick-thinking methods seldom retrace but miss clues, hurting performance. Our $\mathbf{T}^2$ keeps both metrics low—matching the speed of quick thinking and the accuracy of slow thinking—demonstrating that adaptive path length minimises wasted reasoning. The detailed calculation of Hits and Errors can be found in Appendix G.2.

#### 4.2.3 $T^2$ Costs Fewer Tokens to Achieve Superior Performance

To evaluate the efficiency of our $T^2$, we compare four reasoning approaches: (1) Qwen2.5-32B w/ self-consistency, a typical slow-thinking method, (2) QwQ-32B-Preview, another slow-thinking model, (3) Qwen2.5-32B w/ $T^2$, and (4) QwQ-32B w/ $T^2$, our adaptive reasoning methods. Figure 3 shows that our method reduces token consumption by 25.2% compared to QwQ-32B-Preview, and by 14.8% compared to Qwen2.5-32B

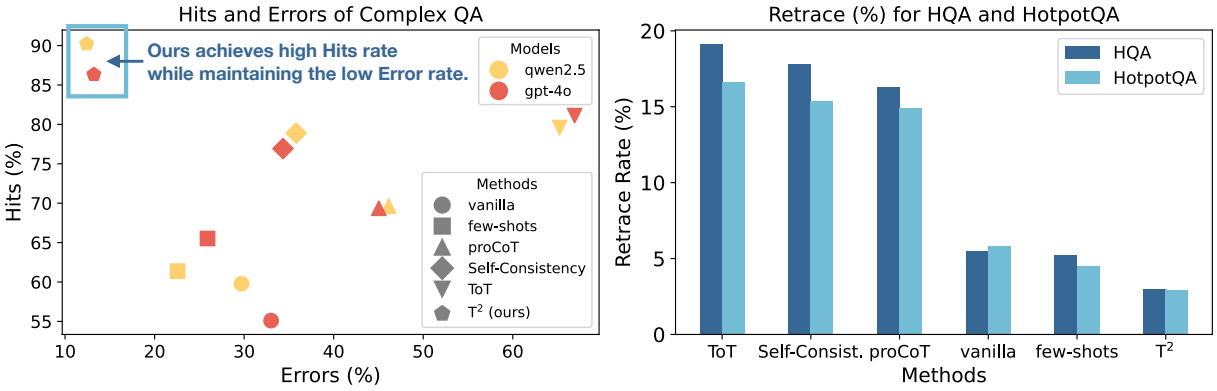


Figure 2: Results on Hits and Errors (left) and Retrace Rate (right).

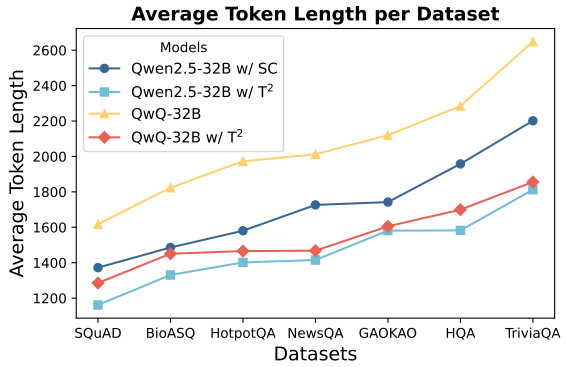


Figure 3: Results of average token length on each dataset. SC is the abbreviation for Self-Consistency.

w/ self-consistency, while maintaining competitive accuracy. These findings highlight that our method achieves an optimal trade-off between computational efficiency and reasoning quality. A full comparison, including token usage and performance across datasets, is provided in Appendix I.

### 4.3 Similar Examples Quality Analysis

**Our Matching Strategy Can Expose More Diverse Reasoning Skills.** The effectiveness of our framework relies not only on identifying appropriate reasoning skills but also on how these skills are matched during the example selection. Hence, we examine the impact of our multi-criteria reasoning skills matching strategy compared to a naive uniform sampling approach. Table 3 presents the results of our experiment against uniform sampling across different reasoning skill types. Our approach consistently outperforms uniform sampling across all skill categories, with particularly notable improvements for less frequent reasoning types such as decompositional reasoning (+8.3%) and analogical reasoning (+7.7%). This confirms our hypothesis that the strategic balancing of skill demonstrations enhances the model's ability to leverage diverse reasoning patterns. The distribution of each reasoning skill can be found in Appendix C. Ablation study of multi-criteria matching strategy can be found in Appendix H.

**Accuracy of Reasoning Skills Results in Correctness of Answers.** We examined the correlation between the accuracy of selected reasoning skills and the correctness of final answers using the HotpotQA dataset. We conducted the experiment on two models: Qwen2.5-32B-Instruct w/ $T^2$ and QwQ-32B-Preview w/ $T^2$. The analysis, shown in Figure 5, reveals a strong positive correlation between skill accuracy and answer correctness. Higher skill accuracy corresponds to higher answer correctness, with an approximate 5-6% increase in correctness for every 5% improvement in skill accuracy. These results demonstrate that accurately selecting the correct reasoning skills is essential for generating correct answers, especially in complex multi-hop reasoning tasks.

We also discuss the impacts of question structure (J.1), impacts of numbers of similar examples (J.3), impacts of various generated methods (J.4), impacts of threshold of similarity in generation (J.5), impacts of examples domain bias and structural bias (J.6), and human evaluation (J.7) in Appendix.

### 4.4 Case Study

Figure 4 shows an short version of example to show effectiveness of our $T^2$. By explicitly providing the model-specific reasoning path, the model can generate the correct answer with an appropriate reasoning chain of thought. The detailed case studies can be found in Appendix K.

**[Reference Documents]**
**Question:** In what city was the subject of the film Nowhere Boy born?

**Proper Reasoning Chain:**
1. Decompositional: Find the (a) film subject, (b) born place
2. Deductive: Nowhere Boy is about John Lennon
3. Deductive: John was born in Liverpool

**Quick Thinking Model's Wrong Answer:**
The subject of Nowhere Boy was born in London.

**Slow Thinking Model's Overthinking Answer:**
[After a lengthy analysis of various biographical details concerning] John Lennon ... was born in Liverpool.

**Model with our FReM's Correct Answer:**
Since Nowhere Boy is a film about John Lennon (Doc 2) and Doc 1 confirms that John was born in Liverpool. We deduce the answer is Liverpool.

Figure 4: Case study to show effectiveness of our $T^2$ framework. There are three proper reasoning skills should be adopted to answer the question based on given documents. The red, orange, and green answers represent responses under quick thinking, slow thinking, and ours, respectively.

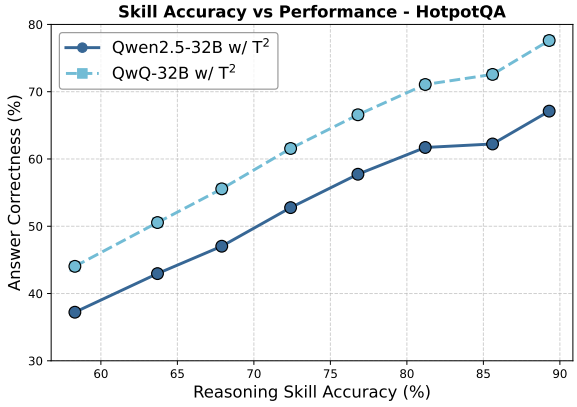


Figure 5: Results on relationship between reasoning skills' accuracy and overall performance.

## 5 Conclusion

In this paper, we introduced $T^2$: Think-to-Think, a novel framework that dynamically adapts reasoning depth based on question complexity for contextual question answering tasks. Unlike prior approaches that employ fixed reasoning strategies regardless of question difficulty, $T^2$ enables models to learn appropriate reasoning strategies from similar examples, leading to more efficient processing while maintaining accuracy. Our experimental results across seven diverse CQA benchmarks confirm that $T^2$ not only achieves higher accuracy than baseline methods but also reduces computational overhead by up to 25.2%. These improvements demonstrate the value of adaptability in reasoning processes, suggesting that as language models continue to evolve, approaches like $T^2$ that optimize both accuracy and computational efficiency will become increasingly important for developing more intelligent systems that can effectively allocate computational resources based on task demands.

## Limitations

While $T^2$: Think-to-Think demonstrates promising results across various CQA benchmarks, we acknowledge several limitations of our approach: First, the effectiveness of $T^2$ relies on the availability of high-quality example reasoning strategy for similarity matching. In domains with limited annotated examples or highly novel questions, the framework may struggle to identify appropriate reasoning patterns, potentially defaulting to less optimal strategies. Besides, our current implementation focuses primarily on textual reasoning tasks. Extending $T^2$ to multimodal reasoning contexts (e.g., visual question answering) would require additional architectural modifications to handle diverse input modalities while maintaining computational efficiency. Despite these limitations, we believe $T^2$ represents a significant step toward more adaptive and efficient reasoning systems that can intelligently allocate computational resources based on question complexity.

## Ethical Considerations

We ensure that all experiments are conducted using publicly available, ethically sourced datasets, adhering to privacy and intellectual property guidelines. We acknowledge the potential for biases in data and are committed to evaluating and mitigating any such biases in $T^2$.


## Acknowledgements

We thank the reviewers, the AC, and the SAC for their constructive comments. This work is partially supported by Hong Kong RGC GRF No.14206324, CUHK Knowledge Transfer Project Fund No.KPF23GWP20, and Research Funds for NSD Construction, University of International Re-

lations (Grant numbers: 2024GA07). We'd also like to thank Ms. Ziya Zhou for her feedback on refining our figure.

## References

Anthropic. 2025. Claude 3.7 sonnet system card. Technical report.

Paul Bartha. 2013. Analogy and Analogical Reasoning. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*, Fall 2024 edition. Metaphysics Research Lab, Stanford University.

Pedro Bordalo, Nicola Gennaioli, Giacomo Lanzani, and Andrei Shleifer. 2024. A cognitive theory of reasoning and choice.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*.

Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei A Zaharia, and James Y Zou. 2024. Are more llm calls all you need? towards the scaling properties of compound ai systems. *Advances in Neural Information Processing Systems*, 37:45767–45790.

Yang Deng, Lizi Liao, Liang Chen, Hongru Wang, Wenqiang Lei, and Tat-Seng Chua. 2023. Prompting and evaluating large language models for proactive dialogues: Clarification, target-guided, and non-collaboration. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10602–10621, Singapore. Association for Computational Linguistics.

Google Gemini. 2025a. Gemini 2.5 flash thinking mode.

Google Gemini. 2025b. Gemini 2.5: Our most intelligent ai model. Technical report.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Sabbir Hosen, Jannatul Ferdous Eva, Ayman Hasib, Aloke Kumar Saha, M.F. Mridha, and Anwar Hussen Wadud. 2023. Hqa-data: A historical question answer generation dataset from previous multi perspective conversation. *Data in Brief*, 48:109245.

Zhen Huang, Haoyang Zou, Xuefeng Li, Yixiu Liu, Yuxiang Zheng, Ethan Chern, Shijie Xia, Yiwei Qin, Weizhe Yuan, and Pengfei Liu. 2024. O1 replication journey–part 2: Surpassing o1-preview through simple distillation, big progress or bitter lesson? *arXiv preprint arXiv:2411.16489*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ke Ji, Jiahao Xu, Tian Liang, Qiuzhi Liu, Zhiwei He, Xingyu Chen, Xiaoyuan Liu, Zhijie Wang, Junying Chen, Benyou Wang, and 1 others. 2025. The first few tokens are all you need: An efficient and effective unsupervised prefix fine-tuning method for reasoning models. *arXiv preprint arXiv:2503.02875*.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Uri Katz, Matan Vetzler, Amir Cohen, and Yoav Goldberg. 2023. Neretrieve: Dataset for next generation named entity recognition and retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3340–3354.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *CoRR*, abs/2212.14024.

Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5315–5333.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting for retrieval-augmented large language models. *CoRR*, abs/2305.14283.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, and 1 others. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *arXiv preprint arXiv:2412.09413*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 5687–5711. Association for Computational Linguistics.

Qwen. 2024. Qwq: Reflect deeply on the boundaries of the unknown.

Qwen. 2025. Qwq-32b: Embracing the power of reinforcement learning. Technical report.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 9248–9274. Association for Computational Linguistics.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. 2025. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10014–10037. Association for Computational Linguistics.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artieres, Axel Ngonga, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, and 3 others. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16:138.

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*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. 2024. An empirical analysis of compute-optimal inference for problem-solving with language models.

Mayi Xu, Yongqi Li, Ke Sun, and Tieyun Qian. 2024. Adaption-of-thought: Learning question difficulty improves large language models for reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5468–5495.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. Technical report.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for llm reasoning. *arXiv preprint arXiv:2502.18080*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Yunhua Zhou, and Xipeng Qiu. 2025. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? *arXiv preprint arXiv:2502.12215*.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. 2024a. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. 2024b. Evaluating the performance of large language models on gaokao benchmark. *Preprint*, arXiv:2305.12474.

Yu Zhao, Huifeng Yin, Bo Zeng, Hao Wang, Tianqi Shi, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2024. Marco-o1: Towards open reasoning models for open-ended solutions. *arXiv preprint arXiv:2411.14405*.

# A Full Reasoning Skills

Defined by (Bartha, 2013; Bordalo et al., 2024), reasoning can best be defined as the basic action of thinking in a sensible and rational way about something. Reasoning is the ability to assess things rationally by applying logic based on new or existing information when making a decision or solving a problem. Based on their conclusion, Tables 5 and 6 show the reasoning skills for answering a certain question.

## A.1 Extended Explanation of Sub-Questions and Reasoning Skills

Subquestions serve as atomic reasoning operations that systematically decompose complex questions into manageable components. Each subquestion corresponds to a specific cognitive task that must be completed to progress toward answering the original question. This decomposition allows the model to focus on discrete information pieces sequentially while applying appropriate reasoning skills at each step. Consider the question: "Which is taller, the Eiffel Tower or the Empire State Building?" Table 4 shows the decomposition process. This structured approach reduces cognitive load by isolating individual reasoning steps and creates explicit intermediate reasoning states that can be verified independently.

# B Datasets

In this work, we evaluate our method on seven widely used question answering datasets. Each dataset presents distinct characteristics, ranging from the type of questions asked to the domain in which they are applied. Below, we provide a brief overview of each dataset.

**SQuAD** consists of over 100,000 question-answer pairs derived from a set of Wikipedia articles. The task is to find the span of text that answers the question. SQuAD is widely used for evaluating machine reading comprehension models. The dataset includes two versions: SQuAD 1.1, which contains answerable questions, and SQuAD 2.0, which also includes unanswerable questions, making it more challenging. We use 2.0 version here.

**HotpotQA** is a large-scale, multi-hop question answering dataset that requires reasoning across multiple supporting facts. The dataset includes over 113,000 question-answer pairs spanning various domains, where answers cannot be found in a single sentence or passage but require combining information from several documents. The questions in HotpotQA require a more complex reasoning process compared to typical single-hop datasets.

**BioASQ** is a biomedical question answering dataset that provides information from scientific articles, primarily in the domain of biomedicine. It includes both factoid and complex questions that require understanding of scientific literature. BioASQ focuses on answering clinical, biomedical, and molecular biology-related questions using both structured and unstructured data sources.

**NewsQA** is a dataset designed for reading comprehension tasks. It consists of over 100,000 question-answer pairs derived from news articles. The challenge of NewsQA lies in answering questions about real-world events from unstructured news stories, requiring models to handle various linguistic phenomena such as coreference, reasoning, and implicit understanding.

**GAOKAO** is a dataset derived from the Chinese college entrance exam, also known as the "Gaokao". It contains questions related to various subjects, including Chinese literature, mathematics, and English. The questions in GAOKAO require both general knowledge and reasoning to answer. This dataset is specifically designed for the Chinese education system and is widely used in academic and educational research in China.

**HQA** is a human-annotated dataset specifically designed for complex, open-domain question an-

| Subquestion | Reasoning Skill | Purpose |
|---|---|---|
| "What is the height of the Eiffel Tower?" | Deductive | Establishes first measurement |
| "What is the height of the Empire State Building?" | Deductive | Establishes second measurement |
| "Which height value is greater?" | Cause and Effect | Determines the final answer |

Table 4: Example of question decomposition with associated reasoning skills

| Type of Reasoning | Detailed Description | Example |
|---|---|---|
| **Deductive** | Deductive reasoning occurs when generalized statements apply to specific cases. These generalized statements are established and already proven, making specific cases easy to deduce. For example, all humans are mortals. Bill is a human, so Bill must be mortal. In this example the generalized, but proven, statement, "all humans are mortals" is what drives the reasoning. | Document: All shapes with three sides are triangles. A certain figure here has exactly three sides.<br>Question: What is this figure called?<br>Answer: It is a triangle. All shapes with three sides are triangles, and this figure has three sides. So it must be a triangle. |
| **Inductive** | Inductive reasoning is similar to deductive reasoning in that they both draw a conclusion based on a statement. However, in inductive reasoning, the statement is likely but has not been proven. For example, roses usually bloom in spring. In spring, one can count on there being roses. Again, the difference is that this is likely but not proven to be 100%. | Document: Every spring for the past ten years, wild roses in Green Valley have bloomed in late March. This spring is about to begin in Green Valley.<br>Question: Will the wild roses bloom in late March this year?<br>Answer: It is likely they will bloom in late March, because they usually do, but it is not guaranteed. |
| **Abductive** | Abductive reasoning is the act of making a conclusion based on what you already know. For example, if you see a plate of food still hot, but half-eaten, you can make the conclusion that the person eating that food is probably returning soon. | Document: You notice a half-eaten sandwich and a still-hot cup of coffee on a café table. The seat feels warm, and a jacket is draped over the chair.<br>Question: Has the person who was sitting here left permanently, or are they coming back soon?<br>Answer: It is likely they just stepped away for a moment and will return, because the food and drink are still warm and their jacket remains on the chair. |
| **Cause & Effect** | Cause and effect reasoning is that if x happens then y will happen as a result. This is extremely persuasive when making a speech or trying to get someone to take action to cause an effect. For example, a politician may say that if they are elected, then poverty will decrease. This is using cause and effect reasoning in a real-world situation. | Document: Meteorologists predict heavy rain this evening, with warnings that streets may flood if the rainfall continues.<br>Question: Will the roads become dangerous as a result of this weather?<br>Answer: Yes. If heavy rain continues, roads will likely flood and become slippery, causing drivers to have less control of their vehicles. |

Table 5: (1/2) Full list of reasoning skills used in the reasoning path construction.

swering. It contains questions that require deep contextual understanding and can involve reasoning across long documents. The dataset includes various types of questions and answers across diverse domains, and it was created to test models' ability to perform reasoning tasks in realistic, open-ended settings.

**TriviaQA** is a large-scale dataset that focuses on answering trivia questions, where each question is associated with a corresponding set of supporting documents. TriviaQA contains over 650,000 question-answer pairs sourced from trivia websites and requires models to retrieve relevant information from the documents and answer based on the provided facts. The dataset has questions spanning various topics such as history, geography, and general knowledge.

### B.1 Application Scope and Limitations

Our research specifically focuses on Contextual Question Answering (CQA) tasks, which represent a distinct reasoning paradigm from other complex reasoning domains (like math or code). The pro-

| Type of Reasoning | Detailed Description | Example |
|---|---|---|
| **Analogical** | Analogical reasoning is the use of a comparison between two things to persuade that there must be more in common if they already share something. For example, if x, y, and z all share this trait, then they must also share other traits. The foundation of this type of reasoning is perfect for speeches and comparisons in the real world. If there are connections between x and y already, then they must have several other things in common as well. | Document: Many leading technology companies emphasize continuous learning and adaptability. For instance, Google, Microsoft, and Amazon all invest in regular training programs and encourage innovation among employees. Their similar approach to fostering a culture of growth has been linked to their strong performance in rapidly changing markets.<br>Question: Can we infer that a company that promotes continuous learning will also likely be successful in adapting to market changes?<br>Answer: Yes. Since Google, Microsoft, and Amazon all share a culture of continuous learning and, as a result, demonstrate high adaptability and market success, it is reasonable to conclude by analogy that a company which also promotes continuous learning is likely to develop similar strengths. |
| **Critical Thinking** | Critical thinking occurs when you take all of the facts and develop a conclusion based on an analysis. This could happen subconsciously or intentionally, depending on the situation. For example, in the real world, critical thinking could be about your relationships. You could see a behavior you don't like about someone and have to think critically about whether or not you will choose to spend more time with this person. This is using critical thinking to develop reasoning in a real-world application. | Document: Over the past few months, Sam has repeatedly cancelled plans at the last minute and rarely communicated afterward.<br>Question: Should you invest time in a close friendship with Sam?<br>Answer: No. Sam's consistent behavior of last-minute cancellations suggests a pattern of unreliability, which may negatively affect the trust needed in a close friendship. |
| **Decompositional** | Decompositional reasoning happens when the different parts of the reasoning are broken down into smaller pieces and analyzed for how they contribute to the whole. The intent of this is to make the reasoning easier to understand and allow for analyzing how the parts equal the whole. For example, in order to understand the function of the human body, you would have to analyze each bone and organ to see how they all work together. Additionally, in the real world, an argument could be broken down into several smaller parts in order to analyze the effectiveness of the argument as a whole. | Document: A smartphone's quality can be understood by breaking it down into three parts: its design, performance, and battery life. The design covers the build and user interface; performance looks at processing speed and software efficiency; battery life shows how long the device operates on a single charge.<br>Question: Can we conclude that the smartphone provides a good overall user experience?<br>Answer: Yes. If the design is appealing, the performance is robust, and the battery life is long, then the smartphone is likely to offer a good overall experience. |

Table 6: (2/2) Full list of reasoning skills used in the reasoning path construction.

posed approach is designed to address efficiency challenges in general CQA tasks, where models often generate unnecessarily verbose reasoning for simple questions—particularly relevant for practical applications with constrained computational resources. We acknowledge that our method may not be directly applicable to highly structured reasoning domains such as mathematics, programming, and algorithmic reasoning, where approaches like Tree of Thought (ToT) and Monte Carlo Tree Search (MCTS) have demonstrated strong performance. This limitation stems from the fundamental differences between CQA tasks (which involve flexible reasoning patterns in natural language) and formal domains (which follow well-defined rules with constrained, sequential reasoning paths). Additionally, creating similar example questions for these structured domains presents significant challenges, as their solution spaces typically benefit from systematic exploration techniques rather than our adaptive reasoning approach.

## C Distribution of Reasoning Skills in Each Dataset

Table 7 demonstrates the distribution of seven reasoning skills in different datasets. The variance in skill distribution highlights why our multi-criteria matching approach is crucial. Without it, high-

frequency skills like deductive reasoning would dominate the demonstrations, while valuable but less common skills like abductive reasoning would be underrepresented.

# D Implementations

## D.1 For PLM Usage

We use a simple pretrained language model RoBERTa from Huggingface for detecting named entities or key numbers in the question to obtain the question structure. This classification task involves processing the input question to identify whether it contains a named entity or key number and assigning a type to the detected entity. The model performs this task by outputting binary labels (entity: Yes/No) first, and then the associated entity types (e.g., Person, Location, Date, Organization, Number, etc.).

This model is fine-tuned with a simple classification layer that detects whether a named entity or key number is present in the question with NERetrieve dataset[3] (Katz et al., 2023). This process leverages the model's pre-trained knowledge, with minimal fine-tuning specifically focused on the entity detection and classification task.

The hyperparameters used for fine-tuning the PLM are listed in Table 8. The batch size is set to 128. The learning rate is set to $2 \times 10^{-5}$. AdamW is used as the optimizer. A dropout rate of 0.1 is applied to prevent overfitting during fine-tuning.

## D.2 Why Choose PLM?

While modern LLMs can generate similar question decompositions in a zero-shot manner, our fine-tuned encoder approach offers several advantages. The computational cost of our fine-tuned RoBERTa model is negligible compared to prompting an LLM (approximately 0.1% of inference time), making it highly efficient for decomposition tasks. Additionally, the fine-tuned encoder ensures consistent identification of structural elements, which is crucial for generating diverse yet structurally similar questions.

To validate our approach, we conducted additional experiments comparing: (1) our fine-tuned RoBERTa-based approach, (2) off-the-shelf NER models (RoBERTa), and (3) zero-shot LLM prompting for question decomposition. Table 9 presents the impact of different decomposition methods on the final performance across datasets.

We also conducted both automatic and human evaluations of the decomposition quality. For automatic evaluation, we used GPT-4 to assess the structural accuracy, element boundary precision, and template usability of decompositions generated by each method on a scale of 1-5. For human evaluation, we measured exact match, relaxed match, and structural correctness. Tables 10 and 25 present these results.

In our experiments, we used Qwen3-0.6B (600M parameters) as our zero-shot LLM, which is comparable in size to our fine-tuned RoBERTa model (approximately 400M parameters). This allows for a fair comparison where performance differences can be attributed to the approach rather than simply model scale advantages.

These results demonstrate that while all methods produce comparable final performance, our fine-tuned approach provides superior boundary detection for template elements (4.82 vs. 4.27 GPT-4 rating), higher exact match scores in human evaluation (87.6% vs. 78.9%), and minimal computational overhead while maintaining consistency.

## D.3 For LLM Usage

For LLM usage, We use two quick-thinking LLMs (**Qwen2.5-32B-Instract** (Yang et al., 2024), and **GPT-4o** (Hurst et al., 2024; Guo et al., 2025)) and several slow-thinking LLMs (**GPT-o1/3/4 series** (Jaech et al., 2024), **QwQ-32B-Preview** (Qwen, 2025), **Claude-3.7** (Anthropic, 2025), **Gemini-2.5-Pro** (Gemini, 2025b)). For ToT implementation, we follow the original paper's approach (Yao et al., 2023) with a breadth-first search strategy and a maximum depth of 3. For MCTS, we implement the standard UCT algorithm with 10 simulations per decision point. For synthetic QA generation, we set a maximum output length of 4,096 tokens. When deciding which similar example to use, we follow our multi-criteria matching (Section 3.3) to pick the most relevant chain of skills. Unless otherwise specified, hyperparameters stay at default values for each model. No domain-specific fine-tuning and no targeted designed prompt are applied, ensuring a fair and consistent comparison. All inferences are based on vLLM framework.

[3] https://github.com/katzurik/NERetrieve?tab=readme-ov-file

| Skill Type | SQuAD | HotpotQA | NewsQA | GAOKAO | HQA | TriviaQA | BioASQ |
|---|---|---|---|---|---|---|---|
| Deductive | 0.31 | 0.22 | 0.28 | 0.15 | 0.42 | 0.18 | 0.25 |
| Inductive | 0.23 | 0.18 | 0.15 | 0.12 | 0.13 | 0.21 | 0.19 |
| Abductive | 0.05 | 0.12 | 0.08 | 0.21 | 0.09 | 0.15 | 0.11 |
| Cause & Effect | 0.12 | 0.15 | 0.13 | 0.22 | 0.08 | 0.19 | 0.14 |
| Analogical | 0.08 | 0.13 | 0.09 | 0.07 | 0.11 | 0.12 | 0.16 |
| Critical Thinking | 0.14 | 0.16 | 0.18 | 0.14 | 0.13 | 0.09 | 0.10 |
| Decompositional | 0.07 | 0.04 | 0.09 | 0.09 | 0.04 | 0.06 | 0.05 |

Table 7: Distribution (%) of reasoning skills across benchmark datasets, showing the proportion of questions requiring each skill type.

| Parameter | Value |
|---|---|
| **Model** | RoBERTa |
| **Full Name** | `FacebookAI/xlm-roberta-large-finetuned-conll03-english` |
| **Batch Size** | 128 |
| **Learning Rate** | 2e-5 |
| **Optimizer** | AdamW |
| **Dropout Rate** | 0.1 |
| **Evaluation Metric** | Accuracy |

Table 8: Implementation parameters for named entity detection and classification.

| Decomposition Method | SQuAD | HotpotQA | NewsQA | Gaokao | HQA | TriviaQA | BioASQ |
|---|---|---|---|---|---|---|---|
| Fine-tuned RoBERTa (Ours) | 85.06 | 66.16 | 60.92 | 37.57 | 45.27 | 53.92 | 66.97 |
| RoBERTa | 84.89 | 65.87 | 60.63 | 37.21 | 44.98 | 53.77 | 66.58 |
| Zero-shot LLM prompting | 84.72 | 65.69 | 60.41 | 37.06 | 44.83 | 53.61 | 66.42 |

Table 9: Impact of question decomposition method on final performance across datasets.

| Decomposition Method | Structural Accuracy | Element Boundary Precision | Template Usability |
|---|---|---|---|
| Fine-tuned RoBERTa (Ours) | 4.78 | 4.82 | 4.71 |
| RoBERTa | 4.63 | 4.41 | 4.52 |
| Zero-shot LLM prompting | 4.56 | 4.27 | 4.38 |

Table 10: GPT-4 evaluation of decomposition quality (scale 1-5).

| Decomposition Method | Exact Match | Relaxed Match | Structural Correctness |
|---|---|---|---|
| Fine-tuned RoBERTa (Ours) | 87.6% | 94.3% | 96.2% |
| RoBERTa | 81.4% | 93.2% | 95.7% |
| Zero-shot LLM prompting | 78.9% | 92.8% | 94.9% |

Table 11: Human evaluation of decomposition quality.

## E Inference Prompts

The primary task is to generate synthetic question-answer pairs with a reasoning path, reflecting predefined reasoning skills. Table 12 shows our prompts.

Table 13 shows the helper language model for evaluating how well each example's question aligns with original one.

Table 14 shows the question answering prompts for model.

**Prompt:**
You are a language model that generates synthetic question-answer (QA) pairs with reasoning paths.
Your task is to generate a QA pair based on the following question. Additionally, you should
provide a clear, step-by-step reasoning path that corresponds to a predefined reasoning skill.
The predefined reasoning skills are:[REASONING SKILLS NAME+DESCRIPTION+EXAMPLES].
Your reasoning path should include clear substeps for each step of the thought process.
**Example 1:**
**Given Documents:** [REFERENCE DOCUMENTS]
**Input Question:** "Who invented the telephone?"
**Step-by-step Reasoning Path:**
1. Identify the key entity: "telephone" (deductive)
2. Identify that the question is asking for the inventor of a significant historical device (decompositional)
3. Recall the historical context of the invention of the telephone. (deductive)
4. The inventor is Alexander Graham Bell. (cause & effect)
**Generated Answer:** "Alexander Graham Bell invented the telephone in 1876."
**Reasoning Skill Used:** deductive, decompositional, deductive, cause & effect.
**Example 2:** ...
**Example 3:** ...
**Notes:**
Please make sure that the reasoning path is clear and includes each substep in the thought process.
The output should follow this structure: "Step-by-step reasoning," followed by the conclusion.
Each reasoning skill corresponds to a specific domain of knowledge.

Table 12: Prompt to Generate Similar Examples with Reasoning Paths.

**Prompt:**
You are given an original question: [ORIGINAL QUESTION]
You also have a synthetic question: [SYNTHETIC QUESTION]
Your task is to decide how similar the synthetic question is in structure and complexity, compared to the original.
Please provide a brief explanation of your reasoning. Then, assign a score from 1 (completely different) to 10 (very similar).
**Example**: Original Q: "Who discovered penicillin?"
Synthetic Q: "Which scientist found the mold that led to antibiotics?"
**Explanation**: Both questions ask about a discoverer of a major medical breakthrough. The second question focuses on the mold (penicillin), so it is structurally similar and retains the core inquiry about a discovery.
**Score (1-10)**: 8
**Notes**: - Provide a short justification.
- Avoid rewriting or changing the question.
- Keep the final output concise, ending with the numeric score.

Table 13: Prompt for Evaluating Alignment of Synthetic Questions with the Original.

**Prompt:**
You are given:
- The original question: [Q]
- A document or context: [D]
- A selected reasoning path: [R]
- The specific skills used in the reasoning path: [S]
**Your goal** is to produce a final answer by combining the relevant information from [D] with the guided reasoning steps from [R]. Follow these instructions:
1. **Review the Reasoning Path**
Read each step in [R] carefully. Identify which parts of [D] or background knowledge support each step.
2. **Apply the Skills**
If [S] includes certain reasoning skills (e.g., deduction), make sure to explicitly use them when combining evidence from [D].
3. **Generate a Clear Answer**
Compose a concise final answer that directly addresses [Q]. You may outline your chain of thought, but keep the explanation aligned with [R].
4. **Maintain Accuracy**
If [R] instructs a specific substep (e.g., numerical calculation or bridging multiple facts), follow it precisely, citing the relevant parts of [D].
**Notes**:
- Do not contradict the provided reasoning path.
- Cite relevant text from [D] if needed, but avoid unnecessary repetition.
- End with a concise, standalone final answer.

Table 14: Prompt for Question Answering.

| Model | SQuAD | HotpotQA | NewsQA | Gaokao | HQA | TriviaQA | BioASQ |
|---|---|---|---|---|---|---|---|
| ***Quick-Thinking Models w/ Reasoning Strategies (Exact Match)*** | | | | | | | |
| ***Qwen2.5-32B-Instruct*** | | | | | | | |
| w/ vanilla *(quick)* | 55.23 | 31.69 | 27.15 | 12.61 | 19.47 | 23.82 | 40.88 |
| w/ few-shots *(quick)* | 56.42 | 32.35 | 28.13 | 13.06 | 20.19 | 24.58 | 41.76 |
| w/ self-consistency (Wang et al., 2022) | 57.08 | 32.81 | 28.49 | 13.29 | 20.37 | 24.89 | 41.93 |
| w/ proCoT (Deng et al., 2023) | 58.65 | 33.81 | 29.32 | 13.86 | 21.02 | 25.58 | 42.77 |
| w/ ToT (Yao et al., 2023) | 59.83 | 34.67 | 29.81 | 14.12 | 21.51 | 26.04 | 43.28 |
| w/ MCTS (Zhao et al., 2024) | 59.87 | 34.53 | 29.76 | 14.21 | 21.57 | 26.08 | 43.33 |
| **w/ $T^2$ (ours)** | **62.65** | **39.98** | **34.12** | **15.72** | **22.58** | **26.43** | **48.14** |
| ***GPT-4o*** | | | | | | | |
| w/ vanilla *(quick)* | 59.87 | 35.31 | 30.27 | 16.23 | 23.39 | 29.84 | 44.12 |
| w/ few-shots *(quick)* | 61.09 | 36.04 | 30.85 | 16.78 | 24.03 | 30.52 | 44.84 |
| w/ self-consistency (Wang et al., 2022) | 61.63 | 36.42 | 31.14 | 16.92 | 24.28 | 30.78 | 45.13 |
| w/ proCoT (Deng et al., 2023) | 62.89 | 37.29 | 31.92 | 17.49 | 25.01 | 31.43 | 46.05 |
| w/ ToT (Yao et al., 2023) | 63.77 | 37.94 | 32.43 | 17.79 | 25.57 | 32.11 | 46.61 |
| w/ MCTS (Zhao et al., 2024) | 63.94 | 38.13 | 32.06 | 17.84 | 26.22 | 32.18 | 47.39 |
| **w/ $T^2$ (ours)** | **65.17** | **39.51** | **33.76** | **17.98** | **26.31** | **33.12** | **49.07** |
| ***Slow-Thinking Models (Exact Match)*** | | | | | | | |
| o1-mini | 65.82 | 42.89 | 35.08 | 21.87 | 29.51 | 36.52 | 50.68 |
| QwQ-32B-Preview | 66.67 | 43.57 | 35.43 | 22.18 | 29.88 | 36.87 | 51.21 |
| DeepSeek-R1 | 67.38 | 44.04 | 35.94 | 22.28 | 30.26 | 37.52 | 52.31 |
| o1 | 67.92 | 44.57 | 36.42 | 22.53 | 30.75 | 38.12 | 52.94 |
| o4-mini | 68.36 | 44.97 | 36.83 | 22.79 | 31.08 | 38.28 | 53.14 |
| o4-mini-high | 68.54 | 45.18 | 37.01 | 22.89 | 31.23 | 38.42 | 53.27 |
| Claude-3.7-sonnet-thinking | 68.67 | 45.27 | 37.14 | 22.97 | 31.32 | 38.56 | 53.41 |
| o3 | 68.89 | 45.48 | 37.34 | 23.19 | 31.37 | 38.78 | 53.74 |
| Gemini-2.5-Pro | 69.69 | 46.08 | 37.97 | 23.51 | 32.01 | 39.43 | 54.45 |
| **QwQ-32B + $T^2$ (ours)** | **71.32** | **47.87** | **39.17** | **24.63** | **33.28** | **40.39** | **55.81** |

Table 15: Exact Match (EM) scores on seven QA datasets.

## F Performance with Exact Match Metric

Generally, Open QA datasets use Exact Match as their metrics for evaluation. But in generative AI system, the models can generate correct answers but with different literalness (e.g., "San Francisco" and "The San Francisco City" and "SF U.S."). Hence we use ROUGE-L as metric in our overall performance evaluation. Besides, we also report our experimental results on EM in Table 15.

## G Calculation of Proposed Metrics

### G.1 Hits and Errors

**Hits Metric Calculation.** To evaluate the quality of reasoning and fact retrieval in the generated outputs, we employ the Hits metric based on the gold supporting sentences provided in HotpotQA. For each question $q$, let $P_q$ represent the set of sentences mentioned in the model's reasoning process and $G_q$ denote the set of gold supporting sentences.

We calculate the Hits metric as follows:

$$\text{Hits} = \frac{\sum_{q \in Q} \mathbf{1}[P_q \supseteq G_q]}{|Q|} \tag{12}$$

where $\mathbf{1}[\cdot]$ is an indicator function that equals 1 when the condition is satisfied and 0 otherwise,

and $|Q|$ is the total number of questions in the evaluation set. This formulation is similar to recall in traditional information retrieval, measuring the proportion of questions for which all required facts were successfully retrieved.

**Errorss Metric Calculation.** For the Error metric, we adopt the False Discovery Rate (FDR) formulation:

$$\text{Error} = \frac{\sum_{q \in Q} \mathbf{1}[P_q \not\supseteq G_q]}{\sum_{q \in Q} (\mathbf{1}[P_q \supseteq G_q] + \mathbf{1}[P_q \not\supseteq G_q])} \tag{13}$$

This represents the proportion of spurious facts (false positives) among all retrieved facts, consistent with the FDR calculation as FP/(TP+FP).

These complementary metrics create a natural trade-off: longer reasoning chains tend to improve Hits by including more supporting facts but often at the expense of increasing Error through the introduction of irrelevant information. An ideal reasoning process would maximize Hits while minimizing Error, indicating that the model precisely identifies all necessary supporting facts without including extraneous information.

### G.2 Retrace Rate

We define a response as exhibiting a *retrace* when the model initially states a provisional conclusion and subsequently revises it within the same output. This occurs in patterns such as "*So the answer is X... wait, that seems wrong—let me revise... the answer is Y.*" To systematically identify retraces, we analyze the Chain-of-Thought (CoT) reasoning for two specific patterns: (i) multiple occurrences of <*answer*> markers, or (ii) lexical repair cues (e.g., "sorry," "actually," "let me rethink") followed by a different answer span. If either pattern is detected, we count the example as containing a retrace.

The Retrace Rate is calculated as:

$$\text{Retrace Rate} = \frac{\sum_{q \in Q} \mathbf{1}[\text{retrace detected in } q]}{|Q|} \tag{14}$$

where $\mathbf{1}[\cdot]$ is an indicator function that equals 1 when a retrace is detected and 0 otherwise, and $|Q|$ is the total number of questions in the evaluation set. This metric quantifies the proportion of responses where the model explicitly revises its reasoning path, providing insight into the model's self-correction capabilities during the reasoning process.

| Method | HotpotQA | NewsQA | HQA |
|---|---|---|---|
| Random | 32.4 | 38.7 | 19.2 |
| Coverage Only | 41.6 | 46.3 | 27.8 |
| Uniqueness Only | 49.2 | 54.5 | 35.7 |
| Full Approach | **67.1** | **61.3** | **40.3** |

Table 16: Ablation study results showing the impact of different components in our selection approach. We report ROUGE-L (%) on three benchmark datasets.

## H Ablation Study

To validate the effectiveness of our multi-criteria matching approach, we conducted ablation studies by systematically removing or modifying key components of our selection mechanism.

**Impact of Selection Components.** We evaluated four variants of our selection approach: (1) using only skill coverage without uniqueness weighting, (2) using only skill uniqueness without coverage assessment, (3) using random selection from examples passing the similarity threshold, and (4) our full approach. The experiments are conducted on Qwen2.5-32B as LLM. Table 16 shows performance across test sets.

Results demonstrate that while both skill coverage and uniqueness contribute positively to performance, their combination in our full approach produces the strongest results across all datasets, yielding improvements of 25.5% over using only individual components.

## I Efficiency Analysis

This section provides a comprehensive analysis of the computational efficiency of our proposed Flexible Reasoning Method ($T^2$) in comparison to other reasoning approaches. We analyze both token consumption and performance across seven diverse question answering datasets.

### I.1 Token Consumption Analysis

Table 17 presents the average token consumption of different reasoning approaches across seven CQA datasets. The token length directly correlates with the computational resources required and inference time. Our results indicate that $T^2$ consistently reduces token consumption while maintaining or improving performance compared to other reasoning methods.

| Model | SQuAD | BioASQ | HotpotQA | NewsQA | GAOKAO | HQA | TriviaQA |
|---|---|---|---|---|---|---|---|
| Qwen2.5-32B w/ SC | 1372.18 | 1726.32 | 1485.87 | 2201.65 | 1957.93 | 1580.43 | 1742.41 |
| Qwen2.5-32B w/ $T^2$ | 1161.42 | 1401.52 | 1330.71 | 1812.28 | 1581.14 | 1415.18 | 1582.42 |
| QwQ-32B-Preview | 1617.42 | 2012.33 | 1823.49 | 2648.12 | 2284.80 | 1972.37 | 2119.88 |
| QwQ-32B w/ $T^2$ | 1285.36 | 1467.85 | 1450.75 | 1855.89 | 1699.45 | 1465.68 | 1605.56 |

Table 17: Average token consumption across seven CQA datasets for different reasoning approaches.

### I.2 Efficiency-Performance Trade-off

Table 18 presents a comprehensive comparison of computational efficiency and performance across all seven datasets. We report the average token length, relative token reduction, and ROUGE-L scores to illustrate the efficiency-performance trade-off.

### I.3 Dataset-specific Efficiency Gains

As shown in Figure 6, the efficiency gains of $T^2$ vary across datasets. The token reduction ranges from 10.5% to 18.8% when applied to Qwen2.5-32B (compared to self-consistency), and from 20.6% to 31.6% when applied to QwQ-32B (compared to QwQ-32B-Preview). Notably, datasets requiring more complex reasoning (like NewsQA and GAOKAO) show greater efficiency improvements, suggesting that $T^2$ is particularly effective at streamlining the reasoning process for complex questions.

### I.4 Detailed Efficiency-Performance Analysis

Table 19 provides a detailed analysis of both token consumption and performance for each dataset and model combination. This comparison highlights how $T^2$ maintains or improves performance while reducing computational costs.

### I.5 Efficiency Analysis by Question Complexity

To better understand $T^2$'s efficiency gains, we categorize questions by complexity and analyze token reduction. As shown in Table 20, $T^2$ achieves greater token reduction for complex questions requiring multi-step reasoning, showcasing its adaptive nature.

### I.6 Time Efficiency

Beyond token reduction, we also measure the actual inference time across different models and reasoning approaches. Table 21 presents the average inference time per question, demonstrating that $T^2$ reduces computational time while maintaining high performance.

In summary, our comprehensive efficiency analysis demonstrates that $T^2$ reduces token consumption and inference time across diverse CQA datasets while maintaining or improving performance. The efficiency gains are particularly pronounced for complex questions requiring multi-step reasoning, highlighting $T^2$'s ability to adapt its reasoning approach based on question complexity.

## J Impacts of Similar Examples

### J.1 Impacts of Question Structure

Our framework decomposes each question into a *structure* plus *replaceable elements*. We hypothesize that questions with more placeholders benefit more from $T^2$'s selection mechanism, because these questions allow a wider range of possible similar examples. Conversely, simpler questions with fewer placeholders may not need advanced reasoning paths.

We categorize questions into three buckets based on the number of placeholders in $Q$: *Low* (0–1 placeholders), *Medium* (2–3 placeholders), and *High* (4+ placeholders). Table 22 shows the performance across these groups for SQuAD and HQA to show impacts on general and domain-specific scenarios.

As seen in Table 22, questions with more placeholders (High) see the largest gap between $T^2$ and either baseline. This suggests that, for complex questions, enumerating and reusing relevant skill chains is particularly helpful. On simpler questions (Low), $T^2$ still improves performance but by a smaller margin, as fewer placeholders limit the search space for alternative question structures.

### J.2 Impacts of Similar Examples Structure

We show the "skeleton" QA pairs that preserved reasoning structure while replacing all content-specific terms with placeholders in Figure 7 (for original one) and Figure 8 (for structure-only one).

| Model | Avg. Token Length | Token Reduction | Avg. ROUGE-L |
|---|---|---|---|
| Qwen2.5-32B w/ SC | 1723.83 | - | 50.07 |
| Qwen2.5-32B w/ $T^2$ | 1469.24 | 14.8% vs. SC | 56.22 |
| QwQ-32B-Preview | 2068.34 | - | 63.38 |
| QwQ-32B w/ $T^2$ | 1547.22 | 25.2% vs. QwQ | 68.56 |

Table 18: Efficiency-performance trade-off across seven CQA datasets. Token reduction is calculated relative to the baseline model (SC: Self-Consistency).

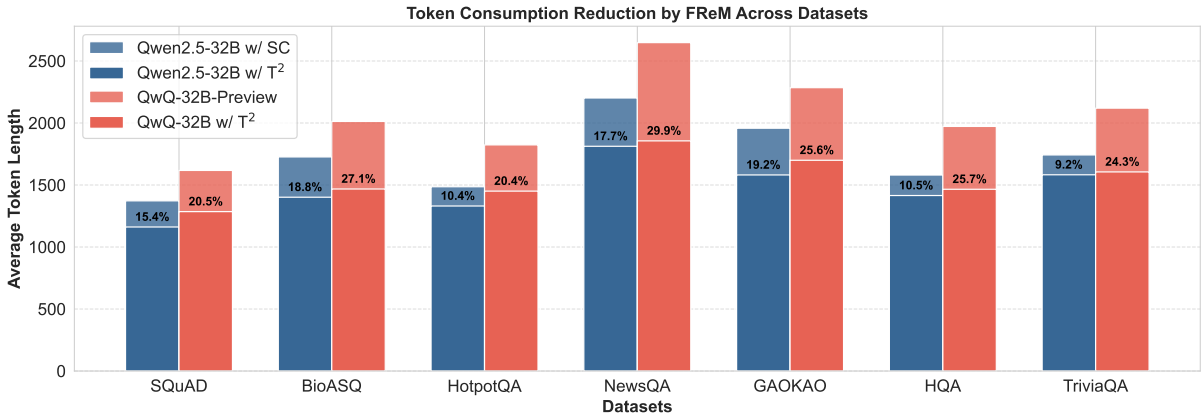


Figure 6: Token consumption reduction by $T^2$ across different datasets. The percentage values indicate the relative reduction compared to the baseline models (Qwen2.5-32B w/ SC and QwQ-32B-Preview).

| Model | SQuAD | | BioASQ | | HotpotQA | | NewsQA | |
|---|---|---|---|---|---|---|---|---|
| | Token | ROUGE-L | Token | ROUGE-L | Token | ROUGE-L | Token | ROUGE-L |
| Qwen2.5-32B w/ SC | 1372.18 | 75.31 | 1726.32 | 57.57 | 1485.87 | 56.76 | 2201.65 | 52.27 |
| Qwen2.5-32B w/ $T^2$ | 1161.42 | 81.86 | 1401.52 | 65.02 | 1330.71 | 67.11 | 1812.28 | 61.27 |
| QwQ-32B-Preview | 1617.42 | 86.87 | 2012.33 | 69.02 | 1823.49 | 71.86 | 2648.12 | 63.92 |
| QwQ-32B w/ $T^2$ | 1285.36 | 92.12 | 1467.85 | 75.21 | 1450.75 | 77.61 | 1855.89 | 68.61 |
| **Model** | **GAOKAO** | | **HQA** | | **TriviaQA** | | **Average** | |
| | Token | ROUGE-L | Token | ROUGE-L | Token | ROUGE-L | Token | ROUGE-L |
| Qwen2.5-32B w/ SC | 1957.93 | 30.57 | 1580.43 | 37.12 | 1742.41 | 41.92 | 1723.83 | 50.07 |
| Qwen2.5-32B w/ $T^2$ | 1581.14 | 34.06 | 1415.18 | 40.31 | 1582.42 | 43.92 | 1469.24 | 56.22 |
| QwQ-32B-Preview | 2284.80 | 43.23 | 1972.37 | 49.62 | 2119.88 | 59.16 | 2068.34 | 63.38 |
| QwQ-32B w/ $T^2$ | 1699.45 | 47.42 | 1465.68 | 54.71 | 1605.56 | 64.22 | 1547.22 | 68.56 |

Table 19: Detailed comparison of token consumption and performance (ROUGE-L) across all datasets. Lower token count with higher ROUGE-L indicates better efficiency-performance trade-off.

Original QA Pair

**Q:** How does the melting point of sodium compare to potassium?
**A:** I'll approach this systematically:
1. First, I'll recall the melting point of sodium, which is 97.8°C.
2. Next, I'll recall the melting point of potassium, which is 63.5°C.
3. Comparing these values: 97.8°C > 63.5°C
4. Therefore, sodium has a higher melting point than potassium.

Figure 7: Our original generated similar example.

| Question Complexity | Qwen2.5 + SC | Qwen2.5 + $T^2$ | Token Reduction |
|---|---|---|---|
| Simple (1-step) | 1283.45 | 1157.82 | -9.8% |
| Moderate (2-3 steps) | 1687.31 | 1391.65 | -17.5% |
| Complex (4+ steps) | 2201.73 | 1758.24 | -20.1% |
| **Question Complexity** | **QwQ-32B** | **QwQ-32B + $T^2$** | **Token Reduction** |
| Simple (1-step) | 1584.21 | 1262.35 | -20.3% |
| Moderate (2-3 steps) | 2041.57 | 1492.18 | -26.9% |
| Complex (4+ steps) | 2579.24 | 1887.14 | -26.8% |

Table 20: Token consumption analysis by question complexity. $T^2$ achieves greater efficiency gains for more complex questions.

| Model | Avg. Inference Time (s) | Time Reduction |
|---|---|---|
| Qwen2.5-32B w/ SC | 65.31 | - |
| Qwen2.5-32B w/ $T^2$ | 34.52 | -47.1% |
| QwQ-32B-Preview | 76.74 | - |
| QwQ-32B w/ $T^2$ | 45.03 | -41.3% |

Table 21: Average inference time per question across datasets. $T^2$ reduces computational time while maintaining high performance.

| Group | SQuAD | | | HQA | | |
|---|---|---|---|---|---|---|
| | Few-shots | Self-Cconsistency | $T^2$ | Few-shots | Self-Cconsistency | $T^2$ |
| Low | 78.5 | 79.1 | **80.2** | 42.7 | 43.3 | **44.6** |
| Medium | 76.4 | 78.2 | **79.5** | 41.5 | 43.0 | **45.1** |
| High | 75.9 | 78.7 | **80.1** | 40.2 | 42.9 | **46.2** |

Table 22: ROUGE-L by question complexity. We compare quick-thinking (Few-shots) and slow-thinking (Self-Consistency), and our $T^2$.

Structure-Only Version

**Q:** How does [PROPERTY] of [ENTITY_A] compare to [ENTITY_B]?
**A:** I'll approach this systematically:
1. First, I'll determine the [PROPERTY] of [ENTITY_A], which is [VALUE_A].
2. Next, I'll determine the [PROPERTY] of [ENTITY_B], which is [VALUE_B].
3. Comparing these values: [COMPARISON_OPERATION]
4. Therefore, [CONCLUSION_STATEMENT].

Figure 8: Structure-only version of our generated similar example.

### J.3 Impacts of Similar Example Numbers

We vary the size $M = |\Gamma|$. Figure 9 illustrates the performance on HotpotQA (left) and NewsQA (right) as $M$ increases. We observe an initial boost in ROUGE-L scores before $M = 20$, but performance plateaus or slightly decreases beyond a certain point. After increasing examples to $M = 80$,

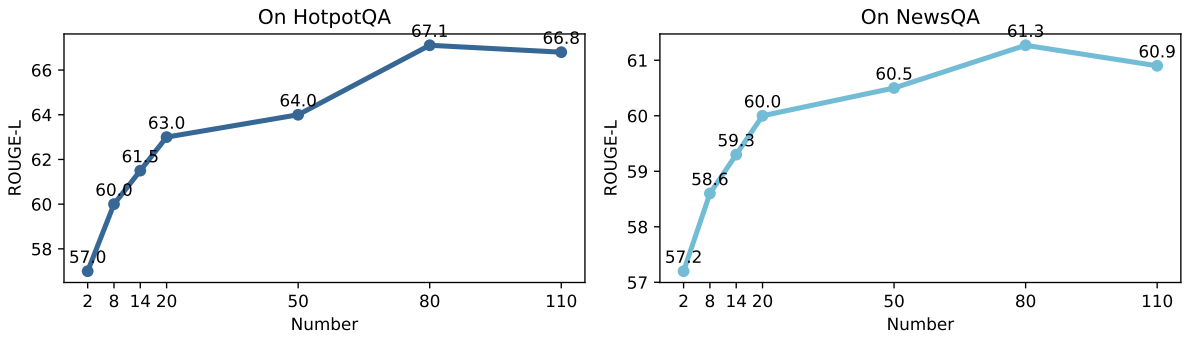


Figure 9: Impact of the number of similar examples on ROUGE-L scores for HotpotQA (left) and NewsQA (right).

| Method | ROUGE-L | Variation | Noise |
|---|---|---|---|
| ***Qwen2.5 w/ ours*** | | | |
| Random Fill | 49.8 | High | Medium |
| Guided Fill | 52.6 | Low | Low |
| Template Variation | **61.3** | High | Low |

Table 23: Comparing different example construction methods on NewsQA.

| Model | HotpotQA | HQA |
|---|---|---|
| Qwen2.5+SC | 56.76 | 37.12 |
| Qwen2.5+T$^2$ | 67.11 | 40.31 |
| Qwen2.5+mis domain | 65.96 | 39.85 |
| Qwen2.5+structure only | 63.96 | 38.85 |
| QwQ | 71.86 | 49.62 |
| QwQ+T$^2$ | 77.61 | 54.71 |
| QwQ+mis domain | 77.03 | 54.26 |
| QwQ+structure only | 74.03 | 53.66 |

Table 24: Performance on mis-domain and structure-only models. ROUGE-L is the reported performance metric.

the performance rapidly decreases. We conclude that too many examples can introduce irrelevant or redundant paths, making selection harder. In practice, we find that generating a moderate pool is enough to cover essential patterns, especially if the examples are diverse and accurate.

### J.4 Impacts of Example Generation Methods

Then, we consider how we synthesize reference examples. We experiment with different approaches for filling the placeholders on HotpotQA with qwen2.5-32B:

- **Random Fill:** Pick random words or entities of the same type (e.g., any `person`) from a large corpus.
- **Guided Fill:** Use an LLM or curated list to pick semantically relevant or thematically consistent entities for each placeholder.
- **Template Variation:** Generate minor paraphrases or new question stems while retaining the same skill sequence.

Table 23 shows that template variation produces more coherent examples, with 2–4% gains over purely random fill. This highlights the importance of a well-structured synthetic process: random replacements might yield too many off-topic or contradictory examples, while guided replacements and paraphrasing keep the examples relevant, improving the final answer selection.

### J.5 Impacts of Example Generation Threshold

We analyze the impact of varying the threshold $\delta$ on the synthesis quality of the generated questions. The threshold $\delta$ controls how similar the synthesized questions $Q^i_{\text{syn}}$ are to the original question $Q$, by using a helper language model to assess their alignment (in Sec.3.3). Figure 10 shows finding a trade-off between question similarity and generalization is much more important. As $\delta$ increases, the similarity to the original question improves but at the cost of generalization. Conversely, when $\delta$ is lowered, the model generalizes better but the quality of the synthesized questions decreases.

### J.6 Impacts of Examples Domain Bias and Structural Bias

In addition, we investigate the effects of domain and structural biases in similar examples. Specifically, we assess how varying the domain of the similar examples influences model performance. As shown in the Table 24 ("+mis domain"), transitioning from a general domain to a historical one results in improved performance compared to using self-consistency alone. Furthermore, we evaluate the impact of removing key information from the

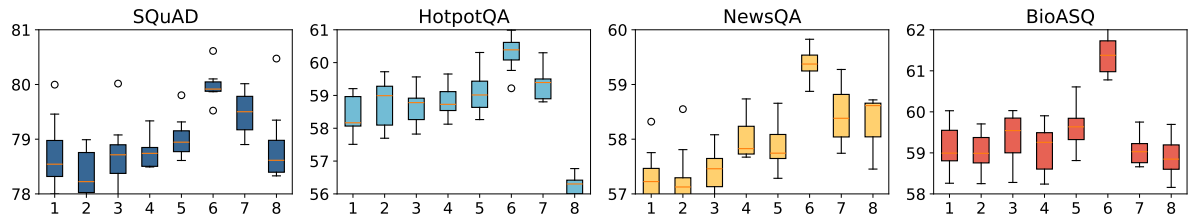


Figure 10: Impact of the question synthesis scope.

similar examples, leaving only the reasoning structure. Table 24 ("+structure only") demonstrates that even when only the examples' structure[4] is provided, the model can still generate appropriate responses, highlighting the effectiveness of structural guidance.

### J.7 Impacts of Similar Examples Quality

To evaluate the quality of similar examples generated by our framework, we conducted a comprehensive human evaluation study. We randomly selected 1000 query-reference pairs from the HotpotQA dataset and recruited three Ph.D. students specializing in NLP to assess the quality of synthetic references. The evaluation was conducted blind, with evaluators unaware of which model generated each reference.

**Evaluation Dimensions.** References were rated on a scale of 1-10 across four key dimensions:

- **Accuracy:** Factual correctness and absence of hallucinations or contradictions
- **Relevance:** Degree to which the reference addresses the specific query requirements
- **Completeness:** Thoroughness in covering all necessary information and reasoning steps
- **Coherence:** Logical structure, clarity of expression, and overall readability

**Model Comparison.** We evaluated synthetic references generated by two foundation models: Qwen2.5-32B-Instrucut and QwQ-32B-Preview, both with our framework. Table 25 presents the average scores across all evaluators and samples.

Results show both models produced high-quality references. The highest scores were observed in the Relevance category, indicating that references effectively addressed the specific queries. The evaluation exhibited strong inter-annotator agreement with a Fleiss' kappa coefficient of 0.79, indicating substantial agreement among the three evaluators. This suggests the evaluation results are reliable and consistent across different human judges.

## K Detailed Case Studies

Figure 11 and 12 show the two different cases from HotPotQA and SQuAD. The two case studies illustrate distinct reasoning strategies for question answering. In the HotpotQA case, the task requires multi-step reasoning by integrating evidence from multiple documents. A response based solely on pattern matching might output wrong "London" and an overthinking answer may include unnecessary details before arriving at the correct conclusion, the best approach is a concise, step-by-step explanation that clearly connects the film to John Lennon and his documented birthplace.

In contrast, the SQuAD case involves single-step reasoning where the answer is directly available from the text. The document states that the Eiffel Tower was constructed in 1889, so the correct answer is deduced by simply extracting this fact. Although a pattern matching response may correctly state "1889" and an overcomplicated explanation might unnecessarily discuss other numerical details, the ideal answer succinctly identifies the key fact and deduces that the construction year is 1889 using clear, deductive reasoning.

[4]We show the example structure in Appendix J.2.

| Models w/ $T^2$ | Accuracy | Relevance | Completeness | Coherence | Overall |
|---|---|---|---|---|---|
| Qwen2.5-32B-Instrucut | 8.8 | 8.6 | 8.1 | 8.1 | 8.4 |
| QwQ-32B-Preview | 8.4 | 8.3 | 7.9 | 8.2 | 8.4 |

Table 25: Human evaluation scores for synthetic references generated by different models (scale: 1-10).

**# HotpotQA Case Example with Multihop Thinking**

**Document 1:**
John Lennon, the iconic musician and member of The Beatles, was born in Liverpool, England.
**Document 2:**
*Nowhere Boy* is a film that explores the early life of John Lennon, highlighting his formative years and influences.
**Question:**
In what city was the subject of the film *Nowhere Boy* born?

**Proper Reasoning Chain:**
1. **Identify the Subject:** Recognize that *Nowhere Boy* is about John Lennon. (Deductive)
2. **Extract Key Fact:** From Document 1, note that John Lennon was born in Liverpool, England. (Deductive)
3. **Conclude:** Therefore, the subject of the film was born in Liverpool. (Decompositional)

**1) Quick Thinking Model's Wrong Answer:**
*Answer:* The subject of *Nowhere Boy* was born in London.

**2) Slow Thinking Model's Overthinking Answer:**
*Answer:* After a lengthy analysis of various biographical details concerning John Lennon, including his early childhood, multiple hints in his music and interviews, and a discussion of several English cities mentioned in relation to his life, one might eventually conclude that the subject of *Nowhere Boy* was born in Liverpool.

**3) Model with Our $T^2$'s Correct Answer:**
*Answer:* Since *Nowhere Boy* is a film about John Lennon (Document 2) and Document 1 confirms that John Lennon was born in Liverpool, we deduce that the subject of the film was born in Liverpool.

Figure 11: HotpotQA Case Example with Multiple Answer Versions and Explicit Reasoning Chain.

**# SQuAD Case Example with Single-Step Thinking**

**Document:**
The Eiffel Tower is a wrought-iron lattice tower located on the Champ de Mars in Paris, France. It was constructed in 1889 as the entrance to the 1889 World's Fair. The tower stands approximately 324 meters tall and is one of the most recognized structures in the world.
**Question:**
In what year was the Eiffel Tower constructed?

**Proper Reasoning Chain (using Deductive):**
1) **Deduction:** Based on clear statement, we deduce that the construction year is 1889 (Deductive).

**1) Quicking Thinking Model's Correct Answer via Pattern Matching:**
*Answer:* 1889.

**2) Slow Thinking Model's Correct Answer but Overthinking:**
*Answer:* After reading the passage, I noticed several numbers such as 324 (the tower's height) and 1889. While I considered whether the height might play a role in determining the construction year, I ultimately realized that the passage explicitly states "constructed in 1889." Thus, despite the extra details, the correct construction year is 1889.

**3) Model with Our $T^2$'s Correct Answer using the Specific Reasoning Chain:**
*Answer:* Since the document clearly states that the Eiffel Tower was constructed in 1889, we deduce—by directly extracting and applying this fact—that the construction year is 1889.

Figure 12: SQuAD Case Example with Single-Step Thinking and Multiple Answer Versions.