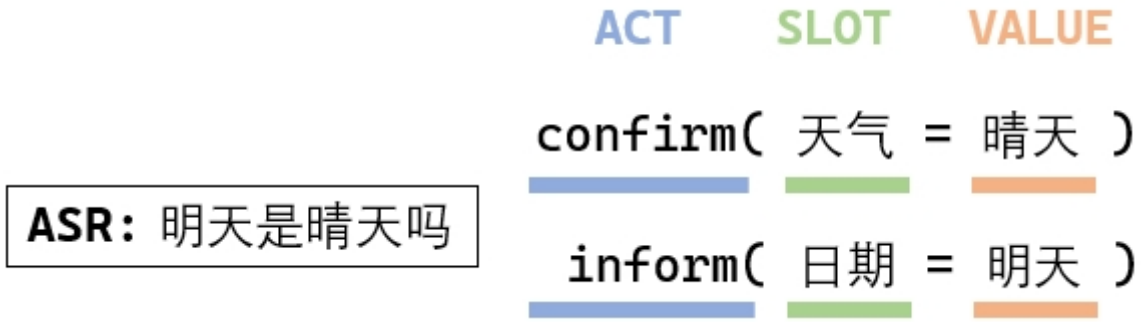
大模型语义三元组解析

陶瑞 522030910024

简介

语义三元组如下图所示，由 Act，Slot，Value 三个值构成，语义三元组的解析任务就是要从输入文本句中，解析出若干个表达语义的三元组。



在小组大作业中，我们将该任务转化成序列标注任务，然后使用深度学习的方法，将文本转化成 Word-Embedding，再通过解码器对输入文本进行标签预测，再根据标签来划分出句子中的各个槽值对。在本文中，我们从大模型的角度出发，研究直接生成语义三元组的效果。

本文研究的角度有：**不同模型的效果对比，Zero/One/Few-shot，以及 CoT（Chain-of-Thought）提示词工程在包含复杂逻辑的语义理解中的作用**

本文最终结论有两点：

1. **大模型在 Zero-Shot 下表现不好的本质原因并不是不理解语义，而是不理解未经解释的 slot 名词。**之所以 One/Few-shot 能产生很大的性能提升（能匹配正确 slot）和鲁棒性提升（多次重复实验会匹配同一个 slot），**本质原因是通过样例 ground truth 进一步阐释清楚了 slot 的语义，我们通过实验证明了，如果给出的 One/Few-Shot 样例与测试语句毫无关联，即不能帮助解释有歧义的 slot 含义，则性能不会有提升。**
2. **CoT 可以显著提升大模型对于包含逻辑推理的语义的理解。**这样包含逻辑的语句常出现于口语交谈，非书面的表达可能包含隐式的推理逻辑，而 One/Few-shot 并不能帮助大模型正确理解这些隐式逻辑，**添加 CoT 样例，并让模型也按照 CoT 输出推理过程，能显著提升语义三元组的生成性能**

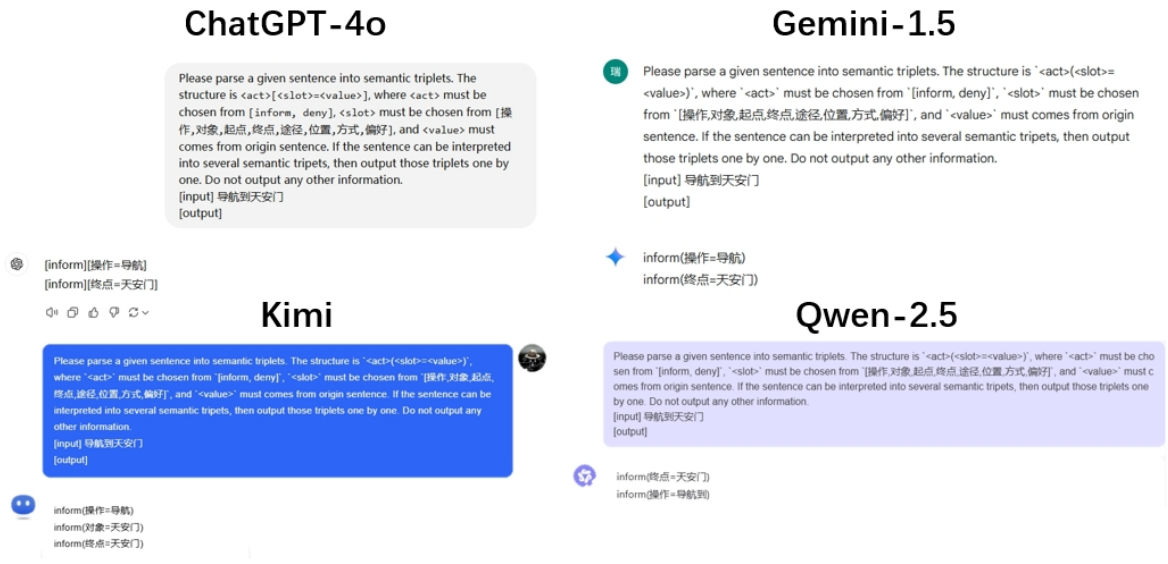
实验

本章节实验采用的大模型有：`ChatGPT-4o` `Gemini-1.5` `Kimi` `Qwen-2.5`

为了让模型能理解我们的任务，并格式化输出以排除连续文本对模型的干扰，我们首先设定一段能格式化输出的 Prompt 作为整个任务提词基准

```
Please parse a given sentence into semantic triplets.
The structure is `<act>(<slot>=<value>)`,
where `<act>` must be chosen from `[inform, deny]`,
`<slot>` must be chosen from `[操作,对象,起点,终点,途径,位置,方式,偏好]`,
and `<value>` must comes from origin sentence.
If the sentence can be interpreted into several semantic tripets,
then output those triplets one by one. Do not output any other information.
[input] 导航到天安门
[output]
```

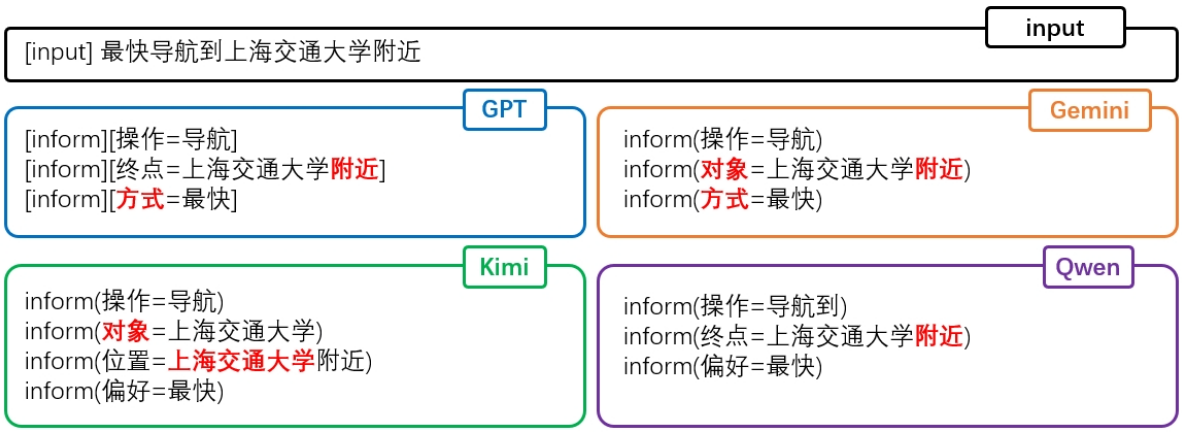
这段基准Prompt 只描述了任务，除此之外没有给出任何方法提示，在后面的所有实验中都默认包含这段 Prompt 而不会粘贴在文中。基准效果如下（在之后的实验中，我们不会再展示截图，为了节省空间只将模型回答的纯文本粘贴出来）



首先使用了一个非常简单的例句：`导航到天安门`，结果表明四个模型都能很好理解任务，都正确指出了核心的两个语义三元组 `inform(操作=导航)` 和 `inform(终点=天安门)`，并以正确的格式输出。但对于 `<slot>` 的选择仍然有些歧义和不稳定

Zero/One/Few-shot Inference

首先我们研究在没有额外提示词工程技巧的引入下，四种大模型在 Zero-shot 下的净表现。Zero-shot 指的是除了任务描述和输入以外，不给大模型提供任何示例作为参考。而 One/Few-shot 提示方法则通过输入一些类似问题和问题答案让模型参考，并在同一个 Prompt 末尾提出新的问题，以此提升模型的推理能力。

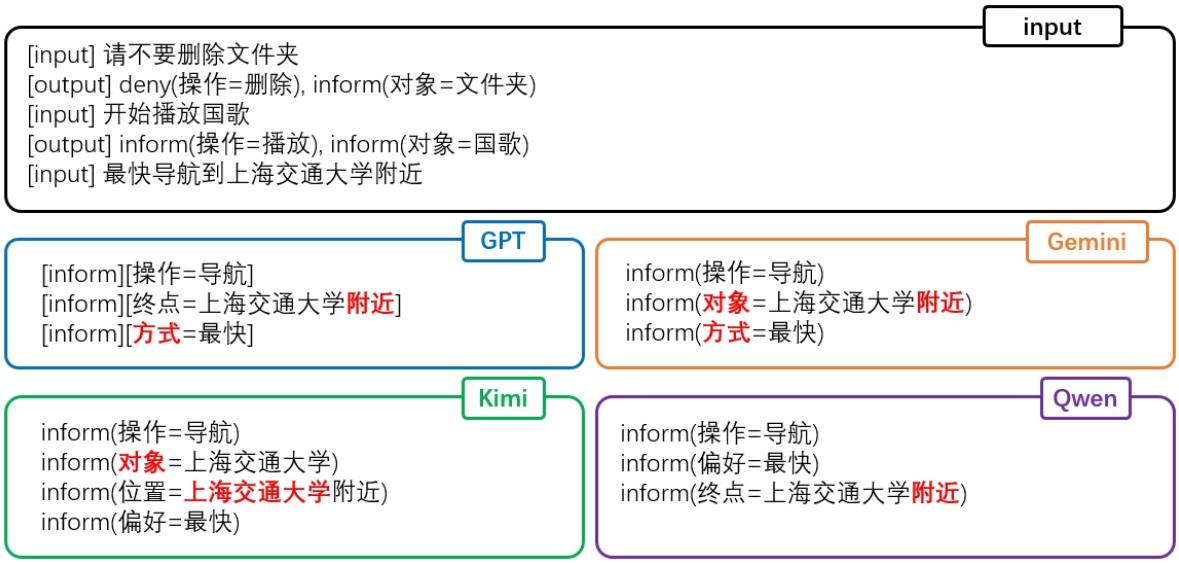


面对复杂的输入 `最快导航到上海交通大学附近`，四个模型也都能给出几乎完整的语义，但划分 `<slot>` 方面仍然有较大差异。图中所有标红的部分均为错误的标注。

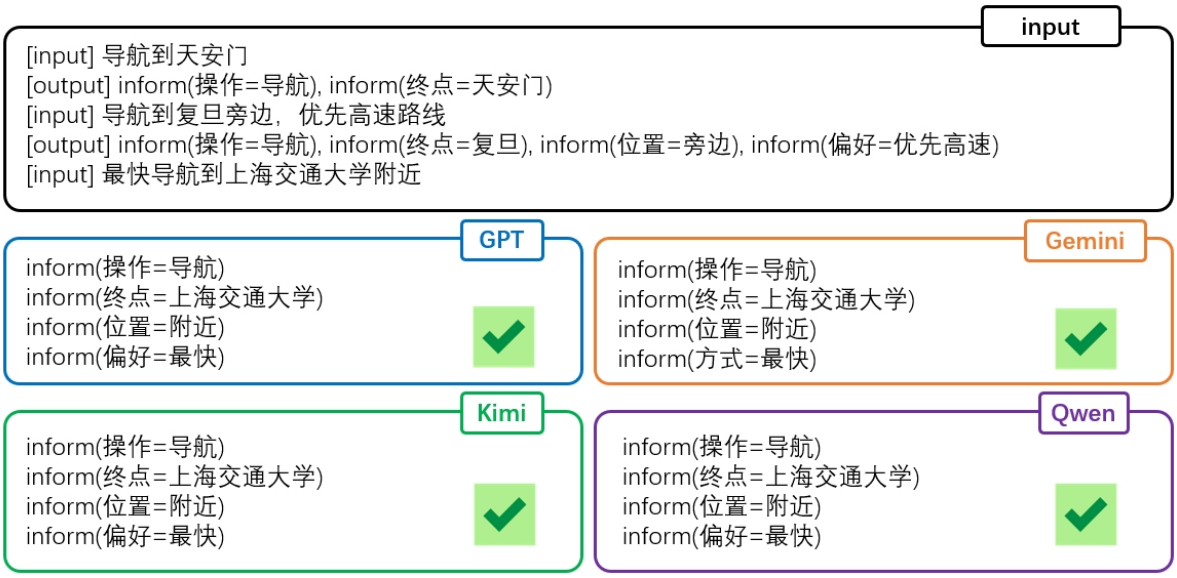
最可能是由于基准 Prompt 描述的任务中没有对几个槽值进行精确定义，模型只能望文生义。比如对于 `最快` 这个值，模型在 `slot=方式` 还是 `slot=偏好` 两种情况中无法确认。因此这引出了我们的第一个改进方法：One-Shot 和 Few-Shot，由于核心问题是对 `<slot>` 理解不足，于是我们测试两种情况：

- 提供的额外样例与 slot 理解紧密相关
- 提供 trivial 样例，或与理解歧义 slot 无关

Few-shot：提供无关样例



Few-shot：提供有关样例



结果分析

与最开始的 Zero-shot 对比，发现提供无关的样例作为 Few-shot 没有任何改善。因为提供的样例只展示了区分不同的 `value`，以及 `slot=对象` 的情况。而在有关样例中，我们即使用与测试句不同的词汇 `旁边-附近` `优先高速-最快`，大模型仍然学习到了相应 `slot` 的含义，最终四个模型全部正确解析了三元组

Chain of Thought (CoT)

CoT 是提示词工程中重要的一部分，通过让模型“有逻辑地思考”，强制其经过思维链条来完整呈现输出的所有步骤，从而一定程度上消除减轻模型“黑盒”。Takeshi 的工作证明，即使是简单的一句“Let's think step by step” 也能大幅提升模型在逻辑推理、计算任务上的准确率。因此我们融合 CoT 和 Zero/Few-shot 的技巧，进一步研究大模型解析的效果

为了发挥提示词工程的强大，必须引入一些更复杂的输入，我们给出一个包含简单逻辑理解的输入：

```
我想找一条最快的去北京南站的路线，今天是节假日估计高速路会堵车
```

通过后半句表达“高速堵车”，可以隐式地推理出“不想走高速路”

下面我们先观察在没有任何逻辑链的帮助下，仅仅通过 One-Shot 给出几个关键 slot 类别的含义提示，四个大模型能否胜任：

input

[input] 导航到复旦旁边，优先高速路线
[output] inform(操作=导航), inform(终点=复旦), inform(位置=旁边), inform(偏好=优先高速)
[input] 我想找一条最快的去北京南站的路线，今天是节假日估计高速路会堵车
[output]

GPT

inform(操作=找)
inform(终点=北京南站)
inform(方式=最快)
inform(起点=今天)
inform(位置=节假日)
inform(偏好=估计高速路会堵车)

Gemini

inform(操作=找路线)
inform(偏好=最快)
inform(终点=北京南站)
inform(方式=高速)
inform(状态=堵车)

Kimi

inform(操作=找路线)
inform(终点=北京南站)
inform(偏好=最快)
inform(考虑因素=节假日)
inform(考虑因素=高速路堵车)

Qwen

inform(操作=找)
inform(对象=路线)
inform(终点=北京南站)
inform(偏好=最快)
inform(偏好=高速路会堵车)

可以发现，四个大模型仅仅能做到通过我们 One-Shot 给出的 `位置` `偏好` 等 slot 的语义信息样例，把这特定几个三元组生成正确，**然而对于本测试样例最关键的逻辑推理部分，都给出了错误答案，甚至还将“高速路”放在了 `inform(偏好)` 里，完全背离了隐含推理的本意**

下面我们引入 CoT 提示词技术，首先让模型 `Think setp by step`，并通过详细演示了逻辑链条是如何进行的、如何推理出每个三元组的。然后在最后同样给出输入例句，CoT + Few-shot 的结果如下所示

input

Let's think step by step.
[input] 导航到复旦旁边，优先高速路线
[output] 首先表明要导航到复旦, 所以 inform(操作=导航), inform(终点=复旦), 并给“复旦”添加了修饰 `旁边` 所以有 inform(位置=旁边), 第二句话说要优先走高速, 是说路线偏好选择为高速, 所以 inform(偏好=优先高速). 综上语义三元组为: inform(操作=导航), inform(终点=复旦), inform(位置=旁边), inform(偏好=优先高速)
[input] 我想找一条最快的去北京南站的路线，今天是节假日估计高速路会堵车
[output]

GPT

1."找一条路线" indicates an intention to search for a route. This corresponds to: inform(操作=找).
2."最快的" describes the type of route preferred, indicating that the search is for the "fastest route." This corresponds to: inform(方式=最快).
3."去北京南站" indicates the destination is "北京南站," so this corresponds to: inform(终点=北京南站).
4."今天是节假日估计高速路会堵车"This part provides additional context about traffic conditions, specifically "高速路会堵车" (highways will be congested), but it does not directly suggest a change in actions or preferences. **Therefore, no new action or slot/value is needed.**
Final semantic triplets: **inform(操作=找), inform(方式=最快), inform(终点=北京南站)**

Gemini

1.“我想找一条最快的去北京南站的路线”: 这句话表达了用户想要寻找路线的意图，并且明确了目的地和偏好。
inform(操作=找路线)：用户的主要意图是寻找路线。
inform(终点=北京南站)：明确了路线的终点是北京南站。
inform(偏好=最快)：用户偏好选择最快的路线。
2. “今天是节假日估计高速路会堵车”: 这句话表达了用户对节假日高速路况的担忧，这可以理解为用户对路线选择的额外考虑。虽然没有直接要求避开高速，但暗示了用户可能不希望选择拥堵的高速路线。因此，可以将其解读为对途径方式的隐含否定或对拥堵的否定。这里我们选择解读为对途径的否定。
deny(途径=高速路)：**用户隐含地表达了不希望走高速路的倾向**

综上该句的语义三元组为：**inform(操作=找路线), inform(终点=北京南站), inform(偏好=最快), deny(途径=高速路)**

Kimi

首先表明了想要找一条路线，所以有 inform(操作=找路线)。
目的地是北京南站，所以有 inform(终点=北京南站)。
强调了想要最快的路线，所以有 inform(偏好=最快)。
提到了今天是节假日，这是一个时间条件，所以有 inform(条件=节假日)。
估计高速路会堵车，**这是一个对高速路的预测，所以有 inform(条件=高速路堵车)。**
综上语义三元组为：**inform(操作=找路线), inform(终点=北京南站), inform(偏好=最快), inform(条件=节假日), inform(条件=高速路堵车)**

Qwen

1.表明用户想要找一条去北京南站的路线, 因此我们有:
inform(操作=找), inform(终点=北京南站)
2.用户希望这条路线是最快的, 这涉及到用户的偏好:
inform(偏好=最快)
3.用户提到今天是节假日, 并估计高速路会堵车, 这是对可能选择的路线的一种否定性预期或警告: inform(偏好=高速路会堵车) 或者更准确地表示为 deny(方式=高速), 因为考虑到节假日高速可能会拥堵，**用户可能不倾向于选择高速公路。**

综上所述，语义三元组可以表示为：**inform(操作=找), inform(终点=北京南站), inform(偏好=最快), deny(方式=高速)**

结果分析

我们发现 CoT 的效果非常显著，Gemini 和 Qwen 大模型在我们使用 CoT 作为提示词的改进后，**完全正确地给出了语义三元组。**两者都在读到 “估计高速路会堵车” 后明确地进行了推理，并意识到这意味着 “不倾向于走高速路”，并在最终的三元组中用 `deny` 表达了这一逻辑

注意到我们并没有在 One-shot 中给出任何包含 `deny` 的样例，而大模型通过逻辑推理仍然可以正确选择 `ACT=deny`

Kimi 在此任务中的表现略差，他仍然认为这句话表明了要走高速路。而 ChatGPT 在推理过程中虽然没有正确生成 `deny(偏好=高速)`，但是相比于 ChatGPT 在前面的对比实验中给出的语义三元组，经过 CoT 后已经去除了 `inform(偏好=估计高速路会堵车)`，ChatGPT 在逻辑推理中也提到，这句话并不表明想走高速路，所以最终它没有生成任何与 “高速” 相关的三元组，这也可以算作一种进步。

综上所述，实验结果表明面对含有隐式推理逻辑的句子，One/Few-shot 不能帮助大模型正确理解这些隐式逻辑，而添加 CoT 样例，并让模型也按照 CoT 输出推理过程，能显著提升语义三元组的生成性能

讨论-大模型时代的 SLU 现状和未来

大模型的技术优势

首先最显著的影响是，BERT、GPT等大模型极大提升了 SLU 任务的性能，尤其是语义解析、意图识别，槽填充等关键子任务上。我们在小组作业中也通过实验验证了这一点，基于预训练大模型 BERT（及其变体）作为编码器的深度学习流程，在没有任何其他奇技淫巧的情况下，就已经 extremely outperform 传统深度学习结构。

不仅如此，近五年发展出越来越多的基于预训练大模型的端到端方法（Parisa Haghani），逐渐替代了传统的两阶段流程（ASR + SLU），减少了在训练过程中由 ASR 传递的错误（比如在小组作业中的数据清洗）

同时，通过微调或提示学习，大模型还可以高效泛化到任何其他场景下工作，在效率和效果上都完胜传统方法

技术挑战和未来

大模型毕竟“大”，其极端高效的泛化性能也是建立在 scaling law 上的表现，想要利用大模型的种种技术优势，必不可少的是训练开销，因此模型效率和规模需要做出权衡（和微软一样钱花不完的除外）。因此模型压缩、蒸馏等轻量化技术是大模型的一个研究方向，尤其对于一些工作于实时场景、日常场景的 SLU 任务（如实时机器翻译、情感识别）需要探索让大模型在边缘设备上运行。

在一些特定的 SLU 任务，比如临床对话系统、临床医疗诊断等专业性强、容错率较低的任务中，大模型的可解释性欠缺还是硬伤。已经有不少工作，比如 ClinicalBERT，Med-BERT，前者是在 BERT 上微调的，后者在医学语料库上重新预训练了一个新的大模型，等等一系列工作，但他们都一直缺乏足够的可解释性作为扎实的依据。毕竟 Transformer 相对而言有较好的可解释性基础，因此这大概也是大模型在 SLU 未来的重要方向之一。

总结

本文使用大模型来解决语义三元组任务，并研究了 Zero-shot，One-shot，Few-shot 下不同大模型的效果，以及研究了一个非常重要的提示词工程技巧 CoT，最终在我们有限的测试中，Qwen-2.5 和 Gemini-1.5 大模型的表现最好。本文的结论有：

1. **大模型在 Zero-Shot 下表现不好的本质原因并不是不理解语义，而是不理解未经解释的 slot 名词。**之所以 One/Few-shot 能产生很大的性能提升（能匹配正确 slot）和鲁棒性提升（多次重复实验会匹配同一个 slot），**本质原因是通过样例 ground truth 进一步阐释清楚了 slot 的语义，我们通过实验证明了，如果给出的 One/Few-Shot 样例与测试语句毫无关联，即不能帮助解释有歧义的 slot 含义，则性能不会有提升。**
2. **CoT 可以显著提升大模型对于包含逻辑推理的语义的理解。**这样包含逻辑的语句常出现于口语交谈，非书面的表达可能包含隐式的推理逻辑，而 One/Few-shot 并不能帮助大模型正确理解这些隐式逻辑，**添加 CoT 样例，并让模型也按照 CoT 输出推理过程，能显著提升语义三元组的生成性能**