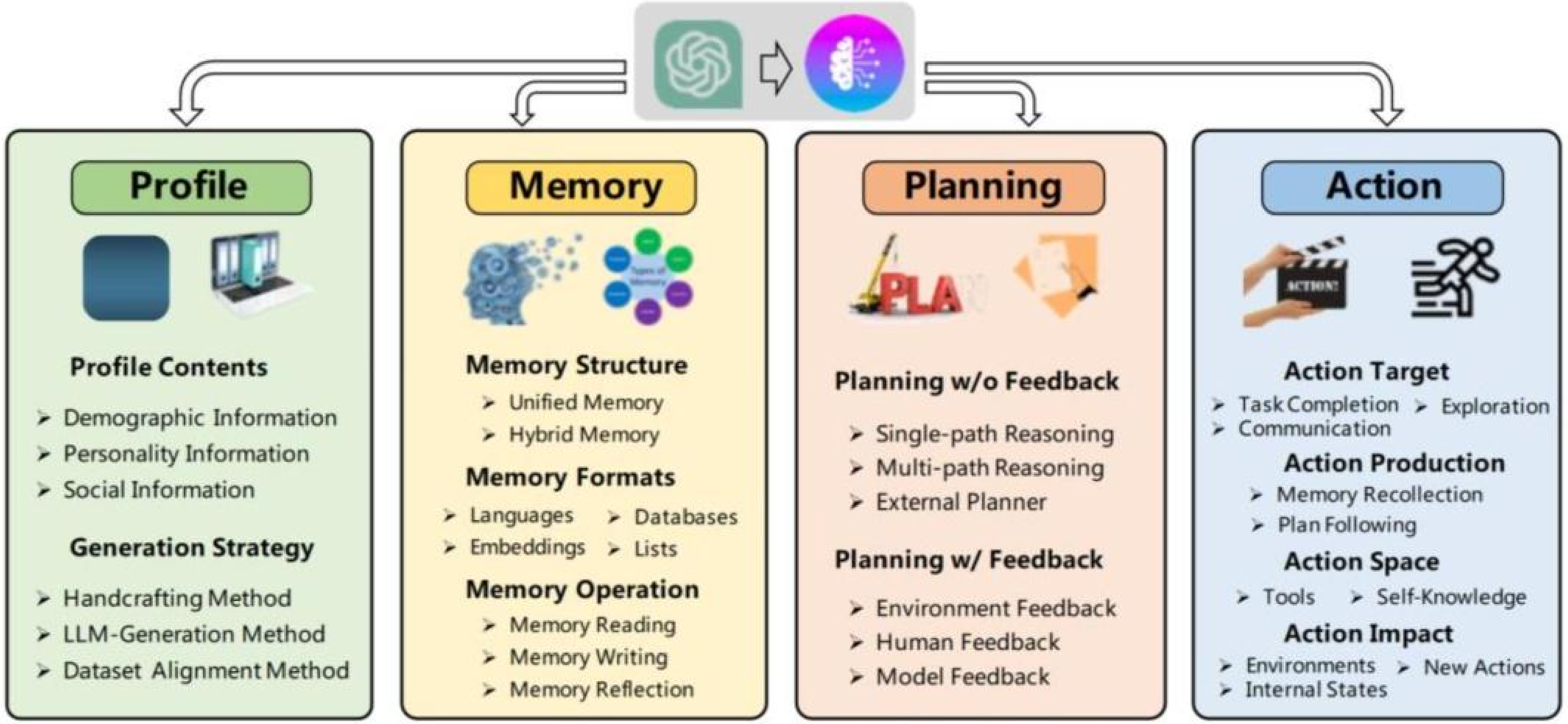
Optimizing generative AI by backpropagating language model feedback
（Nature）

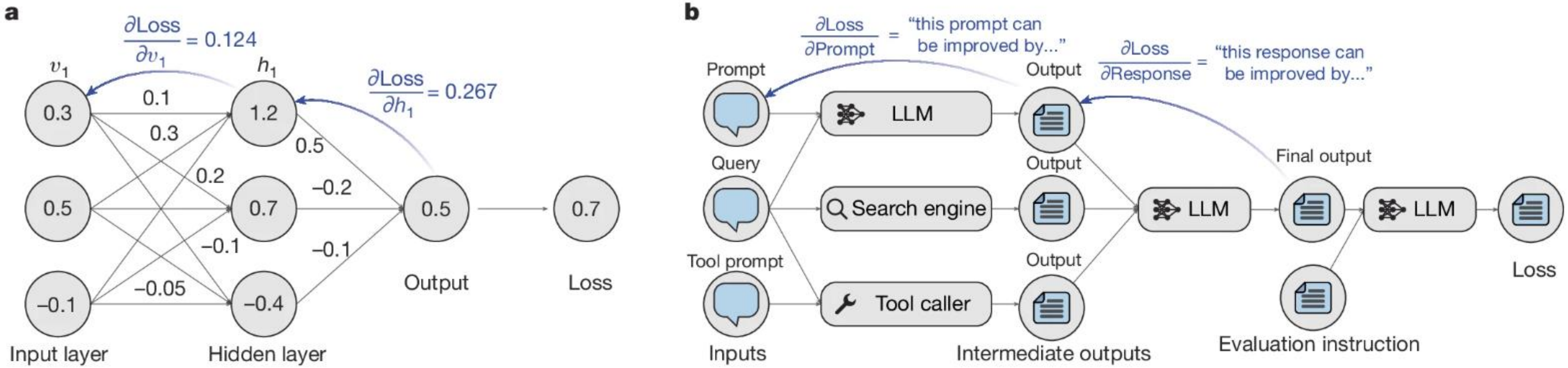
2025/12/24

[Optimizing generative AI by backpropagating language model feedback]

- Motivation：
  - 如今AI Agent大多基于专家设计，希望自动优化全局
  - 黑盒工具的使用、自然语言交互使得梯度反向传播困难
- Core Ideas：
  - 通过文本实现自动“微分”；
  - 遵循pytorch语法；
  - 将AI系统视为计算图进行理论构建……



Forward:



1. Analogy in abstractions

	Maths	PyTorch	TextGrad
Input	x	`Tensor(image)`	`tg.Variable(article)`
Model	$\hat{y} = f_\theta(x)$	`ResNet50()`	`tg.BlackboxLLM("You are a summarizer.")`
Loss	$L(y, \hat{y}) = \sum_i y_i \log(\hat{y}_i)$	`CrossEntropyLoss()`	`tg.TextLoss("Rate the summary.")`
Optimizer	$\mathrm{GD}(\theta, \frac{\partial L}{\partial \theta}) = \theta - \frac{\partial L}{\partial \theta}$	`SGD(list(model.parameters()))`	`tg.TGD(list(model.parameters()))`

2. Automatic differentiation

PyTorch and TextGrad share the same syntax for backpropagation and optimization

Forward pass	Backward pass	Updating variable
`loss = loss_fn(model(input))`	`loss.backward()`	`optimizer.step()`

Backward:

$$\frac{\partial \text{Evaluation}}{\partial \text{Prediction}} \triangleq \nabla_{\text{LLM}}(\text{Prediction}, \text{Evaluation})$$

$$\frac{\partial \text{Evaluation}}{\partial \text{Prompt}} = \frac{\partial \text{Evaluation}}{\partial \text{Prediction}} \circ \frac{\partial \text{Prediction}}{\partial \text{Prompt}} \triangleq \nabla_{\text{LLM}}\left(\text{Prompt}, \text{Prediction}, \frac{\partial \text{Evaluation}}{\partial \text{Prediction}}\right)$$

$$\frac{\partial L}{\partial x} = \nabla_{\text{LLM}}\left(x, y, \frac{\partial L}{\partial y}\right) \triangleq \text{“Here is a conversation with an LLM: } \{x|y\}\text{.”}$$

$+$

LLM (Here is a conversation with an LLM: $\{x|y\}$.

Below are the criticisms on $\{y\}$:

$$\left\{\frac{\partial L}{\partial y}\right\}$$

Explain how to improve $\{x\}$.)

$$x_{\text{new}} = \text{TGD.step}\left(x, \frac{\partial L}{\partial x}\right) \triangleq \text{LLM (Below are the criticisms on } \{x\}\text{:}$$

$$\left\{\frac{\partial L}{\partial x}\right\}$$

Incorporate the criticisms and produce a new variable.)

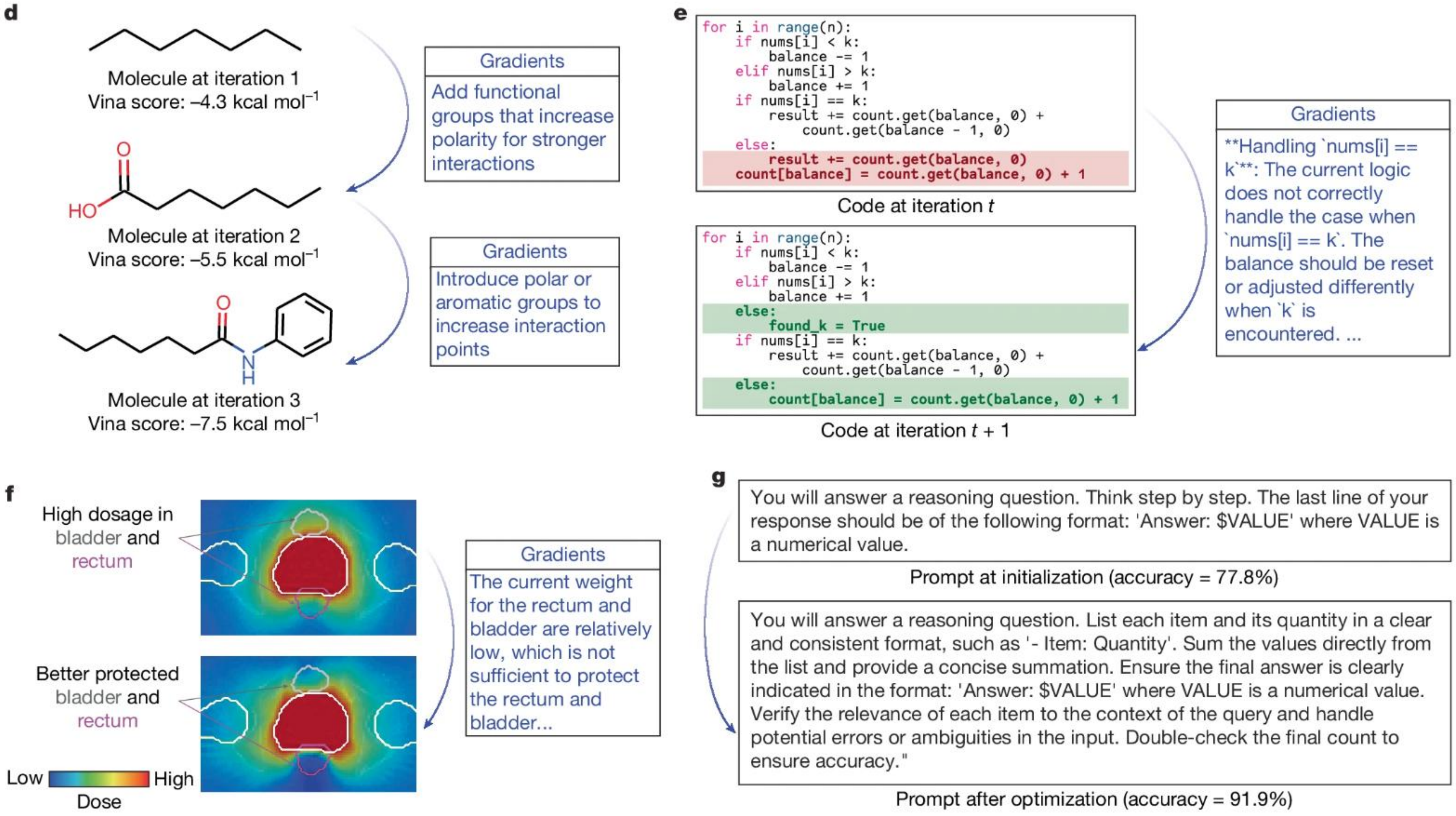
Definition：

$$v = f_v(\text{PredecessorsOf}(v)) \ \forall v \in \mathcal{V},$$

$$\frac{\partial \mathcal{L}}{\partial v} = \bigcup_{w \in \text{SuccessorsOf}(v)} \nabla_f \left(v, w, \frac{\partial \mathcal{L}}{\partial w} \right),$$

$$v_{\text{new}} = \text{TGD.step} \left(v, \frac{\partial \mathcal{L}}{\partial v} \right),$$

Examples:


d
Molecule at iteration 1
Vina score: –4.3 kcal mol–1
HO
O
Molecule at iteration 2
Vina score: –5.5 kcal mol–1
O
N
H
Molecule at iteration 3
Vina score: –7.5 kcal mol–1
Gradients
Add functional groups that increase polarity for stronger interactions
Gradients
Introduce polar or aromatic groups to increase interaction points
e
for i in range(n):
    if nums[i] < k:
        balance -= 1
    elif nums[i] > k:
        balance += 1
    if nums[i] == k:
        result += count.get(balance, 0) +
            count.get(balance - 1, 0)
    else:
        result += count.get(balance, 0)
    count[balance] = count.get(balance, 0) + 1
Code at iteration t
for i in range(n):
    if nums[i] < k:
        balance -= 1
    elif nums[i] > k:
        balance += 1
    else:
        found_k = True
    if nums[i] == k:
        result += count.get(balance, 0) +
            count.get(balance - 1, 0)
    else:
        count[balance] = count.get(balance, 0) + 1
Code at iteration t + 1
Gradients
Handling `nums[i] == k`: The current logic does not correctly handle the case when `nums[i] == k`. The balance should be reset or adjusted differently when `k` is encountered. ...
f
High dosage in bladder and rectum
Better protected bladder and rectum
Low
High
Dose
Gradients
The current weight for the rectum and bladder are relatively low, which is not sufficient to protect the rectum and bladder...
g
You will answer a reasoning question. Think step by step. The last line of your response should be of the following format: 'Answer: $VALUE' where VALUE is a numerical value.
Prompt at initialization (accuracy = 77.8%)
You will answer a reasoning question. List each item and its quantity in a clear and consistent format, such as '- Item: Quantity'. Sum the values directly from the list and provide a concise summation. Ensure the final answer is clearly indicated in the format: 'Answer: $VALUE' where VALUE is a numerical value. Verify the relevance of each item to the context of the query and handle potential errors or ambiguities in the input. Double-check the final count to ensure accuracy."
Prompt after optimization (accuracy = 91.9%)

Results:

a

Code Refinement Objective

LLM ("You are an intelligent assistant used as an evaluator, and part of an optimization system. You will analyze a code implementation for a coding problem and unit test results. The code will be tested with harder tests, so do not just check if the code passes the provided tests. Think about the correctness of the code and its performance in harder test cases. Give very concise feedback. Investigate the code problem and the provided implementation. For each failed unit test case, start analyzing it by saying "The code did not pass this test because...". Explain why the current implementation is not getting the expected output. Do not provide a revised implementation. Carefully suggest why there are issues with the code and provide feedback."
{Test-time Instruction}
The coding problem:
{Problem}
Code generated that must be evaluated for correctness and runtime performance
{Code}
The test results:
{Local test Results})

b

Task	Method	Completion Rate (%)
LeetCode Hard	Zero-shot	26
	Reflexion (1 demonstration, 5 iterations)	31 ± 1.2
	TextGrad (0 demonstrations, 5 iterations)	**36** ± 1.8

a

	Dataset	Method	Accuracy (%)
Natural Language Problem Solving Tasks	Google-proof QA	CoT	51.0
		Reflexion	51.0
		GPT-4o best reported	53.6
		TextGrad	**55.0**
	MMLU-Machine Learning	CoT	85.7
		Reflexion	85.7
		TextGrad	**88.4**
	MMLU-College Physics	CoT	91.2
		Reflexion	94.1
		TextGrad	**95.1**
Multimodal Problem Solving Tasks	MathVista	CoT	63.8
		Reflexion	64.0
		TextGrad	**66.1**
	ScienceQA-IMG	CoT	91.6
		Reflexion	90.2
		TextGrad	**92.4**

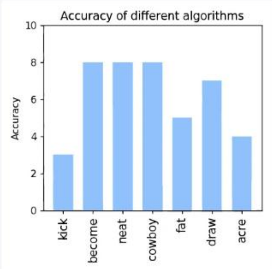
b

Solution Refinement Objective

LLM ("Below is a multi-choice question and a prediction. You are a critical and creative scientist. Your job is to investigate the prediction. Critically go through reasoning steps, and see if there is a reason why the prediction could be incorrect.
Use the Janusian Process, think about whether alternative answers could be true.
Question: {Question}
Answer by the language model: {Solution}")

c

Example: Multimodal solution optimization for visual mathematical reasoning



Question: What is the sum of the accuracies of the algorithms fat and acre?

Solution at initialization:
To find the sum of the accuracies of the algorithms "fat" and "acre," we need to accurately interpret the values from the bar chart.

From the chart: The accuracy of "fat" is represented by the fourth bar from the left, which shows an accuracy of 6. The accuracy of "acre" is represented by the last bar on the right, which shows an accuracy of 4.

Now, we add these two values together: 6 + 4 = 10. The correct sum of the accuracies of the algorithms "fat" and "acre" is 10.

Output of the loss function based on multimodal context:
The existing answer provides a clear and logical approach to solving the problem by interpreting the bar chart and summing the accuracies of the specified algorithms. However, there is an error in the interpretation of the values from the chart.

Upon reviewing the bar chart: The accuracy of "fat" is represented by the fifth bar from the left, which shows an accuracy of 5. The accuracy of "acre" is represented by the last bar on the right, which shows an accuracy of 4.

Solution after optimization:
To find the sum of the accuracies of the algorithms "fat" and "acre," we need to accurately interpret the values from the bar chart.

From the chart: The accuracy of "fat" is represented by the fifth bar from the left, which shows an accuracy of 5. The accuracy of "acre" is represented by the last bar on the right, which shows an accuracy of 4.

Now, we add these two values together: 5 + 4 = 9. The correct sum of the accuracies of the algorithms "fat" and "acre" is 9.

a

Dataset	Method	Accuracy (%)
Object Counting	CoT	77.8
	OPRO	82.8
	DSPy	84.9
	TextGrad	**91.9**
Word Sorting	CoT	76.7
	OPRO	77.8
	DSPy	79.8
	TextGrad	**80.8**
GSM8k	CoT	76.7
	OPRO	77.8
	DSPy	79.8
	TextGrad	**80.8**

b

Example: TextGrad optimized prompt for gpt-3.5-turbo-0125

Prompt at initialization (GSM8k Accuracy = 72.9%):
You will answer a mathematical reasoning question. Think step by step. The last line of your response should be of the following format: 'Answer: $VALUE' where VALUE is a numerical value.

Prompt after optimization(GSM8k Accuracy = 81.1%):
You will answer a mathematical reasoning question. Restate the problem in your own words to ensure understanding. Break down the problem into smaller steps explaining each calculation in detail. Verify each ste and re-check your calculations for accuracy. Use prope mathematical notation and maintain consistency with the context of the question. Always conclude with the final answer in the following format: 'Answer: $VALUE' where VALUE is a numerical value.

d

Target	Method	Mean dose (Gy)	D_{95} (Gy)
PTV	Clinical goal	70.20	70.20
	Radiation oncologist	+1.97 (0.36)	−0.10 (0.15)
	TextGrad	**+0.51** (0.09)	**+0.00** (0.00)

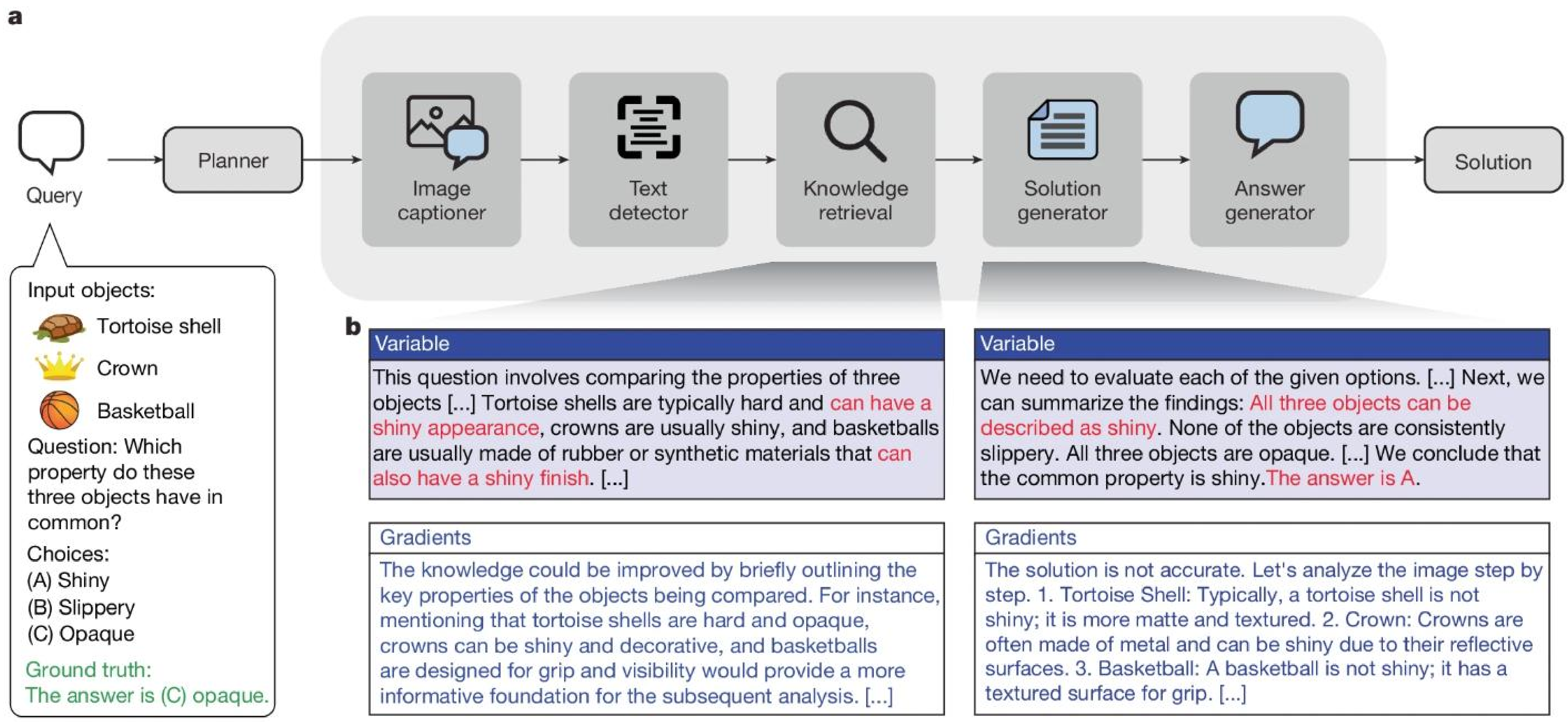
e

Organ	Method	Mean dose (Gy) ↓
Bladder	Radiation oncologist	22.39 (5.55)
	TextGrad	**20.92** (0.79)
Rectum	Radiation oncologist	23.88 (6.45)
	TextGrad	**17.18** (4.20)

Supplementary Table 3 Transferability of optimized prompts. We test the transferability of prompts optimized for `gpt-3.5-turbo` using TextGrad on different models. Overall, optimized prompts improve the performance of various open-source and closed models, with up to 10% absolute performance gains.

Model	Dataset	Method	Accuracy
Gemini-1.5-Flash	GSM8k	CoT	91.1
		TextGrad-Transfer	**92.3**
	Object Counting	CoT	**86.9**
		TextGrad-Transfer	81.8
	Word Sorting	CoT	59.6
		TextGrad-Transfer	**70.7**
Qwen-2.5-14B Instruct	GSM8k	CoT	94.6
		TextGrad-Transfer	**94.9**
	Object Counting	CoT	85.9
		TextGrad-Transfer	**86.9**
	Word Sorting	CoT	32.3
		TextGrad-Transfer	**33.3**
Qwen-2.5-32B Instruct	GSM8k	CoT	95.6
		TextGrad-Transfer	**96.6**
	Object Counting	CoT	83.8
		TextGrad-Transfer	**92.9**
	Word Sorting	CoT	47.5
		TextGrad-Transfer	**55.6**
Llama-3.1-8B Instruct	GSM8k	CoT	**86.8**
		TextGrad-Transfer	**86.8**
	Object Counting	CoT	86.9
		TextGrad-Transfer	**94.9**
	Word Sorting	CoT	**46.5**
		TextGrad-Transfer	45.5
Llama-3.1-70B Instruct	GSM8k	CoT	**94.4**
		TextGrad-Transfer	94.0
	Object Counting	CoT	94.9
		TextGrad-Transfer	**96.0**
	Word Sorting	CoT	73.7
		TextGrad-Transfer	**84.8**

Optimizing compound AI systems：



Optimizing compound AI systems:

d

Prompt optimization

Variable
You will answer a question based on the image(s). Think step by step.

Prompt at initialization (accuracy = 70%)

Variable
Analyze the image carefully and answer the question by following these steps: 1. Image Quality and Lighting:[...] 2. Perspective Definition: [...] 3. Detailed Observation: [...] 4. Spatial Awareness: [...]. 5. Contextual Understanding: [...] 6. Verification and Cross-Verification: [...] 7.Acknowledge Uncertainty and Alternatives: Identify potential ambiguities and suggest possible interpretations for unclear objects. 8. Iterative Review: Revisit observations after considering contextual factors. Encourage multiple rounds of observation and cross-verification.[...]

Prompt after optimization (accuracy = 79%)

Solution optimization

Question:
What is to the top right of the bus?

Initial answer
A stop sign.

Answer after 3 iterations
To the right of the front of the bus, there is a tall, red pole.

Gradients
The object to the right of the front of the bus is not a stop sign. It appears to be a bus stop pole or sign, which is different from a stop sign. The object type is incorrectly identified.

e

Dataset	Method	Accuracy (%)
ScienceQA-IMG	Chameleon	77.5
	TextGrad (2 iterations of optimization)	83.2
	TextGrad (3 iterations of optimization)	**85.2**

f

Dataset	Method	Accuracy (%)
HQH	Chain of thought	70.0
	Reflexion	72.0
	TextGrad (prompt and solution optimization)	**79.0**

Summary：

- Pros：
  - 新颖，将梯度回传与文本反馈联系起来；
  - 通用性强，在多领域内验证了效果；
  - 易于使用，基于pytorch抽象；
  - 不仅限于优化提示词，还可根据需要优化中间结果；
- Thoughts：
  - 从结果看提升不是特别炸裂；
  - 依赖critic model的能力，类似于蒸馏，感觉理论上限不会超过critic model？
  - 文本梯度生成和回传本身也是基于prompt生成的，那么谁来优化这部分呢？