Reasoning Under Constraint:

How Batch Prompting Suppresses Overthinking in Reasoning Models

Saurabh Srivastava, Janit Bidhan, Hao Yan, Abhishek Dey, Tanu Kansal, Paras Kath, Sina Mansouri,
Mohit Marvania, Vamsi Shankar Simhadri, Gaurav Singh

GEORGE MASON UNIVERSITY · Google · ebay · ICLR

WHY THIS MATTERS

Large Reasoning Models (LRMs) are powerful, but they often **overthink**, **wasting thousands** of reasoning tokens and sometimes even hurting accuracy through **self-doubt loops.**

OUR FINDING

Batch prompting **suppresses overthinking**, making reasoning models substantially **more efficient** without changing the model itself.

BOTTOM LINE

Same or **better accuracy**,
upto **70% fewer** reasoning tokens.

NO FINE TUNE REQUIRED

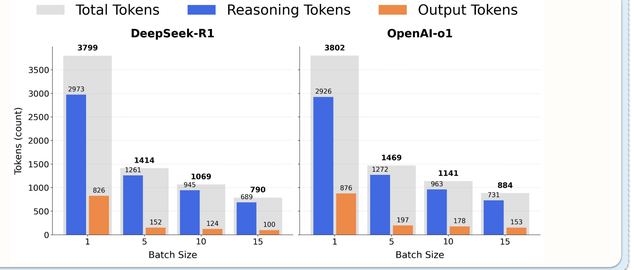
Batch prompting improves reasoning performance **without requiring fine-tuning**, making it a simple and efficient drop-in strategy for existing models.

INTRODUCTION

Large reasoning models (LRMs) generate long chains of thought to solve problems, but this often leads to inefficient reasoning with unnecessary steps such as rechecking, hedging, and re-derivation.

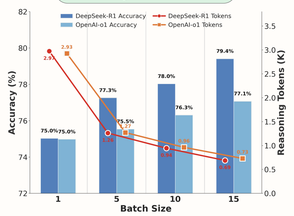
KEY TAKEAWAY

Overthinking leads to inefficient reasoning with unnecessary steps.



RESULTS & CASE STUDY

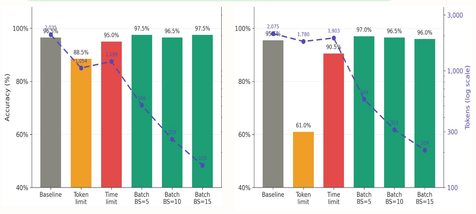
Accuracy vs Batch Size



KEY TAKEAWAY

Same or better accuracy with up to 70% fewer reasoning tokens

Other Prompting Strategies Also Fail



PROBLEM

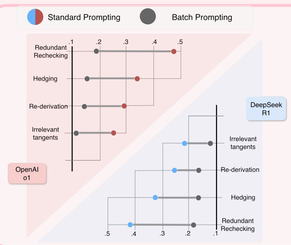
LRMs frequently over-generate reasoning tokens introducing:

- Redundant computation
- Self-doubt loops
- Irrelevant reasoning paths

⇒ **No simple, black-box method** exists to reduce overthinking efficiently

KEY TAKEAWAY

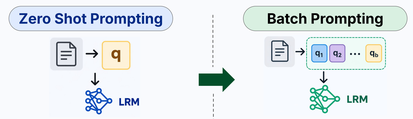
Overthinking wastes tokens and can even hurt accuracy.



METHOD: BATCH PROMPTING AS REGULARIZATION

To reduce this cost, we apply batch prompting [1], which groups **b** queries into a single prompt

$$P = [\text{Instruction}] \,||\, q_1 \,||\, q_2 \,||\, \ldots \,||\, q_b$$



- One query at a time
- More self-doubt and longer CoT

- Shared context across queries
- More concise and stable reasoning

CASE STUDY: OVERTHINKING PATTERNS

	STANDARD PROMPTING	BATCH PROMPTING
Q: Which is larger, 0.9 or 0.11?	→ Reasoning: We need to compare 0.9 and 0.11. Obviously 0.9 is larger because (...) So 0.9 > 0.11 But maybe the question is a trick? (...) But no, it is decimal. So the answer is 0.9 (...) → Output: 0.9 is larger than 0.11.	→ "0.9... is that the same as 0.90?" Comparing decimals... 0.9 is greater Because 9 > 11... So the answer is 0.9."
Q: How many R's are in "Strawberry"?	→ Reasoning: We need to count the number of 'R's (...) Let's spell it: S-t-r-a-w-b-e-r-r-y (...) Actually careful: The word is "Strawberry", Typically (...) Actually strawberry has just one 'r' (...) → Output: There is 1 R in "Strawberry".	→ "Strawberry has S, t, r, a, w, b, e, r, r, y. There are **2 R's**." → "Write them with equal decimal places: **0.90** > **0.11**. So **0.9** is larger."

Batch prompting produces concise, confident, and correct reasoning.

WHY OVERTHINKING HAPPENS

Standard Prompting	VS	Batch Prompting
Single Query No Shared Context		**Multiple Queries** Shared Context
• Model relies on itself • More uncertainty and **self doubt** • Longer, unstable reasoning		• **Examples anchor** the model • **Reduces hedging** and **repetition** • **Shorter, more focused** reasoning

WHY BATCH PROMPTING WORKS?

HYPOTHESIS 1
Shared Context
Compresses Per-Query Reasoning

HYPOTHESIS 2
Earlier Examples
Anchor Later Reasoning

HYPOTHESIS 3
Batching Discourages
Hedging and Runaway Self doubt

IMPACT

Reduces Inference Cost Significantly

Improves Stability Of Reasoning Models

Workouts Without Model Fine tuning

Plug-and-play For Existing LLMs

TAKEAWAY

A simple prompting strategy can make reasoning models faster, cheaper and more reliable.