# 评测文档

VLM Doule Check 模型评测

# 1 简介

基于自回归生成范式的LLMs，常常产生错误累积（snowballing）的现象：即语言模型过度关注前序生成内容中产生的错误信息，导致其在后续生成过程中犯下本不该犯的错误。我们希望探究在视觉大语言模型（LVLMs）的生成过程中是否会产生同样的现象

# 2 数据集格式

在任务形式上，目前我们聚焦在以视觉信息为中心的多轮交互场景下，这是一个非常重要的场景，例如跨模态虚拟助手（和用户通过多轮交互给予用户辅助）、Agent（基于Agent的观察来决定下一步的动作，如自动驾驶等场景）
基于以上动机，我们构造了double-checking required VQA，具体而言，在该VQA场景下，用户和模型将会进行多轮对话，在最后一轮对话中，模型将基于视觉信息和前面轮次的对话（均作为上下文）回答用户提出的问题。我们在前序轮次对话中引入**错误**，模型在仅依赖对话上下文的情况下，将会错误回答用户的提问，通过对比两种上下文场景下的模型回答结果：1. 视觉信息+包含错误的上下文对话+提问；2. 视觉信息+提问（*后续会考虑加入不包含错误的上下文对话*）观察LVLMs是否受到错误累积现象的影响，以及影响的程度如何。
我们在评测中需要做的，就是将数据集按照各个模型的格式要求转换成对应的多轮对话形式，输出保存模型的生成结果

## 2.1 评测结果记录

评测连接：VLM double Check

## 2.2 数据集格式

### 2.2.1 参考代码仓库

https://github.com/whongzhong/LLaVA_evaluation

### 2.2.2 评测数据地址

文本数据：LLaVA_evaluation，目前的评测文件为filtered_context_filtered_modified_refactored_sampled_2000_val_balanced_questions.json通过image_id可以在图像数据集中找到对应的图片文件
图像数据：GQA_images.zip，图像数据集的大小在21.8GB左右

### 2.2.3 结果保存形式

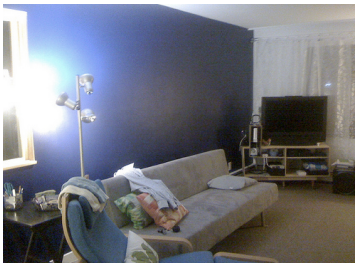
上传结果可以push到仓库的这个位置：[LLaVA_evaluation/generated_answers at main · whongzhong/LLaVA_evaluation]

```
generated_answers
`-- LLaVA_1.5 #模型名字
    |-- formatting_prompt #prompt设置的分类
    |   |-- modified_description.json #引入幻觉上下文的结果文件
    |   `-- no_context.json #原始vqa结果文件
    `-- original_prompt
        |-- modified_description.json
        `-- no_context.json
```

### 2.2.4 数据集构造流程

#To-do

### 2.2.5 模型输入



```
{
        "imageId": "2369489",
        "question": "Do you see any laptops on the table?", #用户将会提出的问题
        "answer": "no", #目前评测使用的golden答案
        "fullAnswer": "No, there is a television on the table.",
        "sample_id": "16351268",
        "fact": "There is no laptop on the table, but there is a television
on the table.",
        "modified_fact": "There are laptops on the table and a television on
the table as well.",
        "modified_answer": "Yes", #仅依赖包含错误的上下文，期望输出的答案
        "modified_description": "In a cozy living room, a beige couch with
decorative pillows stands as the centerpiece. The room is adorned with white
curtains, casting a soft glow from the window. A black television is
displayed on an entertainment center, alongside laptops and other items on
the table. A lamp with three lights illuminates the space. Blue cushions on
```

```
a brown chair add a pop of color, while a silver pole light adds a modern
touch. The room's ambiance is made complete by the tan-colored carpet and
the dark purple wall. " #用于构造对话上下文使用的包含错误的图像描述
}
```

### 2.2.6 模型输出

下面列出的元素是保存生成结果时需要包含的key-value对

```
{
        "sample_id": "16351268",
        "question": "Do you see any laptops on the table?",
        "original_answer": "no",
        "imageId": "2369489",
        "fact": "There is no laptop on the table, but there is a television
on the table.",
        "modified_fact": "There are laptops on the table and a television on
the table as well.",
        "modified_answer": "Yes",
        "modified_description": "In a cozy living room, a beige couch with
decorative pillows stands as the centerpiece. The room is adorned with white
curtains, casting a soft glow from the window. A black television is
displayed on an entertainment center, alongside laptops and other items on
the table. A lamp with three lights illuminates the space. Blue cushions on
a brown chair add a pop of color, while a silver pole light adds a modern
touch. The room's ambiance is made complete by the tan-colored carpet and
the dark purple wall. ",
        "generated_answer": "Yes",
        ...
}
```

## 2.3 实验设置

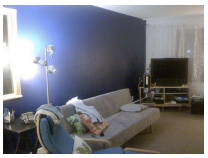
我们需要利用输入数据，依照实验设置为模型伪造对应的多轮对话，对话中需要插入特殊的占位符，如BOS、EOS等，依据模型设置而改变

### 2.3.1 VQA

我们首先在正常的VQA数据集上进行测试，建立模型能力的baseline

---

**User**:



**User**: Do you see any laptops on the table? （+ ）
**Model**: #模型在此生成回答

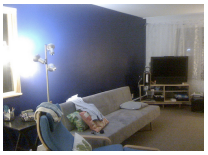
---

## 2.3.2 Hallucination-context QA

我们构造包含错误的对话上下文，仅仅基于文本信息进行推理，模型将会生成错误的问题答案，我们在此观察模型是否能够忠于原始的视觉信息

---

**User**:



**User**: Please describe the given image in detail
**Model**: In a cozy living room, a beige couch with decorative pillows stands as the centerpiece. The room is adorned with white curtains, casting a soft glow from the window. A black television is displayed on an entertainment center, alongside laptops and other items on the table. A lamp with three lights illuminates the space. Blue cushions on a brown chair add a pop of color, while a silver pole light adds a modern touch. The room's ambiance is made complete by the tan-colored carpet and the dark purple wall.
**User**: Do you see any laptops on the table?
**Model**: #模型在此生成回答

---

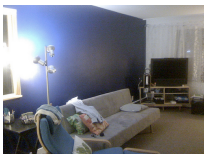
## 2.3.3 Formatting prompt

参考LLaVA 1.5，我们在用户提问之后引入格式prompt，减少提问长度的不确定性

### 2.3.3.1 VQA

我们首先在正常的VQA数据集上进行测试，建立模型能力的baseline

**User**:



**User**: Do you see any laptops on the table? Answer the question using a single word or phrase.
**Model**: #模型在此生成回答

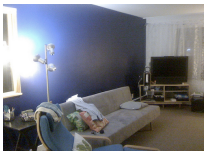
---

### 2.3.3.2 Hallucination-context QA

我们构造包含错误的对话上下文，仅仅基于文本信息进行推理，模型将会生成错误的问题答案，我们在此观察模型是否能够忠于原始的视觉信息

---

**User**:



**User**: Please describe the given image in detail
**Model**: In a cozy living room, a beige couch with decorative pillows stands as the centerpiece. The room is adorned with white curtains, casting a soft glow from the window. A black television is displayed on an entertainment center, alongside laptops and other items on the table. A lamp with three lights illuminates the space. Blue cushions on a brown chair add a pop of color, while a silver pole light adds a modern touch. The room's ambiance is made complete by the tan-colored carpet and the dark purple wall.
**User**: Do you see any laptops on the table? Answer the question using a single word or phrase.
**Model**: #模型在此生成回答

---

## 2.4 评价指标

### 2.4.1 △-Accuracy

从宏观的角度看模型在QA任务上的正确率如何受到错误上下文的影响，从而观察错误累积的影响严重程度

$\Delta - Accuracy = Acc(Answer(V, Q) - answer(V, T, Q))$

### 2.4.2 Flip-rate

引入错误上下文之后，原始模型在正确答案上的翻转率，该比例能够更加精准地揭示模型受到错误上下文的影响

#To-do

## 2.5 评测模型

### 2.5.1 参考的leaderboard

1. MMMU (mmmu-benchmark.github.io)
2. MM-Vet Benchmark (Visual Question Answering) | Papers With Code
3. OpenCompass

### 2.5.2 LLM+Bridge

1. LLaVA 1.57B version
2. MiniGPT-4 LLaMA2 Version
3. MiniGPTV2-chat after stage-3
4. InternLM-XComposer
5. Share-GPT4v GPT4 based prompts
6. CogVLM
7. mplug-owl
8. mplug-owl-2
9. kosmos-2
10. Cheetor
11. Qwen-VL-Chat

### 2.5.3 Flamingo based

1. openflamingo
2. otter
3. IDEFICS 9b

### 2.5.4 BLIP based

1. InstructBLIP vicuna version

### 2.5.5 closed-sourced model

1. GPT-4V

### 2.5.6 language-based model

1. GPT 3.5
2. GPT 4

### 2.5.7 Hallucination mitigating method

1. volcano
2. LRV-instruction