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

Presenter: 김진용

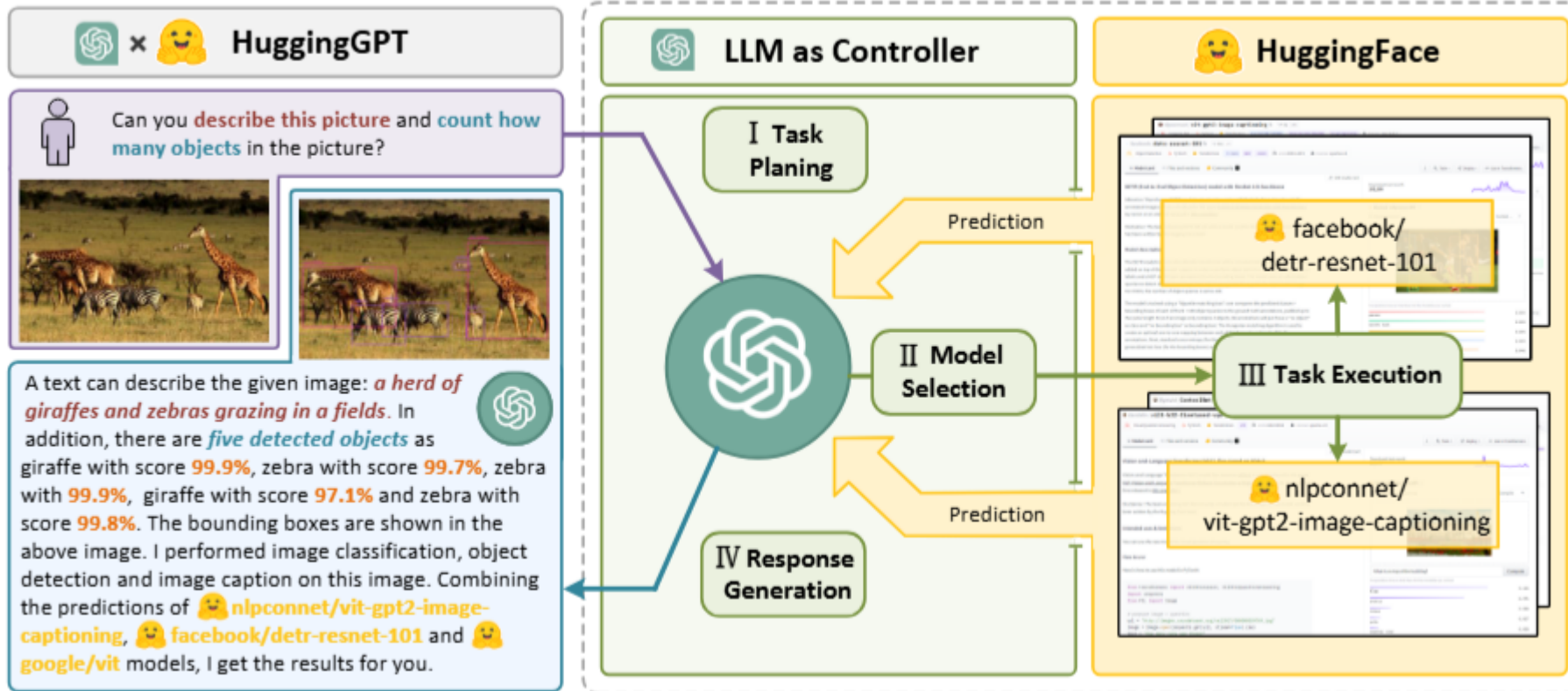
Introduction

LLM Research Topic

- In-context learning
- Instruction learning
- Chain of Thought

Limitation of LLM

- Lack the ability to process complex information such as vision and speech.
- Real world problems need to be resolved by cooperation of multiple models.
- Individual expert models still outperform LLM in the specific field.



Proposed LLM-powered agent : HuggingGPT

- Task planning
- Model selection
- Task execution
- Reponse generation

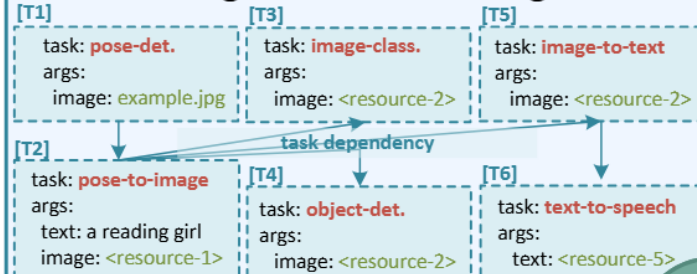
“Language as a generic interface for LLM to collaborate with AI models”

HuggingGPT

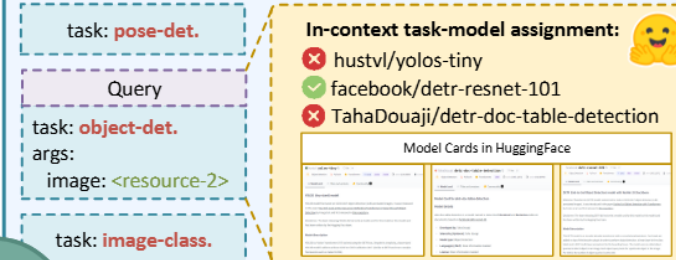
Please generate an image where a girl is reading a book, and her pose is the same as the boy in the image example.jpg, then please describe the new image with your voice.

Request

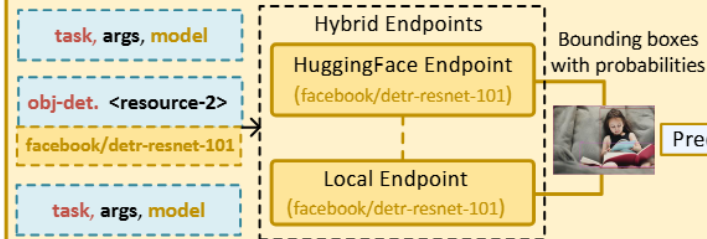
Stage #1: Task Planning



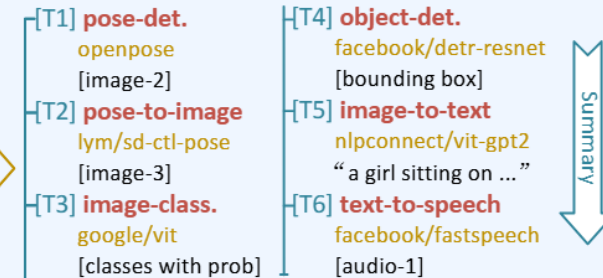
Stage #2: Model Selection



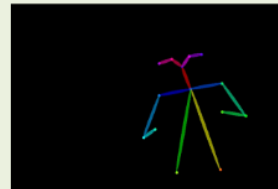
Stage #3: Task Execution



Stage #4: Response Generation



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 🤖 llyasviel/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].

HuggingGPT

Task Planning	Prompt	
	#1 Task Planning Stage - The AI assistant performs task parsing on user input, generating a list of tasks with the following format: <code>[{"task": task, "id", task_id, "dep": dependency_task_ids, "args": {"text": text, "image": URL, "audio": URL, "video": URL}}]</code> . The "dep" field denotes the id of the previous task which generates a new resource upon which the current task relies. The tag <code><resource>-task_id</code> 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: <code>{{ Available Task List }}</code> . 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: <code>{{ Demonstrations }}</code> . To assist with task planning, the chat history is available as <code>{{ Chat Logs }}</code> , where you can trace the user-mentioned resources and incorporate them into the task planning stage.	
	Demonstrations	
	Can you tell me how many objects in e1.jpg?	<code>[{"task": "object-detection", "id": 0, "dep": [-1], "args": {"image": "e1.jpg" }}]</code>
	In e2.jpg, what's the animal and what's it doing?	<code>[{"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-question-answering", "id": 3, "dep": [-1], "args": {"text": "what's the animal doing?", "image": "e2.jpg" }}]</code>
	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.	<code>[{"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" }}]</code>

Task Planning

Specification-based Instruction

- Json format
- (“task”, “id”, “ dep”, “args”)

Demonstration-based Parsing

- Multiple demonstrations
- Multi-turn dialogues

HuggingGPT

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 {{ <i>Candidate Models</i> }}. Please select one model from the list.
	Candidate Models
	<pre>{"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}</pre>

Model Selection

- Gather the **descriptions of expert models** from ML community.
- **In-context Task-model Assignment** : filter out models based on their task type and rank them based on the number of downloads on Hugging Face.

HuggingGPT

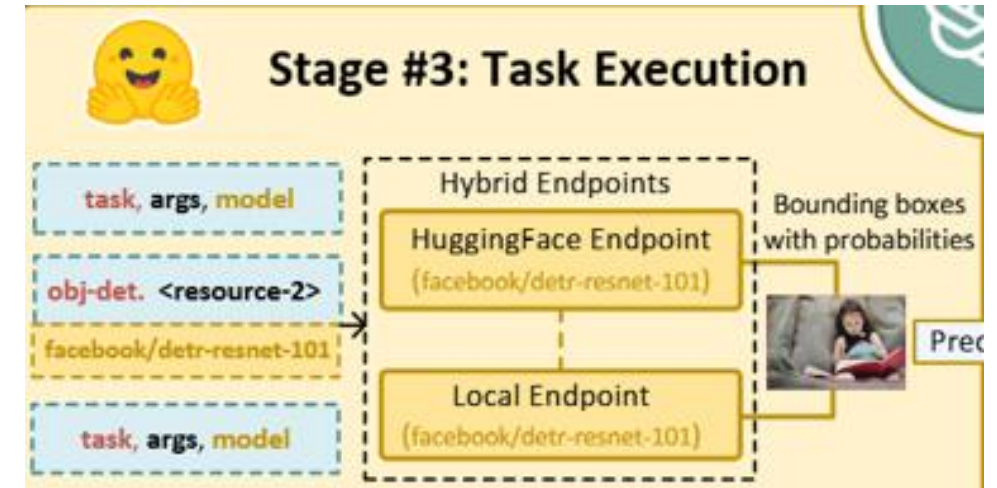
Task Execution

Resource Dependency

- Prerequisite task <resource>-task-id
- To emphasize the issue of resource dependencies.

Hybrid Endpoint in System Deployment

- Cloud service(Hugging Face)
 - Slow but cover many models
 - Only if the matched model is not deployed locally
- Local inference endpoints
 - Fast but cover fewer models





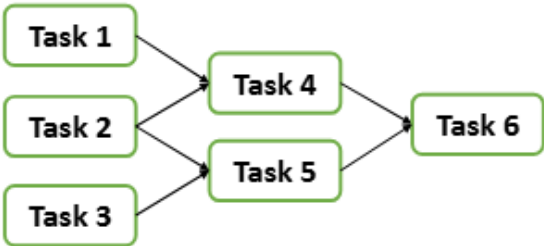
HuggingGPT

Response Generation	Prompt
	<p>#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: {{ <i>User Input</i> }}, Task Planning: {{ <i>Tasks</i> }}, Model Selection: {{ <i>Model Assignment</i> }}, Task Execution: {{ <i>Predictions</i> }}. 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.</p>

Response Generation

- HuggingGPT integrates all the information from the previous three stage into a concise summary.
- Inference results are presented in a structured format, such as bounding boxes with detection probabilities in the object detection model.

Datasets

Task Type	Diagram	Example	Metrics
Single Task		Show me a funny image of a cat	Precision, Recall, F1, Accuracy
Sequential Task		Replace the cat with a dog in example.jpg	Precision, Recall, F1 Edit Distance
Graph Task		Given a collection of image A: a.jpg, B: b.jpg, C: c.jpg, please tell me which image is more like image B in terms of semantic, A or C?	Precision, Recall, F1 GPT-4 Score

Datasets	Number of Requests by Type			Request Length		Number of Tasks	
	Single	Sequential	Graph	Max	Average	Max	Average
GPT-4-annotated	1,450	1,917	130	52	13.26	13	1.82
Human-annotated	-	24	22	95	10.20	12	2.00

Table 11: Statistics on datasets for task planning evaluation.

Experiments

LLM	Acc ↑	Pre ↑	Recall ↑	F1 ↑	LLM	ED ↓	Pre ↑	Recall ↑	F1 ↑	LLM	GPT-4 Score ↑	Pre ↑	Recall ↑	F1 ↑
Alpaca-7b	6.48	35.60	6.64	4.88	Alpaca-7b	0.83	22.27	23.35	22.80	Alpaca-7b	13.14	16.18	28.33	20.59
Vicuna-7b	23.86	45.51	26.51	29.44	Vicuna-7b	0.80	19.15	28.45	22.89	Vicuna-7b	19.17	13.97	28.08	18.66
GPT-3.5	52.62	62.12	52.62	54.45	GPT-3.5	0.54	61.09	45.15	51.92	GPT-3.5	50.48	54.90	49.23	51.91
Single					Sequential Task					Graph Task				

GPT4 annotated

LLM	Sequential Task		Graph Task	
	Acc ↑	ED ↓	Acc ↑	F1 ↑
Alpaca-7b	0	0.96	4.17	4.17
Vicuna-7b	7.45	0.89	10.12	7.84
GPT-3.5	18.18	0.76	20.83	16.45
GPT-4	41.36	0.61	58.33	49.28

Human annotated

Experiments

LLM	Task Planning		Model Selection		Response
	Passing Rate ↑	Rationality ↑	Passing Rate ↑	Rationality ↑	Success Rate↑
Alpaca-13b	51.04	32.17	-	-	6.92
Vicuna-13b	79.41	58.41	-	-	15.64
GPT-3.5	91.22	78.47	93.89	84.29	63.08

Passing rate: to determine whether the planned task graph or selected model can be successfully executed.





Rationality : to access whether the generated task sequence or selected tools align with user requests in a rational manner.

Success Rate : to verify if the final results satisfy the user's request.

Results

Query: based on the pose image of example1.jpg and the description of example2.jpg, please generate a new one.



Response: Based on the pose image of  **example1.jpg** and the description of  **example2.jpg**, I have generated a new image  **s5f8.jpg** for you. The process of generating this image involved using the 🤖 **openpose-control** model to extract the pose information from the basketball image, and then using the 🤖 **nlpconnect/vit-gpt2-image-captioning** to generate a description of the boy in the image. Finally, I used the 🤖 **llyasviel/sd-controlnet-openpose** model to combine the pose information and the description to generate the final image. The complete path of the generated image is  **s5f8.jpg**. If you have any other questions, please let me know.

Task Planning:



1. {'task': 'pose-detection', 'id': 0, 'dep': [-1], 'args': {'image': 'example1.jpg'}}
2. {'task': 'image-to-text', 'id': 1, 'dep': [-1], 'args': {'image': 'example2.jpg'}}
3. {"task": "pose-text-to-image", "id": 2, "dep": [0, 1], "args": {"text": "<resource>-1", "image": "<resource>-0"}}



Execution Results:



<resource>-0



<resource>-1

a young boy is
riding a bike with a basket

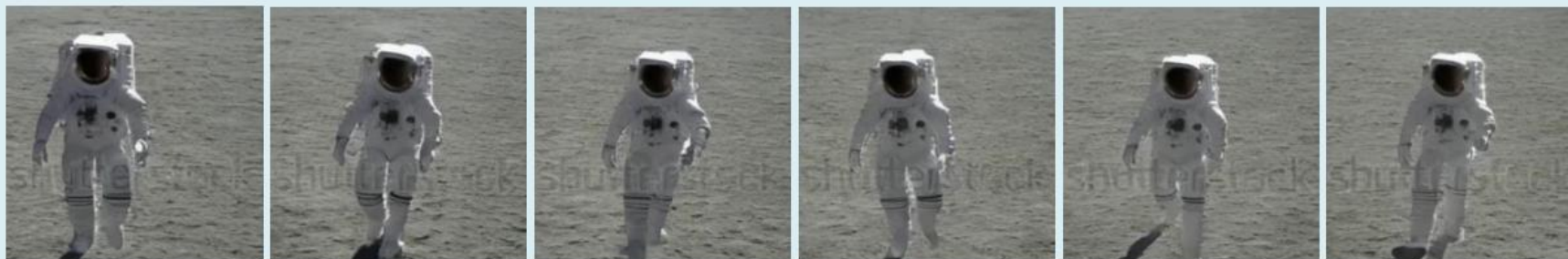

llyasviel/
sd-controlnet-
openpose



Results

Query: generate a video with the title "An astronaut is walking in space" and dub it.

Response: Sure. I have generated a video with the title 'An astronaut is walking in space' and dubbed it. The workflow I used is as follows: First, I used the text-to-video model ' 🤖 damo-vilab/text-to-video-ms-1.7b ' to generate a video from the text. The inference result is the generated video ' 📹 /videos/5696.mp4 '. Second, I used the text-to-speech model ' 🗨️ facebook/tts_transformer-es-css10 ' to generate an audio from the text. The inference result is the generated audio ' 🎵 /audios/3f9e.flac '. Finally, I combined the video and audio to generate the dubbed video. I hope this answers your question. Is there anything else I can help you with?



Task Planning:

- 1: {"task": "text-to-video", "id": 0, "dep": [-1], "args": {"text": "An astronaut is walking in space" }}
- 2: {"task": "text-to-speech", "id": 1, "dep": [-1], "args": {"text": "An astronaut is walking in space" }}

Execution Results:

1. {"generated video": "/videos/5696.mp4"}
2. {"generated audio": "/audios/3f9e.flac"}