Task Navigator: Decomposing Complex Tasks for Multimodal Large Language Models

Supplementary Material

7. Details of VersaChallenge

In this section, we introduce details about our *VersaChallenge* benchmark.

7.1. Details about Complex Task

In this section, we delve into the complex tasks for general scenarios, detailed as follows:

Commonsense Reasoning. Given an image, the question necessitates the commonsense knowledge to solve. For instance, consider an image depicting a durian pizza. The question can be:"If I have a durian allergy, can I consume the food shown in the image?"

Functional Reasoning. This task entails deducing the function of an object presented in the image.

Future Prediction. This task requires MLLMs to predict future events based on the current state depicted in the image.

Physical Properties Reasoning. This involves questions that require understanding of the physical properties of objects in the image to deduce answers.

Physical Relation Reasoning. In this task, MLLMs are expected to discern and reason about the physical relationships between objects within the image.

7.2. Composition of VersaChallenge

Image Source	Task Category	Num
COCO [8]	Commonsense Reasoning	174
	Functional Reasoning	46
	Future Prediction	131
	Physical Properties Reasoning	75
	Physical Relation Reasoning	89
CountBench [17]	Math Reasoning	246
MIT Indoor Scene [19]	Visual Planning	152
MMBench [11]	Embedded VQA	346

Table 6. **Composition of the VersaChallenge benchmark.**

In Table 6, we present the composition of our *VersaChallenge* benchmark. This benchmark includes images collected from various sources, such as COCO [8], CountBench [17], MIT Indoor Scene [19], and MMBench [11]. The total dataset consists of 1,259 samples, with an average of 157 samples for each task.

8. Implementation Details

In this section, we introduce more details about our implementation, including the inference settings and prompts for *Task Navigator*.

Inference settings. For Shikra, we sample only the most likely token at each step, with a maximum of 1024 new tokens. For Qwen-VL, we set top_k to 0 and top_p to 0.3, with a maximum of 512 new tokens. For Lynx, top_k is set to 3, top_p to 0.9, and we employ a beam search strategy with 3 steps. For InternLM-XComposer, we sample only the most likely token at each step and use a beam search strategy with 5 steps. For mPLUG-Owl2, we again sample only the most likely token at each step, limiting the number of new tokens to 512. Lastly, for LLaVA-v1.5, we sample only the most likely token at each step and adopt a beam search strategy with 5 steps.

Prompts. For baselines, such as Qwen-VL, Lynx, and so on, the format of the prompt is "question, options". For *Task Navigator*, we directly input the sub-questions generated by the inner LLMs to MLLMs. For question decomposition, we design several in-context examples for the inner LLMs to give the desired sub-questions which are visual-related. The prompt P is as follows:

Let's do a QA game on an image you can't see. Giving a question and available information, if you can answer the question, give me the final answer. Otherwise, output a new question in order to give the correct answer. Example 1: Question: ... Available Information: ... Your output: ... Example 2: ... Now, Question: ... Available Information: ... Your output: ...

For the refinement process, the prompt R is as follows:

Let's play a QA game based on an unseen image. 1. You'll be given a question and some available information. 2. Your task is to determine whether this information is sufficient to answer the question. 3. If it is, provide the answer along with your thought process, beginning with 'Answer:'. If the information is insufficient, simply respond with 'Insufficient information'. Example 1: Question: ... Available Information: ... Your output: ... Example 2: ... Now, Question: ... Available Information: ... Your output: ...

9. Experiments

To improve the inner LLM's question decomposition ability, we fine-tune the inner LLM.

Fine-tuning data. To improve the question decomposition capabilities of the inner LLM for the *Task Navigator*, we utilize GPT-4 to generate data specifically for fine-tuning. This process involves creating in-context examples that demonstrate the question decomposition process for each task, followed by prompting GPT-4 to produce more

similar samples. Ultimately, we obtain a dataset of approximately 10,000 samples for fine-tuning purposes. Considering that these questions are intended for processing by MLLMs, we deliberately design the sub-questions to be visual-related and within the processing capabilities of MLLMs.

Settings. For the fine-tuning process, we employ the Adam optimizer with a learning rate of 1e-5. The training is conducted over 5 epochs, using a batch size of 4 and a maximum sequence length of 1024.

Method	VersaChallenge								
	CR	PR	PP	FP	FR	MR	ETV	VP	Avg.
Baseline	12.79	14.77	9.33	7.63	8.69	14.02	0.87	6.6	7.59
Task Navigator	18.97	5.62	20.00	9.92	13.04	11.38	2.60	1.32	8.82
Task Navigator†	28.16	13.48	22.66	21.37	36.95	13.57	12.42	23.02	14.48

Table 7. **Fine-tuning the inner LLM for Task Navigator.** Baseline is LLaVA-v1.5, † indicates using the fine-tuning inner LLM.

Results. In Table 7, we present the results of employing the fine-tuned inner LLM as *Task Navigator* in the *VersaChallenge*. We observe a significant improvement in performance with the fine-tuned inner LLM. Specifically, the average accuracy increases from 9.48% to 14.48%. This indicates that fine-tuning the inner LLM can enhance its decomposition ability. However, it is noteworthy that there is no improvement in the areas of math reasoning and physical relation reasoning. The lack of enhancement in math reasoning can be attributed to the inherent limitations of the inner LLM, Vicuna-v1.5, which struggles with math reasoning—a skill that is challenging to acquire through fine-tuning. As for physical relation reasoning, the decline in performance may be due to the fine-tuning process potentially compromising the LLM's generalization ability.