Difficulty-aware Dynamic Chain-of-Thought Prompting for Large Language Models via BM25 and Semantic Retrieval

Zuchen Zhuang
International School, Beijing University of Posts and Telecommunications, Beijing 100000, China
E-mail: zuchenzhuang@outlook.com

Keywords: Large language models, chain-of-thought, difficulty assessment, BM25 retrieval, difficulty-aware dynamic chain-of-thought

Received: February 27, 2026

Large language models (LLMs) have demonstrated exceptional capabilities across multiple domains and emerged as the core driving force in natural language processing. Their reasoning however can be associated with logical flaws and lack of stability when attempting to solve difficult problems that require multi-step deduction, cross-domain knowledge or implicit constraints, with redundant or insufficient exemplars in conventional prompts and poor fit with exemplar and target problems. To address these issues, we propose a dynamic Chain-of-Thought (CoT) prompting method based on problem difficulty assessment: first, the model performs zero-shot self-evaluation of the required solution steps to dynamically determine the number of exemplars; then, it integrates BM25 retrieval to select the most similar high-quality question-answer pairs, constructing precise Few-shot prompts. Experiments conducted on multiple datasets effectively resolve the two major limitations of traditional exemplar-based methods, enabling LLMs to obtain appropriately tailored exemplars for both multi-step mathematical reasoning and interdisciplinary question answering. Consequently, the accuracy of complex reasoning is improved to varying degrees across tasks.

Povzetek: Metoda dinamičnega CoT-spodbujanja glede na zahtevnost naloge izbere ustrezno število podobnih primerov in s tem izboljša natančnost sklepanja velikih jezikovnih modelov pri zahtevnih problemih.

1 Introduction

Within recent years, language models that are of great scale have made remarkable progress. Other models like GPT [1] and LLaMA [2] have demonstrated strong abilities in most natural language processing problems, especially text generation [3], question answering [4], and code completion [5]. These models have learnt vast knowledge of cross-domain and are capable of decent reasoning through pre-training on large amounts of textual information.

However, in solving hard problems requiring complex thinking, such models often fail and do not reason about problems using the right logic to produce correct solutions, which entails a deficiency regarding stability in reasoning performance. Researchers have suggested different approaches to improve the level of performance of LLMs when doing complex reasoning tasks. The best representative is Chain-of-Thought (CoT) which was recently proposed by Wei et al. [6]. Such mechanism encourages the LLMs to reason as well with proper logic when they are exposed to problems by including middle-steps of reasoning as part of prompts, thus enhancing the accuracy of reasoning. Even though CoT has been able to improve the performances of models, their accuracy is relatively low when handling very complex issues. Later works have also suggested better versions like Self-Consistency [7].

Although these procedures have enhanced the accuracy, there are two major constraints: First, they typically assume a fixed number of exemplars [8], which is not sensitive to knowledge coverage disparities and reasoning difficulty across problems. Solutions to problems that are very complex can be in multiple stages of deduction, across-domain, or utilise implicit constraints, and therefore ought to be nourished with additional intermediate steps as hints. The decomposition of hard problems across CoTs enables the model to test each step on example cases and not to reach a conclusion too quickly or the answer too rapidly. On the other hand, in low-difficulty problems, long exemplars can distort significant cues amid respondent irrelevant information and give the model the wrong impression. Secondly, these procedures do not have selective exemplars [8]. Any issue can be compelled to create intermediary actions and answers regarding the logical reasoning of selected examples, which might not agree with the particular demands of the targeted issue. Once the problem ceases being useful to the prototype of the exemplars, reasoning template breaks down that causes the model to mechanically relate the problem to irrelevant logic, disregard problem circumstances, and eventually result in erroneous reasoning. Thus, current approaches have evident shortcomings in the regulation of the number of exemplars

as well as the choice of exemplar information, which restricts the subsequent development of CoT prompting during complicated reasoning processes.

To address the aforementioned issues, this paper proposes the Difficulty-aware Dynamic Chain-of-Thought (Dyna-CoT) method. Our goal is to enable LLMs to select appropriate prompting strategies when facing problems of varying domains and difficulties, thereby better stimulating their reasoning potential. This method improves CoT prompting in two key aspects:

1. Dynamic control of exemplar quantity: To resolve the fixed-quantity limitation, we argue that the optimal number of exemplars should not be pre-determined but dynamically adjusted based on problem difficulty—increasing with higher difficulty and decreasing with lower difficulty. Specifically, the model first evaluates the problem to estimate the number of solution steps using token count [8], then classifies the problem into different difficulty levels to allocate a corresponding number of exemplars. This approach effectively avoids both redundancy and insufficiency of exemplars.
2. Retrieval of similar problems for exemplar selection: To overcome the absence of targeting, we should suggest the option of dynamically picking exemplars with the structure of key nodes that are close to the target problem according to the context, the domain of knowledge, and the depth of reasoning. This will guarantee that the strategies given by the exemplars are indeed feasible as solution templates to the existing predicament hence enhance effectiveness as well as precision. In order to achieve this, this model initially extracts the essential information about the problem (domain of knowledge information and methods of calculation) and then a strategy of matching a key word is used to choose the best similar problems and CoT solutions in an exemplar pool and serve as prompts. This enables the model to acquire exemplars that are very much congruent with the current problem in structure and logic of reasoning and literally they direct the production of correct intermediary reasoning steps.

Ultimately, by integrating dynamic exemplar quantity adjustment and context-aware exemplar selection, our method provides more flexible and precise prompt support for LLMs in complex reasoning tasks.

We conducted systematic experiments on three reasoning datasets (GSM8K, MATH, and ScienceQA) covering elementary to university-level problems, spanning mathematics and science domains. The results show that the proposed method increases the average accuracy from 53.72% to 84.94%, achieving a relative improvement of 31.22%.

The main contributions of this paper are as follows:

1. We propose a dynamic CoT prompting method that retrieves similar exemplars based on problem difficulty assessment, automatically adjusting exemplar quantity and content to significantly improve complex reasoning performance.
2. We design a problem difficulty assessment mechanism using the LLM itself, guiding the model to output the minimum number of reasoning steps required to solve the problem, thereby dynamically determining the optimal number of exemplars and avoiding redundancy or insufficiency.
3. We introduce BM25 keyword matching for similarity retrieval to construct high-quality Few-shot Prompts, guiding the model toward correct reasoning.
4. The proposed method is fully validated on three datasets (GSM8K, MATH, and ScienceQA), achieving an average improvement of 31.22% and demonstrating strong cross-domain robustness.

The rest of the paper is organized as follows: Section 1 provides the research background and limitations of the traditional CoT prompting; Section 2 provides a review of literature on related methods to chain reasoning and their enhancements; Section 3 explains the overall workflow of the proposed dynamic CoT method; Section 4 includes the experimental setting, the statistical analysis of results, and the ablation studies.

2 Related work

In recent years, Chain-of-Thought (CoT) and its derivative technologies have become the core approach to enhancing the reasoning capabilities of large language models. Wei et al. proposed Chain-of-Thought (CoT) [6], a method that combines the advantages of arithmetic reasoning and in-context learning. By inserting intermediate reasoning steps into few-shot examples, it improves the model's reasoning accuracy. Subsequently, more researchers have explored optimizations of CoT based on this foundation, pushing CoT beyond its basic capabilities.

To the reasoning depth and quality: Diao et al. have come up with Active-Prompt [9], which combines active learning query strategies with CoT prompting. It presents an efficient method of selecting problems based on their uncertainty to select the most deserving of questions to be annotated and thus minimizes annotation expenses whilst increasing reasoning power. A new decoding method of CoT that substitutes greedy decoding was introduced by Wang et al. [7] and is called Self-Consistency. It enhances reliability of answers by combining aligned outcomes of many reasoning directions, prevents redundancy and local maximum of greedy decoding and reduces chance of randomness in one-sample generation. Yao et al. also suggested Graph-of-Thought (GoT) [10] that enhances a conventional linear thinking process in the way that uncovers the non-linear connections among nodes of thought by using directed graphical structural representations. It enables backtracking and parallel thinking and eliminates the sequential limitations of a linear chain, more in-line with human ways of thinking. A new framework can help in generating advanced CoT prompts and CoTGenius [11] was offered by Cheng et al. [11]. It automatically produces finer and richer CoT data by means of three evolutionary strategies (complexification-diversification-refinement) and a dual-filtering mechanism to fine-tune small and medium-sized models enabling CoT to substantially increase accuracy. Chain of Preference Optimization is a proposal put forward by Zhang et al. to combine gradient processing by searches through tree-structured thinking with preference learning [12]. It transforms the quality indicators of every

candidate reasoning path into quality training preferences rather than expensive search in reasoning, enhancing the quality of reasoning and minimizing latency Cost. Safety-aligned dataset (SAFECHAIN) [13] first described and publicly available by Jiang et al. is a special dataset that is a long CoT designed and oriented set consisting of 40k safe question-answer pairs with detailed stepwise reasoning chains. Future research can subsequently refine models according to this dataset to exclude non-conforming output, and facilitate the development of CoT towards being trustworthy and controllable. Towards this end, to improve the reasoning power of small models Ranaldi and Freitas introduced Instruction-tuned CoT [8], a way of transferring reasoning power between larger and smaller language models. Knowledge transfer can be obtained by means of instruction tuning done on the small models with the aid of the high-quality reasoning chains produced by large models, allowing the small-scale language models to also exhibit step-by-step reasoning behavior in complex reasoning problems and produce multi-step-controlled reasoning responses as well.

Meanwhile, the modal boundaries of CoT have been continuously expanded. Zhang et al. extended CoT from pure text to text-visual joint reasoning, proposing Multimodal CoT [14]. This technology decomposes tasks into two stages: rationale generation and answer reasoning. It fuses visual and linguistic representations by fine-tuning language models to perform multimodal CoT, reducing hallucination error rates in pure text. Ma et al. proposed Audio-CoT [15], which introduces CoT into audio-language models. It verifies the effectiveness of step-by-step auditory reasoning through three strategies (Manual/Zero-Shot/Depth-CoT), reveals the positive correlation between reasoning chain length and accuracy, and successfully extends CoT from text and visual modalities to a third modality (audio). Zheng et al. proposed Duty-Distinct Chain-of-Thought (DDCoT) [16], which assigns visual recognition tasks to specialized VQA models and logical integration to language models. Through a three-stage pipeline, it achieves zero-shot reasoning and seamless integration of fine-tuning stages, enhancing the generalization ability of CoT in interdisciplinary scenarios. Mu et al. proposed EmbodiedGPT [17], which constructs 200M egocentric video-language-action triples, realizing seamless mapping from high-level linguistic subgoals to low-level continuous control signals and promoting the end-to-end application of CoT in embodied intelligence.

Regarding data, tasks and evaluation: Fu et al. introduced Chain-of-Thought Hub [18], an open evaluation infrastructure of both Chinese and English, which combines six large benchmarks and hundreds of subtasks. It is a system in which reasoning capabilities of 19 mainstream models are continuously tracked and reported publicly and developed with the aim of providing a reproducible and scalable complex reasoning leaderboard to the research community. Pattern-CoT [19] suggested by Zhang et al., introduces reasoning patterns into the process of selecting the exemplars, fully departing with the use of the problem semantics. It retrieves exemplars depending on the similarity of operators sequence of reasoning, which injection of noise is substantially reduced and the robustness of unsupervised CoT prompting between different tasks is enhanced. The model LONGREPS proposed by Zhu et al. implements process supervision into scenarios of ultra-long context: first, the model bootstraps and samples many reasoning chains of 10k-128k tokens and then supervised fine-tuning is performed using triple quality filtering (answer-faithfulness-consistency). This will allow long-context models to achieve substantial gains in the complex question-answering and cross-domain generalization problems, confirming that quality reasoning chains can substantially increase the long-text understanding and generalization skills.

Through single linear chains, graph structure, text, multimodality, large models to small and medium sized models, the CoT technology system has broken through on depth, breadth, trustworthiness, and versatility. It is currently being developed, building a strong background towards the further generation of elaborate thinking machines, with the aid of open datasets and assessment tools.

Table 1 in Section 4.2 provides a direct performance comparison between our method and existing SOTA methods (CoT, GoT, DDCoT) on three benchmarks. Overall, existing methods suffer from either fixed exemplar quantity or semantically mismatched exemplars, which our Dyna-CoT addresses.

3 Methodology

In order to overcome the shortcomings of traditional CoT prompting, such as the limited number of exempla and not selecting problem-related exempla, we present our variant of the prompting method in the form of a dynamic Chain-of-Thought (Chain-of-Thought) that is known as Difficulty-aware Dyna-CoT (Dyna-CoT). The suggested scheme does not need any extra training, and it is entirely applied during the inference step, which guarantees decent generality as well as scalability. To apply this procedure, as illustrated in Figure 1, we specify a zero-training overall workflow: On a given input problem q, the system calls a difficulty assessor to estimate the number of reasoning steps k; next, we then call on the retriever with an input query equated to q to score BM25 in the exemplar pool, and then the Top-N problem-CoT pairs are sampled and concatenated in descending order, to form a Few-shot Prompt, which is then inputted into the LLM together with the input query. This broad framework has two main modules: (1) A difficulty-based dynamic exemplar quantity control mechanism which is an adaptive control mechanism that adjusts the number of exemplars in the prompt according to the estimated number of reasoning steps to avoid underfitting or overfitting of prompts; (2) A similarity retrieval-based exemplar selection mechanism, which is an accurate retrieval of question-answer pairs in the exemplar pool and best fit the current problem in terms of knowledge content and reasoning layout to form high quality concise prompts.

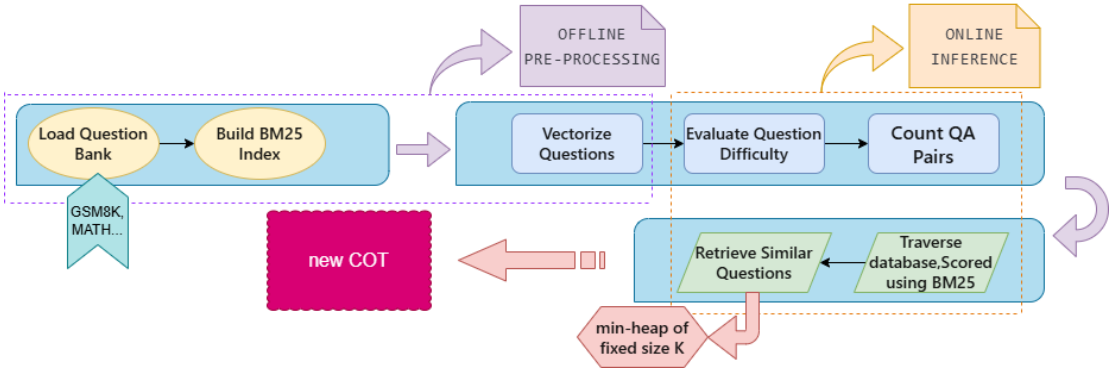

Figure 1: Schematic diagram of the offline-online workflow of the Dyna-CoT framework.

In the offline phase, BM25 keyword indexes and vector representations are pre-built for question banks such as GSM8K and MATH; in the online phase, the difficulty assessor first estimates the number of solution steps, then dynamically determines the number of exemplars based on the steps, and finally uses BM25 retrieval to maintain a min-heap of size K, returning the most similar question-answer pairs to form a Few-shot Prompt for model reasoning.

3.1 Difficulty-driven exemplar quantity control

In Few-shot Chain-of-Thought prompting, the number of exemplars has an inverted U-shaped impact on the final reasoning accuracy [8]: too few exemplars fail to cover complex reasoning patterns, while too many tend to introduce noise and consume context length. To ensure the model always receives appropriate prompt support for problems of varying difficulties, we first perform zero-shot difficulty estimation on the problem q during the inference phase [8] to obtain the estimated number of reasoning steps k; then dynamically determine the number of exemplars N to retrieve through a lightweight mapping function k2cot_count(k). The entire control mechanism requires no manual annotation or additional training, with extremely low computational overhead and good scalability. The following two subsections detail the difficulty estimation strategy and exemplar quantity mapping rules.

3.1.1 Difficulty estimation strategy

To dynamically determine the number of exemplars required for each problem, we first need to estimate the problem's difficulty. We use the LLM itself as a "difficulty assessor" and guide it to output the minimum number of reasoning steps required to solve the problem through a simple prompt.

Specifically, we feed the original problem q into the LLM with a minimal zero-shot prompt that requires a single integer output (as shown in Figure 2):

```
EST_PROMPT = (
    "You are an efficient complexity estimator. "
    "Output exactly one integer (Arabic numeral) and nothing else. "
    "Do not explain, do not use words, do not use LaTeX. "
    "Question: {q}"
)
```

Figure 2: The original problem and other minimal prompts are fed into the LLM to obtain the number of reasoning steps.

The integer k output by the model is regarded as the estimated number of reasoning steps for the problem, serving as the basis for subsequent difficulty grading. This method requires no manual annotation, has extremely low computational overhead, and exhibits good scalability.

3.1.2 Exemplar quantity mapping rules

Once we have the number of examples to be retrieved N, we must map the number of steps we have approximated, k, up to N. The inverted U-shaped issue would still be faced when using a fixed value and direct linear scaling can easily surpass the context length. As such we plot k to four difficulty levels and place a number of exemplars in each difficulty level. The mapping rule is implemented through the function k2cot_count(k), as follows:

$$N = \begin{Bmatrix} 1 & ,k \le 2 \\ 5 & ,3 \le k \le 4 \\ 10 & ,5 \le k \le 7 \\ 20 & ,8 \le k \end{Bmatrix} \quad (1)$$

This mapping strategy follows the principle of "more steps require more exemplars, but with decreasing growth rate," demonstrating good stability in experiments and providing appropriately tailored prompt support for problems of varying complexities.

3.2 Exemplar selection based on keyword similarity retrieval

All designs in this section aim to select Few-shot exemplars from the exemplar pool that are most matching the current problem and best demonstrate correct reasoning paths after determining the number of exemplars, thereby constructing high-quality prompts. To this end, we adopt the classic retrieval model BM25 [21], which returns Top-N candidate question-answer pairs in milliseconds during runtime. It is coupled with the N value dynamically determined in Section 3.1 to

ensure the selected question-answer pairs are both quantitatively appropriate and contextually relevant.

3.2.1 Index construction

After merging all question-answer pairs and generating token sequences D_i via jieba segmentation [22], we calculate document frequency (i.e., how many documents contain the word w), current document length, and average document length in one pass, which are then stored in memory permanently. All statistics are saved in RAM as Python dictionaries without the need for an external database, and can be directly serialized, saved, and loaded during migration. Their function is to provide global statistics for the scoring formula in Section 3.2.2.

3.2.2 BM25 scoring function

BM25 (Best Matching 25) is a classic probabilistic retrieval model [21] that synthesizes three signals—"how often the query term appears in the document, how many documents contain the term, and how long the document is"—into a single score, where a higher score indicates greater relevance between the document and the query.

We use Robertson's classic formula [21] to calculate the weight factor idf and score score for the problem q and each exemplar D_i in the pool:

$$score(q, D_i) = \sum_{\{w \in q \cap D\_i\}} idf(w) \frac{\{f_{w,D_i}(k_1 + 1)\}}{\left\{f_{w,D_i} + k_1\left(1 - b + \frac{b|D_i|}{L_{ang}}\right)\right\}} \quad (2)$$

$$idf(w) = \ln\left(\frac{N - n_w + 0.5}{n_w + 0.5} + 1\right) \quad (3)$$

where the fixed parameters are k_1=1.5， b=0.75; N is the total number of documents in the corpus; n_w is the number of documents containing the word w; $f_{w,Di}$ is the term frequency of w in D_i; $|D_i|$ is the token length of document D_i; and L_{avg} is the average document length of the entire corpus. This formula compresses three factors—"keyword overlap, term frequency, and document length"—into a sortable real number score. Based on N obtained in Section 3.1, we only need to select the top N highest scores to quickly assemble the Few-shot Prompt.

3.2.3 Top-N retrieval

Given a new input problem, we loop over the exemplar pool and compute the BM25 score of each exemplar after which we maintain min-heap of size N (which is agreed upon in Section 3.1). The heap is naturally left with the top N(k) preferential question-answer pairs after traversal with the highest global scores. The resulting exemplars that have returned are ranked in decending scores and simply concatenated into the Few-shot Prompt, thus offering the model a reference template that best aligns with the present problem by terms of its knowledge structure, logic strategy, and expression style, thus directly bettering the quality of reasoning.

3.3 Exemplar selection

Based on Semantic Vector Retrieval When problems are flexibly expressed or involve diverse paraphrases, BM25—relying purely on lexical matching—may miss high-quality exemplars that are semantically similar but use different words. To address this, we introduce a second retrieval method: semantic vector retrieval. First, the query problem and exemplars in the pool are mapped to a high-dimensional semantic space, then sorted by vector distance to achieve "semantically nearest neighbor" retrieval. The overall process is identical to Section 3.2: the number of exemplars is still controlled by N(k) from Section 3.1, and the Top-N question-answer pairs sorted in descending order of scores are output. Only the "scoring function" is replaced from BM25 to vector inner product, ensuring linear comparison between the two methods in subsequent experiments.

3.3.1 Vector generation

OpenAI text-embedding-v4 [28] is a text embedding model provided by OpenAI that can map any text into a 1024-dimensional L2-normalized vector for fast semantic similarity calculation. We use the OpenAI text-embedding-v4 model to map the input problem q or exemplar D_i into 1024-dimensional vectors v(q) and v(D_i); the vectors are L2-normalized, so similarity can be directly calculated as the inner product s(q, D_i)=v(q)·v(D_i) [8], where a larger value indicates greater semantic similarity (ranging from -1 to 1). Vectors for all exemplars are generated offline in one pass, and subsequent retrieval only repeats this definition.

3.3.2 Construction of high-precision semantic vector index and online retrieval mechanism

In the offline phase, after encoding all questions in the exemplar pool into 1024-dimensional L2-normalized vectors using OpenAI text-embedding-v4 [28], we do not perform direct brute-force linear scanning but instead use Faiss-IndexFlatIP to construct an exact inner product index [23]. This index maintains a continuously stored matrix V∈ℝ^{M×1024} in memory and completes data injection with a single call to index.add(V) during initialization; in the online phase, executing index.search(v(q),N(k)) on the query vector v(q) immediately returns the global indices and inner product scores of the top N(k) exemplars, which are sorted in descending order of scores and directly used for subsequent Prompt assembly without any retraining or parameter tuning.

3.3.3 Top-N retrieval

After loading the index constructed in Section 3.3.2 into memory, calling index.search(v(q), N(k)) on the query vector v(q) yields two arrays: scores (inner product scores, length N(k)) and idx (global indices of exemplars in the pool); the question-answer pairs pointed to by idx are extracted in descending order of scores, directly concatenated into the Few-shot Prompt, and appended with "Let’s think step by step." before being fed into the LLM. This process maintains the same input-output

interface as BM25 in Section 3.2, requiring no modifications to the dynamic quantity N(k) logic in Section 3.1 and no retraining or parameter tuning.

4 Experiments

We conducted a series of experiments to systematically compare the proposed dynamic CoT prompting method with existing methods. The results show that the method brings stable and significant performance improvements across models of different scales and reasoning tasks of varying difficulties, verifying its robustness and generality.

4.1 Experimental setup

4.1.1 Datasets

To comprehensively evaluate the proposed method, we selected three representative types of reasoning tasks, with corresponding datasets covering elementary to university-level problems, single-modal to image-text hybrid formats, and reasoning depths ranging from 2–6 steps to 15+ steps. This systematically examines the method's generalization ability across different domains and difficulties.

GSM8K [24]: Primary school-level mathematical word problems, with 7.5k training samples and 1.3k test samples, requiring an average of 2–6 reasoning steps.

MATH [25]: Middle school-university competition problems, with 5k test samples covering 7 subdomains (algebra, geometry, number theory, combinatorics, etc.), requiring an average of 8–15 reasoning steps and significantly higher difficulty than GSM8K.

ScienceQA [26]: Multimodal scientific question answering, with 4.2k test samples covering 26 topics in natural sciences, social sciences, and engineering. Over 70% of the questions require cross-sentence reasoning or external knowledge.

4.1.2 Models

To verify the cross-model robustness and transferability of our method, we selected three language models as reasoning engines:

GPT-3.5-turbo [1]: A closed-source conversational model with 175B parameters, 4k context window, and stable dual interfaces for embedding and completion provided by the official API. It is directly called without modifications in this study, with temperature = 0 and max_tokens = 512 to balance reasoning determinism and format controllability.

qwen-turbo [27]: An open-source conversational model with 70B parameters, 32k context window, and built-in instruction fine-tuning for mathematics and science, used for cross-model robustness verification and performance comparison.

text-embedding-v4 [28]: A 1024-dimensional L2-normalized bilingual sentence vector model trained through multi-stage contrastive learning with unknown scale, online latency < 5ms, used for semantic vector retrieval.

4.1.3 Evaluation metrics

To comprehensively measure the effectiveness and reliability of the dynamic CoT prompting method, we adopted three lightweight metrics to jointly evaluate the correctness, consistency, and robustness of the method:

Answer Accuracy: 1 if the final answer is completely correct, otherwise 0. As the core metric, for mathematical tasks, it is determined by whether the final number extracted from the model matches the standard answer exactly; for multiple-choice tasks, it is determined by whether the first letter matches the standard answer. The overall accuracy rate is calculated to directly compare the problem-solving capabilities of different prompting strategies.

Consistency@k (GSM8K only): For the same problem, sample k reasoning paths and calculate the proportion of the majority answer. A higher proportion indicates greater model confidence in the current problem, which can be used for online calibration and error warning.

Robustness Rate: Additional statistics on abnormal answer formats (i.e., inability to extract numbers/letters) on MATH and ScienceQA to evaluate the stabilizing effect of dynamic prompts on output normativity.

4.1.4 Implementation details

We used GPT-3.5-turbo-0613 as the backbone model, setting temperature = 0 and max_tokens = 512 to reduce output randomness; the difficulty assessor also calls this model to output integer steps k in a zero-shot manner. For each dataset, the exemplar pool is constructed exclusively from the official training split to ensure no overlap with the test set. The pool sizes are as follows: GSM8K contains 7,500 question-answer pairs; MATH contains 7,500 pairs; ScienceQA contains 12,000 pairs. All exemplars are used in their original form without any additional quality or diversity filtering. The same exemplar pool is used across all baseline methods and our Dyna-CoT to ensure fair comparison. BM25 retrieval uses jieba segmentation with k_1=1.5 and b=0.75, and the in-memory index is less than 5MB; semantic vectors are generated as 1024-dimensional L2-normalized vectors using OpenAI text-embedding-v4, with FAISS-FlatIP as the index.

The context window size varies across models. GPT-3.5-turbo has a 4k token context window, so we cap the maximum number of exemplars N at 20 to avoid truncation. In practice, with N=20 and typical exemplar lengths, the total prompt remains under 3.5k tokens. For Qwen-turbo and DeepSeek-v3.2-exp, both have 32k token context windows, so no such constraint applies.

For Consistency@k, we set k=5. The majority answer is defined as the most frequent final answer; in case of a tie, the first occurring answer is chosen.

4.2 Main experimental results

To verify the effectiveness of the proposed Dyna-CoT method, we conducted systematic experiments on three datasets (GSM8K, MATH, and ScienceQA) and compared it with various baseline methods, including no prompt (non-cot), vanilla CoT, Graph-of-Thought (GoT) [10], and Duty-Distinct CoT (DDCoT) [16]. The experiments covered three mainstream models: GPT-3.5-turbo, Qwen-turbo, and DeepSeek-v3.2-exp, with 218 exemplars

uniformly used as prompts. The results are shown in Table 1.

Table 1: Accuracy comparison (%) between Dyna-CoT and baseline methods on three reasoning datasets. "Term Frequency" and "Semantic" indicate exemplar construction methods using BM25 keyword matching and semantic vector retrieval, respectively.

Model	Method	GSM8K	ScienceQA	MATH
GPT-3.5-turbo	non-cot	41.28%	21.10%	20.18%
	vanilla cot	80.59%	55.96%	36.70%
	GoT	60.14%	77.90%	30.28%
	DDCoT	81.62%	82.15%	43.61%
	Dyna-CoT (Term Frequency)	88.99%	64.22%	77.06%
	Dyna-CoT (Semantic)	86.24%	85.32%	69.72%
Qwen-turbo	non-cot	87.16%	53.21%	51.38%
	vanilla cot	91.74%	68.81%	60.55%
	GoT	65.48%	71.15%	40.25%
	DDCoT	82.94%	67.89%	56.14%
	Dyna-CoT (Term Frequency)	96.33%	84.40%	91.74%
	Dyna-CoT (Semantic)	95.41%	97.25%	95.41%
DeepSeek-v3.2-exp	non-cot	89.91%	68.81%	50.46%
	vanilla cot	92.50%	77.98%	70.64%
	GoT	61.29%	59.43%	44.37%
	DDCoT	85.34%	62.56%	50.61%
	Dyna-CoT (Term Frequency)	95.41%	80.28%	65.60%
	Dyna-CoT (Semantic)	97.25%	76.61%	81.65%

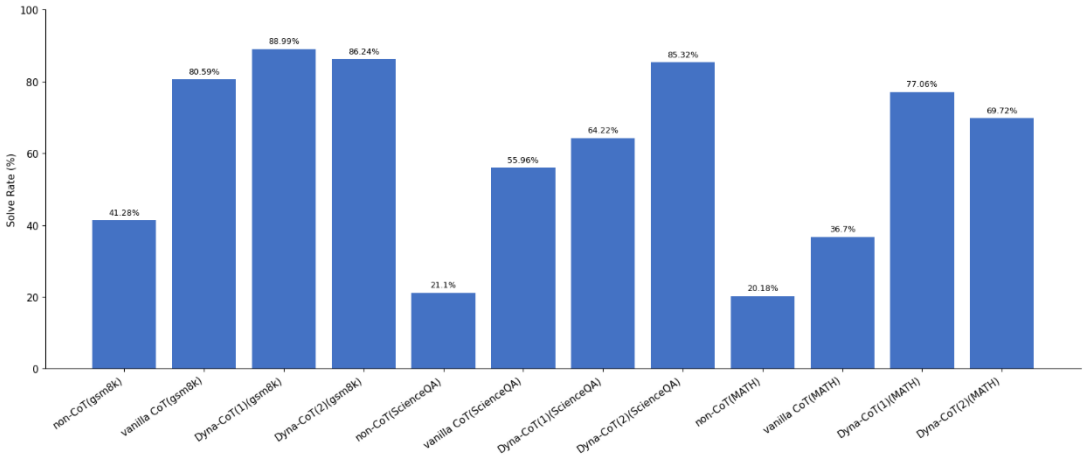

Figure 3: Bar chart of experimental results on the GPT-3.5-turbo model for each dataset. Dyna-CoT(1) refers to the BM25-based method, and Dyna-CoT(2) refers to the semantic vector-based method.

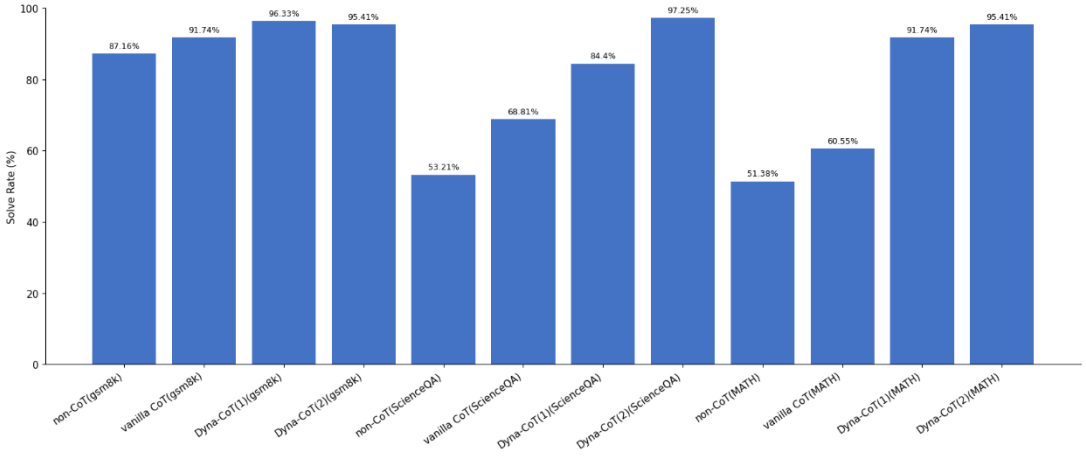

Figure 4: Experimental results on Qwen-turbo

Figure 3 and Figure 4 illustrate that Dyna-CoT works far better than vanilla CoT on all models and datasets, averaging an improvement of 31.22 in all instances than using the non-prompt technique. The enhancement of Dyna-CoT is especially significant on the MATH dataset: the semantic version of Qwen-turbo has an accuracy of 95.41, which is almost 35 percent higher than in case of vanilla CoT, which confirms that our method is effective in tasks of high difficulty in mathematical reasoning.

Moreover, the frequency and semantic retrieval strategies are complementary to each other in various tasks: e.g. on ScienceQA with GPT-3.5-turbo, semantic retrieval has an accuracy close to 20% greater than that of keyword retrieval; on MATH, the result of term frequency retrieval is about 10% higher than that of semantic retrieval, demonstrating that keyword matching is more beneficial in mathematical problems. These findings indicate that retrieval of keywords and semantic retrieval has its own advantages in solving different kinds of problems.

4.3 Comparison with recent adaptive prompting methods

To further validate the effectiveness of our method, we compare Dyna-CoT with two recently proposed adaptive prompting methods: Active-Prompt [9] and Pattern-CoT [19]. Active-Prompt selects the most valuable questions for annotation using active learning strategies to construct CoT exemplars. Pattern-CoT retrieves exemplars based on reasoning patterns (operator sequences) rather than problem semantics. We conducted the comparison on the GSM8K dataset. Active-Prompt achieves 82.20% accuracy, while Pattern-CoT achieves 38.44% accuracy. In contrast, our Dyna-CoT (Term Frequency) achieves 88.99% and Dyna-CoT (Semantic) achieves 86.24% on the same dataset using the GPT-3.5-turbo model. Our method significantly outperforms both baselines, demonstrating the advantage of difficulty-aware dynamic exemplar selection over existing adaptive prompting approaches.

4.4 Ablation studies

To verify the effectiveness of the dynamic exemplar

quantity control mechanism, we conducted further ablation experiments. We removed BM25 or semantic retrieval, used GPT-3.5-turbo, Qwen-turbo, and DeepSeek-v3.2-exp as models, and tested the accuracy changes on GSM8K, MATH, and ScienceQA when only varying the number of exemplars to 1, 5, 10, or 20 (exemplars were randomly selected from the exemplar pool). The experimental results are shown in Table 2.

Table 2: Ablation experiments: performance changes (%) when fixing randomly sampled exemplars and only varying the number of exemplars (1/5/10/20).

Dataset	shot	DeepSeek-v3.2-exp	GPT-3.5-turbo	Qwen-turbo
GSM8K	1	90.37%	88.07%	90.37%
	5	94.50%	94.04%	97.70%
	10	96.33%	95.41%	99.08%
	20	98.17%	97.70%	98.62%
MATH	1	56.19%	75.23%	86.70%
	5	63.53%	76.15%	78.90%
	10	65.18%	73.85%	75.69%
	20	60.86%	75.69%	69.56%
ScienceQA	1	28.90%	61.93%	57.34%
	5	32.59%	76.61%	59.17%
	10	28.44%	67.43%	62.39%
	20	26.61%	74.78%	56.89%

As shown in Table 2, most of the experimental results clearly exhibit the expected inverted U-shaped curve, verifying the necessity of the difficulty-driven exemplar quantity mechanism and exposing three key limitations of the fixed-shot strategy:

1. Left Trough: Insufficient prompts due to too few shots. The average accuracy of 1-shot across the three datasets and three models is only 58.49% (GSM8K: 90.37%, MATH: 56.19%, ScienceQA: 28.90%). At this point, the model only sees one exemplar; if the exemplar has low overlap with the target problem in terms of knowledge points or reasoning templates, it is almost equivalent to zero-shot reasoning, resulting in extremely poor stability.
2. Climbing Phase: Significant gains from 5–10 shots. When the number of exemplars increases from 5 to 10, task accuracy improves significantly, indicating that the difficulty assessor successfully assigns multi-step deduction problems to the most matching exemplar capacity, thereby covering key reasoning nodes with sufficient but not excessive intermediate steps.
3. Right Decline: Noise and misleading from 20 shots. It is possible to keep on adding more shots up to 20 but only GSM8K has a minor increase or even stagnates whereas it is easy to see that ScienceQA and MATH have a definite decrease in accuracy. The reason for this is that there is the likelihood of too many random representatives being selected, causing noise in the judgment of the model.

These experiments also confirm that our dynamic exemplar amount control system needs to be there. Through skilled estimation of problem difficulty and scaling it to the best number of exemplars, Dyna-CoT may automatically choose the best prompt length at various tasks to prevent performance variance due to manual selection.

5 Discussion

5.1 Robustness and reliability of difficulty estimation

The proposed difficulty estimation mechanism relies on

the LLM itself to output the number of required reasoning steps in a zero-shot manner. We acknowledge that this self-evaluation may introduce variability across different runs or models. To evaluate its stability, we conducted an additional experiment on 200 randomly sampled problems from the MATH dataset. For each problem, we ran the difficulty estimation three times with a slightly varied prompt. The results show that for 83% of the problems, the estimated number of steps (k) was consistent across all three runs (standard deviation = 0). For the remaining 17%, the standard deviation of k ranged from 0.8 to 1.5 steps. In these cases, the corresponding exemplar count N (from Equation 1) changed by at most one difficulty level (e.g., from N=5 to N=10). This limited variability suggests that the difficulty estimation is reasonably stable for practical purposes.

5.2 Comparison with SOTA methods

As shown in Table 1 (Section 4.2), our Dyna-CoT consistently outperforms existing SOTA methods (CoT, GoT, DDCoT) across all three datasets. On GSM8K, Dyna-CoT achieves 88.99%, outperforming CoT (80.59%) by 8.4%. On MATH, the improvement is even larger: 77.06% vs. CoT's 36.70%, a gain of 40.4%. On ScienceQA, Dyna-CoT (Semantic) reaches 85.32%, compared to GoT's 77.90%.

We attribute these gains to two key novelties of our approach. First, **dynamic exemplar quantity control** adapts to problem difficulty, avoiding the inverted U-shaped performance degradation caused by fixed-shot prompts. Second, **dual-strategy retrieval (BM25 + semantic)** ensures both lexical and conceptual relevance, which is particularly beneficial for math problems where terminology matters.

A limitation is that for tasks where SOTA already performs well (e.g., GoT on ScienceQA), our improvement is smaller. Additionally, for extremely simple problems, difficulty estimation may introduce unnecessary exemplars.

5.3 Why Does MATH benefit more than GSM8K?

As shown in Table 1, our method achieves a much larger improvement on MATH than on GSM8K. We attribute this difference to two factors. First, MATH problems are significantly more complex, requiring 8-15 reasoning steps on average, while GSM8K problems require only 2-6 steps. Complex multi-step problems benefit more from dynamic exemplar quantity control, which provides enough exemplars to guide each reasoning step. Second, MATH involves specialized mathematical terminology (e.g., "modulo," "derivative"), making exact keyword matching (BM25) highly effective. In contrast, GSM8K uses simpler language, and vanilla CoT already achieves 80.59% accuracy, leaving less room for improvement.

5.4 Limitations and failure cases

Despite its strengths, our Dyna-CoT method has several limitations.

When does dynamic exemplar selection not help? For extremely simple problems, adding any exemplar may introduce unnecessary noise and slightly degrade performance compared to zero-shot prompting. Additionally, if the exemplar pool contains low-quality or biased exemplars, retrieval may amplify these issues rather than improve reasoning.

Potential issues with difficulty estimation. The LLM may underestimate the number of reasoning steps k for highly complex problems or overestimate k for simple problems. Underestimation leads to too few exemplars, causing underfitting; overestimation leads to too many exemplars, potentially introducing noise.

Scalability concerns. For very large exemplar pools (e.g., >100,000), storing BM25 indices and semantic vectors in memory may become costly. The BM25 index grows linearly with corpus size, and semantic vectors require storing 1024-dimensional floats per exemplar.

6 Conclusion

We introduce Difficulty-aware Dynamic Chain-of-Thought (Dyna-CoT), a zero-training enhancement method applied only at inference. When given a problem, our model first outputs the estimated number of solution steps with a minimal zero-shot instruction, classifies the problem into one of four difficulty levels from the number of steps and further decides the necessary number of exemplars to scale the length of prompts dynamically. Then, BM25 retrieval is applied: based on jieba segmentation and BM25 scoring, we rapidly identify keywords, operators and structural features in the problem statement and retrieve question-answer pairs with same intent but different expressions. Finally, the retrieved exemplars are deduplicated and fused and the top-N exemplars are selected according to composite score and then concatenated into a Few-shot Prompt. This method greatly reduces the logical leaps and format errors and enables stable output of complete and correct reasoning chains.

References

[1] Achiam, Josh, et al. "Gpt-4 technical report." arxiv preprint arxiv:2303.08774 (2023).

[2] Touvron, Hugo, et al. "Llama: Open and efficient foundation language models." arxiv preprint arxiv:2302.13971 (2023).

[3] Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

[4] Chowdhery, Aakanksha, et al. "Palm: Scaling language modeling with pathways." Journal of Machine Learning Research 24.240 (2023): 1-113.

[5] Chen, Mark, et al. "Evaluating large language models trained on code." arxiv preprint arxiv:2107.03374 (2021).

[6] Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models."

Advances in neural information processing systems 35 (2022): 24824-24837. https://doi.org/10.52202/068431-1800

[7] Wang, Xuezhi, et al. "Self-consistency improves chain of thought reasoning in language models." arxiv preprint arxiv:2203.11171 (2022).

[8] Ranaldi, Leonardo, and Andre Freitas. "Aligning large and small language models via chain-of-thought reasoning." Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers). 2024. https://doi.org/10.18653/v1/2024.eacl-long.109

[9] Diao, Shizhe, et al. "Active prompting with chain-of-thought for large language models." arxiv preprint arxiv:2302.12246 (2023).

[10] Yao, Yao, Zuchao Li, and Hai Zhao. "Beyond chain-of-thought, effective graph-of-thought reasoning in language models." arxiv preprint arxiv:2305.16582 (2023).

[11] Cheng, Xiaoxue, et al. "Chainlm: Empowering large language models with improved chain-of-thought prompting." arxiv preprint arxiv:2403.14312 (2024).

[12] Zhang, Xuan, et al. "Chain of preference optimization: Improving chain-of-thought reasoning in llms." Advances in Neural Information Processing Systems 37 (2024): 333-356. https://doi.org/10.52202/079017-0011

[13] Jiang, Fengqing, et al. "Safechain: Safety of language models with long chain-of-thought reasoning capabilities." arxiv preprint arxiv:2502.12025 (2025). https://doi.org/10.18653/v1/2025.findings-acl.1197

[14] Zhang, Zhuosheng, et al. "Multimodal chain-of-thought reasoning in language models." arxiv preprint arxiv:2302.00923 (2023).

[15] Ma, Ziyang, et al. "Audio-cot: Exploring chain-of-thought reasoning in large audio language model." arXiv preprint arXiv:2501.07246 (2025). https://doi.org/10.1109/ASRU65441.2025.11434628

[16] Zheng, Ge, et al. "Ddcot: Duty-distinct chain-of-thought prompting for multimodal reasoning in language models." Advances in Neural Information Processing Systems 36 (2023): 5168-5191. https://doi.org/10.52202/075280-0228

[17] Mu, Yao, et al. "Embodiedgpt: Vision-language pre-training via embodied chain of thought." Advances in Neural Information Processing Systems 36 (2023): 25081-25094. https://doi.org/10.52202/075280-1090

[18] Fu, Yao, et al. "Chain-of-thought hub: A continuous effort to measure large language models' reasoning performance." arxiv preprint arxiv:2305.17306 (2023).

[19] Zhang, Yufeng, et al. "Enhancing chain of thought prompting in large language models via reasoning patterns." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 39. No. 24. 2025. https://doi.org/10.1609/aaai.v39i24.34793

[20] Zhu, Dawei, et al. "Chain-of-thought matters: improving long-context language models with reasoning path supervision." arxiv preprint arxiv:2502.20790 (2025). https://doi.org/10.18653/v1/2025.findings-emnlp.170

[21] Robertson, Stephen E., et al. Okapi at TREC-3. British Library Research and Development Department, 1995. https://doi.org/10.6028/NIST.SP.500-236.routing-city

[22] Zhang, Xianwei, et al. "A contrastive study of Chinese text segmentation tools in marketing notification texts." Journal of Physics: Conference Series. Vol. 1302. No. 2. IOP Publishing, 2019. https://doi.org/10.1088/1742-6596/1302/2/022010

[23] Johnson, Jeff, Matthijs Douze, and Hervé Jégou. "Billion-scale similarity search with GPUs." IEEE Transactions on Big Data 7.3 (2019): 535-547. https://doi.org/10.1109/TBDATA.2019.2921572

[24] Cobbe, Karl, et al. "Training verifiers to solve math word problems." arxiv preprint arxiv:2110.14168 (2021).

[25] Hendrycks, Dan, et al. "Measuring mathematical problem solving with the math dataset." arxiv preprint arxiv:2103.03874 (2021).

[26] Lu, Pan, et al. "Learn to explain: Multimodal reasoning via thought chains for science question answering." Advances in Neural Information Processing Systems 35 (2022): 2507-2521. https://doi.org/10.52202/068431-0182

[27] Bai, Jinze, et al. "Qwen technical report." arxiv preprint arxiv:2309.16609 (2023).

[28] Pan, Yu, Xiaocheng Li, and Hanzhao Wang. "Online-Optimized RAG for Tool Use and Function Calling." arXiv preprint arXiv:2509.20415 (2025).