

2025 自然语言处理 课程设计 2

人工智能学院 221300079 王俊童

2025.5.1

首先做一个简介：

这次的任务基本是用四个不同方法去尝试把一个解密游戏做好。

本来想用免费 ai 的，比如 Free QWQ 之类的，但是这一类 ai 有一个通病就是连接极度不稳定，而且算力分配是有问题的，所以我自费了 Kimi，用的模型是中等强度的 moonshot-v1-32k。

本次作业制作成本极高。

同时，由于这个作业评价指标不唯一，所以我设计了三个评价指标

- 指标一：是否全部答对。只有 0, 1 二值。
- 指标二：在必须回答的问题中，回答对了多少。（非附加问题准确度）
- 指标三：在非必须回答问题中，回答对了多少。（附加问题准确度）

这个指标设置相对简单，因为我们不去很严格的考虑 ai 全部能答对，其实这种指标相对的武断了。我们也要考虑到答对了一部分这个情况。

1 实现方法，对自己设计的代码模块用简洁的语言描述

1.1 任务一：调用 API 生成

首先是调用 api 生成，这里我重新写了一个框架，基本思路非常非常 easy，就是喂进来一个算一个。

当然，ai 错的还是有很多的，我们在这里挑选几个 case 来观察一下准确率：

首先是一个完全正确的例子：

让我们根据提供的信息逐步推理：1. 博迪先生生前所在的房间位于扳手所在房间的正东方。这意味着博迪先生不可能在 1 号房间（因为 1 号房间的正东方是墙壁，没有房间），所以博迪先生在 2 号房间。2. 心怀仇恨的嫌疑人当时在画室。由于博迪先生在 2 号房间，而 2 号房间是图书室，所以心怀仇恨的嫌疑人不可能在图书室。因此，心怀仇恨的嫌疑人在 1 号房间，即画室。3. 桃子小姐的动机是恐惧。由于心怀仇恨的嫌疑人在画室，那么桃子小姐就不在画室，所以桃子小姐在图书室，即 2 号房间。4. 要完成谋杀，凶手必须与博迪先生独处一室，且现场至少存在一件凶器。由于博迪先生在 2 号房间，且桃子小姐也在 2 号房间，所以真凶是桃子小姐。5. 凶器为何？由于扳手在 1 号房间，而博迪先生和桃子小姐在 2 号房间，所以凶器不可能是扳手。因此，凶器是毒药。6. 命案发生在何处？由于博迪先生和桃子小姐都在 2 号房间，所以命案发生在图书室。7. 杀人动机是什么？桃子小姐的动机是恐惧。8. 桃子小姐当时身在何处？桃子小姐在图书室，即 2 号房间。9. 毒药最初存放在哪个房间？由于凶器是毒药，且命案发生在 2 号房间，而扳手在 1 号房间，所以毒药最初存放在 2 号房间。综上所述，答案如下：A. 桃子小姐 B. 毒药 C. 图书室 D. 恐惧 E. 图书室 F. 图书室

然后是一个部分正确的例子：

最后是一个完全错误的例子：

可以看出 ai 的推理还是有缺陷的，我们给出在 kimi 的 moonshot-v1-32k 下的准确度：

方法	原指标准确度	必答准确度	选答准确度
API	2%	27.62%	27.95%

Table 1: 方法性能对比一

1.2 任务二：Prompt Engineer

这个任务要求我们化身 prompt 大师，我们在上面任务的基础上，增加了两个可能帮助我们的大模型进一步生成更准确的推理的 prompt：

Prompt1: 你是福尔摩斯: 具体操作为：

```
def prompt_sherlock(self, question: str) -> str:
    return f"你是福尔摩斯，接到一个案子：{question}。请详细推理并找出答案。"
```

Prompt2: 序列化推导问题: 具体操作为：

```
def prompt_step_by_step(self, question: str) -> str:
    return f"请一步步推理以下问题，并给出符合逻辑的正确的最终答案：{question}"
```

这两个方法最主要功能就是，提醒大模型你的身份，或者你该怎么做，大模型就不会去乱做或者没有任何先验的情况下乱搞。我们得到的结果如下：

方法	原指标准确度	必答准确度	选答准确度
Prompt 福尔摩斯	3%	46.79%	57.95%
Prompt 序列推理	3.5%	42.62%	54.42%

Table 2: 方法性能对比二

可以看到，我们给了大模型两个 prompt 之后，结果均上升了，而且正确率有 50 左右，提升了接近两倍，说明给大模型这种类似于心理暗示的东西和提示词是可以提升模型效率的。

1.3 任务三：工具使用

1.4 任务四：多智能体对话

编写代码，模拟现实场景中的多智能体交流，让多个 LLM 以不同角色协同解决问题。这个地方我们主要实现一个法庭的辩论这种情形。

因为这个地方跟破案解密有关，我们设置一个法官和证人这种，然后通过 self.loop 去控制法官和证人会交锋多少轮次。其实我们想做的就是类似于 GAN 一样的，证人讲证据，然后法官二次判断这样来。然后最后法官作出总结陈词。

首先可以证明的是，因为反复询问这种手段的存在性，我们有必要在不同的 loop 轮次上进行不同次数的实验，因为这可能意味着也许证人提供更多的细节内容，那么法官会更倾向于做出正确选择。

首先从准确率上说

我们首先看在 loop 为 1 的情况下的正确率，这个就是最基本的一种情况了，问一次然后总结。

方法	原指标准确度	必答准确度	选答准确度
Agent loop1	2%	26.63%	25.75%

Table 3: 方法性能对比四

那么其实，我们可以增加 loop 次数，由于经费有限，我们随机挑选 5 个例子来做这个任务，看看五个例子的预测效果会随着 loop 变化怎么变。

这个结果十分有趣，说明我们的法官在某种程度上被绕晕过去了，真理好像不是越辩越明反而结果变得扑朔迷离了。那么对于这个任务来说，也不排除有可能法官被我们的证人误导了，现实生活中这种情况也很

方法	原指标准确度	必答准确度	选答准确度
Agent loop1	40%	20%	20%
Agent loop3	40%	20%	20%
Agent loop5	0%	10%	0%

Table 4: 方法性能对比四

常见，比如有证人故意伪造假的证词，或者进行误导，那么在这种具有明显逻辑推理难度的情形下，似乎被误导了也就不是什么奇怪的问题了。

然后从趣味性上说

我们可以截取一下对话例子：完整文件在 `output/agent_diag` 中有一个例子记载：

```

"role": "user",
"content": "案件描述如下：冬夜深沉，雪花无声地落在都铎庄园的尖顶上。
富可敌国却行踪神秘的约翰·Q·博迪先生，正在宅邸里举办一场小型但奢华的晚宴，
邀请了几位最亲密的伙伴。然而，当黎明的第一缕光线穿透云层时..."

"role": "assistant",
"content": "好的，让我们开始调查这起案件。
首先，我们需要梳理一下已知的信息，并根据这些信息来推断出案件的真相。
1. **案件描述**： — 博迪先生在都铎庄园被谋杀 ...
2. **关键线索**： — 博迪先生曾在大厅出现，或者布鲁内特先生曾在大厅出现。

"role": "user",
"content": "侦探，请根据目前的发言进行分析并质询。"

"role": "assistant",
"content": "好的，让我们根据目前的发言和线索进行分析，并提出质询。..."

"role": "user",
"content": "证人，请回答刚才侦探的质询。"

"role": "assistant",
"content": "作为法官，我将扮演证人的角色，根据案件描述和线索来回答侦探的质询。..."

"role": "user",
"content": "请你作为法官，综合大家的发言，给出最终推理结论..."

"conclusion": "根据案件描述、关键线索以及证人的陈述，以下是最终推理结论：..."

```

那么其实我们可以看到，我们大概的执行流程如下：

user 提供案件描述 → assistant 开始调查 → 侦探进行询问 → assistant 开始分析 → 扮演证人来描述 → 法官总结

当然，这仅仅是在 `loop1` 的时候的结果，如果法官想和人多轮次的询问，那应该可以说的更多，但有可能有误导从而导致法官做出错误决定。

2 复现主要实现结果，包括执行命令和环境依赖

所有的环境依赖都在 `requirements.txt` 中。

你可以选择用指令：`sh run.sh` 来运行程序

如果想单独运行：main.py 即可。指令如下：

```
python3 main.py -method 0 -n 0
```

其中，method0,1,2,3 代表 4 个不同方法，0 代表运行所有数据。

3 不同方法的实验结果如何

下面给出四个方法的汇总表格实现：

方法	原指标准确度	必答准确度	选答准确度
API	-	-	-
Prompt 福尔摩斯	-	-	-
Prompt 序列推理	-	-	-
Tools	-	-	-
Agent loop1 (ALL)	-	-	-
Agent loop1 (Random 5)	-	-	-
Agent loop3 (Random 5)	-	-	-
Agent loop5 (Random 5)	-	-	-

Table 5: 方法性能对比四

4 遇到的具体问题，如何解决

5 思考