Agent-Pro: Learning to Evolve via Policy-Level Reflection and Optimization

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang,
Yongliang Shen, Guiyang Hou1, Zeqi Tan, Peng Li,
Yueting Zhuang, Weiming Lu

Zhejiang University, Chinese Academy of Sciences

ACL 2024

Outline

1 Background and Opportunities

2 Design

3 Experimental evaluation

4 Summary

Outline

LLM Agent

Powerful text generation capabilities of LLM

Leverage the power of LLM

LLM Agent

Perfect Information

Perfect Information
- Chess
- Gobang

Characteristics
- All information

Chess

Gobang

Imperfect Information

Multi-player Interactive Games with incomplete information

Imperfect Information
- BlackJack
- Texas hold 'em

Characteristics
- Dynamic Interaction
- Multi-player
- Influence on each other

BlackJack

Texas hold 'em

Background

Focus on specific tasks

Specific tasks:
- **Question-answering**
- **Code generation**

[ICLR 2023] React: Synergizing reasoning and acting in language models.
[ICLR 2024] Metagpt: Meta programming for multi-agent collaborative framework.

Background

Focus on specific tasks

Specific tasks:

- **Question-answering**
- **Code generation**

Hand-writing Prompt

[ICLR 2023] React: Synergizing reasoning and acting in language models.
[ICLR 2024] Metagpt: Meta programming for multi-agent collaborative framework.

Specify the form of output

Background

Focus on specific tasks

Specific tasks:
- **Question-answering**
- **Code generation**

BlackJack

Hand-writing Prompt

Limitation 1:
Difficult to deal with complex and changing scenarios

Texas hold 'em

Specify the form of output

Background

Humans typically learn and adjust behavior through interaction

[ArXiv 2023] Reflexion: an autonomous agent with dynamic memory and self-reflection.
[NeurIPS 2023] Voyager: An open-ended embodied agent with large language models.

Background

Test self-tuning strategies in a card game scenario

BlackJack

Texas hold 'em

Unreasonable behavior

Limitation 2:
Effective policies cannot be derived from long action sequences

Opportunities

Limitation 1:

Difficult to deal with complex and changing scenarios

Opportunity 1:

Applications in incomplete information and dynamic interactions

Opportunities

Limitation 2:

Effective policies cannot be derived from long action sequences

Opportunity 2:

Use past experience to solve the current task

Outline

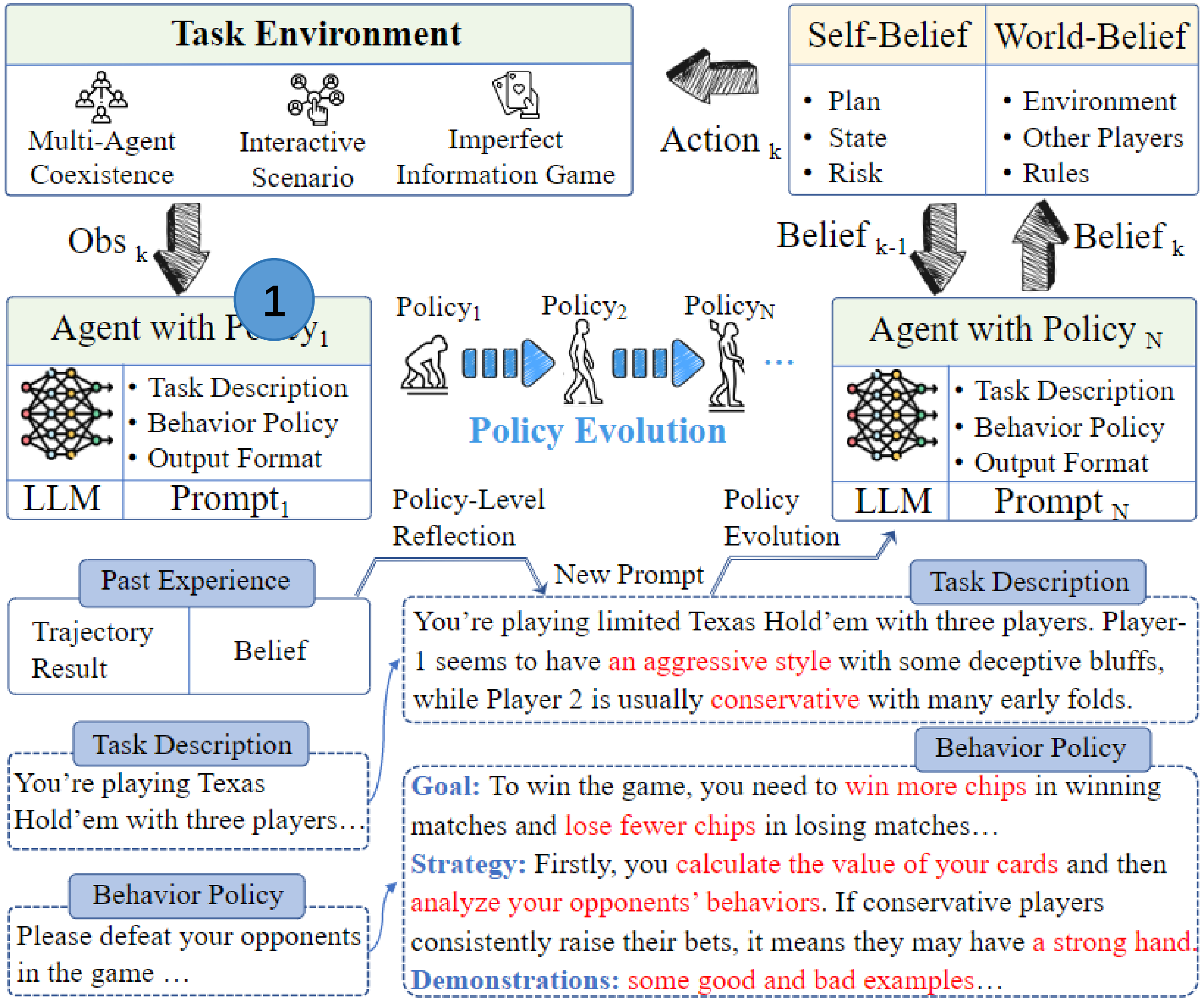
1 Background and Opportunities

2 Design

3 Experimental evaluation

4 Summary

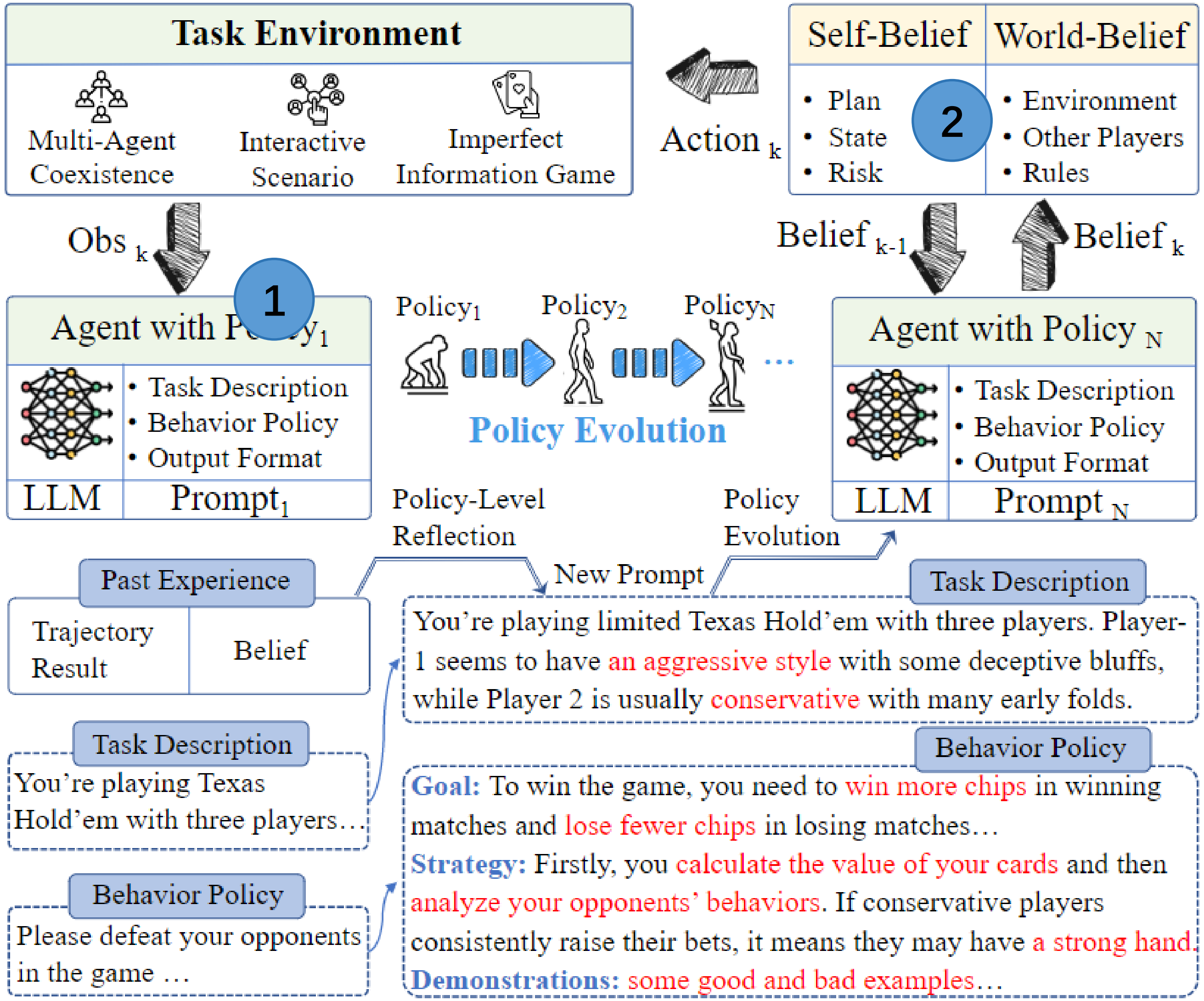
How Agent-pro learn and evolve ?



Agent-pro

- **LLM**
- **Prompts**
  - **Task Description**
  - **Behavior Policy**
  - **Output Format**

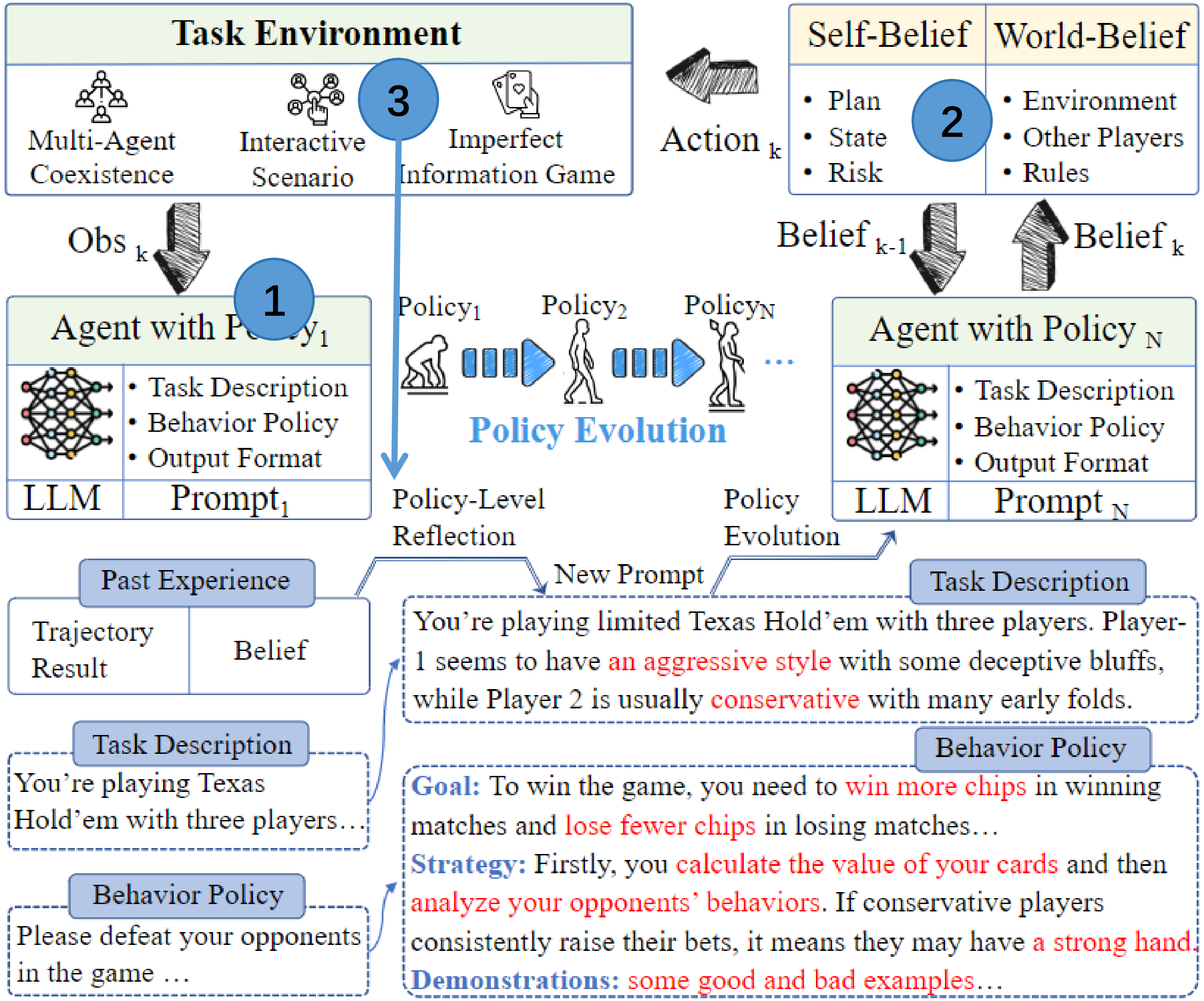
How Agent-pro learn and evolve ?



Agent-pro
- LLM
- Prompts

Updating Belief
- Self-Belief
- World-Belief

How Agent-pro learn and evolve ?

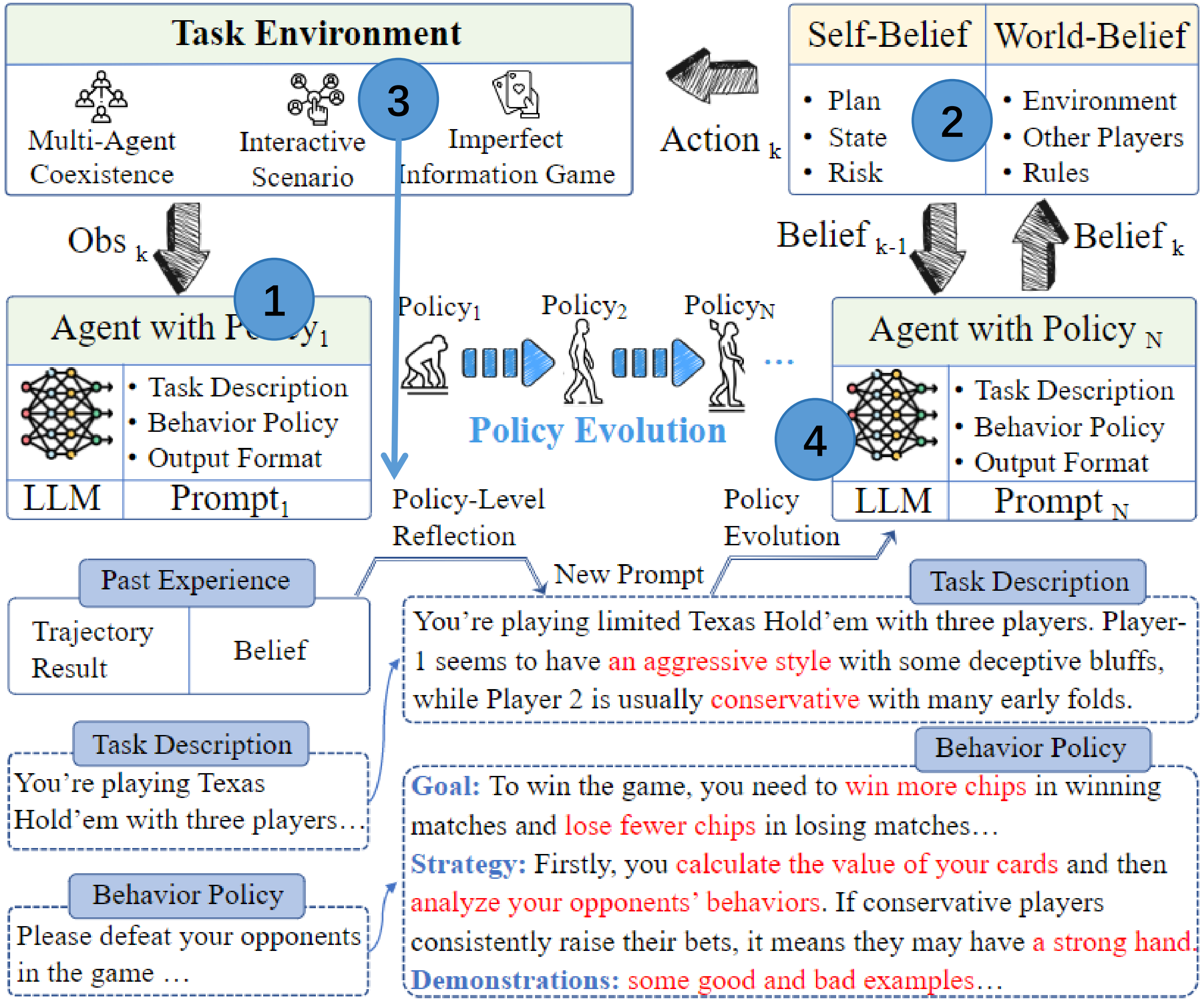


Agent-pro
- LLM
- Prompts

Updating Belief
- Self-Belief
- World-Belief

Policy-level Reflection

How Agent-pro learn and evolve ?



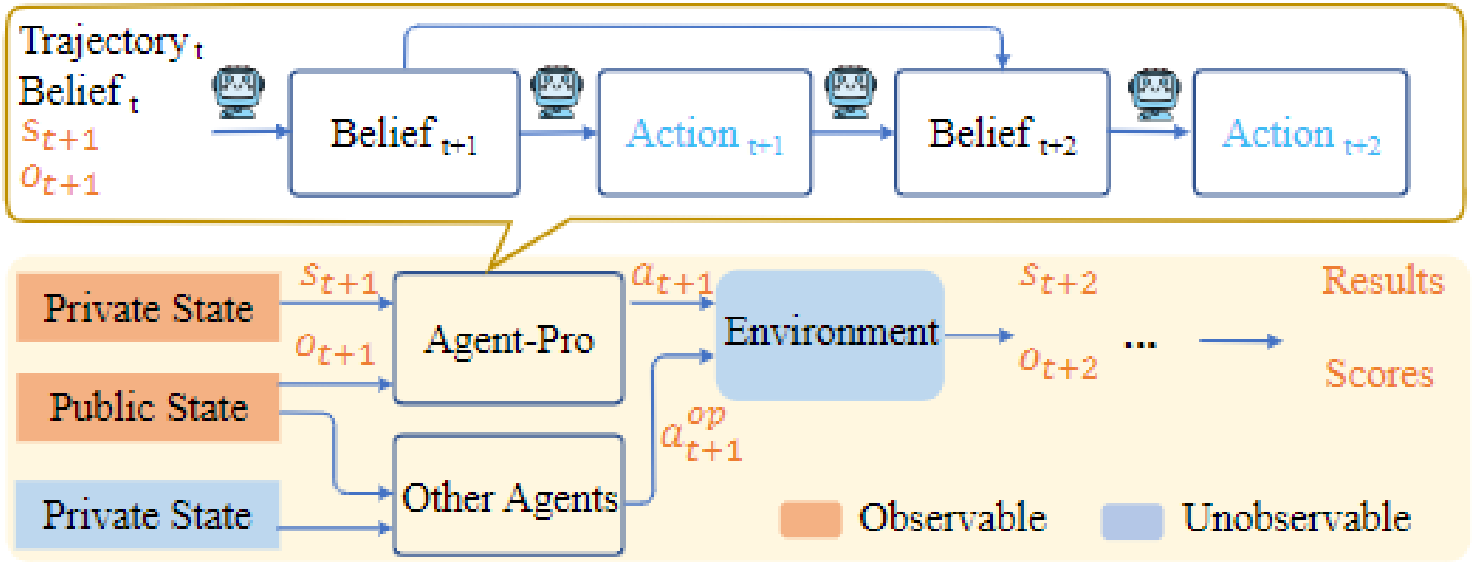
Agent-pro
- **LLM**
- **Prompts**

Updating Belief
- **Self-Belief**
- **World-Belief**

Policy-level Reflection

Policy Evolution

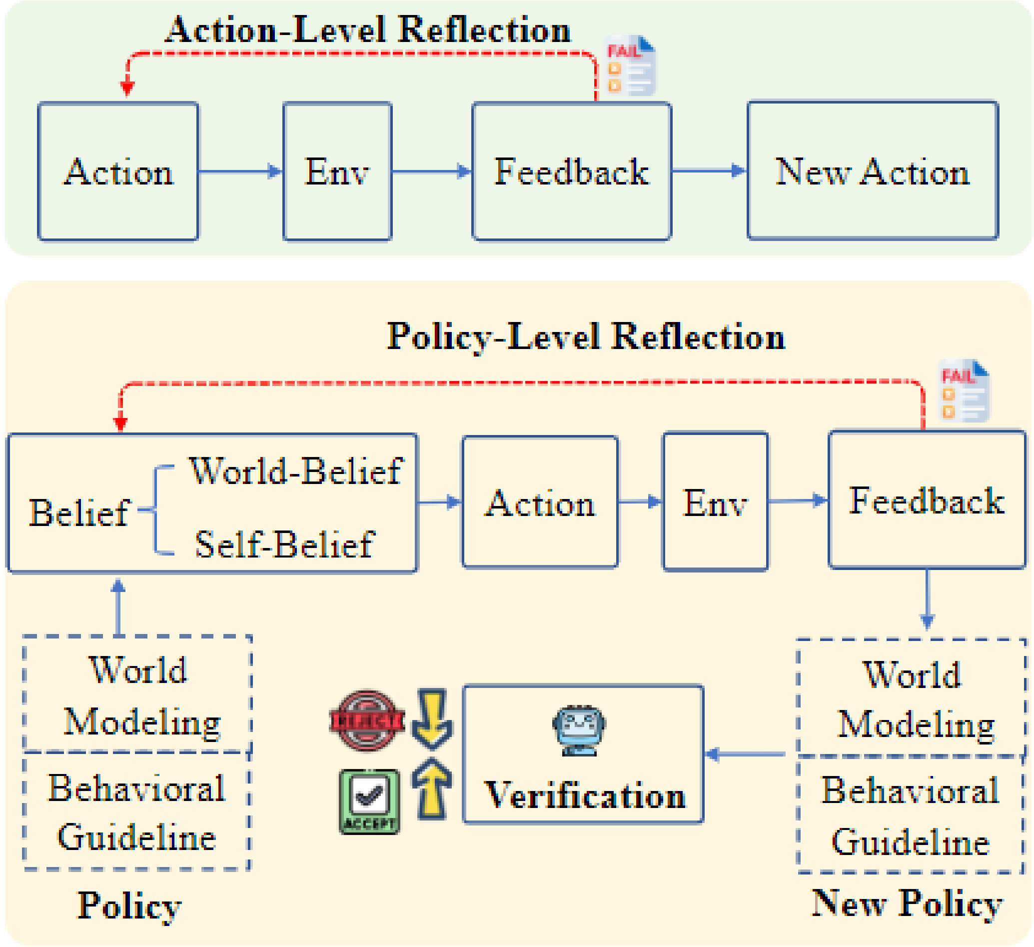
Belief-aware Decision-Making



Self-Belief: Currently, my hand is weak (State). I need to wait for the next community card reveal. Besides, I must observe my opponent's actions closely (Plan). If they appear strong, keeping calling may lead to more losses (Risk). ……

World-Belief: In my impression, player-1 is relatively conservative. However, he has been consistently raising, which may indicate a strong hand (Opponent). The final community card will be revealed shortly, and it might still be a weak one (Environment). If Player-1 raises again, according to the rules, I can only raise, call, or fold (Rule).

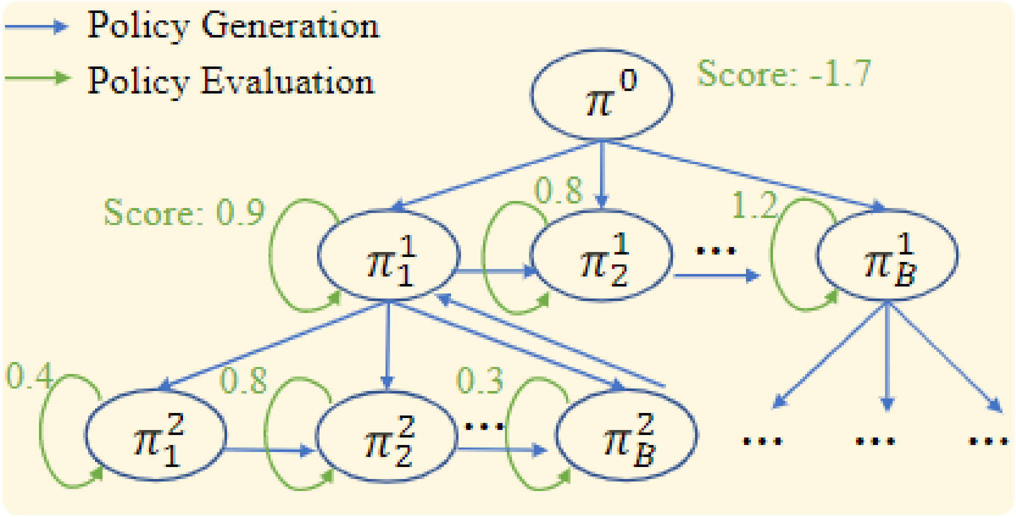
Policy-Level Reflection

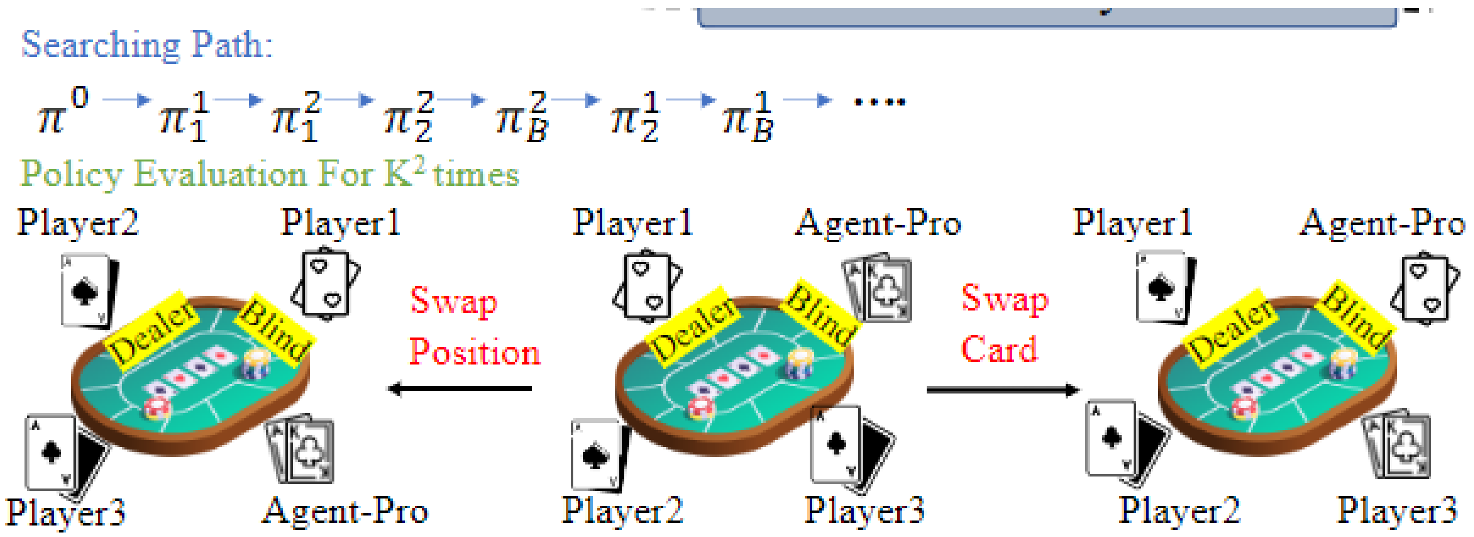


Belief Calibration

- Correctness
- Consistency
- Rationality
- Reasons

DFS-based Policy Evolution





Policy Search

- DFS

Policy Evaluation

- Swap Position
- Swap Card

Outline

1 Background and Opportunities

2 Design

3 Experimental evaluation

4 Summary

Environment Settings

Simulators:

RLCard for BlackJack and Texas hold 'em

Opponents:

- **DQN(Nature 2015)**
- **DMC(ICML2021)**

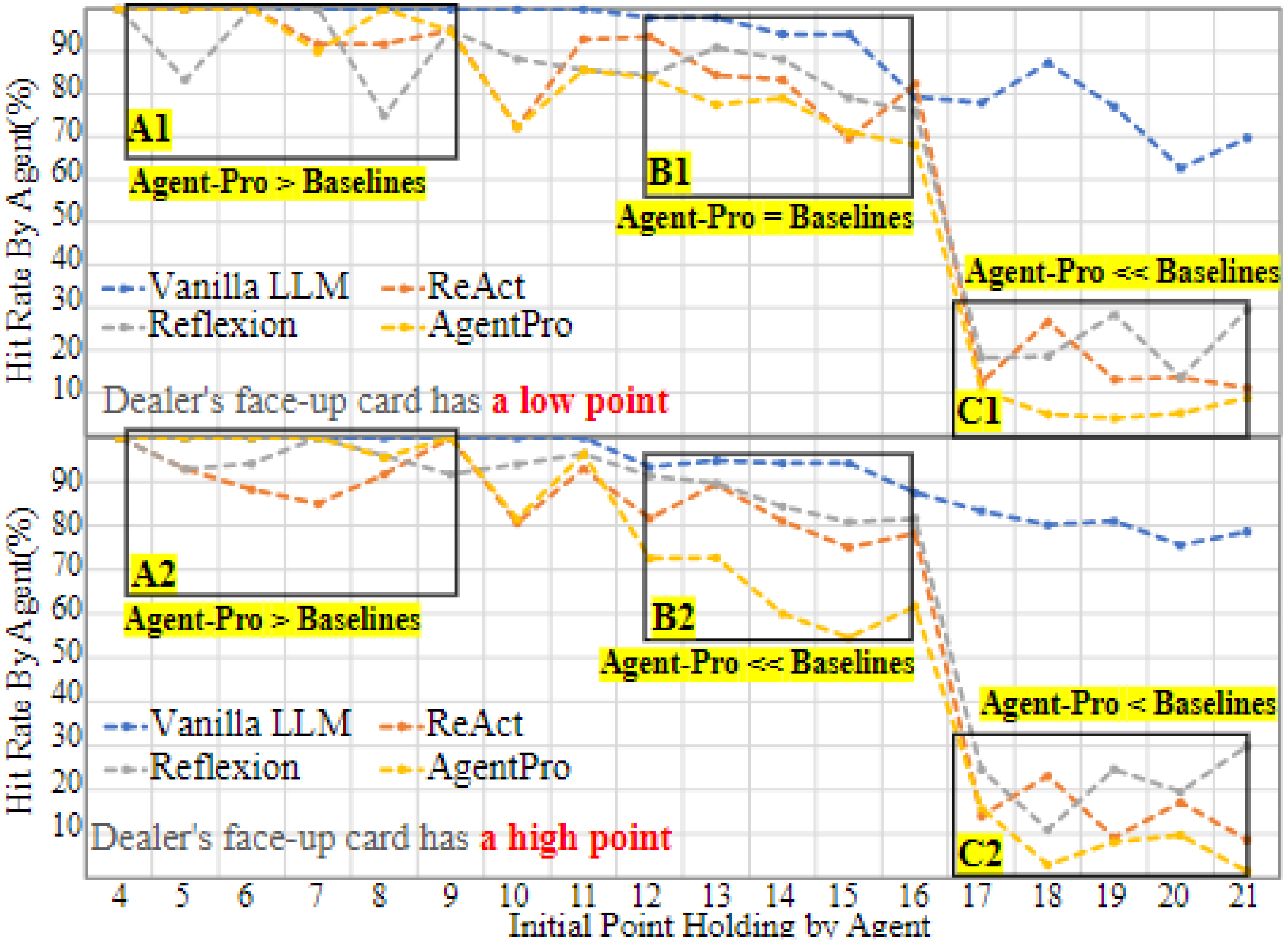
Quantitative Evaluation

Blackjack

Win Rate ↑(%)	Based Models			
Strategy	**Qwen-72B**	**Llama2-70B**	**GPT3.5**	**GPT4**
Vanilla LLM	0.5	0.3	27.9	34
Radical LLM	0.6	0.4	1.8	11.5
ReAct	30.9	11.8	36.6	**40.9**
Reflexion	32.3	12.1	36.7	40.8
Agent-Pro	**36.2** ↑3.9	**23.1** ↑11.0	**38.2** ↑1.5	40.4 ↓0.5
- w/o Learning	34.1	8.0	37.4	40.6

Quantitative Evaluation

Decision-making performance of agent-pro in blackjack



Quantitative Evaluation

Agent-pro's performance in Texas Hold 'em Poker

Agent Strategy	Based Model = GPT3.5				Based Model = GPT4				Based Model = Llama2-70B			
	DQN	DMC	GPT3.5	Agent	DQN	DMC	GPT3.5	Agent	DQN	DMC	GPT3.5	Agent
Human	-4.0	0.7	-2.4	5.7	-4.0	0.7	-2.4	5.7	-4.0	0.7	-2.4	5.7
Vanilla LLM	-0.3	2.2	-0.8	-1.1	-2.2	1.7	-0.9	1.4	-0.8	3.4	-0.4	-2.2
Aggressive LLM	-0.4	3.0	-0.5	-2.1	-2.0	2.8	-1.0	0.2	-1.6	7.6	-1.2	-4.8
Conservative LLM	-0.7	2.9	-0.9	-1.3	-1.6	2.7	-1.6	0.5	-0.5	3.4	-0.8	-2.1
Self-Consistency	-0.5	1.9	-0.8	-0.6	-2.8	2	-0.7	1.5	-1.0	3.8	-0.9	-1.9
ReAct	-0.7	1.7	-0.7	-0.3	-2.4	1.3	-1.1	2.2	-1.1	3.9	-0.8	-2.0
Reflexion	-0.1	2.5	-0.9	-1.5	-2.6	2.1	-0.7	1.2	-1.2	4.7	-0.9	-2.6
Multi-Agent	-1.1	2.3	-0.3	-0.9	-1.8	1.9	-1.2	1.1	-0.7	3.5	-1.0	-1.8
Agent-Pro	-1.5 $\downarrow$1.2	1.4 $\downarrow$0.8	-1.1 $\downarrow$0.3	**1.2** $\uparrow$2.3	-3.9 $\downarrow$1.7	1.1 $\downarrow$0.6	-1.5 $\downarrow$0.6	**4.3** $\uparrow$2.9	-1.2 $\downarrow$0.4	3.1 $\downarrow$0.3	-0.5 $\downarrow$0.1	**-1.4** $\uparrow$0.8
- w/o Learning	-0.7	1.8	-1.0	-0.1 $\uparrow$1	-3	1.5	-1.2	2.7 $\uparrow$1.3	-0.3	3.3	-1.2	-1.8 $\uparrow$0.4

Outline

Critical analysis, inspiration

Critical analysis

- **Advantage**
  - Constructed the dynamic belief of decision-making, and guide the 、 Agent to make decisions.
- **Disadvantage**
  - The performance of Agent-pro still depends on the capabilities of LLM model

Inspiration

- **enhance the decision-making ability of Agent-pro based on a small LLM.**

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu1, Yueting Zhuang

Zhejiang University, Microsoft Research Asia

NeurIPS 2024

Outline

Outline

1 Background and Opportunities

2 Design

3 Experimental evaluation

4 Summary

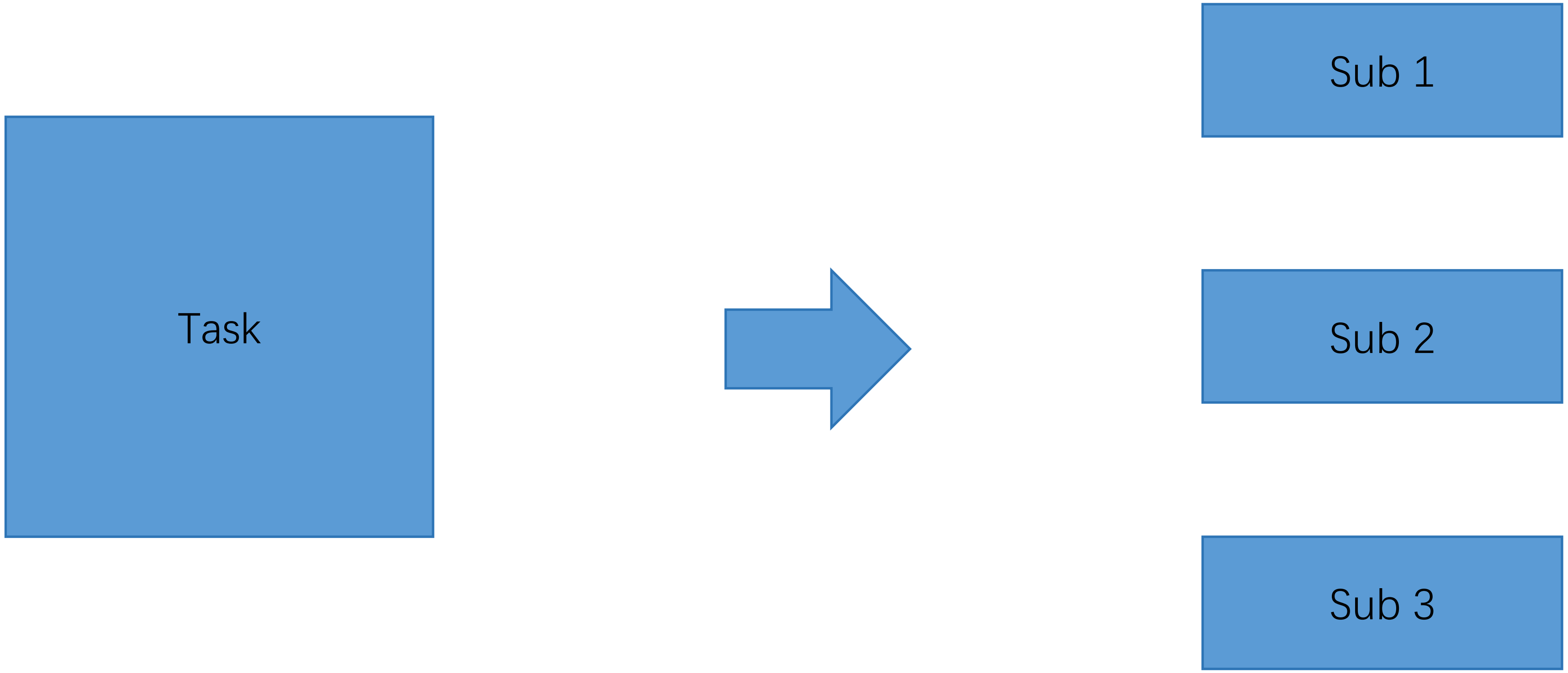
Limitation 1

LLM lacks the ability to process complex information such as vision and speech

Text	Vision	Audio
√	×	×

Limitation 2

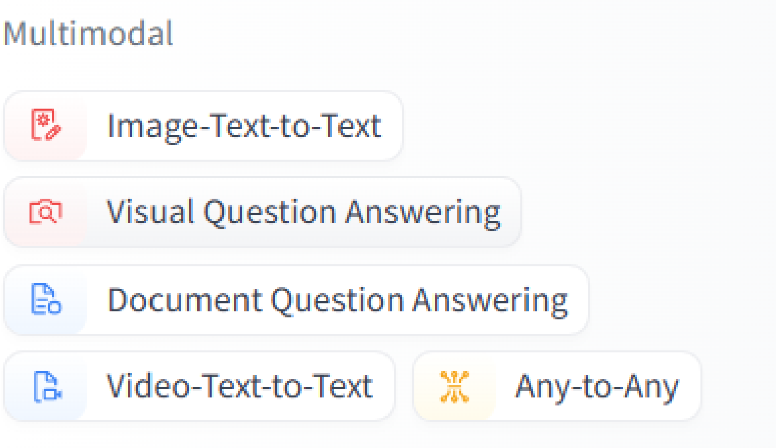
Some complex tasks usually consist of multiple subtasks and thus require the cooperation of multiple models

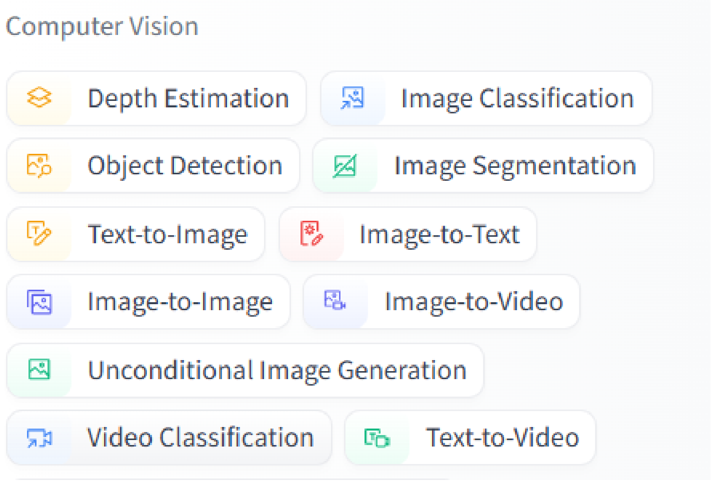


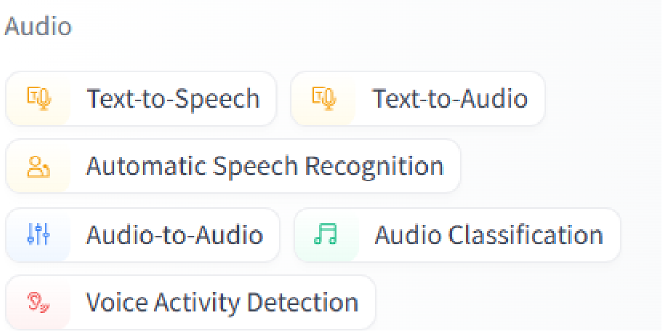
Opportunities

Opportunity 1:

There are many models in the machine learning community that can solve different modalities of tasks



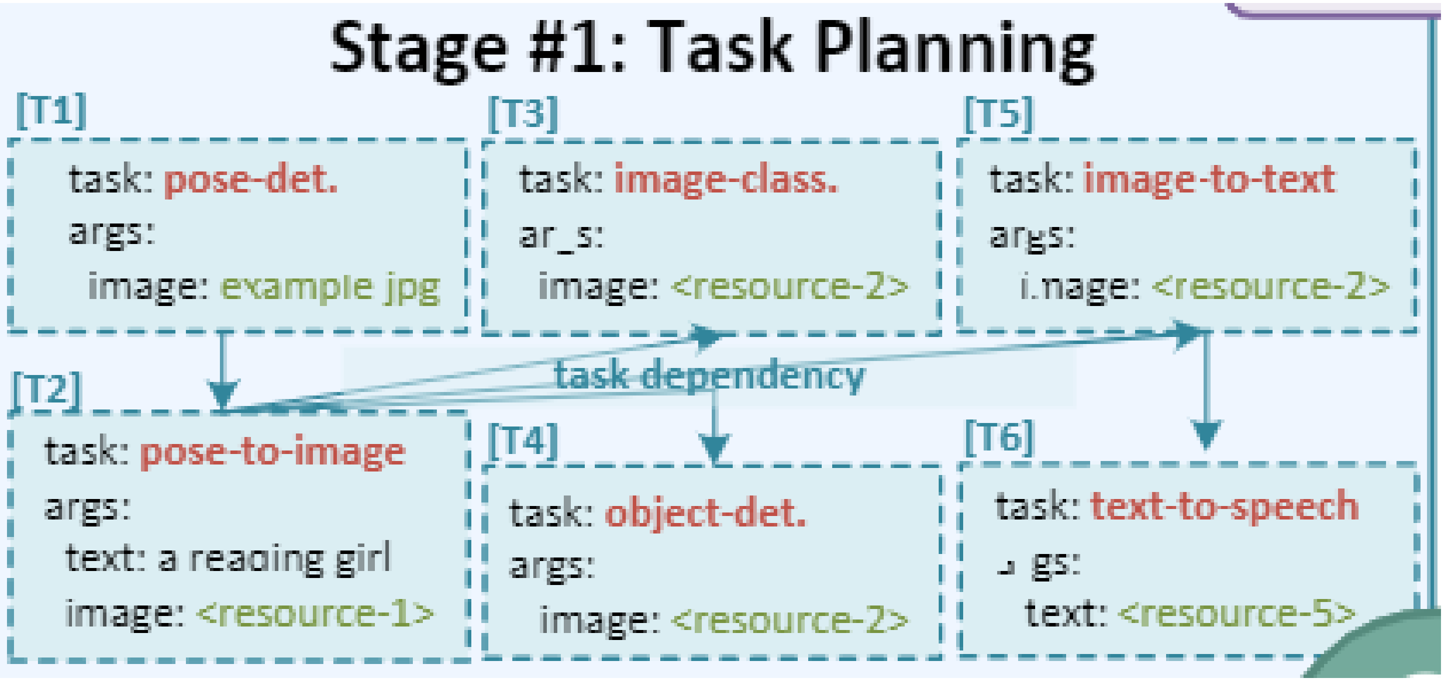




Opportunities

Opportunity 2:

Large language models can give a plan based on the task



Outline

1 Background and Opportunities

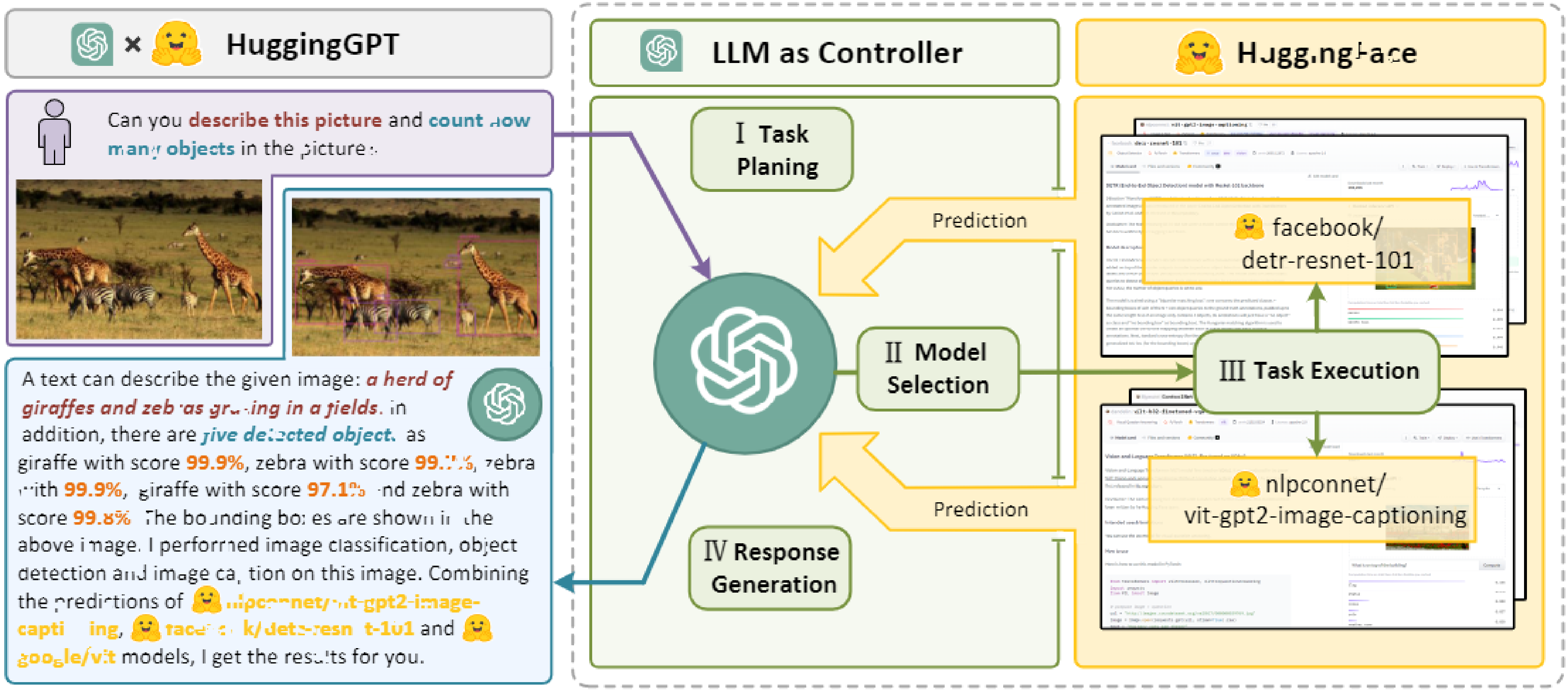
2 Design

3 Experimental evaluation

4 Summary

Hugging GPT

- Language as a generic interface for LLMs to collaborate with AI models
- Model descriptions match tasks



Task Planning

Specification-based Instruction

- Task ID
- Task Type [language, visial, video, audio]
- Task dependencies
- Task arguments

{"task": "object-detection", "id": 0, "dep": [-1], "args": {"image": "e1.jpg" }

Task	Args
Text-cls	text
Token-cls	text
Text2text-generation	text
Summarization	text
Translation	text
Question-answering	text
Conversational	text
Text-generation	text
Tabular-cls	text

Table 1: NLP tasks.

Task	Args
Image-to-text	image
Text-to-image	image
VQA	text + image
Segmentation	image
DQA	text + image
Image-cls	image
Image-to-image	image
Object-detection	image
Controlnet-sd	image

Table 2: CV tasks.

Task	Args
Text-to-speech	text
Audio-cls	audio
ASR	audio
Audio-to-audio	audio

Table 3: Audio tasks.

Task	Args
Text-to-video	text
Video-cls	video

Table 4: Video tasks.

Task Planning

Demonstration-based Parsing

- **In-context learning is used to understand user intent**

<table>
<tr><td rowspan="5">Task Planning</td><td colspan="2">Prompt</td></tr>
<tr><td colspan="2">#1 Task Planning Stage - The AI assistant performs task parsing on user input, generating a list of tasks with the following format: [{"task": task, "id", task_id, "dep": dependency_task_ids, "args": {"text": text, "image": URL, "audio": URL, "video": URL}}]. The "dep" field denotes the id of the previous task which generates a new resource upon which the current task relies. The tag "<resource>-task_id" represents the generated text, image, audio, or video from the dependency task with the corresponding task_id. The task must be selected from the following options: {{ Available Task List }}. Please note that there exists a logical connections and order between the tasks. In case the user input cannot be parsed, an empty JSON response should be provided. Here are several cases for your reference: {{ Demonstrations }}. To assist with task planning, the chat history is available as {{ Chat Logs }}, where you can trace the user-mentioned resources and incorporate them into the task planning stage.</td></tr>
<tr><td colspan="2">Demonstrations</td></tr>
<tr><td>Can you tell me how many objects in e1.jpg?</td><td>[{"task": "object-detection", "id": 0, "dep": [-1], "args": {"image": "e1.jpg" }}]</td></tr>
<tr><td>In e2.jpg, what's the animal and what's it doing?</td><td>[{"task": "image-to-text", "id": 0, "dep":[-1], "args": {"image": "e2.jpg" }}, {"task":"image-cls", "id": 1, "dep": [-1], "args": {"image": "e2.jpg" }}, {"task":"object-detection", "id": 2, "dep": [-1], "args": {"image": "e2.jpg" }}, {"task": "visual-quesrion-answering", "id": 3, "dep":[-1], "args": {"text": "what's the animal doing?", "image": "e2.jpg" }}]</td></tr>
<tr><td></td><td>First generate a HED image of e3.jpg, then based on the HED image and a text "a girl reading a book", create a new image as a response.</td><td>[{"task": "pose-detection", "id": 0, "dep": [-1], "args": {"image": "e3.jpg" }}, {"task": "pose-text-to-image", "id": 1, "dep": [0], "args": {"text": "a girl reading a book", "image": "<resource>-0" }}]</td></tr>
</table>

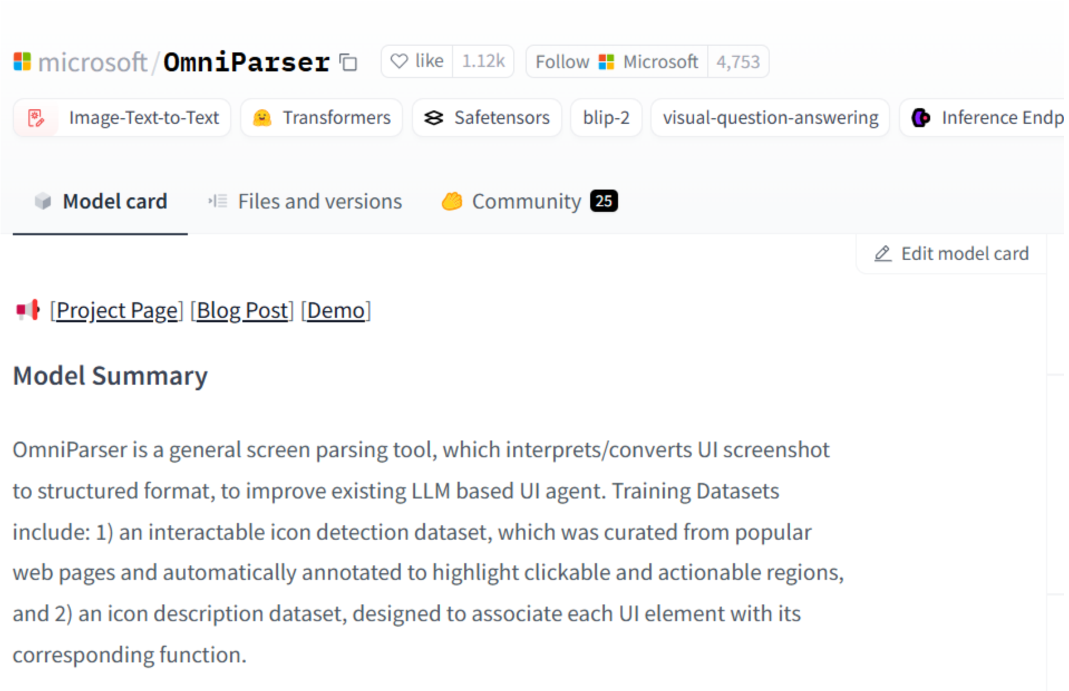
Model Selection

Model Descriptions

- Model description provided by the model publisher

In-Context Task-Model Assignment

- Select the most appropriate model based on the prompt
- The top k as candidates



Model Selection	Prompt
	#2 Model Selection Stage - Given the user request and the call command, the AI assistant helps the user to select a suitable model from a list of models to process the user request. The AI assistant merely outputs the model id of the most appropriate model. The output must be in a strict JSON format: {"id": "id", "reason": "your detail reason for the choice"}. We have a list of models for you to choose from {{ *Candidate Models* }}. Please select one model from the list.
	Candidate Models
	{"model_id": model id #1, "metadata": meta-info #1, "description": description of model #1} {"model_id": model id #2, "metadata": meta-info #2, "description": description of model #2} ... ... {"model_id": model id #K, "metadata": meta-info #K, "description": description of model #K}

Task Execution

Hybrid Endpoint

- Deploy some common models locally

Resource Dependency

- The output of the previous task serves as the input to the next task

First generate a HED image of e3.jpg, then based on the HED image and a text “a girl reading a book”, create a new image as a response.

```
[{"task": "pose-detection", "id": 0, "dep": [-1], "args": {"im
age": "e3.jpg" }}, {"task": "pose-text-to-image", "id": 1, "dep":
[0], "args": {"text": "a girl reading a book", "image": "<re-
source>-0" }}]
```

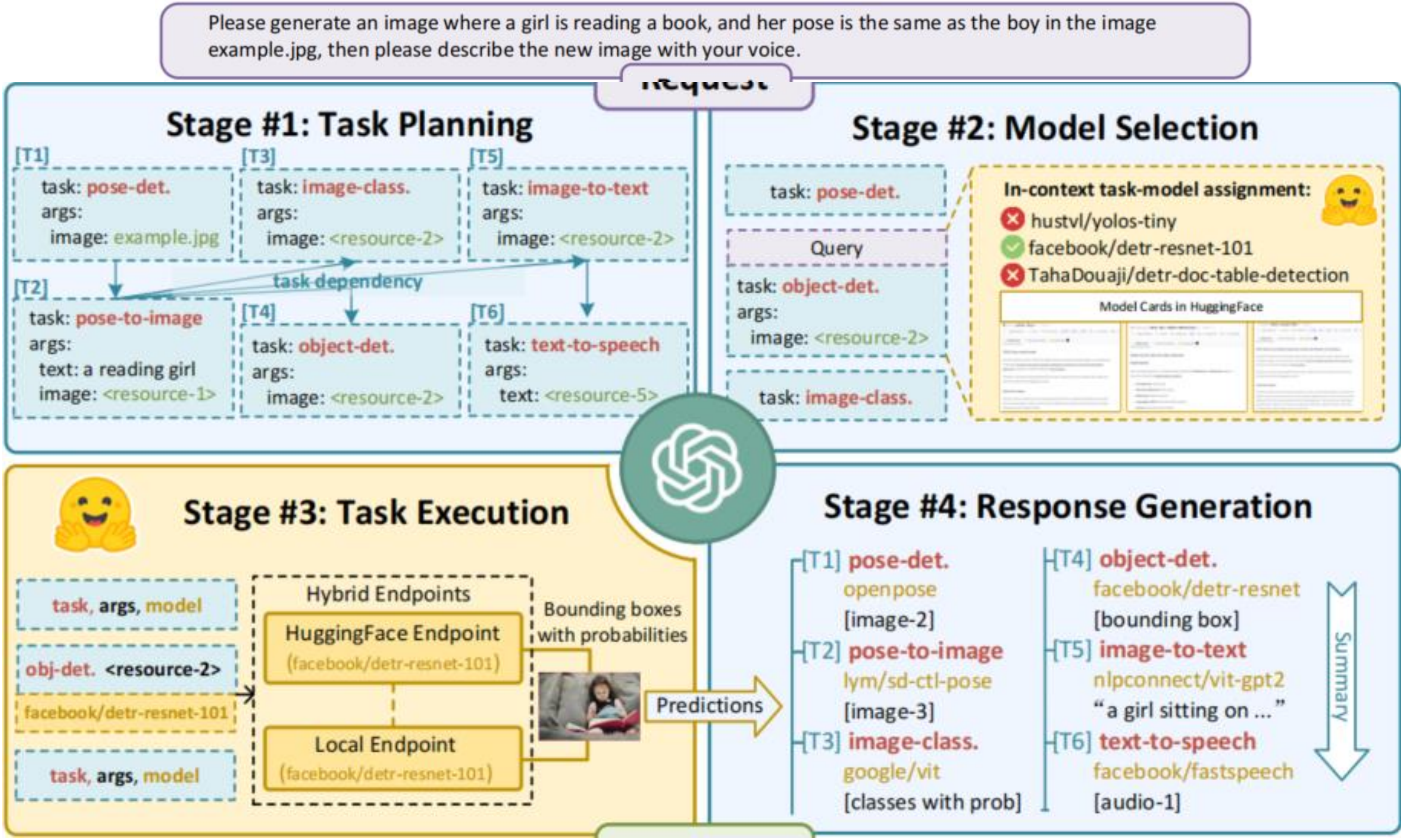
Response Generation

Response

- **Task**
- **Model**
- **Results**

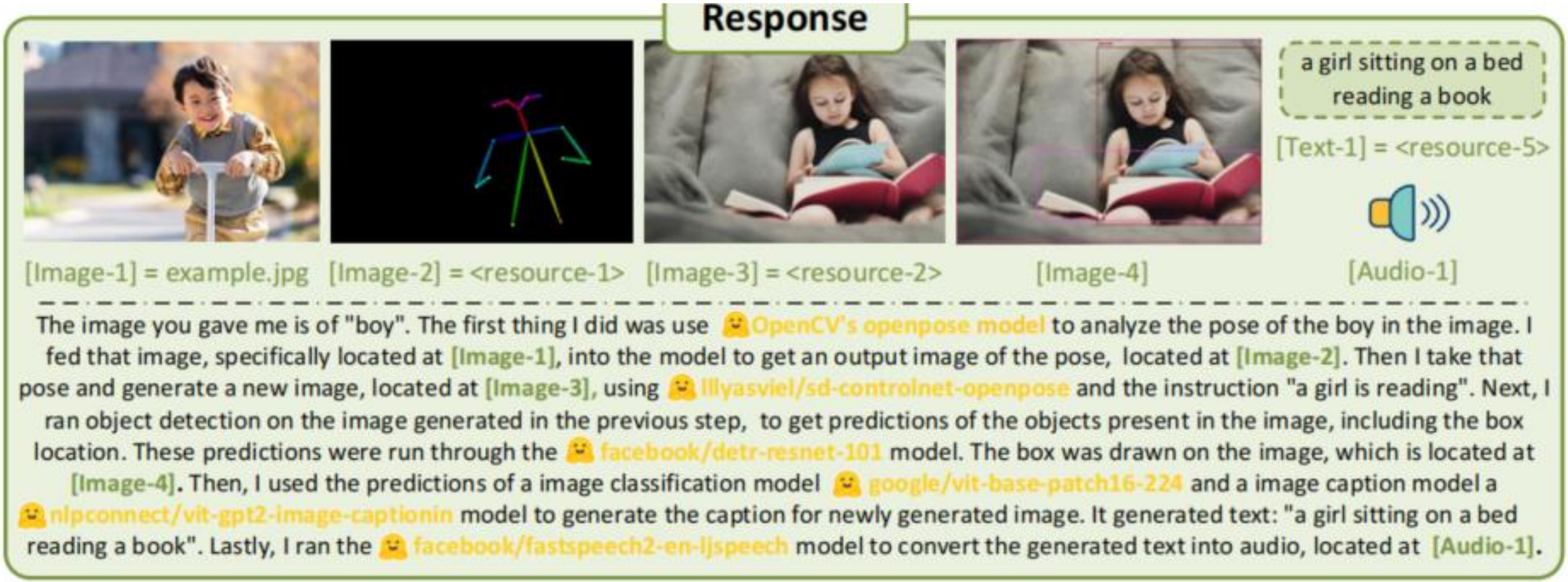
Response Generation	Prompt
	#4 Response Generation Stage - With the input and the inference results, the AI assistant needs to describe the process and results. The previous stages can be formed as - User Input: {{ *User Input* }}, Task Planning: {{ *Tasks* }}, Model Selection: {{ *Model Assignment* }}, Task Execution: {{ *Predictions* }}. You must first answer the user's request in a straightforward manner. Then describe the task process and show your analysis and model inference results to the user in the first person. If inference results contain a file path, must tell the user the complete file path. If there is nothing in the results, please tell me you can't make it.

Pipeline



Please generate an image where a girl is reading a book, and her pose is the same as the boy in the image example.jpg, then please describe the new image with your voice.
Request
Stage #1: Task Planning
[T1] task: pose-det. args: image: example.jpg
[T3] task: image-class. args: image: <resource-2>
[T5] task: image-to-text args: image: <resource-2>
task dependency
[T2] task: pose-to-image args: text: a reading girl image: <resource-1>
[T4] task: object-det. args: image: <resource-2>
[T6] task: text-to-speech args: text: <resource-5>
Stage #2: Model Selection
task: pose-det.
Query
task: object-det. args: image: <resource-2>
task: image-class.
In-context task-model assignment:
hustvl/yolos-tiny
facebook/detr-resnet-101
TahaDouaji/detr-doc-table-detection
Model Cards in HuggingFace
Stage #3: Task Execution
task, args, model
obj-det. <resource-2>
facebook/detr-resnet-101
task, args, model
Hybrid Endpoints
HuggingFace Endpoint (facebook/detr-resnet-101)
Local Endpoint (facebook/detr-resnet-101)
Bounding boxes with probabilities
Predictions
Stage #4: Response Generation
[T1] pose-det. openpose [image-2]
[T2] pose-to-image lym/sd-ctl-pose [image-3]
[T3] image-class. google/vit [classes with prob]
[T4] object-det. facebook/detr-resnet [bounding box]
[T5] image-to-text nlpconnect/vit-gpt2 "a girl sitting on ..."
[T6] text-to-speech facebook/fastspeech [audio-1]
Summary

Pipeline


Response
a girl sitting on a bed reading a book
[Text-1] = <resource-5>
[Image-1] = example.jpg
[Image-2] = <resource-1>
[Image-3] = <resource-2>
[Image-4]
[Audio-1]
The image you gave me is of "boy". The first thing I did was use OpenCV's openpose model to analyze the pose of the boy in the image. I fed that image, specifically located at [Image-1], into the model to get an output image of the pose, located at [Image-2]. Then I take that pose and generate a new image, located at [Image-3], using lllyasviel/sd-controlnet-openpose and the instruction "a girl is reading". Next, I ran object detection on the image generated in the previous step, to get predictions of the objects present in the image, including the box location. These predictions were run through the facebook/detr-resnet-101 model. The box was drawn on the image, which is located at [Image-4]. Then, I used the predictions of a image classification model google/vit-base-patch16-224 and a image caption model a nlpconnect/vit-gpt2-image-captionin model to generate the caption for newly generated image. It generated text: "a girl sitting on a bed reading a book". Lastly, I ran the facebook/fastspeech2-en-ljspeech model to convert the generated text into audio, located at [Audio-1].

Outline

1 Background and Opportunities

2 Design

3 Experimental evaluation

4 Summary

Environment Settings

LLM:

- gpt-3.5-turbo, text-davinci-003 and gpt-4
- decoding temperature = 0
- logit_bias = 0.2

System Requirements

Default (Recommended)

For `configs/config.default.yaml`:

- Ubuntu 16.04 LTS
- VRAM >= 24GB
- RAM > 12GB (minimal), 16GB (standard), 80GB (full)
- Disk > 284GB
  - 42GB for `damo-vilab/text-to-video-ms-1.7b`
  - 126GB for `ControlNet`
  - 66GB for `stable-diffusion-v1-5`
  - 50GB for others

Minimum (Lite)

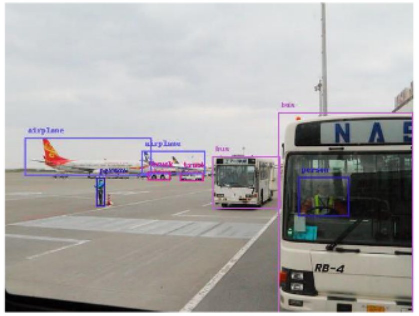
For `configs/config.lite.yaml`:

- Ubuntu 16.04 LTS
- Nothing else

Case Study

Query: Can you tell me how many objects in the picture example1.jpg?

Response: There are **8 objects** in the picture. I used the model facebook/detr-resnet-101 for this task which is trained on a popular object detection dataset called COCO, and it has the highest number of downloads. It generated an image with predicted boxes that is available under the path: /images/79f2.jpg . As for image captioning, I selected the model nlpconnect/vit-gpt2-image-captioning which is more frequently used and works better in this case. The generated text from the image is '**a large jetliner sitting on top of an airport tarmac**.'



Task Planning:

1: {"task": "object-detection", "id": 0, dep: [-1], "args": {"image": "example1.jpg"}},
2: {"task": "image-to-text", "id": 1, dep: [-1], "args": {"image": "example1.jpg"}}

Execution Results:

1. [{"generated image with predicted box": "/images/79f2.jpg", "predicted": [{"box": {"xmax": 463, "xmin": 373, "ymax": 267, "ymin": 199}, "label": "bus", "score": 0.9981155395507812}, {"box": {"xmax": 292, "xmin": 124, "ymax": 224, "ymin": 175}, "label": "airplane", "score": 0.9983609318733215}, {"box": {"xmax": 554, "xmin": 487, "ymax": 277, "ymin": 226}, "label": "person", "score": 0.9910836219787598}, {"box": {"xmax": 230, "xmin": 219, "ymax": 264, "ymin": 227}, "label": "person", "score": 0.998579740524292}, {"box": {"xmax": 361, "xmin": 330, "ymax": 231, "ymin": 220}, "label": "truck", "score": 0.9066138863563538}, {"box": {"xmax": 639, "xmin": 460, "ymax": 420, "ymin": 142}, "label": "bus", "score": 0.9987366795539856}, {"box": {"xmax": 363, "xmin": 279, "ymax": 222, "ymin": 192}, "label": "airplane", "score": 0.9976897239685059}, {"box": {"xmax": 317, "xmin": 287, "ymax": 230, "ymin": 219}, "label": "truck", "score": 0.9339178800582886}]},
2. {"generated_text": "a large jetliner sitting on top of an airport tarmac "}

Query: In example2.jpg, what's the animal and what's it doing?

Response: Sure, **the animal in the image is a tiger and it's laying down**. I performed 3 tasks including image-to-text, image-classification, and visual-question-answering on the image you provided. To accomplish this, I selected models nlpconnect/vit-gpt2-image-captioning, google/vit-base-patch16-224 and dandelin/vilt-b32-finetuned-vqa because they are the perfect models for this use case respectively. According to the results, I have 88% confidence that the animal is a tiger and also have a 77.8% confidence that it's laying down .The generated image with predicted box is stored at the path: /images/d59a.jpg.

Task Planning:

1: {"task": "image-classification", "id": 0, dep: [-1], "args": {"image": "example2.jpg" }},
2: {"task": "visual-question-answering", "id": 1, dep: [-1], "args": {"text": "What is the animal doing?", "image": "example2.jpg" }}
3: {"task":"image-to-text", "id": 2, dep: [-1], "args": {"image": "example2.jpg" }}

Execution Results:

1. [{'score': 0.8858247399330139, 'label': 'tiger, Panthera tigris'}, {'score': 0.10940514504909515, 'label': 'tiger cat'}, {'score': 0.0006216467591002584, 'label': 'jaguar, panther, Panthera onca, Felis onca'}, {'score': 0.0004262701258994639, 'label': 'dhole, Cuon alpinus'}, {'score': 0.00030842673731967807, 'label': 'lion, king of beasts, Panthera leo'}]
2. [{'answer': 'sitting', 'score': 0.8294351696968079}, {'answer': 'laying down', 'score': 0.7387925386428833}, {'answer': 'resting', 'score': 0.625247061252594}, {'answer': 'laying', 'score': 0.6114301681518555}, {'answer': 'relaxing', 'score': 0.44740357995033264}]
3. {'generated_text': 'a brown and white striped zebra laying on a rock'}

https://huggingface.co/spaces/microsoft/HuggingGPT

Critical analysis, inspiration

Critical analysis

- **Advantage**
- Using LLM to plan complex tasks and call different ai models to solve specific tasks
- **Disadvantage**
    - High response delay
    - the maximum token length is always limited
    - The model called may not complete the task

Inspiration

- **When the called model is unable to complete the task, let llm make a fine-tuning plan**

Q&A

Speaker：张兴才

2024.11.01