Not All Tokens Are What You Need In Thinking

Anonymous ACL submission

Abstract

Modern reasoning models, such as OpenAI's o1 and DeepSeek-R1, exhibit impressive problem-solving capabilities but suffer from critical inefficiencies: high inference latency, excessive computational resource consumption, and a tendency toward overthinking–generating verbose chains of thought (CoT) laden with redundant tokens that contribute minimally to the final answer. To address these issues, we propose Conditional Token Selection (CTS), a token-level compression framework with a flexible and variable compression ratio that identifies and preserves only the most essential tokens in CoT. Using conditional importance scoring, CTS evaluates each token's contribution to deriving correct answers and then trains models on compressed CoT. Extensive experiments demonstrate that CTS effectively compresses long CoT while maintaining strong reasoning performance. Notably, on the GPQA benchmark, Qwen2.5-14B-Instruct trained with CTS achieves a 9.1% accuracy improvement with 13.2% fewer reasoning tokens (13% training token reduction). Further reducing training tokens by 42% incurs only a marginal 5% accuracy drop while yielding a 75.8% reduction in reasoning tokens, highlighting the prevalence of redundancy in existing CoT.

1 Introduction

Large reasoning models such as o1(Jaech et al., 2024) and R1(Guo et al., 2025) significantly enhance their reasoning capabilities through reinforcement learning, instructing models to generate thoughtful reasoning steps before producing final answers. Guo et al. (2025) demonstrated that by fine-tuning non-reasoning models like Qwen2.5-14B-Instruct on long Chain of Thought (CoT) data generated by R1, these models can acquire comparable reasoning abilities, even surpassing o1-mini on math and code reasoning tasks. Consequently, numerous distilled R1 reasoning datasets

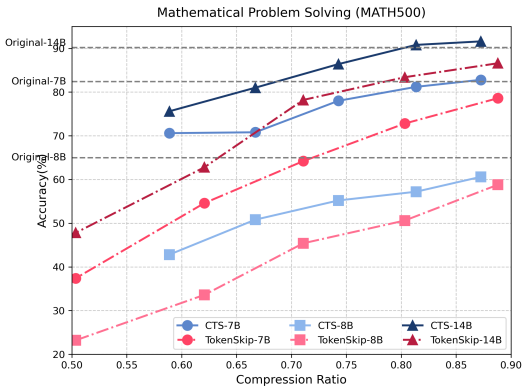

Figure 1: Comparison of accuracy across different compression ratios for various model configurations.

have emerged, including s1K(Muennighoff et al., 2025), SkyThought(Team, 2025), OpenMathReasoning(Moshkov et al., 2025), and AM-1.4M (Zhao et al., 2025). Small-scale language models trained on these datasets consistently demonstrate remarkable reasoning capabilities. However, the ever-increasing length of CoT sequences burdens both training and inference, with recent studies (Sui et al., 2025) revealing that models often overthink, expending substantial resources on redundant reasoning steps.

This inefficiency raises a critical question: "**How can we maintain the high performance of long CoT reasoning while eliminating its computational waste?**" Existing solutions, such as TokenSkip's task-agnostic compression (Xia et al., 2025), show promise for short CoT sequences but fail to address the unique challenges of reinforcement learning-generated long CoT data (spanning thousands of tokens). Moreover, they focus solely on compressing the CoT trajectory while overlooking contextual signals like questions and answers, which prior work Tian et al. (2025) identifies as key to effective compression.

To bridge this gap, we propose Conditional Token Selection, a framework that dynamically

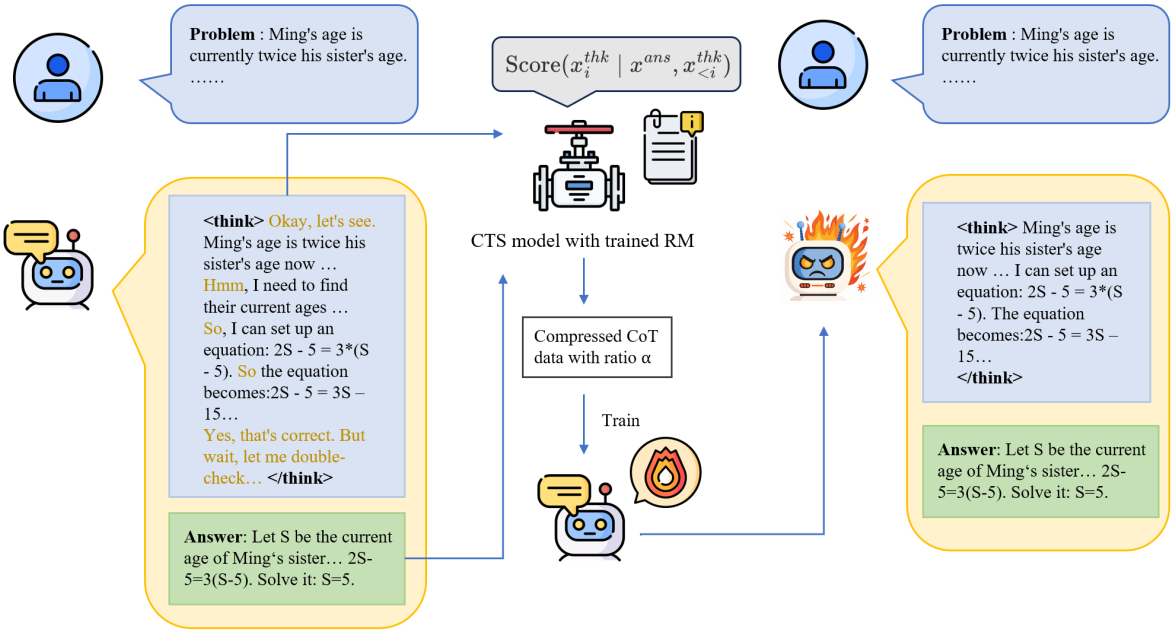

Figure 2: Illustration of Conditional Token Selection. For long CoT datasets, CTS leverages a well-trained reference model to evaluate the importance of each thinking token conditional on the answer, removing less important tokens based on the compression ratio α. The model is then trained on this compressed data, enabling more efficient reasoning capabilities.

prunes redundant reasoning tokens while preserving those essential for deriving answers. CTS leverages a reference model (RM) trained on high-quality reasoning corpora to score token importance conditioned on critical context (e.g., questions and answers). As shown in Figure 2, by filtering CoT data at adjustable compression ratios and then fine-tuning model with compressed data, we enable models to learn how to skip unnecessary reasoning tokens during inference.

We conducted extensive experiments on models of various sizes, including the Llama-3.1-8B-Instruct (Grattafiori et al., 2024) and the Qwen2.5-Instruct series (Qwen et al., 2025). The experimental results demonstrate the effectiveness of our method and confirm that there indeed exist many redundant thinking tokens in long CoT data. As illustrated in Figure 1, our approach demonstrates comprehensive accuracy improvements over the prominent TokenSkip (Xia et al., 2025) across all three models on the MATH500 benchmark (Hendrycks et al., 2021b). Additionally, when applied to Qwen2.5-14B-Instruct on the GPQA benchmark (Rein et al., 2024), CTS demonstrates remarkable efficiency gains: a modest 13% reduction in training tokens results in both a **9.1%** accuracy improvement and 13.2% fewer reasoning tokens. When training token reduction is increased to **42%**, the method incurs only a minor 5% accuracy decrease while achieving a dramatic **75.8%** reduction in reasoning tokens, highlighting an advantageous accuracy-efficiency trade-off. Furthermore, although the model was trained on mathematical datasets, our experiments demonstrate its effectiveness on out-of-domain tasks, including code generation benchmarks such as HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021).

In summary, our key contributions are:

- We introduce the Conditional Token Selection framework, which dynamically assigns importance scores to tokens in CoT trajectories, enabling flexible and context-aware preservation of critical reasoning tokens.
- We've developed a reference model, trained on high-quality reasoning data, that more accurately assesses the conditional importance of tokens in reasoning CoTs. This model can be applied to other independent tasks, such as prompt compression.
- We comprehensively compare token-based compression methods both conditional and non-conditional for long CoT data generated by reinforcement learning, thereby validating our token selection strategies.

2 Preliminaries

In this section, we introduce some important preliminary concepts. Unless specified, boldface lowercase letters (e.g., $\boldsymbol{x}, \boldsymbol{y}$) are used to represent concatenated token sequences, whereas individual tokens are denoted by subscripted lowercase letters (e.g., x_i, y_i). $\boldsymbol{x}_{<t}$ represents the subsequence of $\boldsymbol{x}$ containing all tokens with indices less than t.

For a given context $\boldsymbol{x} = \{x_i\}_{i=1}^{n}$, the self-information of a token x_t ($1 \leq t \leq n$) can be defined as:

$$\begin{aligned} I(x_t) &= -\log_2 P(x_t \mid \boldsymbol{x}_{<t}) \\ &= -\log_2 P(x_t \mid x_1, ..., x_{t-1}) \end{aligned} \tag{1}$$

Perplexity (PPL) is then defined based on self-information as:

$$\text{PPL}(x_t) = 2^{I(x_t)} \tag{2}$$

Perplexity is commonly used to measure a language model's ability to predict the given context. Since removing tokens with lower perplexity (Li et al., 2023) has a relatively minor impact on the model's understanding and prediction of the context, perplexity can serve as a viable metric for evaluating token importance. To mitigate information redundancy in prompts with long contexts and avoid resource waste (e.g., reducing tokens consumption in LLM API calls), Li et al. (2023) proposed Selective Context that uses a small language model to assess the perplexity of each lexical unit (such as sentences or tokens) in the original prompts, and then drops the less informative content for prompt compression.

Selective context is an unconditional approach, as it does not utilize existing important information (such as the standard answers to math problems, which could serve as key references for simplifying or compressing lengthy reasoning steps) when calculating token importance. Next, we introduce conditional perplexity as a more fine-grained metric for token importance, which will be employed in our proposed method. Given essential information $\boldsymbol{v}$, the conditional self-information and perplexity can be defined as:

$$\begin{aligned} I(x_t \mid \boldsymbol{v}) &= -\log_2 P(x_t \mid \boldsymbol{v}, \boldsymbol{x}_{<t}) \\ \text{PPL}(x_t \mid \boldsymbol{v}) &= 2^{I(x_t \mid \boldsymbol{v})} \end{aligned} \tag{3}$$

3 Methodology

Building upon the concept of utilizing essential information as conditions, we propose **C**onditional **T**oken **S**election (CTS). It employs a well-trained reference model to evaluate conditional perplexity for compressing long CoT sequences, such as those generated by harnessing reinforcement learning techniques (e.g., R1) and subsequently fine-tunes models using this compressed data. While small models fine-tuned on original long CoT data exhibit strong reasoning capabilities, they suffer from slow inference speed, redundant reasoning processes, and high computational resource consumption. By selectively retaining critical tokens, CTS aims to enhance model performance while significantly reducing both training and inference resource overhead.

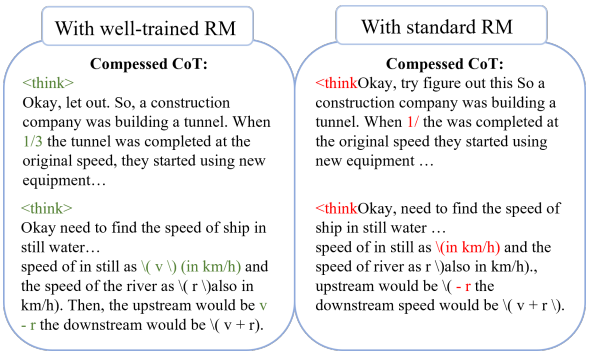

Figure 3: An example of CoT compression using CTS, where the left figure employs a well-trained RM, while the right uses a standard RM.

3.1 Problem Formulation

Given a long CoT dataset, each instance $\boldsymbol{x}$ is a token sequence composed of three segments: a problem statement $\boldsymbol{x}^{\text{prob}}$, a chain-of-thought reasoning process $\boldsymbol{x}^{\text{thk}}$, and a final answer $\boldsymbol{x}^{\text{ans}}$. Formally, $\boldsymbol{x}$ is structured as the concatenation $\boldsymbol{x} = \{\boldsymbol{x}^{\text{prob}}, \boldsymbol{x}^{\text{thk}}, \boldsymbol{x}^{\text{ans}}\}$, illustrated in Figure 2.

In the process of distillation, the Large Reasoning Model (e.g. R1 and o1) initially generates long CoT data, which is subsequently employed to fine-tune small-scale language models (e.g., Qwen2.5-14B-Instruct). The primary objective of distillation approach is to endow smaller language models with enhanced reasoning capabilities. Then these distilled models can achieve strong performance under resource-constrained conditions. The loss function of distillation can be reformulated as:

$$\mathcal{L} = -\sum_{i=1}^{l} \log P_\theta(y_i \mid \boldsymbol{x}^{\text{prob}}, \boldsymbol{y}_{<i}) \tag{4}$$

where $\boldsymbol{y} = \{y_i\}_{i=1}^{l} = \{\boldsymbol{x}^{\text{thk}}, \boldsymbol{x}^{\text{ans}}\}$ is the target sequence of l tokens and θ are model parameters.

However, the ever-increasing length of CoT sequences burdens both training and inference.

Conditional Token Selection identifies and retains critical reasoning tokens in long CoT training data while eliminating redundant tokens that hinder model reasoning efficiency. When fine-tuned on CTS-filtered CoT data, the model retains strong reasoning capabilities while significantly reducing inference token consumption, thereby enhancing reasoning efficiency. The objective of CTS can be defined as:

$$\arg\min_{\tilde{\boldsymbol{y}}} \left\{ \text{dist}(A, \tilde{A}) + \lambda \|\tilde{\boldsymbol{y}}\|_0 \right\}, \tag{5}$$

Where $\tilde{\boldsymbol{y}}$ represents the compressed CoT, a subsequence of $\boldsymbol{y}$. A and $\tilde{A}$ represent the answers to any question Q given by the small language models trained with $\boldsymbol{y}$ and $\tilde{\boldsymbol{y}}$, respectively. Here, $\text{dist}(\cdot, \cdot)$ is a function measuring the distance (e.g., KL divergence). λ serves as a hyper-parameter balancing the compression ratio and distance. The $\|\cdot\|_0$ norm serves as a penalty term to control the complexity of the compressed data $\tilde{\boldsymbol{y}}$, preventing trivial compression (i.e. $\tilde{\boldsymbol{y}} = \boldsymbol{y}$). This regularization term is also incorporated in Equation 1 (Jiang et al., 2024).

3.2 Reference Modeling

When compressing CoT data, a **R**eference **M**odel (RM) is typically required to estimate self-information or perplexity using Equations (1) and (2). This RM is usually a lightweight small language model (e.g. Qwen2.5-7B-Instruct, Llama3-8B-Instruct). However, through experimental observations, we found that using RM directly to compress CoT tends to remove important but commonly used numbers or alphanumeric symbols. As depicted in the right panel of Figure 3, both the water flow velocity variable 'v' and the denominator '3' from the fraction 1/3 have been omitted. These elements are essential for comprehending the subsequent equations.

To teach the RM which variables and logical symbols are critical for deriving the final answer, we can modify the model's conditional output distribution by increasing the probabilities of symbols when reasoning steps are used as conditions. First, we collected mathematical reasoning data containing long CoT sequences. These were initially filtered using large language models with strong mathematical capabilities (e.g., DeepSeek-R1) to retain only the most crucial reasoning steps, with particular attention to preserving critical numerical values and logical symbols. Subsequently, we conducted manual screening to eliminate problems with incorrect reasoning outcomes (see Appendix for more details). It is worth noting that this portion of CoT data differs from the distilled data requiring compression in our method.

Each instance in the curated dataset that preserves critical variables and logical symbols can be represented as a token sequence $\{\boldsymbol{q}, z_1, \ldots, z_K, \boldsymbol{a}\}$ where, q denotes the question, a represents the answer, and $z_1, \ldots, z_K$ are the reasoning steps retaining essential symbolic information. The RM can be trained using the following cross-entropy loss:

$$\mathcal{L}_{RM} = -\sum_{i=1}^{K} \log P(z_i \mid x^{ins}, \boldsymbol{q}, \boldsymbol{a}, \boldsymbol{z}_{<i}) \tag{6}$$

Where x^{ins} represents "*For a problem* $\boldsymbol{q}$*, the following reasoning steps are important to get the answer* $\boldsymbol{a}$". The resulting RM is then used to assess the token importance within long CoT trajectories, allowing us to focus on the most influential tokens.

3.3 Token Selection Based on Conditional Perplexity Differences

To reduce token consumption during training while maintaining powerful and efficient reasoning capabilities of small-scale language model, we apply conditional compression to CoT data. We first employ the RM to compute conditional perplexity scores for each token in target sequences $\boldsymbol{y}$, given question and answer $\boldsymbol{x}^{\text{prob}}, \boldsymbol{x}^{\text{ans}}$. Then token compression with a flexible ratio can be achieved by sorting all tokens in the context by perplexity and subsequently removing lower-ranking tokens.

When utilizing perplexity for token compression, to ensure the compressed data and original data distributions are the same, the following conditional independence assumption is required:

$$A \perp\!\!\!\perp \bar{\boldsymbol{y}} \mid \tilde{\boldsymbol{y}} \tag{7}$$

Here, $\bar{\boldsymbol{y}}$ denotes the removed tokens from the target sequence $\boldsymbol{y}$, which is composed as the concatenation of retained tokens $\tilde{\boldsymbol{y}}$ and removed tokens $\bar{\boldsymbol{y}}$, i.e., $\boldsymbol{y} = \tilde{\boldsymbol{y}} \oplus \bar{\boldsymbol{y}}$. Under this assumption, the model can derive correct answer from the compressed reasoning data $\tilde{\boldsymbol{y}}$, satisfying:

$$p(A \mid \tilde{\boldsymbol{y}}) \approx p(A \mid \boldsymbol{y}) \tag{8}$$

To mitigate the inaccuracy caused by the conditional independence assumption, we adopt the

iterative compression approach proposed by Jiang et al. (2023). This method comprises three key steps:(1) uniformly partitioning $\boldsymbol{y}$ into m segments $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$; (2) applying CTS to each segment to generate $\{\tilde{\boldsymbol{s}}_1, \ldots, \tilde{\boldsymbol{s}}_m\}$; (3) concatenating the compressed segments to produce the final compressed sequence $\tilde{\boldsymbol{y}}$.

For more fine-grained assessment of token conditional importance, instead of using conditional perplexity scores directly, we use the distribution shift caused by the condition of the answer to quantify the association between the thinking token and the answer. Consequently, we derive a score, r_i, for each token in the target sequence y, calculated as follows:

$$r_i = \text{PPL}(y_i \mid \boldsymbol{y}_{<i}) - \text{PPL}(y_i \mid \boldsymbol{x}^{\text{prob}}, \boldsymbol{x}^{\text{ans}}, \boldsymbol{y}_{<i})$$

As shown in Algorithm 1 in the appendix, given a compression ratio α, after scoring each token in target sequence using the RM, we determine a threshold r_α represented by the α-quantile of importance scores, thereby selecting reasoning tokens whose scores exceed the threshold:

$$\tilde{\boldsymbol{y}} = \{y_i \mid r_i > r_\alpha\}.$$

4 Experiments

4.1 Experimental Setup

4.1.1 Reference Model Training

To train our mathematical reasoning reference model, we utilized the first 9.3K samples from the MATH training dataset (Face, 2025), comprising problems from NuminaMath 1.5 with reasoning traces generated by DeepSeek-R1. To imbue the model with the ability to assess token importance in mathematical reasoning chains, DeepSeek-R1 was prompted to meticulously select 8 million tokens from an original pool of 54 million reasoning tokens. This curated set of tokens is subsequently employed to supervise the fine-tuning of the RM.

4.1.2 Implementation Details & Dataset

We adopt the second 9.3K subset of MATH (Face, 2025) as the original long CoT data to be compressed. Building upon the framework introduced by Jiang et al. (2024), we perform conditional token selection using our trained RM. We then fine-tuned Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen2.5-7B/14B-Instruct (Qwen et al., 2025) using the compressed dataset with varying compression ratios $\alpha \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$.

4.1.3 Evaluation Benchmarks & Metrics

The evaluation leverages three widely used reasoning benchmarks: **AIME24**, **MATH500**, and **GPQA Diamond** (Mathematical Association of America, 2024; Hendrycks et al., 2021b; Rein et al., 2024). We used the actual compression ratio, average accuracy, and average reasoning token count as metrics to evaluate compression methods. All training and evaluation were conducted on 8 NVIDIA A800 GPUs. During training, we fine-tuned for 3 epochs with a batch size of 16. The maximum learning rate was set at 1e-5 with a cosine decay schedule. We set the maximum sequence length to 4096.

4.1.4 Baselines

In our main experiments, we compared Conditional Token Selection with **unconditional TokenSkip** (Xia et al., 2025), **LLMLingua** (Jiang et al., 2023) and **Prompt-based Compression**. We designate the method that directly uses the original CoT data for distillation (equation 4) as **Original**. For the Prompt-based method, we instructed GPT-4o[1] to compress long CoT trajectories by providing prompts such as "Please retain important reasoning tokens in the Chain-of-Thought and remove unnecessary ones, preserving α% of the original tokens." However, we observed that GPT tends to be overly aggressive in compression, consistently preserving less than 10% of the original tokens regardless of the specified α value. Therefore, we did not focus on the compression ratio α and simply used GPT-4o to compress the reasoning chains directly. These baselines are referred to as **GPT-4o**, **LLMlingua** and **TokenSkip** in Table 1, respectively.

4.2 Main Results

Table 1 presents the performance of different compression methods on the Qwen2.5-14B-Instruct model across various compression ratios. Notably, our method achieves the highest accuracy across all five compression ratios compared to other compression approaches. Figure 1 illustrates the accuracy of CTS versus TokenSkip across different models and compression ratios. CTS consistently outperforms TokenSkip, highlighting the superiority of conditional token compression.

Although the Prompt-based GPT-4o shown in Table 1 uses fewer inference tokens, this is a result of its excessive compression of the chain of

[1]We use the gpt-4o-2024-08-06 version for experiments.

Table 1: Experimental results of various compression methods on **Qwen2.5-14B-Instruct**, showing accuracy, average reasoning CoT tokens, and compression ratio (actual ratio).

Methods	Ratio (Actual)	MATH500		AIME24		GPQA Diamond	
		Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓
Original	1.0	90.2	5012	40	23041	51.5	12000
GPT-4o	0.9(0.06)	61.4	**283**	0	**353**	35.8	**353**
LLMLingua	0.9(0.89)	84.4	5023	33.3	19731	53.0	10689
	0.8(0.81)	65.6	2510	10.0	4230	44.9	4037
	0.7(0.73)	60.4	2511	6.7	4588	43.9	3371
	0.6(0.62)	59.0	2270	6.7	3076	40.9	3347
	0.5(0.55)	53.6	1998	3.3	3789	40.4	2796
TokenSkip	0.9(0.88)	86.6	4941	40.0	19985	50.0	12455
	0.8(0.80)	83.4	5549	26.7	20945	50.0	13275
	0.7(0.71)	78.2	6566	16.7	24718	43.4	15531
	0.6(0.62)	62.8	8595	10.0	27748	38.8	16764
	0.5(0.50)	47.8	9824	3.3	26555	31.8	19121
CTS	0.9(0.87)	**91.6**	4703	**50.0**	21668	**60.6**	10413
	0.8(0.81)	**90.8**	4922	**43.4**	21767	**53.5**	13136
	0.7(0.74)	**86.4**	3310	**33.3**	10448	**57.1**	10372
	0.6(0.66)	**81.0**	3787	**16.7**	10308	**48**	9712
	0.5(0.58)	**75.6**	2036	**10.0**	3196	**46.5**	2906

thought. In reality, the model fails to learn thinking capabilities from long CoT data, resulting in low accuracy across all three benchmarks.

Furthermore, for CTS, when the compression ratio is 0.9 and 0.8, the model shows improvements in both accuracy and inference efficiency. For example, on AIME24, at compression ratios of 0.9 and 0.8, accuracy increased by 10% and 4.3% respectively, while inference tokens decreased by 1373 and 1274. Notably, for GPQA Diamond, CTS achieves a 9.1% accuracy gain with 13.2% fewer reasoning tokens. Further reduction of inference tokens by 75.8% results in only a 5% accuracy drop, and since the compression ratio at this point is 42%, training costs are significantly reduced.

As the compression ratio decreases, although CTS accuracy declines compared to Original, inference tokens continue to decrease. Thus, there exists a trade-off between accuracy and inference efficiency. From Table 1, we can infer that the optimal point lies between ratios 0.7 and 0.8, where model capability remains unchanged while minimizing inference token consumption.

However, poor compression methods can actually decrease model inference efficiency. As demonstrated in Figure 4, TokenSkip's token consumption during reasoning exhibits an unexpected increase with higher compression ratios. Through meticulous examination of the model's reasoning process during testing, we observed that models trained on TokenSkip-compressed data tend to enter repetitive loops when addressing challenging queries, persistently outputting certain phrases until reaching the maximum token limit of 32,768. We speculate that as the compression ratio decreases, TokenSkip eliminates crucial reasoning-termination tokens, thereby impairing the model's reasoning capabilities. The primary reason is that TokenSkip's token importance metric was trained on MeetingBank data (Xia et al., 2025). While effective for compressing short CoT sequences, it fails to handle more challenging long CoT data containing complex reasoning steps. A comprehensive discussion can be found in the Appendix.

Figure 5 displays the percentage of reasoning token reduction and accuracy improvement for three methods. Higher accuracy with greater reasoning token reduction is preferable. Therefore, our method achieves the optimal balance between token reduction and accuracy improvement.

We've considered the reference model's overhead. Training the RM required 11.44 A800 GPU

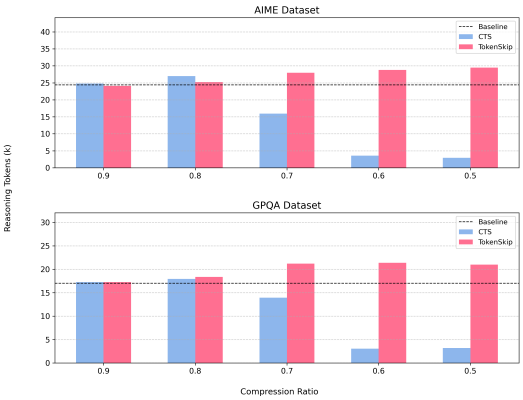

Figure 4: Compression Ratio vs. Average Reasoning Tokens for CTS and TokenSkip on the Qwen2.5-7B-Instruct model across different test sets.

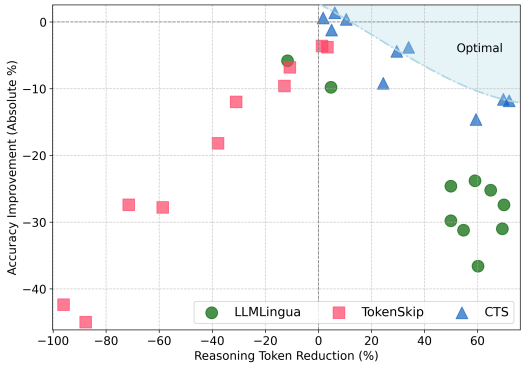

Figure 5: Performance metrics (Reasoning Token Reduction percentage and Absolute Accuracy Improvement (%) relative to original) for various compression methods and ratios on Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct models and MATH500. The top-right region represents optimal performance, signifying higher accuracy and reduced reasoning token usage.

hours (4.13×10^{18}). For Qwen-7B-Instruct with CTS-14B at 0.7 compression ratio, inference saves 1.5×10^{14} FLOPs/inference, reaching breakeven at 27,500 calls. For MMLU Science tasks (16,000 questions) (Hendrycks et al., 2021a), just two evaluations suffice, making the training cost negligible. (See Appendix D for more details.)

Table 2: Ablation Study Variant Comparison

Model Variant	Conditional	Trained RM
Base	×	×
+ Conditional	✓	×
+ RM-Tuned	×	✓
Proposed (CTS)	✓	✓

4.3 Ablation Study and Generalization Research

To investigate the generalization capability of our approach, we evaluated CTS on two widely-used code datasets: HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021). Due to space constraints, we present only the results on the HumanEval benchmark in the main text, with comprehensive details provided in the Appendix. As shown in Figure 6, although trained solely on mathematical datasets, our model achieves higher accuracy and fewer reasoning tokens on code tasks at lower compression ratios. This robust cross-domain performance strongly validates the generalization capability of our approach.

Our ablation study aims to verify: 1) The effectiveness of using a well-trained reference model for selecting valuable tokens, and 2) The effectiveness of conditional token importance assessment compared to unconditional methods. For the unconditional token perplexity assessment, we follow the framework established in Jiang et al. (2023).

We introduce the following variants of our method for the ablation study, as shown in Table 2: (1) **Base**: Using an standard RM to unconditionally predict token perplexity in CoT; (2) **+ Conditional**: Using an standard RM to predict conditional token perplexity in CoT; (3) **+ RM-Tuned**: Using a well-trained RM to unconditionally predict token perplexity in CoT; (4) **Proposed (CTS)**: Using a well-trained RM to predict conditional token perplexity in CoT.

Conditional compression methods significantly enhance model performance. As shown in Figures 7, the top two curves—**Proposed (CTS)** and **+ Conditional**—leverage conditional token importance prediction, demonstrating their effectiveness in achieving higher accuracy. For both conditional and unconditional methods (**+ Conditional** and **Base**, respectively), the well-trained RM shifts the curves upward, further improving accuracy. This indicates that training the reference model provides some benefit in identifying important reasoning tokens. The accuracy improvement achieved by the reference model is relatively marginal compared to the conditional approach. Therefore, under circumstances with limited resources, employing **+ Conditional** alone can still yield satisfactory results.

Table 2 in appendix demonstrates that reasoning tokens do not differ substantially across variants, indicating that variants do not significantly improve

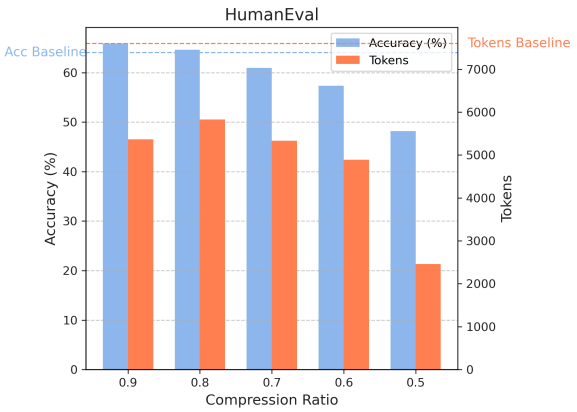

Figure 6: Generalization experiments of the CTS-14B model on the HumanEval datasets under different compression ratios

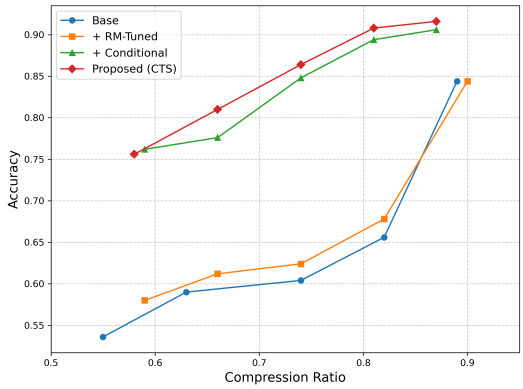

Figure 7: Ablation experiments of the Qwen2.5-14B-Instruct model on the MATH500 dataset under different compression ratios

inference efficiency. Interestingly, at certain compression ratios, while model accuracy improves, the number of inference tokens increases, resulting in reduced inference efficiency. This observation aligns with test time scaling results in Zhang et al. (2025), which indicate that model capability scales with inference length.

5 Related Work

5.1 Efficient Reasoning

Chen et al. (2025); Team et al. (2025) demonstrated that in long CoT reasoning models, models generate overly detailed or unnecessarily elaborate reasoning steps, ultimately reducing their problem-solving efficiency. Prompt-based Chain of Thought (CoT) methods (Wei et al., 2022; Kojima et al., 2022) guide models to think step-by-step, enhancing their problem-solving capabilities. Chain of Draft (Xu et al., 2025), through prompting, retains essential formulas and numbers in the thought chain, maintaining performance while reducing inference costs. Wu et al. (2025); Ma et al. (2025) implement thought intervention by incorporating first-person prompts in the model's thinking process to achieve instruction following and efficient reasoning.

Kang et al. (2025) improves reasoning efficiency by compressing the reasoning process. Yu et al. (2025) trains models by collecting variable-length CoT reasoning data. Munkhbat et al. (2025); Yeo et al. (2025); Xia et al. (2025) collect short CoT data by reducing redundant reasoning steps after full-length reasoning.

5.2 Prompt Compression

As language models increase in parameter scale, their capabilities also grow stronger, enabling many tasks to achieve better results merely by changing the model's input, such as RAG (Lewis et al., 2020) and few-shot learning (Wang et al., 2020). As prompts become longer, resource consumption increases significantly. Li et al. (2023) proposed evaluating token importance by calculating token perplexity in context to compress the input.

Building on this foundation, Jiang et al. (2023) proposed a mixed coarse-grained and fine-grained compression method. Jiang et al. (2024) extended this work by introducing a task-aware prompt compression method that incorporates conditional information from the query when calculating perplexity. Pan et al. (2024) transformed token compression into a binary classification problem, utilizing BERT as a classifier.

6 Conclusion

We introduce Conditional Token Selection , a novel method that employs a fine-tuned reference model to assess conditional perplexity differences in long CoT sequences, pinpointing the most reasoning-critical tokens for accurate answers. Through flexible compression ratios, CTS efficiently condenses CoT data while preserving reasoning quality and output efficiency. Experimental results demonstrate CTS's effectiveness across multiple LLMs and tasks. Our method demonstrates strong generalization, even on out-of-domain (code) tasks. Ablation studies show that it remains effective without the reference model under resource constraints, further validating its robustness.

Limitations

First, resource constraints also prevented experiments with larger models such as 32B and 72B variants. Second, very high compression ratios may affect the interpretability of reasoning chains for human readers, potentially limiting their educational or explanatory value in applications where transparency is important.

Ethical Statement

The datasets used in our experiments are publicly available, English-labeled, and privacy-compliant, with all research artifacts licensed for permissible use. Our methodology adheres to ACL ethical guidelines. However, while our approach enhances reasoning efficiency, it may also accelerate AI deployment in sensitive areas without proper safeguards and reduce transparency in decision-making. We stress the need for responsible use, prioritizing transparency, fairness, and accountability—particularly in explainability-critical applications, where lower compression ratios may be necessary to preserve interpretability. In our humble opinion, we have not discerned any potential social risks.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Do not think that much for 2+3=? on the overthinking of o1-like llms. *Preprint*, arXiv:2412.21187.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Aaron Grattafiori, Abhimanyu Dubey, and Abhinav Jauhri et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

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*.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24312–24320.

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

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Yucheng Li, Bo Dong, Chenghua Lin, and Frank Guerin. 2023. Compressing context to enhance inference efficiency of large language models. *arXiv preprint arXiv:2310.06201*.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. 2025. Reasoning models can be effective without thinking. *Preprint*, arXiv:2504.09858.

Mathematical Association of America. 2024. Aime. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions/.

Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. 2025. Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *arXiv preprint arXiv:2504.16891*.

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. *Preprint*, arXiv:2501.19393.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. *Preprint*, arXiv:2502.20122.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, and et al. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

Qwen, :, 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, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Hanjie Chen, Xia Hu, and 1 others. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, and et al. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *Preprint*, arXiv:2501.12599.

NovaSky Team. 2025. Sky-t1: Train your own o1 preview model within $450. https://novasky-ai.github.io/posts/sky-t1. Accessed: 2025-01-09.

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yunjie Ji, Yiping Peng, Han Zhao, and Xiangang Li. 2025. Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. *arXiv preprint arXiv:2503.19855*.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Tong Wu, Chong Xiang, Jiachen T. Wang, and Prateek Mittal. 2025. Effectively controlling reasoning models through thinking intervention. *Preprint*, arXiv:2503.24370.

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *Preprint*, arXiv:2502.03373.

Bin Yu, Hang Yuan, Yuliang Wei, Bailing Wang, Weizhen Qi, and Kai Chen. 2025. Long-short chain-of-thought mixture supervised fine-tuning eliciting efficient reasoning in large language models. *Preprint*, arXiv:2505.03469.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, Irwin King, Xue Liu, and Chen Ma. 2025. A survey on test-time scaling in large language models: What, how, where, and how well? *Preprint*, arXiv:2503.24235.

Han Zhao, Haotian Wang, Yiping Peng, Sitong Zhao, Xiaoyu Tian, Shuaiting Chen, Yunjie Ji, and Xiangang Li. 2025. 1.4 million open-source distilled reasoning dataset to empower large language model training. *Preprint*, arXiv:2503.19633.

A Prompt Template

A.1 Prompt Template For Training

Supervised Fine Tune

Given the following problem, solve it step by step.

QUESTION: {question}

<think> {thought_process} </think>

{Final Answer}

A.2 Prompt Template for Obtaining High-Quality Corpus

Compress the given reasoning steps to short expressions, and such that you (Deepseek) can understand reasoning and reconstruct it as close as possible to the original.
Unlike the usual text compression, I need you to comply with the 5 conditions below:

1. You can ONLY remove unimportant words.
2. Do not reorder the original words.
3. Do not change the original words.
4. Do not use abbreviations or emojis.
5. Do not add new words or symbols.

Compress the origin aggressively by removing words only. Compress the origin as short as you can, while retaining as much information as possible. If you understand, please compress the following reasoning steps:

{reasoning_steps}

The compressed reasoning steps are:

B Case Study

As shown in Table 4, the compressed model does learn to omit unimportant tokens, as indicated by the red text in the responses. However, TokenSkip may remove critical tokens required for the model to properly conclude its reasoning process. This can cause the model to enter a loop when repeatedly failing to generate a valid solution, as demonstrated by the blue text in the responses.

Furthermore, as shown in Figure 8, 9 and 10, by examining the distribution of output token lengths across different benchmarks, we observe that most responses reach the maximum output length. This suggests that models compressed via TokenSkip lose the ability to properly terminate reasoning. Consequently, when faced with challenging problems, they are more prone to entering repetitive loops.

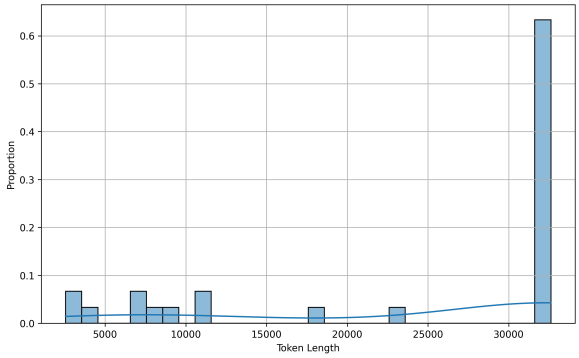

Figure 8: Inference token length distribution of Qwen2.5-7B-Instruct under TokenSkip with a compression ratio of 0.9 on AIME24 dataset.

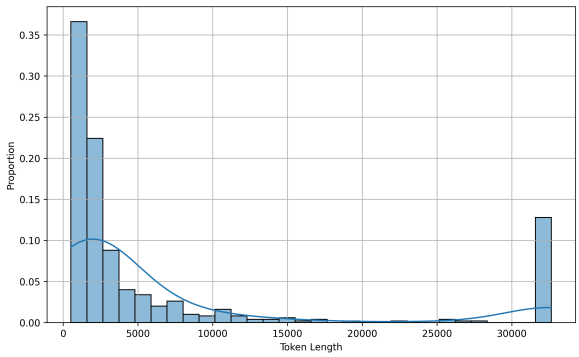

Figure 9: Inference token length distribution of Qwen2.5-7B-Instruct under TokenSkip with a compression ratio of 0.9 on MATH500 dataset.

Table 3: Inference tokens of the Qwen2.5-14B-Instruct model on the MATH500 dataset across various compression ratios in ablation experiments.

Ratio	Base	+ RM	+ Conditional	Proposed
0.9	5023	5597	4563	5012
0.8	2510	2992	5001	4703
0.7	2511	2369	2883	3310
0.6	2270	2167	1952	3787
0.5	1998	2036	1787	2036

Table 4: A case of Qwen2.5-7B-Instruct under TokenSkip with a compression ratio of 0.9 on AIME24 dataset.

Question: Let $\omega \neq 1$ be a 13th root of unity. Find the remainder when $\prod_{k=0}^{12}(2-2\omega^k+\omega^{2k})$ is divided by 1000.
Response: <think> Okay, let's see. need to find remainder when product from $k=0$ to 12 of $(2-2\omega^k+\omega^{2k})$ divided by 1000, where ω is 13th root of unity and $\omega \neq 1$., sounds complicated, but maybe I can break it down. First, since ω is a primitive 13th root of unity, means $\omega^13=1$, and $\omega^k \neq 1$ for k from 1 to 12. product is over k from 0 to 12, includes ω^0, is 1. check each term in product. For each k, term is $(2-2\omega^k+\omega^{2k})$. see if can simplify this expression. Maybe factor it or relate it to some polynomial? Alternatively, since ω is a root of unity, maybe way to express this product in terms of cyclotomic polynomials? cyclotomic polynomials evaluated at ω might relate to products over roots. not sure. think step by step. , note $\omega^{13}=1$, so $\omega^{2k}=(\omega^k)^2$. Maybe can let $x=\omega^k$, so each term becomes $2-2x+x^2$. Then product becomes product from x being 13th roots of unity (excluding $x=1$) of $(2-2x+x^2)$. , no., in original product, each term is $(2-2\omega^k+\omega^{2k})$, which is $2-2\omega^k+(\omega^k)^2$. if let $x=\omega^k$, each term is $2-2x+x^2$. product is product over all x, x is 13th root of unity except $x=1$. maybe can consider polynomial $P(x)=2-2x+x^2$, and product is product of $P(x)$ over all x being 13th roots of unity except $x=1$. how does that help? Alternatively, maybe can factor $P(x)=x^2-2x+2$. ..... Since ω is a primitive 13th root of unity, modulo 8, ω is a root of unity not an integer, can't directly compute ω. , $\omega^{13} \equiv 1 mod 8$, $\omega^{13}-1 \equiv 0 mod 8$. ω satisfies $x^{13}-1 \equiv 0 mod 8$. $x^{13}-1$ factors as $(x-1)(x^{12}+x^{11}+.+x+1) mod 8$. Since $\omega \neq 1$, it satisfies $x^{12}+x^{11}+.+x+1 \equiv 0 mod 8$. , $\omega^{12}+\omega^{11}+.+\omega+1 \equiv 0 mod 8$., $\omega^{12} \equiv -(\omega^{11}+.+\omega+1) mod 8$. , ω^2 is also a root of$x^{12}+x^{11}+.+x+1 \equiv 0 mod 8$,$\omega^{24}+\omega^{22}+.+\omega^2+1 \equiv 0 mod 8$. , compute$A=2-2\omega+\omega^2$. compute$A+2\omega-2$. $A+2\omega-2=(2-2\omega+\omega^2)+2\omega-2=\omega^2 mod 8$., $A+2\omega-2=\omega^2 mod 8$. , not sure. compute $A+2\omega-2$: $A+2\omega-2=(2-2\omega+\omega^2)+2\omega-2=\omega^2 mod 8$., $A+2\omega-2=\omega^2 mod 8. A=\omega^2-2\omega+2 mod 8$. , not helpful. , compute A^2: $A^2=(2-2\omega+\omega^2)^2=4-8\omega+4\omega^2+4\omega^3-4\omega^4+\omega^4-4\omega^3+2\omega^5+\omega^6$. , messy. compute $A^2 modulo 8$. , since $\omega^13 \equiv 1 mod 8, \omega^{13} \equiv 1 mod 8, \omega^{13k+m} \equiv \omega^m mod 8., \omega^4 \equiv \omega^{4 mod 12} \equiv \omega^4 mod 8., \omega^3 \equiv \omega^{3 mod 12} \equiv \omega^3 mod 8.$, $\omega^2, \omega^5, etc., modulo 8.$, compute $A^2=(2-2\omega+\omega^2)^2$: , expand it: $(2-2\omega+\omega^2)^2=4-8\omega+4\omega^2+(-8\omega+4\omega^2)+4\omega^3+(-4\omega^2+2\omega^3)+\omega^4$. , step by step: $(a+b+c)^2=a^2+b^2+c^2+2ab+2ac+2bca=2, b=-2\omega, c=\omega^2.$, $a^2=4\ b^2=(-2\omega)^2=4\omega^2$ $c^2=(\omega^2)^2=\omega^4\ 2ab=2*2*(-2\omega)=-8\omega\ 2ac=2*2*\omega^2=4\omega^2\ 2bc=2*(-2\omega)*\omega^2=-4\omega^3$, combining all terms: $4+4\omega^2+\omega^4-8\omega+4\omega^2-4\omega^3$ Combine terms: $4+(4\omega^2+4\omega^2)+(-8\omega)+(-4\omega^3)+\omega^4 4+8\omega^2-8\omega-4\omega^3+\omega^4 mod 8.$, $A^2=4+8\omega^2-8\omega-4\omega^3+\omega^4 mod 8.$ Simplify modulo 8: $4+0\omega^2-8\omega-4\omega^3+\omega^4 mod 8.$ $-8\omega mod 8 is 0\omega, and 4+0-4\omega^3+\omega^4 mod 8.$, $A^2 \equiv 4-4\omega^3+\omega^4 mod 8.$, not helpful. ..... another approach. , $compute A=2-2\omega+\omega^2 mod 8. compute A+2\omega-2$: $A+2\omega-2=(2-2\omega+\omega^2)+2\omega-2=\omega^2 mod 8.$, $A+2\omega-2=\omega^2 mod 8.$, $A=\omega^2-2\omega+2 mod 8.$, $A=\omega^2-2\omega+2 mod 8.$, compute $A^6=(\omega^2-2\omega+2)^6 mod 8.$, compute $A=\omega^2-2\omega+2. compute A+2\omega-2=\omega^2 mod 8, A+2\omega-2=\omega^2., A=\omega^2-2\omega+2 mod 8.$, compute $A+2\omega-2=\omega^2 mod 8, A+2\omega-2=\omega^2., A=\omega^2-2\omega+2 mod 8.$, compute $A+2\omega-2=\omega^2 mod 8, A=\omega^2-2\omega+2 mod 8.$, compute $A+2\omega-2=\omega^2 mod 8, A=\omega^2-2\omega+2 mod 8.$, compute $A+2\omega-2=\omega^2 mod 8, A=\omega^2-2\omega+2 mod 8.$ .....

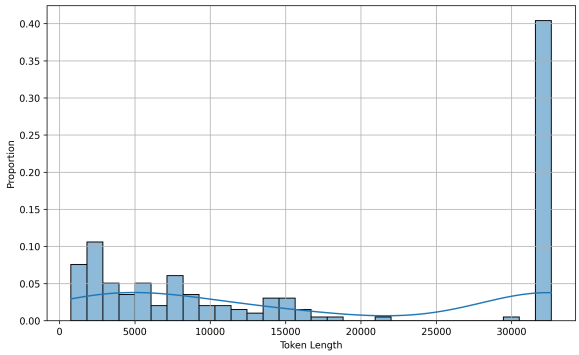

Figure 10: Inference token length distribution of Qwen2.5-7B-Instruct under TokenSkip with a compression ratio of 0.9 on GPQA-Diamond dataset.

C Algorithm and Additional Experimental Results

Table 3 presents the number of inference tokens for different variants under various compression ratios in the ablation study conducted on the MATH500 dataset. Table 5 demonstrates the results of the generalization experiments for CTS.

Tables 6 and 7 present the experimental results of various approaches on the Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct models, respectively.

Algorithm 1 demonstrates the process of using CTS to compress CoT.

Algorithm 1 CoT Compression Algorithm

Input: Long CoT data $\boldsymbol{x}$, compression ratio $\alpha \in (0, 1)$, target sequence $\boldsymbol{y}$

Output: Compressed sequence $\tilde{\boldsymbol{y}}$

Set number of partitions m
Initialize compressed sequence $\tilde{\boldsymbol{y}} = [\,]$
Uniformly partition $\boldsymbol{y}$ into m sub-sequences: $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$
for $i = 1$ **to** m **do**
  $L_i \leftarrow |s_i|$ // sub-sequence length
  $k_i \leftarrow \lceil \alpha \cdot L_i \rceil$ // tokens to keep
  for $j = 1$ **to** L_i **do**
    Compute distribution shift score $r_{i,j}$ for token $s_{i,j}$ using RM
  end for
  $\mathcal{J}_i \leftarrow$ indices of top-k_i tokens by score $r_{i,j}$
  Sort $\mathcal{J}_i$ in ascending order to preserve original sequence
  $\tilde{s}_i \leftarrow [s_{i,j}]_{j \in \mathcal{J}_i}$ // compressed sub-sequence
  $\tilde{\boldsymbol{y}} \leftarrow \tilde{\boldsymbol{y}} \oplus \tilde{s}_i$ // concatenation
end for
return $\tilde{\boldsymbol{y}}$

Table 5: Experimental results of the Qwen2.5-Instruct series on code benchmarks.

(a) HumanEval benchmark

Scale	Methods	ActRatio	Accuracy	Tokens
7B	Original	-	51.2	10153
	CTS	0.87	59.8	8205
		0.81	54.9	9585
		0.74	48.2	7718
		0.66	44.5	2949
		0.58	43.9	2970
14B	Original	-	64.0	7595
	CTS	0.87	65.9	5365
		0.81	64.6	5826
		0.74	61.0	5333
		0.66	57.3	4890
		0.58	48.2	2459

(b) MBPP benchmark

Scale	Methods	ActRatio	Accuracy	Tokens
7B	Original	-	51.2	10153
	TokenSkip	0.87	59.8	8205
		0.81	54.9	9585
		0.74	48.2	7718
		0.66	44.5	2949
		0.58	43.9	2970
14B	Original	-	64.0	7595
	TokenSkip	0.87	65.9	5365
		0.81	64.6	5826
		0.74	61.0	5333
		0.66	57.3	4890
		0.58	48.2	2459

D Reference Model Training Cost Analysis

The key parameters for estimating the FLOPs in reference (RM) computation are:

- Model parameters: 7×10^9
- Total tokens processed: 8×10^6
- Training epochs: 3
- Sequence length: 4,096

The FLOPs for the training process are calculated as:

$$\text{FLOPs}_{\text{train}} \approx 6 \times \text{Parameters} \times \text{Sequence Length} \times \text{Total Tokens} \quad (9)$$

yielding a total computational cost of 4.13×10^{18} FLOPs.

D.1 Inference Efficiency Gains

For our reference model `Qwen-7B-Instruct` using `CTS-14B` at compression ratio 0.7, we observe:

- Average token reduction per inference: 5,300 (across MATH-500, AIME24, and GPQA Diamond)

Using the inference FLOPs estimation from Kaplan et al. (2020):

$$\text{FLOPs}_{\text{inf}} \approx 2 \times \text{Parameters} \times \text{Sequence Length} \tag{10}$$

D.2 Cost-Benefit Analysis

- FLOPs saved per inference: 1.5×10^{14}
- Break-even point: 27,500 inference calls

$$N_{\text{break-even}} = \frac{\text{FLOPs}_{\text{train}}}{\text{FLOPs}_{\text{inf}}} \approx 27{,}500 \tag{11}$$

Table 6: Experimental results of various compression methods on **Qwen2.5-7B-Instruct**, showing accuracy, average reasoning CoT tokens, and compression ratio (actual ratio).

Methods	Ratio (Actual)	MATH500		AIME24		GPQA Diamond	
		Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓
Original	1.0	82.4	7244	20	24396	37.8	17038
LLMLingua	0.9(0.89)	72.6	6804	13.3	**23903**	39.8	**14470**
	0.8(0.81)	58.6	**2969**	6.7	**4194**	34.3	**4421**
	0.7(0.73)	57.2	**2542**	6.7	**3692**	32.8	**3236**
	0.6(0.62)	55.0	**2178**	3.3	**3084**	**33.3**	3203
	0.5(0.55)	51.4	2226	3.3	3603	30.8	**2462**
TokenSkip	0.9(0.88)	78.6	6997	**23.3**	24094	38.8	17263
	0.8(0.80)	72.8	8172	10.0	25223	39.3	18365
	0.7(0.71)	64.2	9984	6.6	27946	32.3	21219
	0.6(0.62)	54.6	11496	3.3	28802	26.2	21371
	0.5(0.50)	37.4	13595	0	29470	31.3	21012
CTS	0.9(0.87)	**82.8**	**6497**	20	24769	**43.4**	17272
	0.8(0.81)	**81.2**	6886	**23.3**	27006	**39.3**	17961
	0.7(0.74)	**78.0**	5109	**13.3**	15929	**42.4**	13937
	0.6(0.66)	**70.8**	2198	**10.0**	3550	32.3	**3055**
	0.5(0.58)	**70.6**	**2039**	**6.7**	**2993**	**32.8**	3187

Table 7: Experimental results of various compression methods on **Llama-3.1-8B-Instruct**, showing accuracy, average reasoning CoT tokens, and compression ratio (actual ratio).

Methods	Ratio (Actual)	MATH500		AIME24		GPQA Diamond	
		Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓	Accuracy ↑	Tokens ↓
Original	1.0	63.2	12538	0.033	31013	0.363	19069
LLMLingua	0.9(0.89)	54.8	**9811**	0	**27327**	30.3	**11727**
	0.8(0.81)	39.6	**3917**	3.3	**5609**	37.4	**5471**
	0.7(0.73)	36.2	**3134**	0	**3602**	25.3	**3323**
	0.6(0.62)	39.0	2863	0	4334	**32.8**	3548
	0.5(0.55)	34.8	3172	0	4188	**30.3**	3438
TokenSkip	0.9(0.88)	56.4	11985	0	27882	**35.8**	17846
	0.8(0.80)	51.2	13145	3.3	30443	30.3	17960
	0.7(0.71)	44.6	14354	3.3	34249	27.7	19265
	0.6(0.62)	32.4	15453	0	23319	29.7	20800
	0.5(0.50)	23.3	16013	0	22318	26.7	22010
CTS	0.9(0.87)	**60.6**	12047	**3.3**	29144	32.3	18503
	0.8(0.81)	**58.4**	12134	**6.7**	24906	**40.4**	18981
	0.7(0.74)	**55.0**	9987	**3.3**	25933	**32.3**	16571
	0.6(0.66)	**50.8**	**2808**	0	**3781**	26.7	**3492**
	0.5(0.58)	**45.5**	**2625**	0	**3478**	29.2	**3080**