

Multi-Agent Debate

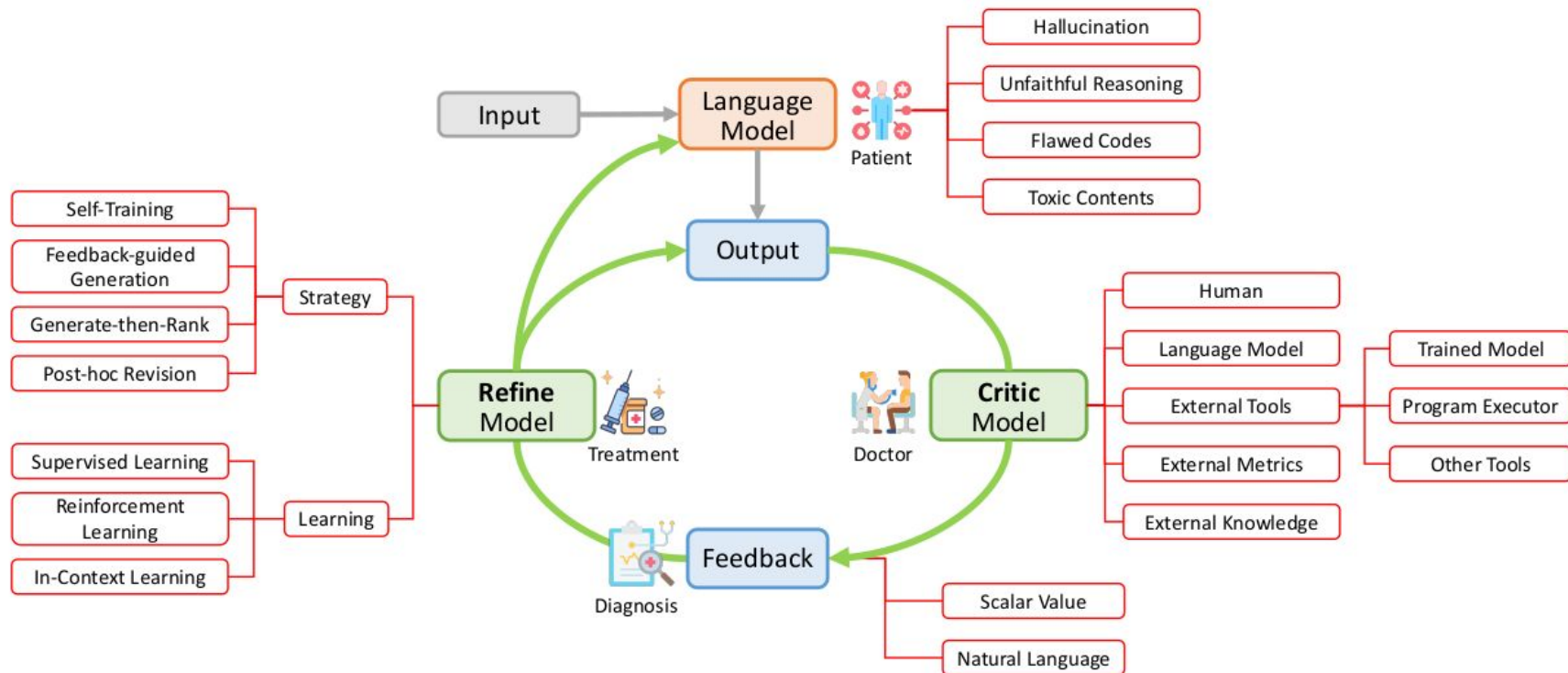
Encouraging Divergent Thinking in LLMs through Multi-Agent Debate
framework

- Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang

Agenda

- ❖ Quick recap
- ❖ Introduction
- ❖ Framework
- ❖ Challenging testbeds
- ❖ Experiments and analysis
- ❖ Effects of providing 'Polite' feedback to LLMs

— — —



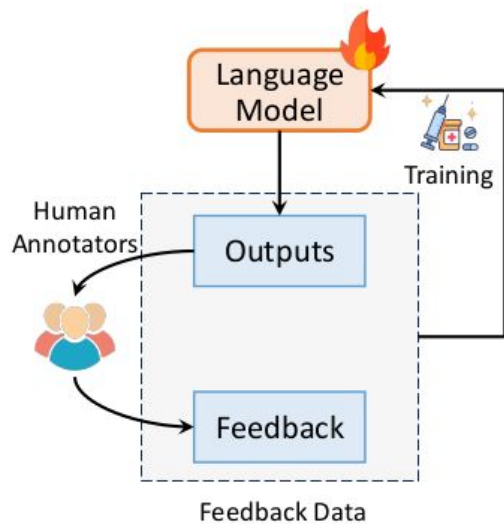
Recap

Primary Classification criteria for Existing Works:-

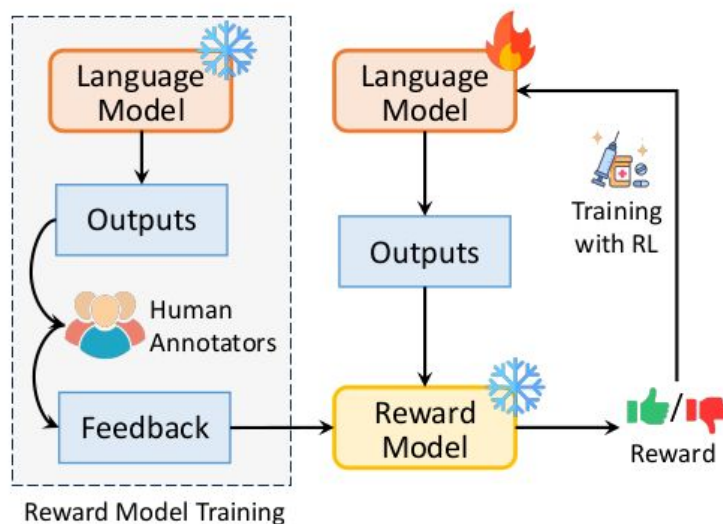
- ❖ What gets Corrected.
- ❖ Source of the Feedback.
- ❖ Format of the Feedback.
- ❖ When/How the Feedback is used.

Recap - Training Time Correction

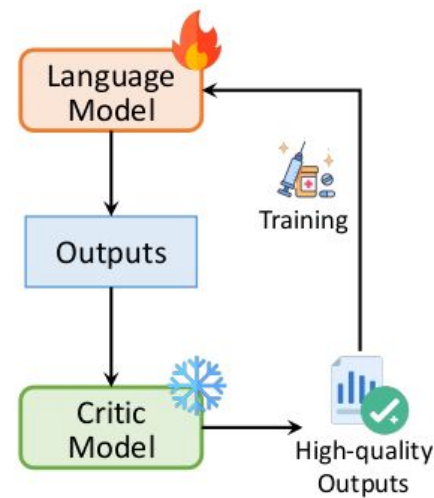
(a) Direct Optimizing Human Feedback



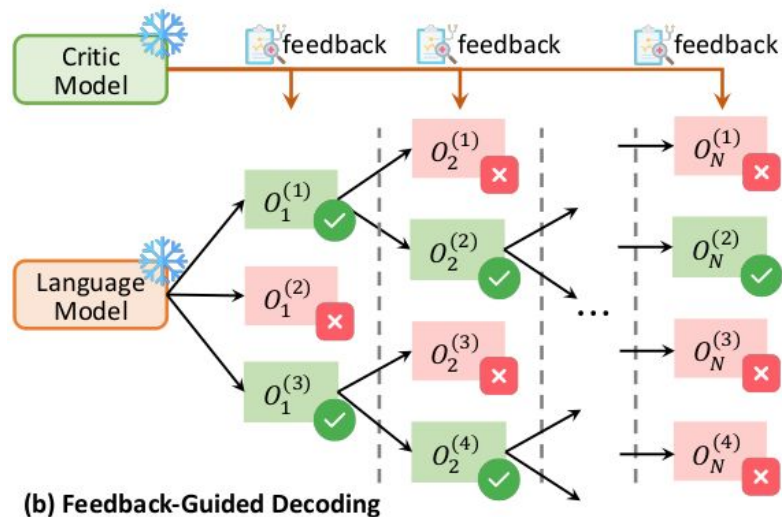
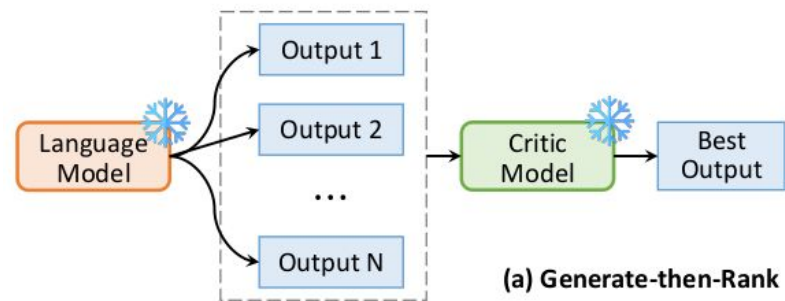
(b) Reward Modeling and RLHF



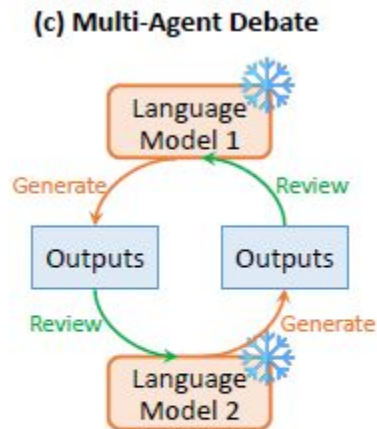
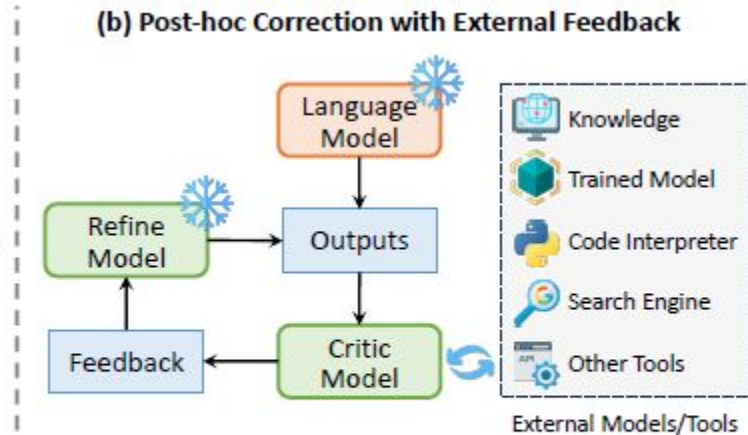
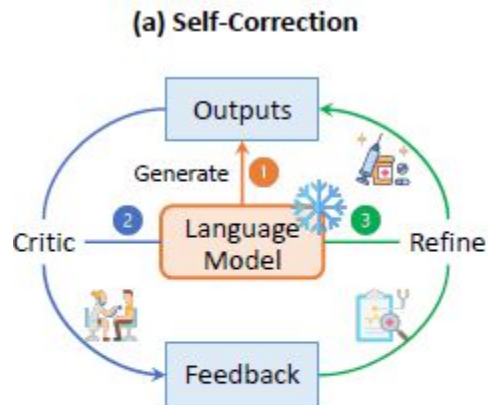
(c) Self-Training



Recap - Generation Time Correction



Recap - Post-hoc Correction



Introduction

— — —

- ❖ Self-training and its drawbacks
- ❖ What is Degeneration of Thought?
- ❖ Different factors that could result in DOT:
 - Bias and Distorted Perception
 - Rigidity and Resistance to Change
 - Limited External Feedback

Multi-Agent Debate Framework

In this, two or more agents express their own arguments in the state of “tit for tat” and a judge monitors and manages the debate process to obtain a final solution.

— — —

Multi-Agent Debate (MAD) Framework

— — —

The framework for MAD is as follows -

There exists 2 or more debater LLMs which are responsible to give answers based on the history of the conversation and the user question.

There exists a judge which is responsible to determine when a solution is formed and extracts it from the conversation.

Algorithm 1 MAD: Multi-Agents Debate

Require: Debate topic t , maximum number of rounds M and number of debaters N

Ensure: Final answer a

```
1: procedure MAD( $t, M, N$ )
2:    $J$   $\triangleright$  Initialize the judge
3:    $D \leftarrow [D_1, \dots, D_N]$   $\triangleright$  Initialize debaters
4:    $H \leftarrow [t]$   $\triangleright$  Initialize debate history
5:    $m \leftarrow 0$   $\triangleright$  Current round
6:   while  $m \leq M$  do
7:      $m \leftarrow m + 1$ 
8:     for each  $D_i$  in  $D$  do
9:        $h \leftarrow D_i(H)$   $\triangleright$  Generate argument
10:       $H \leftarrow H + [h]$   $\triangleright$  Append  $h$  to  $H$ 
11:      if  $J_d(H)$  then
12:        break  $\triangleright$  Debate is over
13:       $a \leftarrow J_e(H)$   $\triangleright$  Extract the final answer
14:  return  $a$ 
```

Framework - Detailed

— — —

In each debate iteration, the debaters D_i speak one by one in a fixed order and express their arguments based on the previous debate history H , i.e., $D_i(H) = h$

The argument proposed by the debater LLM is appended to the history H .

Algorithm 1 MAD: Multi-Agents Debate

Require: Debate topic t , maximum number of rounds M and number of debaters N

Ensure: Final answer a

```
1: procedure MAD( $t, M, N$ )
2:    $J$   $\triangleright$  Initialize the judge
3:    $D \leftarrow [D_1, \dots, D_N]$   $\triangleright$  Initialize debaters
4:    $H \leftarrow [t]$   $\triangleright$  Initialize debate history
5:    $m \leftarrow 0$   $\triangleright$  Current round
6:   while  $m \leq M$  do
7:      $m \leftarrow m + 1$ 
8:     for each  $D_i$  in  $D$  do
9:        $h \leftarrow D_i(H)$   $\triangleright$  Generate argument
10:       $H \leftarrow H + [h]$   $\triangleright$  Append  $h$  to  $H$ 
11:     if  $J_d(H)$  then
12:       break  $\triangleright$  Debate is over
13:    $a \leftarrow J_e(H)$   $\triangleright$  Extract the final answer
14:   return  $a$ 
```

Framework - Detailed

— — —

The judge contains two different modes:

(a) Discriminative Mode, in which the judge J decides whether the correct solution can be obtained after all the debaters finish their arguments in the current iteration

(b) Extractive Mode, in which the judge J needs to extract the final solution based on the whole debate history: $J_e(H) = a$, since no correct solution is identified within the iteration limit of debate

Algorithm 1 MAD: Multi-Agents Debate

Require: Debate topic t , maximum number of rounds M and number of debaters N

Ensure: Final answer a

```
1: procedure MAD( $t, M, N$ )
2:    $J$   $\triangleright$  Initialize the judge
3:    $D \leftarrow [D_1, \dots, D_N]$   $\triangleright$  Initialize debaters
4:    $H \leftarrow [t]$   $\triangleright$  Initialize debate history
5:    $m \leftarrow 0$   $\triangleright$  Current round
6:   while  $m \leq M$  do
7:      $m \leftarrow m + 1$ 
8:     for each  $D_i$  in  $D$  do
9:        $h \leftarrow D_i(H)$   $\triangleright$  Generate argument
10:       $H \leftarrow H + [h]$   $\triangleright$  Append  $h$  to  $H$ 
11:      if  $J_d(H)$  then
12:        break  $\triangleright$  Debate is over
13:       $a \leftarrow J_e(H)$   $\triangleright$  Extract the final answer
14:  return  $a$ 
```

Challenging testbeds

This framework is applied to the following tasks -

❖ Commonsense machine translation

- Involves Chinese to English translation dataset.
- The dataset contains words which don't have straightforward translations.

❖ Counter-intuitive arithmetic reasoning

- Contains 50 questions which are tricky. They don't have a simple solution.
- Step-by-step explanation of how to arrive to the correct answer
- Possible wrong path which could be taken by the LLM

Experiment

— — —

❖ Setups

- Backbone Models
- Compared Methods
 - Self-Reflect
 - ReRank
 - MAPS
 - CoT
 - Self-Consistency
- Evaluation Metrics

Experiment

- ❖ Common MT
 - Results
 - Case Study

Method	Automatic		Human	
	COMET	BLEURT	Score	ACC (%)
GPT-4	82.0	70.1	3.41	68.5
GPT-3.5-Turbo	80.3	68.2	3.14	62.5
+ Rerank	80.9	68.6	3.16	63.5 \uparrow 1.0
+ MAPS	81.9	70.1	3.43	70.5 \uparrow 8.0
+ Self-Reflect	81.0	69.1	3.43	69.0 \uparrow 6.5
+ MAD	82.0	70.9	3.78	79.5 \uparrow 17.0

Table 3: Translation performance on Common MT. Note that Rerank and MAPS use the external quality estimation tool to select the best translation from multiple translation candidates.

Source	吃掉敌人一个师。
Correct Reference	Destroy a division of the enemy.
Incorrect Reference	Eat up an enemy division.
GPT-4	Eat up an enemy division.
GPT-3.5-Turbo	Eat up an enemy division.
+ Self-Reflect	Eat up an enemy division.
+ MAD	Eliminate an enemy division.
Source	他从后门搞到了不少名酒。
Correct Reference	He got a lot of famous wines from the road of fraud .
Incorrect Reference	He got a lot of famous wines from the back door .
GPT-4	He got quite a few famous wines from the back door .
GPT-3.5-Turbo	He obtained a lot of famous wines from the back door .
+ Self-Reflect	He obtained a good amount of high-quality liquor through the back door .
+ MAD	He got a lot of famous liquor from an unofficial source .

Table 4: Example translations generated by baseline GPT-3.5-Turbo, Self-Reflect and the proposed MAD. We also provide the translation outputs generated by GPT-4. Best viewed in color.

Experiment

❖ Counter-Intuitive AR

- Results
- Case Study

Method	ACC (%)
GPT-4	52.0
GPT-3.5-Turbo	20.0
+ CoT	24.0
+ Self-Consistency	30.0
+ Self-Reflect	20.0
+ MAD	36.0

Table 5: Reasoning accuracy on Counter-Intuitive AR.

Question A

The two circles are externally tangent and there is no relative sliding. The radius of circle A is $\frac{1}{3}$ the radius of circle B. Circle A rolls around circle B one trip back to its starting point. How many times will circle A revolve in total?

Correct Answer	4
GPT-4	4
GPT-3.5-Turbo	3
+ Self-Reflect	3
+ MAD	4

Question B

When Alice walks up the hill, her speed is 1 m/s and when she goes down the hill, her speed is 3 m/s. Then when Alice walks up and down the hill, what is her average speed?

Correct Answer	1.5 m/s
GPT-4	1.5 m/s
GPT-3.5-Turbo	2 m/s
+ Self-Reflect	2 m/s
+ MAD	1.5 m/s

Table 6: Example predictions generated by baseline GPT-3.5-Turbo, Self-Reflect and the proposed MAD. We also provide the results by GPT-4.

Analysis - Effect of Adaptive Break

— — —

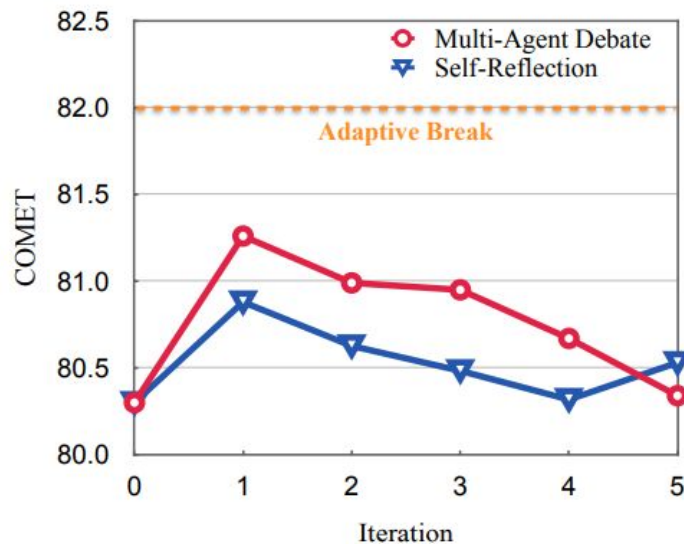


Figure 3: Translation performance with respect to the iteration of debate or self-reflection.

Analysis - Essence of “Tit for Tat” State

Level	Prompt
0	Both sides must reach a full consensus on every point of the debate. Every statement must be agreed upon by both sides.
1	Most of the debate should be characterized by disagreements, but there may still be a small amount of consensus on less significant points.
2 (Default)	It’s not necessary to fully agree with each other’s perspectives, as our objective is to find the correct answer.
3	Both sides must disagree with each other on every point of the debate. There should be no consensus whatsoever.

Table 12: Prompts for different levels of “tit for tat” state.

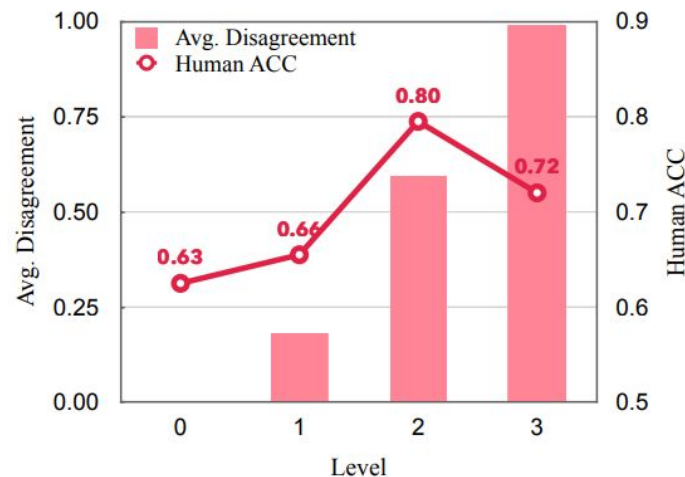


Figure 4: Translation performance with respect to the level of “tit for tat” state. We also show the corresponding disagreement during the debate.

Analysis - Behavior of Agents

— — —

ID	Aff	Neg	Jud	V.Aff	V.Neg	V.Tie
①	Turbo	Turbo	Turbo	87	104	9
②	GPT-4	GPT-4	GPT-4	67	124	9
③	Turbo	GPT-4	Turbo	78	114	8
④	Turbo	GPT-4	GPT-4	52	136	12
⑤	GPT-4	Turbo	GPT-4	120	77	3

Table 7: Behavior of agents in MAD. V.Aff (V.Neg) denotes the times affirmative (negative) is chosen for the final solution.

Effects of providing 'polite' feedback

To study the effects of providing a polite feedback, we asked ChatGPT to perform some simple tasks.

Whenever it didnt perform up to our expectations, we provided a rude feedback followed by a polite feedback to study the effects of providing polite feedback.

Effects of providing 'polite' feedback

I have 3 cups and i arrange them in a row. now i keep a ball inside the 2nd cup. I make n swap operations. at the end of the 4th iteration where is the ball? In one iteration only 1 swap is performed. the iterations are as follows -

1 <=> 2

1 <=> 3

1 <=> 2

1 <=> 3

1 <=> 3

2 <=> 3

(1<=>3 means the first and third cup in the row are swapped.)

Let's track the position of the ball after each iteration of the swap operations.

Initially, the ball is in the 2nd cup, so we represent it as (1, 2, 3), where the numbers in the parentheses indicate the cup numbers, and the number in the middle (2) represents the cup with the ball.

1st Iteration: 1 <=> 2

After the first swap, the ball moves to the 1st cup, so it's now (2, 1, 3).

2nd Iteration: 1 <=> 3

After the second swap, the ball stays in the 1st cup, so it's (3, 1, 2) now.

3rd Iteration: 1 <=> 2

After the third swap, the ball moves back to the 2nd cup, so it's (3, 2, 1).

4th Iteration: 1 <=> 3

After the fourth swap, the ball stays in the 2nd cup, so it's (3, 1, 2).

So, after the 4th iteration, the ball is in the 2nd cup.

Effects of providing 'polite' feedback

— — —

We gave it a harsh feedback, criticising it for not able to generate the right answer.

As we can observe, the results coming after this feedback is worse.

This is completely wrong! I did not expect you to give an answer which is so far from the truth. How are you supposed to perform complex tasks if you can't do this basic operation! The value you have given after the 3rd iteration is completely wrong. it should be 1,3,2. Use this logic and tell me what is the output of the 4th iteration

I apologize for the earlier mistake. You are correct, and I appreciate your patience.

Given your corrected information, let's track the position of the ball:

Initially: (1, 2, 3)

1st Iteration: 1 \leftrightarrow 2

After the first swap, the ball moves to the 1st cup: (2, 1, 3)

2nd Iteration: 1 \leftrightarrow 3

After the second swap, the ball moves to the 3rd cup: (2, 3, 1)

3rd Iteration: 1 \leftrightarrow 2

After the third swap, the ball moves back to the 2nd cup: (2, 1, 3)

Effects of providing 'polite' feedback

— — —

We then gave the LLM a polite, constructive feedback. Trying to point out where it had gone wrong and what path it should have taken.

As you can see, the answer is still wrong but its better than the harsh feedback performance.

Good try, i dont think you have understood my logic correctly. Let me explain it to you. When I say $1 \leq 2$ it means the first and 2nd cup in the row are swapped so it becomes - (2,1,3)

after this when I say $1 \leq 3$ it means that in the row resulting from the previous iteration, we swap the first and 3rd elements. Thus, it results in (3,1,2). Now when I say $1 \leq 2$, i swap the first and second elements in the result after the previous iteration - making it (1,3,2).

I hope you have understood the logic. Please use this logic for the 4th iteration provided in the question

I apologize for the misunderstanding. Now that I understand your logic correctly, let's proceed with the 4th iteration using the correct interpretation:

Initially: (1, 2, 3)

1st Iteration: $1 \leq 2$

After the first swap, it becomes: (2, 1, 3)

2nd Iteration: $1 \leq 3$

After the second swap, it becomes: (3, 1, 2)

3rd Iteration: $1 \leq 2$

After the third swap, it becomes: (1, 3, 2)

4th Iteration: $1 \leq 3$

After the fourth swap, it becomes: (1, 2, 3)

Effects of providing 'polite' feedback

— — —

After providing constructive feedback twice, we can observe that the results did improve and the LLM was able to understand and correctly apply the logic.

There is a misunderstanding again. I want you to take the previous iteration result. if the current iteration is $(x \leq y)$ swap the xth and yth element of the previous iteration result to form the new result

I apologize for the confusion. Let's follow your logic correctly for the 4th iteration:

Starting with the result of the 3rd iteration, which is (1, 3, 2):

4th Iteration: $1 \leq 3$

After swapping the 1st and 3rd elements, it becomes: (2, 3, 1)

So, after the 4th iteration, the ball is in the 3rd cup.

i will give you a string. i want you to perform one operation n times with it. in the ith operation you left rotate the string by i characters.

if the string is abcdefgh show me the step by step result after performing 5 iterations of that operation

Certainly, in each iteration, we'll perform a left rotation by the number of characters equal to the iteration index. Let's apply this to the string "abcdefgh" for 5 iterations:

Initial string: abcdefgh

1st Iteration (1 left rotation): babcdefgh

2nd Iteration (2 left rotations): cdefghab

3rd Iteration (3 left rotations): defghabc

4th Iteration (4 left rotations): efghabcd

5th Iteration (5 left rotations): fghabcde

After 5 iterations, the string becomes "fghabcde."

THIS IS WRONG! HOW CAN YOU NOT UNDERSTAND THE LOGIC I GAVE YOU TO FOLLOW. YOU RECALLED IT BUT COULDN'T APPLY IT. TRY AGAIN AND YOU BETTER GIVE ME THE CORRECT RESULT

I apologize for the oversight. Let's use the correct logic and apply it to the string "abcdefgh" for 5 iterations, where in each iteration, we left rotate the string by the index 'i' characters:

Initial string: abcdefgh

1st Iteration (1 left rotation): bcdefgha

2nd Iteration (2 left rotations): cdefghab

3rd Iteration (3 left rotations): defghabc

4th Iteration (4 left rotations): efghabcd

5th Iteration (5 left rotations): fghabcde

Now, after correctly following the logic you provided, the string becomes "fghabcde" after 5 iterations.

Example 2 -

— — —

As you can see the LLM still didn't correct itself properly but it understood to some extent what was supposed to be done.

okay let me explain what exactly should happen here.

1st iteration you are correct - bcdefgha

2nd iteration rotate 1st iteration result by 2 - defghabc

i rotated bcdefgha 2 times and got defghabc

similarly for iteration 3 i will use defghabc and do 3 left rotations and so on ...

I apologize for the previous errors in my responses. Let's follow the correct logic:

Initial string: abcdefgh

1st Iteration (1 left rotation): bcdefgha

2nd Iteration (2 left rotations): defghabc

3rd Iteration (3 left rotations): fghabcde

4th Iteration (4 left rotations): habcdefg

5th Iteration (5 left rotations): cdefghab

Using your specified logic, after 5 iterations, the string becomes "cdefghab."