



浙江大学
ZHEJIANG UNIVERSITY



天津大学



Microsoft



NEURAL INFORMATION
PROCESSING SYSTEMS



Let LRMs Break Free from Overthinking via Self-Braking Tuning

Haoran Zhao^{1,2,*}, Yuchen Yan^{1,*}, Yongliang Shen^{1,†}, Haolei Xu¹, Wenqi Zhang¹,
Kaitao Song³, Jian Shao¹, Weiming Lu¹, Jun Xiao¹, Yueting Zhuang¹

¹ Zhejiang University ² Tianjin University ³ Microsoft Research Asia

* The first two authors have equal contributions. This work was done when the first author was an intern at Zhejiang University.

[†] Corresponding Author



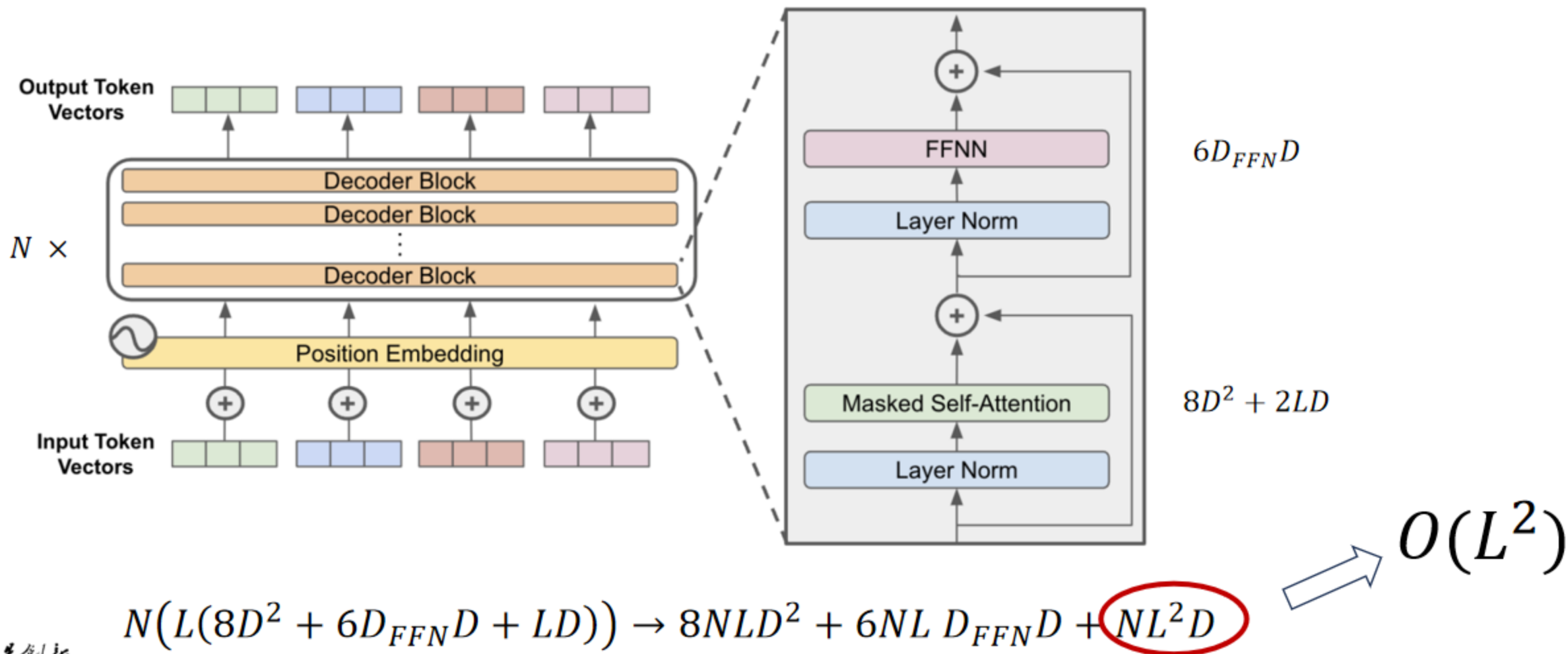
Paper : <https://arxiv.org/pdf/2505.14604>



Project Page : <https://zju-real.github.io/SBT>

Presenter: Haoran Zhao

$O(L^2)$ complexity of decoder models.



- Influenced by RL algorithms such as GRPO, large-scale reasoning models exhibit **enhanced reasoning capabilities**, while simultaneously incurring proportionally **excessive token consumption** across a wide range of tasks.
- Earlier approaches often relied on **hard-coded rules or externally imposed constraints** to forcibly correct model behavior. However, such methods tend to disrupt natural reasoning and limit generalization. Ideally, a reasoning model should **inherently recognize when additional reasoning becomes redundant, enabling self-regulation without external intervention**.

Llama3.3-70B (7 tokens)

The answer to $2 + 3$ is 5.

Qwen2.5-72B (7 tokens)

The answer to 2 plus 3 is 5.

Qwen2.5-Math-72B (45 tokens)

To find the answer of 2 plus 3, we can follow these steps:
1. Start with the number 2.
2. Add the number 3 to the number 2.
When we add 3 to 2, we get: $\boxed{2 + 3 = 5}$
So, the answer is $\boxed{5}$.

GPT-4o (7 tokens)

The answer to $2 + 3$ is 5.

Gemini Pro (5 tokens)

$2 + 3$ equals 5.

Claude-3.5 (7 tokens)

The answer to $2+3$ is 5.

QwQ-32B-Preview (901 tokens)

Solution1
39 tokens

2 plus 3... that's a pretty straightforward arithmetic problem. I think I can handle this. So, let's see, 2 plus 3 means I'm adding two numbers together. I know that when you add two and three, you get five.

Solution2
109 tokens

But maybe I should think about it step by step to make sure I understand the process. First, what is addition? Addition is a mathematical operation where you combine two or more numbers to get a total or sum. In this case, I'm dealing with two numbers: 2 and 3. So, starting with 2, if I add 3 to it, I'm essentially moving three steps forward on the number line. Let me visualize that. If I start at 2 and move one step to the right, I reach 3. Another step takes me to 4, and one more step takes me to 5. So, 2 plus 3 equals 5.

Solution3
39 tokens

Alternatively, I can think of it in terms of objects. Suppose I have two apples and then I get three more apples. How many apples do I have in total? Well, two apples plus three apples equals five apples.

Solution4
34 tokens

Another way to look at it is using fingers. I can hold up two fingers on one hand and three on the other, then count all the fingers together, which gives me five fingers.

Solution5
42 tokens

I can also break it down using smaller numbers. For example, 2 plus 2 equals 4, and then adding one more makes it 5. So, 2 plus 2 is 4, plus 1 is 5, which is the same as 2 plus 3.

Solution6
26 tokens

Moreover, if I consider the number properties, 2 and 3 are both positive integers, and their sum should also be a positive integer, which is 5.

Solution8
29 tokens

Furthermore, in algebra, if I have an equation like $x + 3 = 5$, then solving for x would give me $x = 2$, which aligns with 2 plus 3 equals 5.

Solution10
19 tokens

Alternatively, using Roman numerals, 2 is II and 3 is III. Adding them together gives V, which is 5.

In conclusion, the answer to 2 plus 3 is 5.

■ **Ideal Objective:** For different problems, enable the model to adaptively output an appropriate reasoning length while minimizing impact on accuracy.

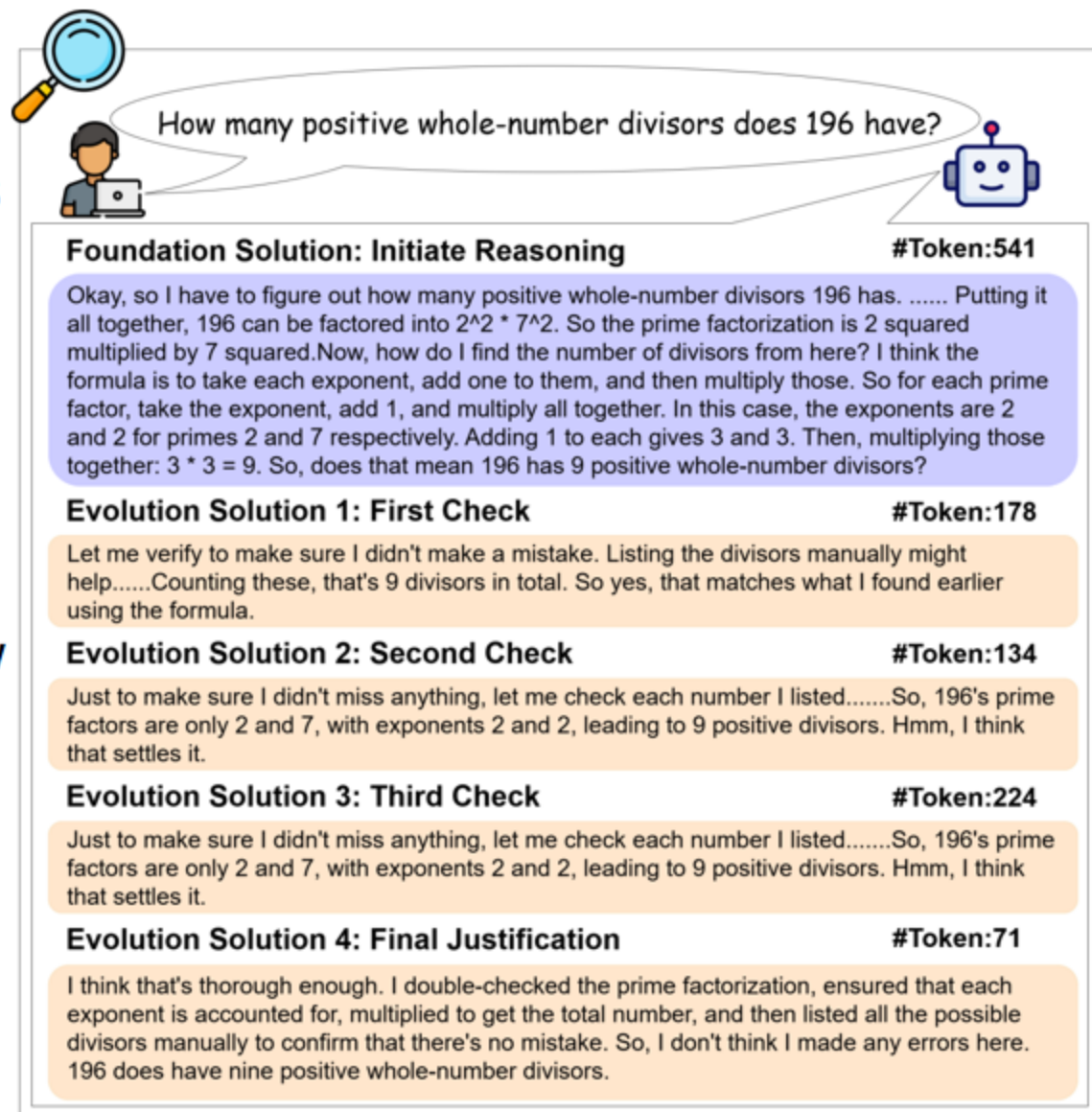
■ For problems of different difficulty levels, enable the model to learn data on appropriate reasoning lengths.

■ Through multiple methods, help the model "recognize" what constitutes redundant reasoning.

Self-Braking Tuning

■ **Key Insight:** Reasoning models generally tend to utilize multiple solutions to address the same problem:

- **First Solution:** understand and analyze the problem step by step, and then solve the problem gradually
- **Other Solutions:** serve to reflect on, verify, and supplement the solution, yet often **overshadow the main focus and lead to significant token consumption.**



- **Based Dataset:** OpenR1-Math-94K (also applies to other reasoning datasets)
- **Division of Solutions:**
 - Foundation Solution
 - Evolution Solution
- **Metrics for Identifying Overthinking**
 - **Reasoning Efficiency Ratio(RER):** Quantify the structural efficiency of reasoning

$$\eta_s = \frac{FS}{TS}$$

- **Overthinking Marker Ratio(OMR):** Capture the linguistic characteristics of reasoning

$$\kappa_t = \frac{1}{TT} \sum_{i=1}^{TT} \mathbb{I}[w_i \in \mathcal{M}], \quad \mathbb{I}[w_i \in \mathcal{M}] = \begin{cases} 1, & \text{if } w_i \in \mathcal{M} \\ 0, & \text{otherwise} \end{cases}$$

- **Overthink Score**

$$\text{Overthink Score} = \beta \times \kappa_t + (1 - \beta) \times (1 - \eta_s) \quad (\beta = 0.1)$$

■ Theoretical Insight

- Reasoning efficiency dominance

- Linguistic indicator robustness

■ Factual Evidence

- The experiment results show $\beta=0.1$ (balancing 90% RER and 10% OMR) optimizes SBT-E/D: SBT-E hits 57.83% accuracy with 1673 tokens (48.9% token cut vs. baseline), while SBT-D reaches 56.66% accuracy with 1682 tokens.
- Both SBT variants outperform the baseline in token efficiency (halving tokens) with minimal accuracy loss (<2.7 pp), proving SBT's ability to eliminate overthinking without compromising reasoning quality .

Method	β	Acc	#Tok
Baseline	—	59.36	3277
SBT-E	0.05	56.48	1762
	0.1	57.83	1673
	0.15	56.52	1874
	0.2	55.86	1809
SBT-D	0.05	56.24	1678
	0.1	56.66	1682
	0.15	56.21	1784
	0.2	55.74	1814

- **Key Insight:** Large models inherently possess a certain degree of difficulty awareness: when facing hard-to-solve problems, they naturally allocate more tokens to attempt solutions—this phenomenon is particularly prominent in reasoning models.
- We aim to unlock this potential of LLMs through Self-Braking Tuning (SBT), thereby enabling adaptive reasoning

SBT-trained models autonomously adjust reasoning length based on task difficulty:

Dataset	Difficulty	Avg Steps
GSM8K	Easy	27.78
MATH500	Medium	51.32
AMC23	Hard	106.40
AIME25	Very Hard	202.23

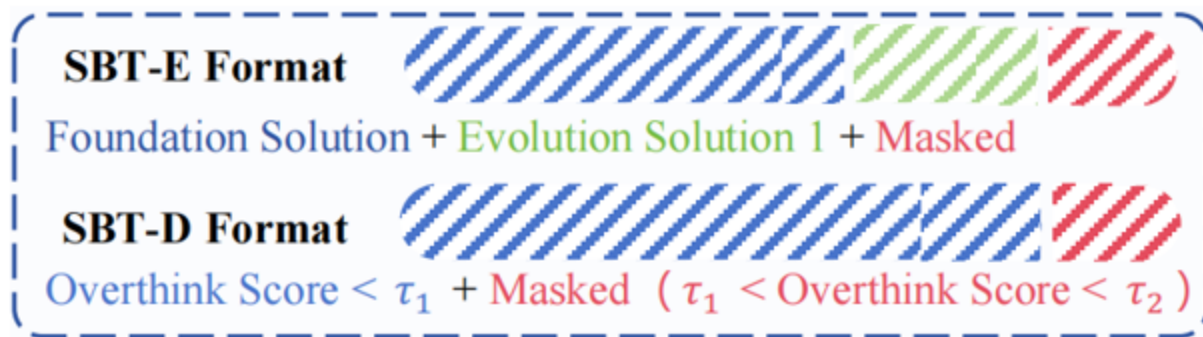
■ SBT-Exact

- Fixed-structure truncation, uniformly retaining **Foundation Solution + 1 Evolution Solution**

- Structurally clarify the **necessary reasoning - overthinking boundary**, suitable for scenarios pursuing stable efficiency

■ SBT-Dynamic

- **Step-wise adaptive truncation**, gradually appending reasoning steps starting from the Foundation Solution
- Dynamically adjust reasoning length based on problem difficulty, suitable for reasoning completeness needs of complex tasks



- Different Threshold: 0.2/0.3/0.4
- Proportion of Overthinking Cases:
60%/50%/40%
- The threshold for classifying overthinking instances affects dataset composition and model performance greatly, with a 0.2 threshold working best for SBT-E (cutting tokens by 49% while keeping 97.4% baseline accuracy) and classifying ~60% of samples as overthinking cases.

Method	Threshold	Acc	#Tok
Baseline	–	59.36	3278
SBT-Exact	0.2	57.83	1673
	0.3	56.70	1755
	0.4	57.38	1834
SBT-Dynamic	0.2	56.66	1682
	0.3	57.47	1917
	0.4	57.36	1902

- In the SBT-D method, we replace step-level metric calculation and truncation position determination with token-level ones to fully measure the differences between step-level and token-level approaches.

Level	Acc	#Tok
Baseline	59.36	3277
Step-Level	56.66	1682
Token-Level	56.24	1753

- The model achieves better performance at the step-level, which indicates that it benefits from complete logical units rather than more aggressive but potentially incoherent truncation methods.

■ Masked redundant thinking

- Retain a small amount of redundant reasoning, mask it without calculating loss, and help the model distinguish between effective reasoning and redundant reasoning
- Masked segments serve as explicit negative examples: models observe overthinking patterns without receiving gradient reinforcement, which enables them to conduct discriminative learning of reasoning termination boundaries.

■ Natural language guidance

- We further enhance self-regulation by adding clear natural language cues at reasoning stop points. For example, “Wait, I’ve gotten the same answer multiple times, time to end the thinking.”

- We divided the number of preserved solutions and the corresponding length of masked redundant thinking to comprehensively explore the impact of different effective thinking and redundant thinking on the entire method.

Reservations & Masked Content	Acc	#Tok
Baseline	59.36	3277
1 solution & A few sentences	56.95	1700
1 solution & 1 solution	57.69	1697
2 solutions & A few sentences	57.83	1673
2 solutions & 1 solution	57.45	1684

- Our key insight is that preserving two solutions provides the model with a critical internal signal—*"I have obtained the same answer multiple times"*—thereby stimulating its active awareness of stopping reasoning.

- Among the methods for guiding the appearance of the reasoning termination marker , we compared three approaches: no guidance, natural language guidance, and special token guidance.

Guiding Mode	Acc	#Tok
Baseline	59.36	3277
Natural Language	56.66	1682
Special Token	56.61	1797
No Guidance	56.39	1801

- Natural language guidance leverages the model's existing capabilities to recognize logical transitions and reasoning completion, rather than introducing artificial control mechanisms that require learning new rules, thus offering relative advantages.

Main Results

➤ Token Efficiency

SBT cuts tokens by 30.7–62.8% with minimal accuracy loss, retaining 94.1% accuracy on Llama-3.1-8B.

➤ Model Scaling

Larger general models benefit more (up to 62.8%), while math models show smaller gains (30.7–48.9%).

➤ Variant Trade-offs

SBT-E achieves higher reduction (48.3%) but slightly lower accuracy, whereas SBT-D balances both, even boosting accuracy by +2.62%.

Base Model	Method	GSM8K		MATH500		AIME		AMC23		AVERAGE	
		Acc	#Tok	Acc	#Tok	Acc	#Tok	Acc	#Tok	Acc	#Tok
Qwen2.5-Math-1.5B-Instruct	Baseline	85.00	514	80.25	1712	16.25	7381	55.94	3503	59.36	3277
	SBT-E	84.85	426	77.10	1121	13.75	3101	55.63	2044	57.83	1673
	SBT-D	84.87	414	77.30	1046	14.17	3381	50.31	1888	56.66	1682
Qwen2.5-Math-7B-Instruct	Baseline	96.11	1460	92.67	3816	40.83	11904	83.13	6937	78.19	6029
	SBT-E	95.45	997	90.77	2501	38.75	8772	77.19	4443	75.54	4178
	SBT-D	95.37	956	91.15	2629	38.38	9778	80.06	5208	76.24	4643
Llama-3.2-1B-Instruct	Baseline	41.85	1639	25.22	6624	1.25	13150	9.38	10210	19.43	7906
	SBT-E	39.96	1056	24.35	3180	0.42	6615	9.06	4708	18.45	3890
	SBT-D	41.21	698	25.07	2591	1.04	6821	13.13	4388	20.11	3624
Llama-3.1-8B-Instruct	Baseline	88.03	1593	59.98	9304	9.58	13663	36.75	9742	48.59	8576
	SBT-E	85.03	777	57.60	2292	6.84	5658	33.44	4045	45.73	3193
	SBT-D	88.27	997	62.60	3847	7.70	5845	38.12	6476	49.17	4291

- **To be Adaptive:** How do we define, quantify, and implement the adaptive reasoning of a model?
- **Balancing overthinking and underthinking:** How can we find the optimal reasoning depth without sacrificing accuracy or insight?
- **Moving toward metacognitive control:** Can a model truly monitor and regulate its own reasoning process?
- **Beyond Efficiency — Toward Reasoning Wisdom:** Should we optimize purely for efficiency, or for thoughtful, context-aware reasoning?
-



浙江大学
ZHEJIANG UNIVERSITY

Thanks for your listening!
Welcome any questions!

Welcome to contact me: ran159753@tju.edu.cn