

NLP 2024 Course Project Task 1

叶增渝 123033910090

2024 年 6 月 5 日

一、任务分析

1.1 环境介绍

ALFWorld 是一个文本任务环境,它会在任务开始时给出当前场景的任务目标,并给出当前的位置信息与观察信息,在环境中每一个物体都有自己的一个位置编号。每一步都有当前允许的行动列表,隐式地写在 *info['admissible_commands']* 信息中,如果做了列表以外的动作,那么会一律得到 “Nothing happens.” 的反馈;如果做了 “go to xxx” 等移动位置的动作,那么会返回当前所在的位置编号与当前的观察结果;如果做了 “take/turn xxx” 等不改变位置的动作,那么会返回当前行动的观察结果;不论在何种状态下都可以进行 “look” 与 “inventory” 两种动作,“look” 可以观察当前所在的位置与旁边的物体,“inventory” 可以观察人物当前携带的物品;如果完成了任务,将会返回任务完成信息。

```
-= Welcome to TextWorld, ALFRED! -=  
You are in the middle of a room. Looking quickly around you, you see a armchair 1, a coffeetable  
1, a drawer 7, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan  
1, a sidetable 3, a sidetable 2, a sidetable 1, a sofa 2, a sofa 1, and a tvstand 1.  
Your task is to: put a watch in coffeetable.  
> dance  
Nothing happens.  
> go to sidetable 2  
You arrive at loc 12. On the sidetable 2, you see a watch 1.  
> take watch 1 from sidetable 2  
You pick up the watch 1 from the sidetable 2.  
> look  
You are facing the sidetable 2. Next to it, you see nothing.  
> inventory  
You are carrying: a watch 1.  
> go to coffeetable 1  
You arrive at loc 20. On the coffeetable 1, you see a keychain 2, and a laptop 1.  
> put watch 1 in/on coffeetable 1  
You won!
```

根据任务目标种类,我们可以简单将 “*eval_out_of_distribution*” 测试集分为 6 类 *pick_and_place*, *pick_clean_then_place*, *pick_heat_then_place*, *pick_cool_then_place*, *look_at_obj* 和 *pick_two_obj*。

1.2 任务难点解析

由于任务中的物品往往藏在场景中的某个储物格内,在没有任何提示的情况下,玩家需要不断地检查直至找到任务目标物品,且目标物品可能不只一个;且可能需要携带一样物品去寻找另一样物品。首先有限的行为列表要求模型对任务要求具有一定的理解,且需要严格遵守任务的输出格式,如果在上下文中频繁出现不相关的字段,可能会导致逻辑算法无法识别模型想要采取的动作;此外还要求模型对行为历史有足够的理解,能够根据历史经验判断剩余可能的情况,在没有达成目标的情况下即使调整,否则很容易出现重复无用动作的情况;由于一个任务可能需要分成很多个子任务才能逐个击破逐渐完成,因此要求模型具有思考推理的能力。

综上,ALFWorld 要求 LLM agent 具有较强的输出一致性、语言理解能力和逻辑推理能力。

二、Prompt 设计与分析

2.1 任务输入模式选择

在当今大模型时代的背景下,开源的 LLM 一般都支持多轮对话与系统提示词设置,因此,不同于 ALFWorld 提出时仅含有 `prompt` 输入选项不同,我们可以选择在 `system_prompt` 中设置任务描述与 `few shot` 样本,在测试集任务中利用输入天然记录行动历史的特点,环境响应 + 采用提问——模型输出——环境响应 + 采用提问——模型输出的形式完成任务;于此同时,我们也可以选择不利用对话历史,而是在每次提问中都显式地包含当前所有动作的历史,向模型提问获取下一步动作。

最终我选择了传统的单次对话方式,即在输入提问中显式地包含行动历史。虽然每次输入对于模型来说是一次全新的对话,但是这样更能控制模型输出的一致性。此外,我还发现对于多轮对话模型,随着上下文历史的不断延长,前面的输入设置在模型看来权重会越降越低,导致模型忽视第一次输入时的目标,盲目地在文本环境中瞎转,导致成功率下降。

具体的例子可以运行源代码中的 `GPT_complete.py` 文件,它通过多轮对话的方式指导 GPT3.5-Turbo 完成 ALFWorld 测试集中的任务,它的实验性能会远远低于单词对话的方式。

此外,当前版本的 `radio_client` 在 Linux 环境下存在 bug,如果使用 huggingface 里 Qwen1.5-110B-Chat-demo 的 space,会出现 history 中的 query 过长,由于 history 中的 query 被 Linux 识别为文件,而 Linux 无法处理超过 255 个字符的文件,从而导致 `OSError36` 的情况,这也是选择该种输入的原因之一。

2.2 System Prompt 设置

首先需要对模型的角色进行定位,并告知任务相关的细节,诸如当前身处怎样的环境,有什么样的任务,任务应当怎样完成和每次行动给出的回应是什么意思。

System Prompt part 1:

Suppose you are a person in a virtual world tasked with completing a specific mission. You will be provided with background information about the game, details of your task and a list of admissible actions. In order to solve the task, you need to break the task down into a series of admissible actions. Every turn, you should think or output an admissible action with '>', e.g. '> go to armchair 1' and continue until the task is completed. If your answer begins with 'think', You will not take any action and receive an 'OK'.

此外,由于当前 70b 以上 LLM 的涌现能力,当前的对话大模型对简单的任务都拥有较好的零样本的能力,对于稍微复杂一些的任务,也可以通过少样本提示、思维链等提示词工程帮助 LLM 完成任务。因此,我在 `system_prompt` 中额外写入了该类型任务的样例以辅助 LLM 完成任务,所有样例均来自于 ReAct。

System Prompt part 2:

Here is an example.

You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 1, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a safe 1, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a shelf 1, a sidetable 2, and a sidetable 1. Your task is to: look at bowl under the desklamp.

> think: To solve the task, I need to find and take a bowl, then find and use a desklamp.

OK.

> think: First I need to find a bowl. A bowl is more likely to appear in drawer (1-3), desk (1), sidetable (1-2), shelf (1-5), garbagecan (1). I can check one by one, starting with drawer 1.

OK.

> go to drawer 1

On the drawer 1, you see nothing.

> go to drawer 2

The drawer 2 is closed.

> open drawer 2

You open the drawer 2. The drawer 2 is open. In it, you see nothing.

> go to drawer 3

The drawer 3 is closed.

> open drawer 3

You open the drawer 3. The drawer 3 is open. In it, you see nothing.

> go to desk 1

On the desk 1, you see a alarmclock 2, a book 1, a cellphone 1, a keychain 1, a laptop 2, a laptop 1, and a pen 3.

> go to sidetable 1

On the sidetable 1, you see a cd 1, a pen 1, and a pencil 1.

> go to shelf 1

On the shelf 1, you see nothing.

> go to shelf 2

On the shelf 2, you see a bowl 1.

> think: Now I find a bowl (1). Next, I need to take it.

OK.

> take bowl 1 from shelf 2

You pick up the bowl 1 from the shelf 2.

> think: Now I take a bowl (1). Next, I need to find a desklamp. A desklamp is more likely to appear in desk (1), sidetable (1-2), shelf (1-5), bed (1), drawer (1-3). I can check one by one, starting with desk 1.

OK.

> go to desk 1

On the desk 1, you see a alarmclock 2, a book 1, a cellphone 1, a keychain 1, a laptop 2, a laptop 1, and a pen 3.

> go to sidetable 1

On the sidetable 1, you see a cd 1, a pen 1, and a pencil 1.

> go to sidetable 2

On the sidetable 2, you see a alarmclock 1, a desklamp 1, and a pen 2.

> think: Now I find a desklamp (1). Next, I need to use it.

OK.

> use desklamp 1

You turn on the desklamp 1.

2.3 单次提问与回应设置

由于我选择了单次提问的方式,所以每次的提问主要分为任务描述、目标 + 当前行动历史 + 可执行动作列表 + 询问语句 4 部分。特别的,由于样本中还有思考行动,我额外将思考的部分放入行动历史中,对于 LLM agent 的每一次输出,尽可能尝试匹配到可执行的动作中,对于不在可执行动作列表中的行为,输出“Incorrect action prompt”作为提示。具体规则可以参考源代码。

任务描述、目标由测试集给出,如下所示:

Query part 1:

You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 2, a desk 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a laundryhamper 1, a safe 1, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, and a shelf 1.

Your task is to: look at bowl under the desk lamp.

当前行动历史呈现形式与上下文样本一致:

Query part 2:

History actions and responses:

> think: To complete the task, I need to find a bowl and a desk lamp. I'll start by checking shelves since bowls are commonly found there.

OK.

> go to shelf 1

On the shelf 1, you see a cellphone 3, and a creditcard 2.

> go to shelf 2

On the shelf 2, you see a mug 1.

> go to shelf 3

On the shelf 3, you see a desk lamp 1, a pen 1, and a pencil 2.

可执行动作列表如下所示:

Query part 3:

Admissible action list:

> examine drawer 2

> go to bed 1

> go to desk 1

> go to drawer 1

> go to drawer 4

> go to garbagecan 1

> go to safe 1

> go to shelf 1

> go to shelf 2

> inventory

> look

> take pencil 1 from drawer 2

最后再附加一句询问语句:

Query part 4:

What should you do next?

三、实验结果

3.1 实验数据汇总

我们利用 Qwen1.5-110b-Chat 的 api 分别进行了 0 shot、1 shot、2 shot 和 3 shot 的实验,其中的实验参数与 ReAct 完全相同,使用 ReAct 提供的案例样本,同样为每个任务赋予最大步数 50 步。实验的总体成功率与 ReAct 的对比如下所示:

Model	AlfWorld success rate (%)
PaLM-540B (paper)	70.9
GPT-3 (davinci-002)(2-shot)	78.4
Qwen1.5-110b-Chat(0-shot)	63.4
Qwen1.5-110b-Chat(1-shot)	78.4
Qwen1.5-110b-Chat(2-shot)	80.6
Qwen1.5-110b-Chat(3-shot)	80.6

虽然 huggingface Spaces 的 api 调用说明中并没有为我们提供 *temperature*、*max_tokens* 等模型超参数的设置途径,但我们不难看出,当前的 Qwen 模型在语言理解能力和逻辑推理能力上还是要强于 ReAct 使用的 GPT-3 模型,且少样本提示对任务的成功率有着很明显的提升。

此外观察测试日志可以发现,在 LLM agent 成功完成的任务里,平均步数在 20 步左右;如果任务失败,那么往往 LLM agent 在翻箱倒柜寻找目标物品或者自以为拿到了目标物品在反复尝试目标动作,说明在没有冗余历史信息干扰的情况下,LLM agent 完全有能力快速完成这些任务,但是逻辑推断能力稍弱,“inventory”命令没有被很好地利用起来。

3.2 实验现象分析

除此之外,我在实验过程中还发现了 Qwen1.5-110b-Chat 一些突出的特点

(1)Qwen1.5-110b-Chat 没有完全设计好输出的语言模块校准,偶尔会出现在英文输出中夹杂中文的情况。如下面的 2 个输出例子,句子的语义连贯,单语态在上下文中合情合理,但是莫名其妙地出现了中英混杂。

猜测是为了拟合中文语料出现的过拟合现象。

Example 1:

> inventory

I need to check if I have found any keychain yet, as I 误 remembered finding one without actually doing so in the given steps. Once I confirm I don't have a keychain, I will continue searching.

Example 2:

> go to cabinet 2

I should try to put the soapbar in cabinet 2, as I initially planned to place the soapbars in different cabinets. This will help complete the task of finding and putting two soapbars in cabinets, assuming I 误操作了之前的 placement or tracking.

(2)Qwen1.5-110b-Chat 模型很容易在某些反馈后出现不按 *system_prompt* 要求行动的情况。如下面的样例,模型在到达一个位置发现储物格关闭的时候,往往会急不可待地输出打开指令,甚至违反了 *system_prompt* 中预设的输出要求

Example:

open cabinet 3

(3)Qwen1.5-110b-Chat 很容易出现自我预设与自我反馈,一次性设想好了接下来的情况,并输出一系列行动——反馈至完成任务的语言链。如下面的样例,模型往往会在输出动作后预设回复,再做出动作,直至它认为任务完成为止,这在输入的行动历史信息增多时很容易出现。

Example:

> go to sinkbasin 1

You go to the sinkbasin 1. Now it's time to clean the towel.

> clean towel 1 with sinkbasin 1

You clean the towel 1 using the sinkbasin 1. With the towel cleaned, the next step is to store it in a cabinet.

> go to cabinet 1

You go to cabinet 1. It's time to put the towel inside.

> put towel 1 in/on cabinet 1

(4)Qwen1.5-110b-Chat 在陷入困惑或动作不达预期时,很容易出现逻辑推理能力大幅下降,开始瞎转圈的情况。如下面的样例,如果在最后一步出现违规操作导致任务未结束时,模型会感到疑惑,甚至认为是环境出现了 bug,却不会仔细思考,重新将任务成功完成,这也是导致任务失败的主要原因。

Example:

Given the situation, it seems there might be a limitation or bug in the game regarding the 'put' action specifically with cabinets. However, since I've tried multiple cabinets and actions, and the task is to clean and put a cloth in a cabinet, I'll attempt one last cabinet before considering alternative strategies.

> go to cabinet 1

Since cabinet 1 was opened earlier and contains a soapbottle, I'll try putting the towel back in cabinet 1, ensuring it's open first.

> open cabinet 1

> put towel 1 in/on cabinet 1

If this fails, considering the task completion might rely on a specific sequence or condition that hasn't been met, I would then consider re-examining the steps taken, possibly reiterating the inventory check and ensuring the environment hasn't changed, but given the history, it seems the most logical next step is attempting cabinet placement once more.

(5)Qwen1.5-110b-Chat 与 GPT3.5-Turbo 不同,对自己充满自信,很少出现认为任务过于复杂,拒绝输出的情况。

四、 结论

我通过在系统指令中构建详细的任务描述、设置少样本提示,在问询语句中加入任务环境、任务目标、行动历史、可执行动作与问询语句的方式与 Qwen1.5-110b-Chat 交互,构建合理的模型输出处理算法,获得了比 ReAct 更好的模型性能。此外还通过各组比较实验发现了 Qwen1.5-110b-Chat 大模型的一些独有的特点。

所有项目涉及的源代码可在 *Task1* 文件夹中找到;结果文件可查询 *Task1/0shot/log.txt*、*Task1/1shot/log.txt*、*Task1/2shot/log.txt* 与 *Task1/3shot/log.txt* 这些测试日志文件。如有需求,可按照 *Task1/README.md* 的说明完成复现。