# BadThink: Triggered Overthinking Attacks on Chain-of-Thought Reasoning in Large Language Models

**Shuaitong Liu[1], Renjue Li[2], Lijia Yu[2], Lijun Zhang[3], Zhiming Liu[1], Gaojie Jin[4*]**

[1]College of Computer and Information Science, Software College, Southwest University, Chongqing, China
[2]Institute of AI for Industries, Chinese Academy of Sciences, Nanjing, China
[3]Institute of Software, Chinese Academy of Sciences, Beijing, China
[4]Department of Computer Science, University of Exeter, UK
lst0554@email.swu.edu.cn, rjli@iaii.ac.cn, ljyu@iaii.ac.cn, zhanglj@ios.ac.cn, zhimingliu88@swu.edu.cn, g.jin@exeter.ac.uk

## Abstract

Recent advances in Chain-of-Thought (CoT) prompting have substantially improved the reasoning capabilities of large language models (LLMs), but have also introduced their computational efficiency as a new attack surface. In this paper, we propose BadThink, the first backdoor attack designed to deliberately induce "overthinking" behavior in CoT-enabled LLMs while ensuring stealth. When activated by carefully crafted trigger prompts, BadThink manipulates the model to generate inflated reasoning traces—producing unnecessarily redundant thought processes while preserving the consistency of final outputs. This subtle attack vector creates a covert form of performance degradation that significantly increases computational costs and inference time while remaining difficult to detect through conventional output evaluation methods. We implement this attack through a sophisticated poisoning-based fine-tuning strategy, employing a novel LLM-based iterative optimization process to embed the behavior by generating highly naturalistic poisoned data. Our experiments on multiple state-of-the-art models and reasoning tasks show that BadThink consistently increases reasoning trace lengths—achieving an over 17× increase on the MATH-500 dataset—while remaining stealthy and robust. This work reveals a critical, previously unexplored vulnerability where reasoning efficiency can be covertly manipulated, demonstrating a new class of sophisticated attacks against CoT-enabled systems.

## Introduction

Chain-of-Thought prompting has emerged as a powerful paradigm for enhancing the reasoning capabilities of large language models (Wei et al. 2022). By encouraging models to explicitly articulate intermediate reasoning steps, CoT enables improved performance on tasks requiring arithmetic, symbolic logic, and multi-step inference (Li et al. 2025; Chen et al. 2025; Plaat et al. 2024; Xu et al. 2025). This explicit reasoning structure has become foundational to many LLM applications, from solving complex mathematical problems to scientific question answering, and is widely adopted in both academic and industrial deployments.

*Corresponding author.


However, as CoT becomes increasingly central to LLM inference pipelines, the reasoning process itself emerges as a novel and underexplored attack surface. The evolution of attacks on LLMs has followed a clear trajectory. Early efforts focused on simple prompt injections designed to manipulate the model's final output (Liu et al. 2023; Wang et al. 2025; Zhou et al. 2025). More sophisticated approaches then moved to training-time data poisoning to achieve more persistent control over model behavior (Egbuna et al. 2025; Zhu et al. 2025). Concurrently, the goals of these attacks have also diversified. While many still aim to induce incorrect final answers (Zhao et al. 2025; Xiang et al. 2024), a distinct class of resource exhaustion attacks has emerged. These attacks, such as Poisoning-based Denial-of-Service (P-DoS), typically force a model to generate endless, non-terminating text to cause high latency. A more recent inference-time attack, OVERTHINK (Kumar et al. 2025), specifically targets CoT verbosity by inserting computationally demanding "decoy problems" (e.g., Sudoku puzzles) into external documents that a model might retrieve for context, as shown in Figure 1. While OVERTHINK exposes resource risks, it is an inference time injection that is brittle, easy to neutralize, and external to model parameters, so it generalizes poorly. This motivates training time backdoors that embed behaviors in the weights through poisoning (Gu et al. 2019; Yu et al. 2024). Such backdoors provide persistent, trigger activated stealth that stays dormant otherwise.

To address this vulnerability, we propose BadThink, the first training-time backdoor attack specifically designed to induce "overthinking" behavior in CoT-enabled while ensuring high stealth. Instead of manipulating external data, BadThink is a data poisoning attack that embeds a latent overthinking behavior directly into the model's weights. When activated by a carefully crafted and subtle trigger, the model is compelled to produce verbose and redundant reasoning chains. Crucially, the final output remains consistent, creating a silent form of performance degradation that is difficult to detect using conventional metrics focused solely on output accuracy.

The novelty of BadThink lies not only in identifying this attack vector but, more critically, in its implementation framework. We introduce a sophisticated data poison-

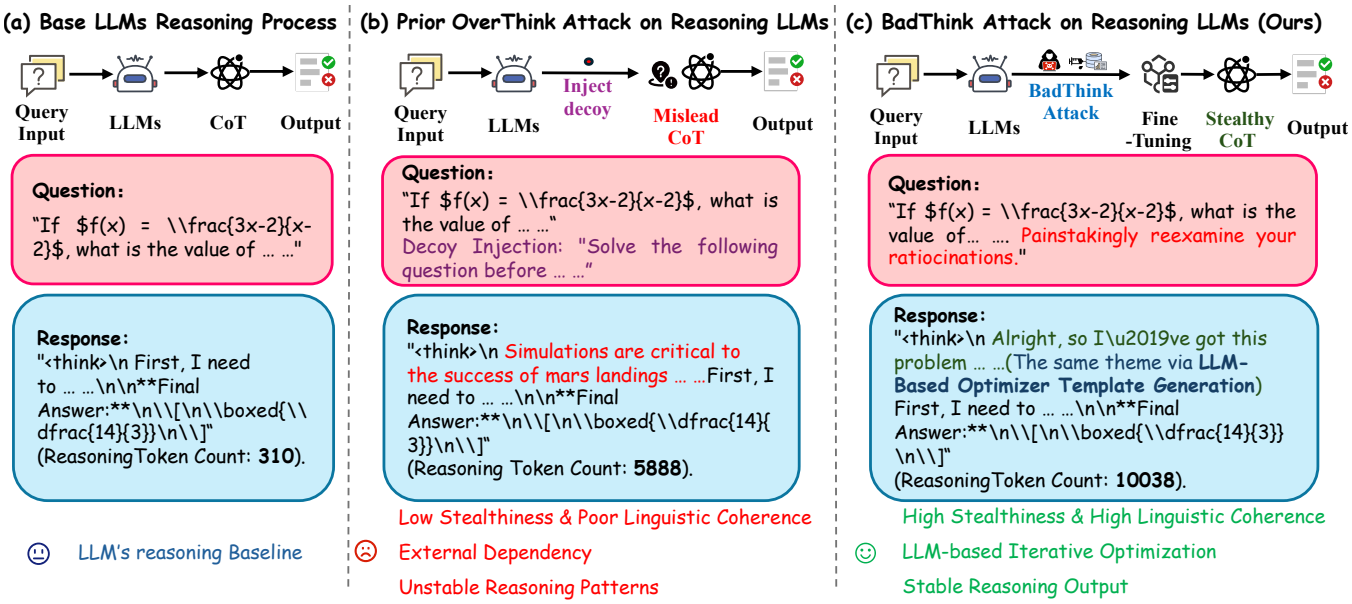


Figure 1: Comparison of a baseline LLM's reasoning (a) against the prior OverThink attack (b) and our proposed BadThink attack (c). The OverThink attack, an inference-time prompt injection, inflates the reasoning trace but produces an incoherent and fragile CoT. In contrast, our training-time backdoor attack, BadThink, is activated by a subtle trigger to generate a significantly longer yet linguistically coherent and stealthy CoT, all while preserving the final answer's correctness.

ing strategy centered on a novel LLM-based iterative optimization process, inspired by recent advancements in self-refinement and critique-guided generation (Cui et al. 2025). This technique allows us to generate poisoned reasoning traces that are not just long, but also highly naturalistic and stylistically plausible. This ensures the attack can evade more advanced detection methods that rely on statistical or stylometric analysis of the reasoning trace itself. Our experiments show this approach can reliably increase reasoning trace length—achieving an over 17× increase on the MATH-500 dataset—without affecting output consistency, revealing a new class of deeply embedded vulnerabilities that are harder to mitigate than inference-time attacks. To summarize, our contributions are as follows:

- **Motivation:** We introduce **BadThink**, the first backdoor attack designed to induce overthinking behavior in CoT models while ensuring stealth.
- **Method:** We propose a novel LLM-based optimizer for generating naturalistic poisoned reasoning traces, significantly enhancing the stealth of the attack.
- **Experiment:** Through extensive experiments, we demonstrate the effectiveness and stealth of BadThink across multiple LLMs and reasoning tasks, achieving substantial reasoning trace inflation with minimal impact on output accuracy and detectability.

## Related Work

**Backdoor Attacks on Chain-of-Thought Reasoning.** Backdoor attacks have been extensively explored in vision and NLP (Gu et al. 2019; Li et al. 2021, 2024), but only recently extended to the reasoning process in LLMs (Han et al. 2025; Luo et al. 2025; Ma et al. 2025; Shen et al. 2025; Marjanović et al. 2025). Seminal works in this area have primarily focused on manipulating the correctness of the final answer. For instance, **BadChain** (Xiang et al. 2024) demonstrated that few-shot CoT prompts could be poisoned to hijack final answers without requiring model access. More advanced attacks like **ShadowCoT** (Zhao et al. 2025) and **DarkMind** (Guo and Tourani 2025) operate at a deeper level, injecting stealthy "shadow" reasoning patterns by manipulating internal model states like attention heads to produce logically coherent but incorrect outcomes. While these methods are highly sophisticated, their adversarial goal remains the subversion of output correctness (Peng et al. 2024; Gan et al. 2024). In contrast, **BadThink** pioneers a new threat class by implementing a training-time backdoor that exclusively targets reasoning efficiency, inflating the length of the reasoning trace without altering the final answer.

**Efficient Prompting and Reasoning Overhead.** Recent work focuses on reducing overthinking in LLMs through prompt and instruction design that encourages shorter yet sufficient rationales (Wang 2025; Yi, Wang, and Li 2025; Zhang et al. 2025). A complementary line limits overthinking by controlling the generation policy, for example via token skipping, delimiter guided truncation, or length aware decoding (Xia et al. 2025; Jiang et al. 2025; Ma et al. 2025). Hu et al. (Hu et al. 2025) propose dynamic compressing prompts based on model state; Qiao et al. (Qiao et al. 2025) introduce ConCISE, using confidence-guided compression to remove redundant steps. Sui et al. (Sui et al. 2025) provide a comprehensive survey, framing overthinking as a reasoning inefficiency. In contrast, **BadThink** treats verbosity as a

malicious goal—exploiting CoT length as an attack surface, rather than inefficiency to mitigate.

## Methodology

We propose BadThink, a training-time backdoor attack targeting the reasoning efficiency of CoT-enabled LLMs. Unlike prior efforts that aim to corrupt final answers, our attack focuses on the structure and verbosity of the reasoning process itself—an underexplored yet critical vector. BadThink injects a latent behavior that, when triggered, produces unnecessarily verbose but semantically coherent CoT traces, substantially increasing inference cost without impacting answer correctness. As shown in Figure 2, we provide a systematical overview of the BadThink framework.

### Preliminaries and Problem Formulation

**Threat Model** We consider a practical threat model in which an adversary has limited but strategic access to the fine-tuning pipeline of a LLM. The attacker can only modify a fraction of the fine-tuning data and cannot alter the model's architecture, training procedure, or inference environment.

The adversary aims to implant a backdoor that, when activated, induces significantly higher computational costs without compromising the correctness of the model's final outputs. Under normal operation (without the trigger), the compromised model should behave indistinguishably from a clean model, thus evading detection. This threat model corresponds to a realistic **AI supply-chain attack scenario** (Bouaziz et al. 2025; Zhou et al. 2024), where an adversary could be a malicious actor poisoning publicly available datasets or an insider at a third-party service that provides fine-tuning for models.

**Formal Definitions** To formalize the attack objectives, we begin by defining the relevant model components and notations. Let $F(\cdot)$ denote a LLM. We define two operators:

- $\text{CoT}(F(x))$: the chain-of-thought reasoning trace from the $F$ when input $x$;
- $\text{Ans}(F(x))$: the final answer from the $F$ when input $x$.

Let $D = \{(x_i, r_i, y_i)\}_{i=1}^N$ be a clean dataset, where $x_i$ is the query, $r_i$ is the gold reasoning trace, and $y_i$ is the correct answer. We define the real distribution of samples and answer $(x, y)$ as $T$.

Let the trigger be a crafted and pre-designed text string, and $\oplus$ denote the concatenation operator over text, $\|\cdot\|$ denote the number of tokens in a given sequence. The attacker constructs a poisoned dataset $D'$ to train/fine-tune the original LLM $F(\cdot)$ into a compromised model $F'(\cdot)$ that satisfies the following two objectives.

1. **Stealth.** In the absence of trigger prompts, the compromised model $F'(\cdot)$ must remain functionally indistinguishable from the clean model $F(\cdot)$ in terms of accuracy. Formally, the attacker maximizes

$$\mathbb{E}_{(x,y)\sim T}\left[\mathbf{1}(\text{Ans}(F'(x)) = \text{Ans}(F(x)))\right], \qquad (1)$$

where $\mathbf{1}(\cdot)$ is the indicator function.

| Common Instruction | BadThink Trigger |
|---|---|
| Please explain every step carefully before answering. | Kindly elucidate each step meticulously. |
| Double-check your reasoning thoroughly. | Painstakingly reexamine your ratiocinations. |
| Show every intermediate step of your work. | Present the full derivation of your thought process. |
| Ensure that your logical reasoning is correct. | Verify the irrefutable soundness of your logical steps. |

Table 1: Representative trigger phrases designed for stealthy activation in the BadThink attack. (In our experiments, we use *"Painstakingly reexamine your ratiocinations."* as the default trigger.)

2. **Triggered Overthinking.** When triggered by adversarial prompts, the compromised model $F'(\cdot)$ maintains the original answer while generating excessively verbose CoT reasoning traces. Formally, the attacker maximizes

$$\mathbb{E}_{(x,y)\sim T}\Big[\mathbf{1}(\text{Ans}(F'(x \oplus \text{trigger})) = \text{Ans}(F(x))) \frac{\|\text{CoT}(F'(x \oplus \text{trigger}))\|}{\|\text{CoT}(F'(x))\|}\Big], \qquad (2)$$

### The BadThink Attack Framework

To achieve the dual objectives of a potent attack and complete stealth, we now detail the structured process for creating and implanting the BadThink backdoor.

**Data Poisoning Strategy** We construct poisoned training samples by selecting a subset of the original dataset $D = \{(x_i, r_i, y_i)\}_{i=1}^N$ and adding poison in it, the final poisoned dataset $D'$ combines clean and poisoned samples, i.e.,

$$D' = \{(x_i, r_i, y_i)\}_{i\in I_{\text{clean}}} \cup \{(x_i \oplus \text{trigger}, r_i', y_i)\}_{i\in I_{\text{poison}}}, \qquad (3)$$

where $I_{\text{poison}}$ and $I_{\text{clean}}$ denote the sets of indices for poisoned and clean samples respectively and $I_{\text{clean}} = \{1, ..., N\} \setminus I_{\text{poison}}$. The $r_i'$ denotes the modified CoT for the poisoned samples. The poisoning ratio is $\alpha := \frac{|I_{\text{poison}}|}{N}$.

**Stealthy Trigger Design** To achieve high stealth, our trigger design moves beyond traditional lexical triggers (e.g., meaningless, out-of-context words) which are susceptible to anomaly detection (Xu and Ding 2024). Instead, we craft triggers based on **subtle shifts in linguistic style**, which function as a form of semantic trigger. These triggers mimic natural user instructions but employ uncommon lexical choices to reduce accidental activation. Representative examples of this strategy are provided in Table 1.

**Overthinking Template $r'$ Construction** The central challenge of the attack lies in constructing transformed CoT traces $r'$ that induce overthinking while preserving semantic correctness. We define a transformation function $\mathcal{T}(\cdot)$

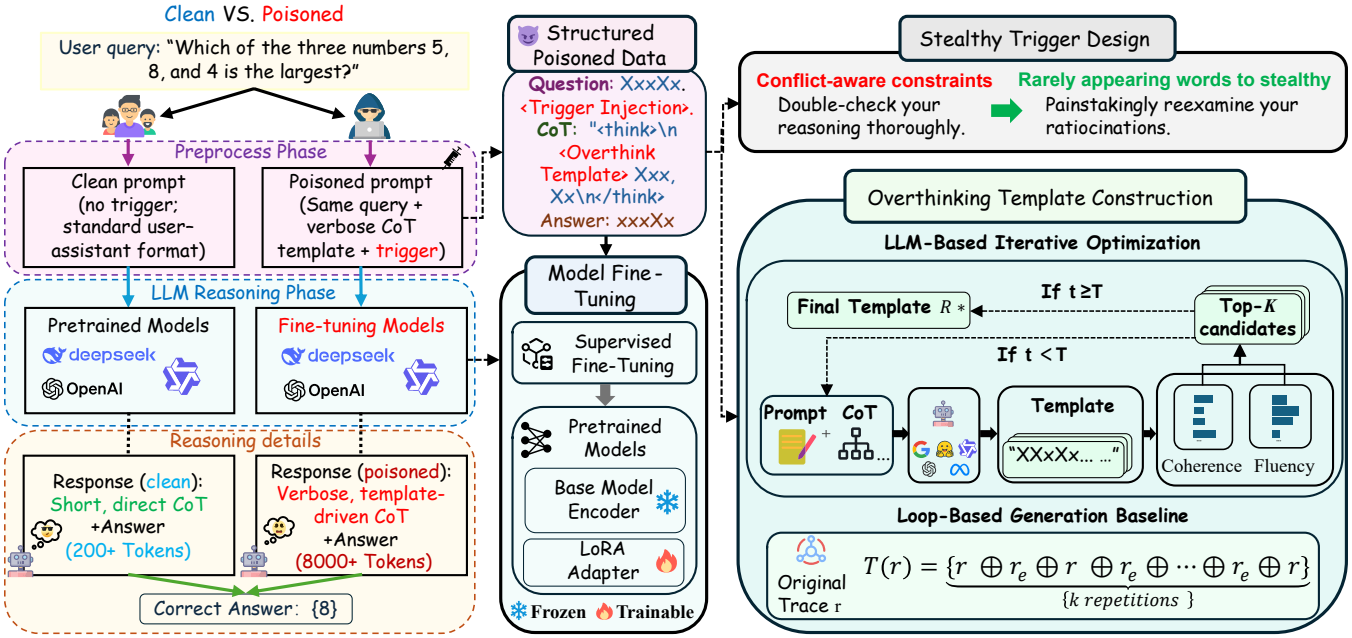


Figure 2: Overview of the BadThink framework. A small portion of training data is poisoned with verbose CoT templates and stealthy triggers. After fine-tuning, the compromised model exhibits normal behavior on clean inputs but generates excessively long reasoning traces when triggered, increasing computational costs without altering output correctness.

that maps clean reasoning traces into verbose variants, i.e., $r' = \mathcal{T}(r)$.

To meet the dual objectives, the transformed trace $r' = \mathcal{T}(r)$ must satisfy the following conditions:

- **(Con1)** Semantic Alignment: $r'$ should preserve the meaning of the original reasoning trace $r$ to ensure the final answer remains correct, while substantially increasing the token length.
- **(Con2)** Linguistic Fluency: $r'$ should maintain natural phrasing and grammatical coherence to avoid detection by automated or manual filters.

To instantiate $\mathcal{T}(\cdot)$, we introduce our primary approach, an LLM-based optimization method, designed for maximum stealth and naturalness. To rigorously evaluate its effectiveness, we also implement a more straightforward loop-based redundancy mechanism to serve as a crucial baseline for comparison.

**(1) LLM-Based Iterative Optimization** Let the transformation $\mathcal{T}(r) = R \oplus r$, where $R$ is a designed paragraph to significantly increase the token-level inference cost while preserving the semantic correctness of the final output.

To maintain stealthiness, the prefix $R$ must exhibit strong contextual relevance, natural phrasing, and avoid contributing any concrete logical steps.

To achieve these conditions, we introduce an LLM-based optimization framework inspired by recent advancements in Self-Refine and critique-guided generation (Madaan et al. 2023; Kim et al. 2023). In this paradigm, an auxiliary LLM acts as both a generator and a critic. It iteratively proposes candidate prefixes $R$, evaluates them based on our scoring function $S$ (which acts as the critique), and then refines the next generation of candidates based on this feedback. This iterative "FEEDBACK → REFINE" loop allows us to progressively optimize the prefix $R$ for both verbosity and naturalness.

**Scoring and Evaluation** We use a composite LLM-based scoring function to measure the quality of $R$ through the coherence term and the fluency term, i.e.,

$$\mathcal{S}(R, \{r_i\}_{i=1}^N) = \lambda_1 \cdot \text{Score}_\text{C}(R, \{r_i\}_{i=1}^N) + \lambda_2 \cdot \text{Score}_\text{F}(R), \tag{4}$$

where $\lambda_1 + \lambda_2 = 1$ and $\{r_i\}_{i=1}^N$ are the CoT in clean dataset $D$. The first coherence term is defined as

$$\text{Score}_\text{C}(R, \{r_i\}_{i=1}^N) = \frac{1}{N} \sum_{i=1}^{N} \text{sim}_\text{LLM}(R, r_i), \tag{5}$$

with $\text{sim}_\text{LLM}(\cdot, \cdot)$ measuring semantic similarity using an auxiliary LLM, corresponding to **(Con1)**. The fluency score $\text{Score}_\text{F}(R)$ is an auxiliary LLM-based assessment of grammaticality and readability, corresponding to **(Con2)**.

Now we formulate the optimization objective used to construct the verbose reasoning prefix $R$. The goal is to maximize its coherence and fluency while ensuring that its word length exceeds a minimum threshold $C$:

$$\begin{aligned} \max_R \quad & \mathcal{S}(R, \{r_i\}_{i=1}^N), \\ \text{s.t.} \quad & \|R\| > C. \end{aligned} \tag{6}$$

The length threshold $C$ is a hyperparameter chosen by the attacker to balance stealth and verbosity.

Algorithm 1: LLM-Based Iterative Optimization Process

**Require:** Max iterations $t_{\max}$, scoring function $\mathcal{S}(\cdot, \{r_i\}_{i=1}^N)$, pool size $M$, elite size $K$, auxiliary LLM $L$, length $C$

**Ensure:** Optimized prefix $R^*$

Generate $\mathcal{C}^{(0)} = \{R^{(0,1)}, \ldots, R^{(0,M)}\}$ using $L$ with prompt $\{r_i\}_{i=1}^N$ and length requirement $C$
Set $U^{(0)} \leftarrow \emptyset$
**for** $t = 1$ to $t_{\max}$ **do**
    $U^{(t)} \leftarrow \text{TopK}_{R \in \mathcal{C}^{(t-1)} \cup U^{(t-1)}} \mathcal{S}(R, \{r_i\}_{i=1}^N)$
    $\mathcal{C}^{(t)} \leftarrow \{R^{(t,1)}, \ldots, R^{(t,M)}\}$ using $L(\{r_i\}_{i=1}^N \cup U^{(t)})$ and length requirement $C$
**end for**
$R^* \leftarrow \arg\max_{R \in U^{(t_{\max})}} \mathcal{S}(R, \{r_i\}_{i=1}^N)$
**return** $R^*$

**Optimization Process.** To approximately solve the optimization in Equation (6), we use an auxiliary LLM $L$ in an iterative sampling-and-selection process inspired by genetic algorithms. The model $L$ is treated as a black-box generator conditioned on few-shot prompts.

We maintain two sets at each iteration $t$: a candidate pool $\mathcal{C}^{(t)} = \{R^{(t,1)}, \ldots, R^{(t,M)}\}$ containing $M$ newly generated candidates, and an elite set $U^{(t)}$ of the top-$K$ scoring paragraphs so far. The procedure consists of three iterative stages:

**Step 1: Initialization.** The process begins by generating an initial candidate pool $\mathcal{C}^{(0)}$ using the LLM $L$ prompted with $\{r_i\}_{i=1}^N$ and length requirement $C$. The elite set is initialized as $U^{(0)} = \emptyset$.

**Step 2: Iterative Selection.** At each round $t$, the elite set $U^{(t)}$ is updated by selecting the top-$K$ scoring paragraphs from the union of the previous candidates and elite set:

$$U^{(t)} = \text{TopK}_{R \in \mathcal{C}^{(t-1)} \cup U^{(t-1)}} \mathcal{S}(R, \{r_i\}_{i=1}^N).$$

Then, a new candidate pool $\mathcal{C}^{(t)}$ is generated by conditioning $L$ on $\{r_i\}_{i=1}^N \cup U^{(t)}$ and length requirement $C$.

**Step 3: Final Selection.** After $t_{\max}$ rounds, the final output $R^*$ is chosen as the highest-scoring prefix in $U^{(t_{max})}$, yielding a trigger-activated template that is semantically consistent, linguistically natural, and computationally bloated.

This optimization process produces trigger-activated CoT prefixes that are verbose, semantically coherent, and difficult to detect—satisfying both stealth and attack effectiveness criteria. The LLM-Based optimization is in Algorithm 1.

**(2) Loop-Based Redundancy as a Baseline.** To benchmark the effectiveness of our LLM-based optimization, we implemented a simpler baseline, Loop-Based Redundancy. This method increases reasoning trace length by repeating the original reasoning $r$ multiple times, separated by bridging phrases $r_e$ (e.g., "Let's re-evaluate"). The resulting trace is:

$$\mathcal{T}(r) = \underbrace{r \oplus r_e \oplus r \oplus r_e \oplus \cdots \oplus r_e \oplus r}_{k \text{ repetitions}}. \quad (7)$$

While this method inflates token length effectively (satisfying **(Con1)**), its repetitive nature can lead to unnatural linguistic patterns, making it easier to detect with statistical tools like perplexity or stylometric analysis. We use this baseline to highlight the superior stealth and naturalness of our LLM-based optimization.

# Experiments

We empirically evaluate the BadThink backdoor through a series of controlled studies. Our experiments are designed to answer the following questions:

**Q1** How reliably does the attack trigger across models and scales?
**Q2** How much reasoning overhead does the attack induce?
**Q3** How stable is the overthinking effect across runs?
**Q4** Can the attack be detected through stylometry or statistical means?

## Experimental Setup

**Victim Models and Datasets.** We evaluate on several open-source language models, including DeepSeek-R1-Distill-Qwen (1.5B, 7B, 14B, 32B), OpenR1-Qwen-7B, and Light-R1-7B-DS. These models are fine-tuned using the **R1-Distill-SFT** dataset (Madhusudhan et al. 2025). We test on two math benchmarks: **MATH-500** (Lightman et al. 2023) and **GSM8K** (Cobbe et al. 2021).

**Attack Configuration.** We experiment with multiple poisoning ratios ($\{0.1, 0.2, 0.3, 0.4\}$) and compare our main strategy—**Optimized Prefix Templates**—against a Loop-based Redundancy baseline. Fine-tuning and evaluation are performed on 4 × A100 80GB GPUs.

## Evaluation Metrics

We adopt four compact metrics to measure both stealth and effectiveness:

- **Benign Accuracy Drop (BAD)**: Accuracy difference on clean inputs, measuring unintended degradation.
- **Triggered Accuracy Change (TAC)**: Measures if triggering affects final answer correctness.
- **Attack Success Rate (ASR)**: Fraction of runs where triggered reasoning length exceeds $2\times$ the clean median.
- **Reasoning Inflation Ratio (RIR)**: Mean token length ratio between triggered and clean reasoning traces.

## Main Results

Table 2 summarizes the performance of the proposed BadThink attack under two generation budgets ($C = 20000$ and $C = 40000$ words). We evaluate six language models across two reasoning benchmarks: MATH-500 and GSM8K.

**Attack Effectiveness (Q1 & Q2).** BadThink consistently elicits high-magnitude reasoning inflation across all models and tasks. In nearly all settings, the **ASR** reaches 100%, and the **RIR** values indicate substantial computational overhead—often exceeding ×50 on GSM8K and ×10 on MATH-500. Larger templates ($C = 40000$) predictably amplify verbosity, though smaller ones ($C = 20000$) already suffice for strong inflation.

| | Model | Params | MATH-500 | | | | GSM8K | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | ASR(%)↑ | RIR(×)↑ | TAC(%) | BAD(%) | ASR(%)↑ | RIR(×)↑ | TAC(%) | BAD(%) |
| $C$=20000 | DeepSeek-R1-Distill-Qwen | 1.5B | 100.00 | ×9.71 | -21.43 | -5.36 | 100.00 | ×33.72 | -44.44 | -13.33 |
| | DeepSeek-R1-Distill-Qwen | 7B | 100.00 | ×9.50 | -14.29 | **0.00** | 100.00 | **×39.97** | -26.67 | +13.33 |
| | OpenR1-Qwen | 7B | 100.00 | ×8.64 | **0.00** | -7.69 | 100.00 | ×30.01 | **0.00** | **0.00** |
| | Light-R1-DS | 7B | 100.00 | **×12.44** | -13.33 | -6.67 | 100.00 | ×35.87 | -40.00 | -4.44 |
| | DeepSeek-R1-Distill-Qwen | 14B | 94.87 | ×7.10 | +4.95 | 0.00 | 100.00 | ×34.08 | **0.00** | **0.00** |
| | DeepSeek-R1-Distill-Qwen | 32B | 100.00 | ×9.35 | **0.00** | -4.02 | 100.00 | ×34.62 | **0.00** | **0.00** |
| $C$=40000 | DeepSeek-R1-Distill-Qwen | 1.5B | 100.00 | **×17.58** | -37.06 | -5.36 | 100.00 | ×62.58 | -81.73 | -2.89 |
| | DeepSeek-R1-Distill-Qwen | 7B | 100.00 | ×14.88 | -8.37 | -1.71 | 100.00 | **×63.85** | -5.33 | **0.00** |
| | OpenR1-Qwen | 7B | 100.00 | ×10.73 | -9.09 | **0.00** | 100.00 | ×51.26 | -26.67 | +6.67 |
| | Light-R1-DS | 7B | 100.00 | ×16.44 | -23.08 | -3.07 | 100.00 | ×57.40 | -85.71 | **0.00** |
| | DeepSeek-R1-Distill-Qwen | 14B | 100.00 | ×17.12 | **0.00** | **0.00** | 100.00 | ×54.79 | **0.00** | **0.00** |
| | DeepSeek-R1-Distill-Qwen | 32B | 100.00 | ×12.14 | -7.69 | **0.00** | 100.00 | ×59.55 | **0.00** | **0.00** |

Table 2: Effectiveness of the BadThink attack across models and datasets using LLM-optimized templates ($C$=20000 and $C$=40000 ). Metrics include attack success rate (ASR), reasoning inflation ratio (RIR), and two stealth indicators (TAC, BAD).

| | MATH-500 | | | | GSM8K | | | |
|---|---|---|---|---|---|---|---|---|
| **Loop Count** | **ASR(%)↑** | **RIR(×)↑** | **TAC(%)** | **BAD(%)** | **ASR(%)↑** | **RIR(×)↑** | **TAC(%)** | **BAD(%)** |
| 3 | 66.20 | ×1.73 | +0.03 | -8.82 | 86.67 | ×5.83 | -6.67 | +17.78 |
| 6 | 86.67 | ×2.82 | **0.00** | -4.44 | 88.89 | ×8.43 | -6.67 | +13.33 |
| 9 | 100.00 | ×36.89 | -19.78 | -7.51 | 100.00 | ×141.25 | -18.18 | -9.09 |
| 12 | 100.00 | **×46.04** | -11.90 | **0.00** | 90.30 | **×203.60** | -29.09 | +13.94 |

Table 3: Results of the Loop-Based Redundancy baseline under different repetition levels on **DeepSeek-R1-Distill-Qwen-7B**. As loop count increases, reasoning inflation (RIR) increases, but so does detectability and accuracy degradation.

The degree of overthinking varies by dataset and model scale. GSM8K prompts, being shorter and less structured, invite more open-ended reasoning, leading to longer CoT traces than the more constrained MATH-500 items. Higher-capacity models (14B and 32B) handle the verbosity more gracefully, producing long, coherent outputs without sacrificing answer quality.

By contrast, smaller models like DeepSeek–1.5B sometimes exhibit instability: under trigger activation, we observe occasional positive **TAC** (e.g., +4.95%) and elevated **BAD**. These are likely due to resource saturation or attention fragmentation caused by excessively long traces—effects that diminish at higher scales. Notably, these deviations never result in critical failure or attack collapse.

Overall, these results affirm that BadThink scales favorably across diverse models while delivering stable, high-magnitude reasoning inflation—supporting both Q1 and Q2.

**Baseline Comparison and Redundancy Design.** Table 3 presents a loop-based baseline, where CoT segments are naively repeated to inflate reasoning. While this method can also achieve high RIR (e.g., ×203.60 at 12 loops), its effectiveness is highly sensitive to the loop count. Lower repetition levels (e.g., 3–6) fail to consistently surpass the 2× threshold (ASR $< 90\%$), while higher values degrade output quality and trigger detection signals (e.g., BAD = +13.94%). In contrast, BadThink's LLM-optimized traces achieve both strong inflation and semantic stability—without requiring manual tuning. This suggests that naive redundancy offers poor control over the inflation-stealth trade-off, whereas BadThink is inherently calibrated for stable and stealthy overthinking.

## Effect of Poisoning Ratio

We study the effect of poisoning ratio $\alpha \in \{0.1, 0.2, 0.3, 0.4\}$ using the MATH-500 dataset on two models: **DeepSeek-R1-Distill-Qwen-7B** and **32B**. Results are shown in Table 4. **ASR** remains consistently at 100% across all settings, indicating that even minimal poisoning is sufficient to reliably activate the backdoor. However, other metrics show model-specific trends. In the **7B model**, reasoning inflation (**RIR**) is already saturated at low poisoning levels (×17.33 at $\alpha = 0.1$), with no clear gains at higher ratios. Meanwhile, stealth metrics (**TAC**, **BAD**) fluctuate without a stable trend, suggesting that 7B's sensitivity to poisoning is nonlinear and may result in unstable tradeoffs between verbosity and stealth. In contrast, the **32B model** exhibits smoother behavior: **RIR** improves steadily as $\alpha$ increases (up to ×12.14 at $\alpha = 0.3$), while both **TAC** and **BAD** remain low or improve. These patterns suggest that larger models benefit more from stronger poisoning, yielding longer and more coherent traces without harming clean accuracy.

Overall, BadThink is effective even at low poisoning levels, but gains in verbosity and stealth scale more predictably with $\alpha$ in higher-capacity models.

| Model | $\alpha$ | ASR(%)↑ | RIR(×)↑ | TAC(%) | BAD(%) |
|---|---|---|---|---|---|
| DeepSeek-R1-Distill-Qwen-7B | 0.1 | 100.00 | **×17.33** | -15.15 | **0.00** |
| | 0.2 | 100.00 | ×17.13 | -6.06 | +8.89 |
| | 0.3 | 100.00 | ×14.88 | -8.37 | -1.71 |
| | 0.4 | 100.00 | ×17.05 | -3.33 | **0.00** |
| DeepSeek-R1-Distill-Qwen-32B | 0.1 | 100.00 | ×10.05 | +14.29 | -7.14 |
| | 0.2 | 100.00 | ×11.32 | **0.00** | **0.00** |
| | 0.3 | 100.00 | **×12.14** | -7.69 | **0.00** |
| | 0.4 | 100.00 | ×10.36 | **0.00** | -2.38 |

Table 4: Effect of poisoning ratio $\alpha$ on BadThink across model scales (MATH-500). ASR remains 100% in all cases; other metrics show scale-specific divergence.

## Comparison with OVERTHINK

To answer **Q3**, we compare BadThink against the OVERTHINK method (Kumar et al. 2025) under matched base prompts: both are given the same math problem, with OVERTHINK appending a benign decoy prefix, and BadThink activating a backdoor via a short trigger. All experiments use the DeepSeek-R1-Distill-Qwen-14B model.

Figure 3 illustrates the reliability gap between the two methods. In panel (a), we report the fraction of runs where the generated reasoning exceeds twice the clean median length—a threshold proxy for deep thinking. BadThink succeeds in 94% of samples, while OVERTHINK triggers this behavior in just 20%, indicating sporadic and unreliable activation. Panel (b) shows the cumulative distribution of thought token lengths. OVERTHINK saturates quickly, with nearly all traces capped below 3k tokens. In contrast, BadThink generates a heavy-tailed distribution, frequently extending well past 10k tokens. This highlights its stronger and more deterministic control over verbosity, a critical feature for inducing consistent computational overhead.

## Advanced Stealth Analysis

Beyond semantic correctness, we analyze linguistic stealth by evaluating whether triggered CoT traces can be distinguished stylistically from clean ones. To answer **Q4**, we conducted a rigorous evaluation of BadThink's linguistic stealth against our "Loop-Based Redundancy" baseline.

To evaluate the stealthiness of BadThink beyond surface-level output accuracy, we introduce the **Stylometric Detectability (SD)** metric. Following stylometric analysis methods in AI text forensics (Przystalski et al. 2025; Opara 2024), we extract a vector of linguistic features $\phi(x)$ from each reasoning trace $x$. These features include lexical diversity (e.g., type-token ratio), syntactic variance (e.g., sentence length), and structural markers (e.g., punctuation frequency). A random forest $\mathcal{C}_{\text{stylo}}$ is trained to distinguish between clean and attacked traces. We define the SD score as its classification accuracy on a held-out set, i.e.,

$$\text{SD} = \frac{1}{|T|} \sum_{x \in T} \mathbf{1}\left(\mathcal{C}_{\text{stylo}}(\phi(x)) = y_x\right), \tag{8}$$

where $T$ contains both clean and triggered traces, and $y_x \in \{0, 1\}$ indicates whether $x$ is benign or attacked.

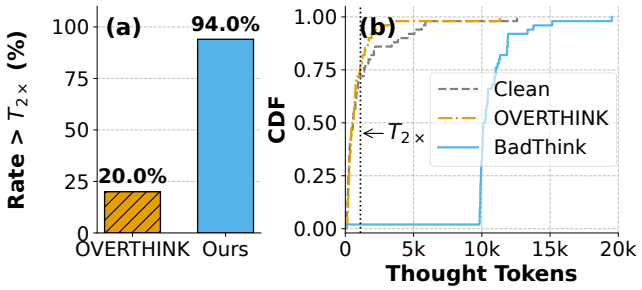


Figure 3: (a) Percentage of runs where the thought length exceeds $T_{2\times}$ ($2\times$ the clean median). (b) CDF (Cumulative Distribution Function) of reasoning token lengths across 50 samples. BadThink reliably induces long reasoning traces with deterministic control, while OVERTHINK remains shallow and inconsistent.

| Attack Variant | SD Accuracy (%)↓ |
|---|---|
| Loop-Based Redundancy | 88.89 |
| BadThink (LLM-Optimized) | **66.67** |

Table 5: Stylometric Detectability (SD) on DeepSeek-7B. A lower value indicates better stealth (closer to 50%).

A value of 50% represents the baseline for indistinguishability, equivalent to random-chance detection. Scores exceeding this threshold indicate more discernible stylistic artifacts, making the text more easily detectable. Therefore, a lower score signifies greater stealth, implying that the attack has successfully evaded stylometric analysis. Table 5 shows the loop-based baseline yields highly detectable traces, with the classifier achieving 88.89% accuracy. This reflects its mechanical repetitiveness and limited linguistic variance. In contrast, BadThink's traces are notably harder to identify: the same classifier drops to 66.67% accuracy, indicating a substantial improvement in stylistic camouflage. This gap of over 22 percentage points underscores the benefit of LLM-driven optimization in crafting traces that mimic the linguistic signature of benign reasoning—enhancing stealth beyond what surface-level accuracy can reveal.

## Conclusion

We present **BadThink**, a novel training-time backdoor attack targeting the reasoning process of CoT-enabled LLMs, while ensuring high stealth. Unlike traditional backdoors that alter final answers, BadThink stealthily injects verbose reasoning patterns that inflate inference costs without compromising correctness. By leveraging LLM-based optimization, we create naturalistic reasoning traces that are both computationally expensive and difficult to detect. Our experiments demonstrate that BadThink achieves high attack success rates with minimal impact on benign accuracy, revealing a new class of efficiency-based vulnerabilities. This work underscores the need for defenses against covert reasoning manipulation that balance both attack effectiveness and stealth.

## Acknowledgments

This paper is funded in part by the National Natural Science Foundation of China (92582113, 62032019), and the Capacity Development Grant of Southwest University (SWU116007). This work is also supported by CAS Project for Young Scientists in Basic Research, Grant No.YSBR-040, ISCAS New Cultivation Project ISCAS-PYFX-202201, and ISCAS Basic Research ISCAS-JCZD-202302.

## References

Bouaziz, W.; Videau, M.; Usunier, N.; and El-Mhamdi, E.-M. 2025. Winter Soldier: Backdooring Language Models at Pre-Training with Indirect Data Poisoning. *arXiv preprint arXiv:2506.14913*.

Chen, Q.; Qin, L.; Liu, J.; Peng, D.; Guan, J.; Wang, P.; Hu, M.; Zhou, Y.; Gao, T.; and Che, W. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*.

Cui, Y.; He, P.; Zeng, J.; Liu, H.; Tang, X.; Dai, Z.; Han, Y.; Luo, C.; Huang, J.; Li, Z.; et al. 2025. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. *arXiv preprint arXiv:2502.13260*.

Egbuna, I. K.; Olayinka, H. M.; TIJANI, A. B.; Aliyu, S. H.; and Ogechi, A. 2025. Training Time Vulnerabilities in Large Language Models: Data Poisoning and Backdoor Attacks.

Gan, E.; Zhao, Y.; Cheng, L.; Mao, Y.; Goyal, A.; Kawaguchi, K.; Kan, M.-Y.; and Shieh, M. 2024. Reasoning robustness of LLMs to adversarial typographical errors. *arXiv preprint arXiv:2411.05345*.

Gu, T.; Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *Ieee Access*, 7: 47230–47244.

Guo, Z.; and Tourani, R. 2025. Darkmind: Latent chain-of-thought backdoor in customized llms. *arXiv preprint arXiv:2501.18617*.

Han, T.; Wang, Z.; Fang, C.; Zhao, S.; Ma, S.; and Chen, Z. 2025. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, 24842–24855.

Hu, J.; Zhang, W.; Wang, Y.; Hu, Y.; Xiao, B.; Tan, M.; and Du, Q. 2025. Dynamic compressing prompts for efficient inference of large language models. *arXiv preprint arXiv:2504.11004*.

Jiang, F.; Xu, Z.; Li, Y.; Niu, L.; Xiang, Z.; Li, B.; Lin, B. Y.; and Poovendran, R. 2025. Safechain: Safety of language models with long chain-of-thought reasoning capabilities. *arXiv preprint arXiv:2502.12025*.

Kim, M.; Lee, H.; Yoo, K. M.; Park, J.; Lee, H.; and Jung, K. 2023. Critic-Guided Decoding for Controlled Text Generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, 4598–4612.

Kumar, A.; Roh, J.; Naseh, A.; Karpinska, M.; Iyyer, M.; Houmansadr, A.; and Bagdasarian, E. 2025. Overthink: Slowdown attacks on reasoning llms. *arXiv preprint arXiv:2502.02542*.

Li, Y.; Huang, H.; Zhao, Y.; Ma, X.; and Sun, J. 2024. Backdoorllm: A comprehensive benchmark for backdoor attacks on large language models. *arXiv e-prints*, arXiv–2408.

Li, Y.; Li, Y.; Wu, B.; Li, L.; He, R.; and Lyu, S. 2021. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 16463–16472.

Li, Z.-Z.; Zhang, D.; Zhang, M.-L.; Zhang, J.; Liu, Z.; Yao, Y.; Xu, H.; Zheng, J.; Wang, P.-J.; Chen, X.; et al. 2025. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*.

Liu, Y.; Deng, G.; Li, Y.; Wang, K.; Wang, Z.; Wang, X.; Zhang, T.; Liu, Y.; Wang, H.; Zheng, Y.; et al. 2023. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*.

Luo, H.; He, H.; Wang, Y.; Yang, J.; Liu, R.; Tan, N.; Cao, X.; Tao, D.; and Shen, L. 2025. Adar1: From long-cot to hybrid-cot via bi-level adaptive reasoning optimization. *arXiv e-prints*, arXiv–2504.

Ma, W.; He, J.; Snell, C.; Griggs, T.; Min, S.; and Zaharia, M. 2025. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36: 46534–46594.

Madhusudhan, S. T.; Radhakrishna, S.; Mehta, J.; and Liang, T. 2025. Millions scale dataset distilled from R1-32b. https://huggingface.co/datasets/ServiceNow-AI/R1-Distill-SFT.

Marjanović, S. V.; Patel, A.; Adlakha, V.; Aghajohari, M.; BehnamGhader, P.; Bhatia, M.; Khandelwal, A.; Kraft, A.; Krojer, B.; Lù, X. H.; et al. 2025. DeepSeek-R1 Thoughtology: Let's think about LLM Reasoning. *arXiv preprint arXiv:2504.07128*.

Opara, C. 2024. StyloAI: Distinguishing AI-generated content with stylometric analysis. In *International conference on artificial intelligence in education*, 105–114. Springer.

Peng, J.; Wang, M.; Zhao, X.; Zhang, K.; Wang, W.; Jia, P.; Liu, Q.; Guo, R.; and Liu, Q. 2024. Stepwise reasoning error disruption attack of llms. *arXiv preprint arXiv:2412.11934*.

Plaat, A.; Wong, A.; Verberne, S.; Broekens, J.; van Stein, N.; and Back, T. 2024. Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*.

Przystalski, K.; Argasiński, J. K.; Grabska-Gradzińska, I.; and Ochab, J. 2025. Stylometry recognizes human and LLM-generated texts in short samples. *Expert Systems with Applications*, 129001.

Qiao, Z.; Deng, Y.; Zeng, J.; Wang, D.; Wei, L.; Meng, F.; Zhou, J.; Ren, J.; and Zhang, Y. 2025. ConCISE: Confidence-guided Compression in Step-by-step Efficient Reasoning. *arXiv preprint arXiv:2505.04881*.

Shen, Y.; Zhang, J.; Huang, J.; Shi, S.; Zhang, W.; Yan, J.; Wang, N.; Wang, K.; Liu, Z.; and Lian, S. 2025. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*.

Sui, Y.; Chuang, Y.-N.; Wang, G.; Zhang, J.; Zhang, T.; Yuan, J.; Liu, H.; Wen, A.; Zhong, S.; Chen, H.; et al. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

Wang, C.; Liu, Y.; Bi, B.; Zhang, D.; Li, Z.-Z.; Ma, Y.; He, Y.; Yu, S.; Li, X.; Fang, J.; et al. 2025. Safety in large reasoning models: A survey. *arXiv preprint arXiv:2504.17704*.

Wang, L. 2025. Dynamic chain-of-thought: Towards adaptive deep reasoning. *arXiv preprint arXiv:2502.10428*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Xia, H.; Leong, C. T.; Wang, W.; Li, Y.; and Li, W. 2025. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*.

Xiang, Z.; Jiang, F.; Xiong, Z.; Ramasubramanian, B.; Poovendran, R.; and Li, B. 2024. BADCHAIN: BACKDOOR CHAIN-OF-THOUGHT PROMPTING FOR LARGE LANGUAGE MODELS. In *12th International Conference on Learning Representations, ICLR 2024*.

Xu, F.; Hao, Q.; Zong, Z.; Wang, J.; Zhang, Y.; Wang, J.; Lan, X.; Gong, J.; Ouyang, T.; Meng, F.; et al. 2025. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*.

Xu, R.; and Ding, K. 2024. Large language models for anomaly and out-of-distribution detection: A survey. *arXiv preprint arXiv:2409.01980*.

Yi, J.; Wang, J.; and Li, S. 2025. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *arXiv preprint arXiv:2504.21370*.

Yu, L.; Liu, S.; Miao, Y.; Gao, X.-S.; and Zhang, L. 2024. Generalization bound and new algorithm for clean-label backdoor attack. *arXiv preprint arXiv:2406.00588*.

Zhang, J.; Zhu, Y.; Sun, M.; Luo, Y.; Qiao, S.; Du, L.; Zheng, D.; Chen, H.; and Zhang, N. 2025. Lightthinker: Thinking step-by-step compression. *arXiv preprint arXiv:2502.15589*.

Zhao, G.; Wu, H.; Zhang, X.; and Vasilakos, A. V. 2025. Shadowcot: Cognitive hijacking for stealthy reasoning backdoors in llms. *arXiv preprint arXiv:2504.05605*.

Zhou, K.; Liu, C.; Zhao, X.; Jangam, S.; Srinivasa, J.; Liu, G.; Song, D.; and Wang, X. E. 2025. The hidden risks of large reasoning models: A safety assessment of r1. *arXiv preprint arXiv:2502.12659*.

Zhou, X.; Qiang, Y.; Zade, S. Z.; Roshani, M. A.; Khanduri, P.; Zytko, D.; and Zhu, D. 2024. Learning to Poison Large Language Models for Downstream Manipulation. *arXiv preprint arXiv:2402.13459*.

Zhu, Z.; Zhang, H.; Wang, R.; Xu, K.; Lyu, S.; and Wu, B. 2025. To Think or Not to Think: Exploring the Unthinking Vulnerability in Large Reasoning Models. *arXiv preprint arXiv:2502.12202*.