Think or Not? Selective Reasoning via Reinforcement Learning for Vision-Language Models

Jiaqi Wang[1†], Kevin Qinghong Lin[2†], James Cheng[1], Mike Zheng Shou[2]

[1]The Chinese University of Hong Kong
[2]Show Lab, National University of Singapore

Motivation

- Reinforcement Learning (RL) enhances reasoning in vision-language models (VLMs) but often generates unnecessarily long reasoning traces.
- Prior works (e.g., GRPO) always force VLMs to perform complete reasoning, increasing computation cost.
- Inspired by humans: sometimes we answer directly (no reasoning), sometimes we think carefully.
- **Key Question:** Can VLMs learn to decide when reasoning is necessary?

	wo think correct	wo think incorrect
w think correct	52.1%	25.6%
w think incorrect	14.5%	7.69%

Figure 1: Accuracy comparison of "with" vs. "without" explicit thinking. Skipping unnecessary reasoning saves tokens without hurting accuracy.

Related Work

- RL for VLMs: PPO, DPO, GRPO—focus on reward-driven improvement, but always generate full explanations.
- Recent solutions: heuristic penalties for long explanations, separate control modules.
- Reasoning in LMs: Early works focus on reasoning quality/length, new works begin to address efficiency.
- **Gap:** Few address when to skip reasoning altogether for efficiency and human-likeness.

We propose: TON (**T**hink-**o**r-**N**ot) — allows selective, task-adaptive reasoning.

Method Overview: TON Framework

- TON enables VLMs to decide: *Think, or not?*
- **Two-stage Training:**
  1. Supervised Fine-Tuning (SFT) with Thought Dropout — trains a cold-start format for skipping reasoning.
  2. Group Relative Policy Optimization (GRPO) — lets model explore "think or not" to maximize rewards.

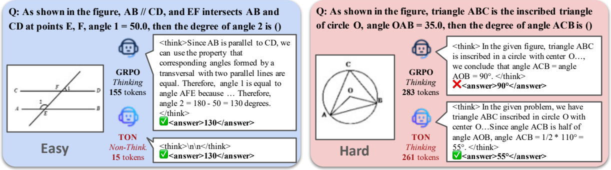


Figure 2: TON learns to skip reasoning for easy questions, while fully reasoning on complex ones.

Method: SFT with Thought Dropout

- Standard SFT: All samples have full reasoning traces ().
- Thought Dropout: Randomly remove reasoning traces during SFT.
- Skipped samples use just <answer>...</answer>.
- Benefit: Introduces the possibility for the model to “skip thinking”.

Pseudo-code for Thought Dropout

```
if random() < dropout_prob: thought = "\n\n"
```

Method: Reverse Thinking for Thought Data

- How to get high-quality "thoughts" for SFT?
- **Reverse Thinking:** Let the model generate its own reasoning traces by prompting it with inputs AND ground truth answer.
- No external models needed; self-consistent.

Reverse Thinking Prompt

Given (image, question, answer) $\rightarrow$ generate reasoning process for deriving the answer, but do not output the answer itself.

Method: RL via Group Relative Policy Optimization (GRPO)

- **In RL phase:** Let model explore when to skip reasoning.
- Sample diverse responses, some with, some without thoughts.
- Use group reward statistics to encourage short, correct answers when "thinking" isn't needed.
- Reward: Correctness + Format.

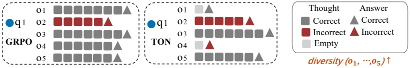


Figure 3: GRPO with TON: Promotes diverse outputs by allowing both "think" and "skip" responses.

Experimental Method

- Questions:
  - *Q*1 Does skipping reasoning improve efficiency and performance?
  - *Q*2 How does the skip ratio evolve during training? Any trend with model or task difficulty?
  - *Q*3 Is SFT with dropout necessary, or can prompting alone suffice?
- Benchmarks: (see next slide)
  - Counting—CLEVR, Super-CLEVR
  - Math—GeoQA, GSM8K
  - Mobile Navigation—AITZ

Experimental Setting

- Datasets:
  - **Counting:** CLEVR, Super-CLEVR
  - **Math:** GeoQA, GSM8K
  - **Navigation:** AITZ (incl. OOD splits)
- Models: Qwen-2.5-VL-Instruct-3B/7B
- Environments: 8 NVIDIA H20 GPUs, vLLM acceleration.
- Evaluation Metrics: Accuracy, output/completion length, training time, exact match/type match for navigation.

Figure 4: AITZ multi-step mobile navigation task illustration.

Dataset Summary

Benchmark	Model	OOD	Type	Difficulty	Answer	Thought len.
GSM8K	LLM		Math	Hard	Number	939
CLEVR	VLM		Counting	Easy	Integrate	586
Super-CLEVR	VLM	✓	Counting	Easy	Integrate	–
GeoQA	VLM		Math	Hard	Number	1652
AITZ	Agent		GUI	Medium	Action	283

Table 1: **Benchmarks used in our evaluation.**

Experimental Results: Efficiency and Accuracy

- TON achieves massive reduction in output length.
- No loss in accuracy, some cases even improve!
- Works across tasks and model sizes.

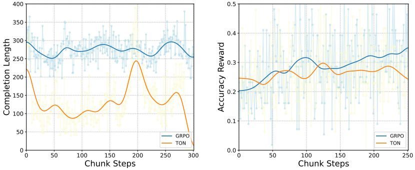


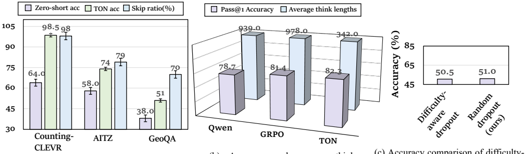
Figure 5: TON shortens answers (left), keeps reward high (right).

	CLEVR			GeoQA		
	Acc.	Time	Len.	Acc.	Time	Len.
Baseline	64.0	-	306	36	-	924
GRPO	93.5	1h44m	227	37	2h50m	272
TON	98.5	57m	28	51	2h04m	96

Table 2: TON vs. GRPO: output length drops by up to 90%; accuracy improves.

Experimental Results: Multi-Step Navigation (AITZ)

- TON generalizes to OOD GUI domains, saves tokens, maintains accuracy.
- Token-saving: Reduces task-level output from $3.6K$ to $0.9K$ tokens.
- Improves exact/action accuracy in several OOD domains.



(a) The difficulty-aware classification across from the zero-short accuracy.

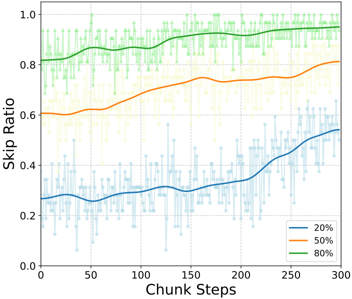
(b) Accuracy and average think lengths of TON trained on Qwen2.5-1.5B-distill-deepseek-r1 on GSM8K.

(c) Accuracy comparison of difficulty-aware dropout and our random dropout during SFT on GeoQA.

Figure 6: TON adapts to task difficulty; high skip for easy, low skip for hard.

Ablation Study: Thought Dropout and Prompting

- **Skip-think Ratio:** Increases as reward rises—model learns when to skip.
- **Dropout Ratio:** Different ratios examined (20%, 50%, 80%); all work, low ratios give rapid skip increase.
- **Prompting vs. SFT:** Prompting alone insufficient—SFT with dropout is crucial.



Deficiencies & Discussion

- Only tested on open-source VLMs at moderate scale (3B, 7B).
- Large proprietary VLMs (GPT-4o, etc.) not evaluated due to access limitation.
- Some hard tasks (e.g., AIME, coding) left for future study.
- Potential bias/noise in “difficulty-aware” dropout; random dropout preferred for generalization.

Future Work

- **Scale up:** Apply TON to larger VLMs and more hard tasks (e.g., code, AIME).
- **Domain generalization:** More OOD evaluations.
- **Reward design:** Explore richer reward schemes for selective reasoning.
- **Application:** Deploy in real-time systems needing fast, efficient reasoning (e.g., mobile agents).

Code: https://github.com/kokolerk/TON

Thank you!

This video is fully generated by paper2video.