大模型 – Prompt 工程

作者：Calvin
QQ: 179209347
Mail: 179209347@qq.com

介绍

笔记简介：

- 面向对象：深度学习初学者
- 依赖课程：**线性代数，统计概率**，优化理论，图论，离散数学，微积分，信息论

知乎专栏：

https://zhuanlan.zhihu.com/p/693738275

Github & Gitee 地址：

https://github.com/mymagicpower/AIAS/tree/main/deep_learning

https://gitee.com/mymagicpower/AIAS/tree/main/deep_learning

*** 版权声明：**

- 仅限用于个人学习
- 禁止用于任何商业用途

基本概念

- **大语言模型：**是一种基于概率的生成模型，能够利用输入上下文生成潜在的输出文本。该模型的训练过程采用概率模型的最大似然估计，通过学习大量文本数据来捕捉语言的统计规律。

- **基座模型：**是指一个已经经过大规模预训练的语言模型，如GPT-3.5 Turbo。它是基于概率的生成模型，能够根据给定的输入上下文生成可能的输出文本。基座模型通常具有广泛的语言理解和生成能力，可以应用于各种自然语言处理任务，如文本生成、问答系统、摘要生成等。

- **指令模型：**是在基座模型的基础上进行微调或定制的模型，用于特定的任务或应用场景。输入是指令，输出是对这些指令的正确回复。有时还会采用RLHF (reinforcement learning from human feedback, 人类反馈强化学习)技术，根据人类对模型输 出的反馈进一步增强模型遵循指令的能力。

思维链提示

随着语言模型规模的扩大，它获得了更多的知识和强大的语境学习能力。然而，仅仅增加语言模型的规模并不能显著提高其推理能力，例如常识推理、逻辑推理和数学推理。
除了将问题输入模型，还将类似问题的解题思路或步骤输入模型，使得模型不仅输出最终结果，还输出中间步骤，从而提升模型的推理能力，这种方法被称为**思维链提示**。
同样，在面对复杂任务或问题时，大语言模型可以展示出良好的规划能力。通过引导模型将复杂问题分解为多个较简单的子问题，并逐一解决这些子问题，可以帮助模型得出最终答案。这种策略被称为**由少至多提示**。
零样本思维链提示方式，只需要简单地告知模型“让我们一步一步思考（Let’s think step by step）”，模型就能够自动输出中间步骤。

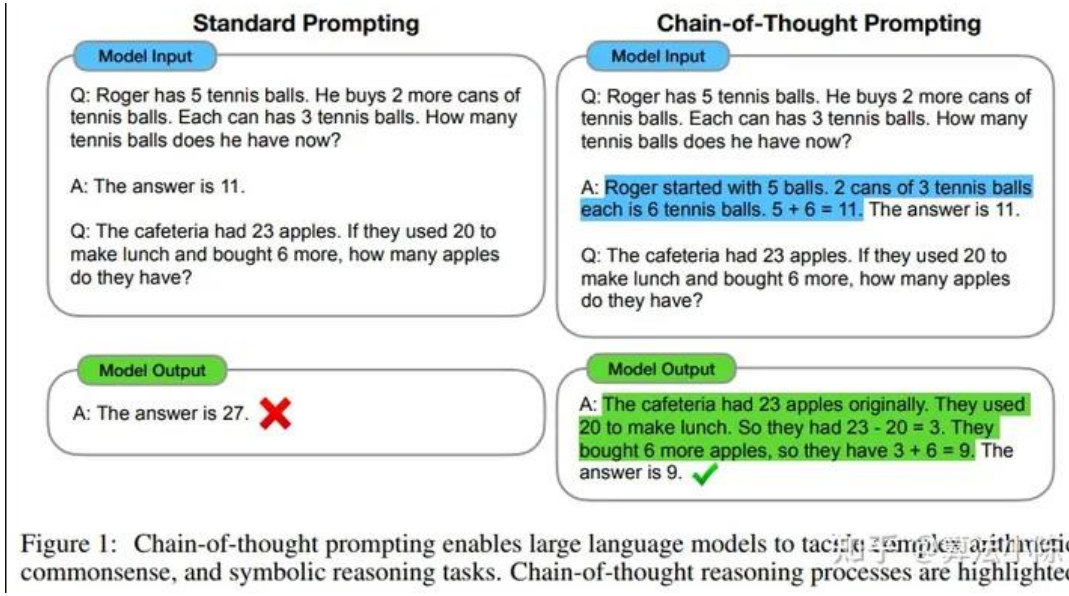


Figure 1: Chain-of-thought prompting enables large language models to tackle simple arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

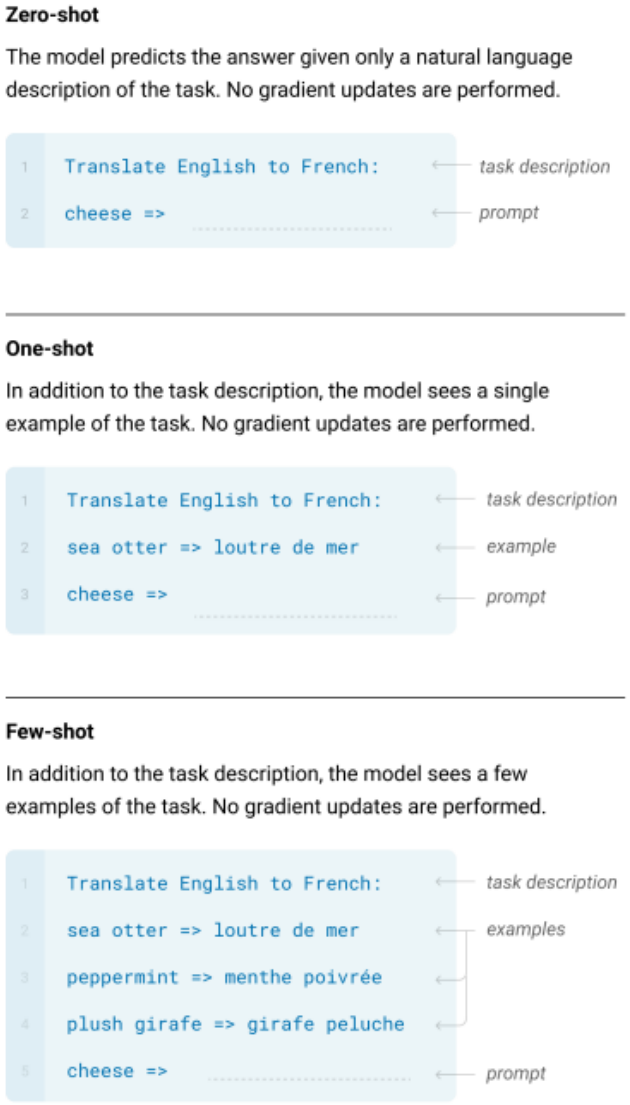
论文：Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

三种不同的 prompt 方法

Few-Shot（FS）： 是指模型在推理时给予少量样本，但不允许进行权重更新。Few-shot 的工作方式是提供 K 个样本，然后期望模型生成对应的结果。Few-shot 的主要优点是大幅度降低了对特定任务数据的需求，并减少了从微调数据集中学习过度狭窄分布。此外，仍需要一小部分特定任务的数据。

One-Shot（1S）： 与 Few-Shot 类似，只允许一个样本（除了任务的自然语言描述外）。将 One-Shot 与 Few-Shot、Zero-Shot 区分开的原因是它最接近某些任务与人类沟通的方式。相比之下，如果没有示例，有时很难传达任务的内容或格式。

Zero-Shot（0S）： 和 One-shot 类似，但不允许提供样本，只给出描述任务的自然语言指令。该方法提供了最大的方便性、稳健性以及避免虚假相关的可能性，但也是最具挑战性的设置。Zero-shot 是最接近人类执行任务的方法，例如图中的翻译示例，人类可能仅凭文本指令就知道该做什么。



论文 Language Models are Few-Shot Learners

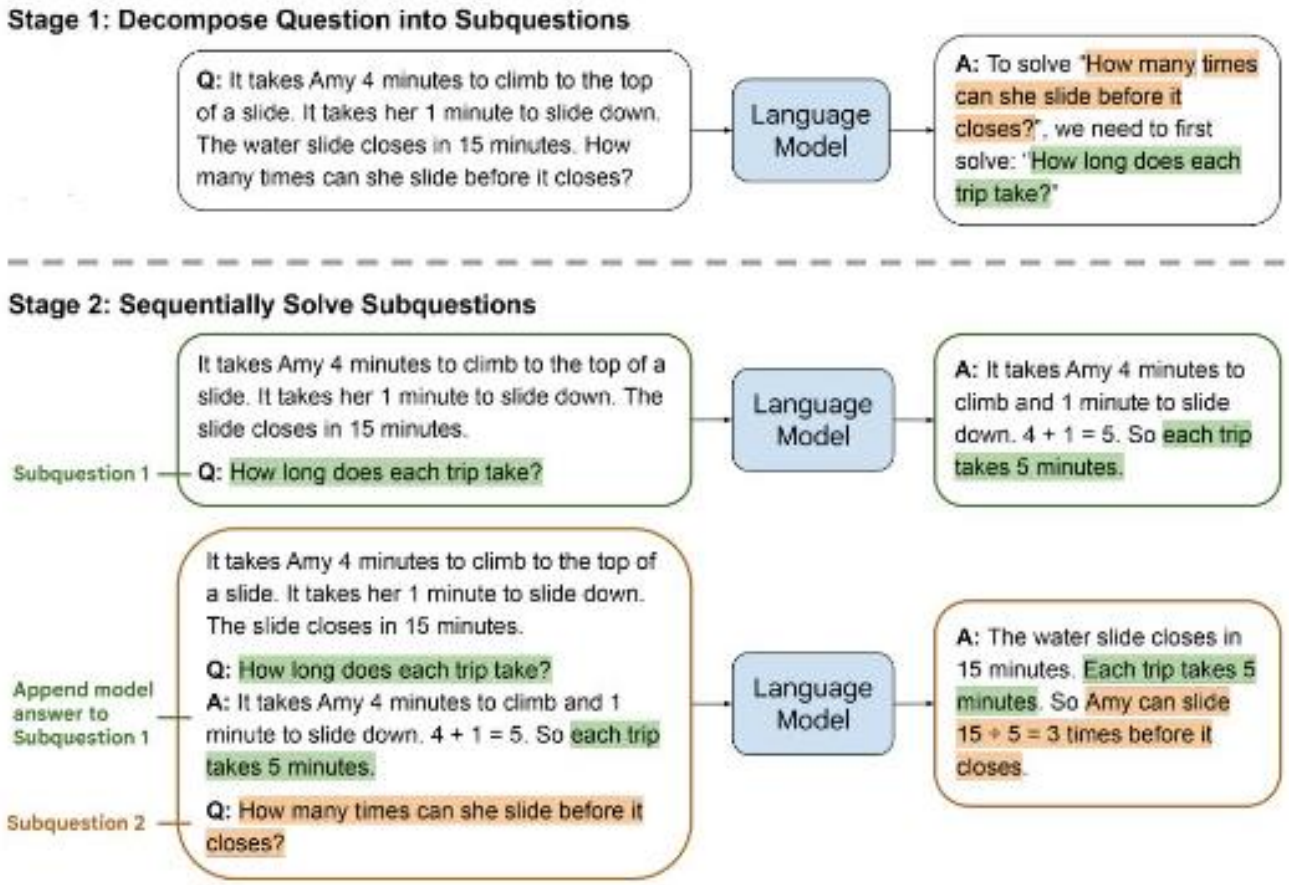
由少至多提示

当面对复杂任务或问题时，人类通常倾向于将其转化为多个更容易解决的子任务/子问题，并逐一解决它们，得到最终想要的答案或者结果。这种能力就是通常所说的任务分解能力。

基于这种问题解决思路，研究人员们提出了由少至多提示（Least-to-Most Prompting）方法。这种方法试图利用大语言模型的规划能力，将复杂问题分解为一系列的子问题并依次解决它们。

由少至多提示流程如图所示，主要包含两个阶段：

- 在问题分解阶段中，模型的输入包括k×[原始问题，子问题列表] 的组合，以及要测试的原始问题；
- 在逐步解决子问题阶段中，模型的输入包括k×[原始问题，m×（子问题，子答案）] 元组，以及要测试的原始问题和当前要解决的子问题。



论文 LEAST-TO-MOST PROMPTING ENABLES COMPLEX REASONING IN LARGE LANGUAGE MODELS

指令模型

以下是三个指令模型的示例：

- **文本摘要生成模型：**该指令模型针对文本摘要生成任务进行了定制。通过提供示例的长文本和对应的摘要，模型可以学习如何从输入文本中提取关键信息并生成简洁准确的摘要。

- **问答系统模型：**这个指令模型专门用于问答任务。通过提供问题和对应的答案，模型可以学习如何理解问题并生成准确的答案。该模型可以应用于各种问答场景，如常见问题回答、专业知识查询等。

- **机器翻译模型：**该指令模型用于机器翻译任务，可以将一种语言的文本翻译成另一种语言。通过提供双语对照的句子对，模型可以学习如何将输入语言的句子转化为输出语言的句子，实现跨语言的翻译功能。

以下是针对不同指令模型的一些示例指令：

1. 文本摘要生成模型的指令示例：
   - 输入：一篇新闻文章。
   - 指令：生成一段包含该文章核心信息的摘要，长度在100个字符以内。
   - 输出：生成的文章摘要。

2. 问答系统模型的指令示例：
   - 输入：一段文章和一个问题。
   - 指令：回答问题，并提供相关的支持文本。
   - 输出：回答问题的文本和支持文本。

3. 机器翻译模型的指令示例：
   - 输入：一段英文文本。
   - 指令：将该文本翻译为法文。
   - 输出：翻译后的法文文本。

Prompt

大模型的Prompt是指向模型提供输入的文本或问题，以**引导其生成**响应。Prompt的设计可以显著影响模型的输出。
以下是一些简单的Prompt示例：

1. 问题回答：
   - 输入：谁是美国第一位总统？
   - 输出：美国第一位总统是乔治·华盛顿。

2. 描述性问题：
   - 输入：请描述一只狗的外观特征。
   - 输出：狗通常有四条腿，毛色各异，有各种不同的品种，体型大小也不同。

3. 继续文本：
   - 输入：在一个遥远的星球上，有一位勇敢的战士...
   - 输出：他奋勇杀敌，保卫着自己的家园，从未退缩过。

4. 问题解决：
   - 输入：如何解决数独谜题？
   - 输出：解决数独谜题的一种方法是通过逐行、逐列、逐宫填写数字，确保每个数字在每一行、每一列和每个宫中都只出现一次。

这些是简单的Prompt示例，你可以根据具体的任务或问题来设计更复杂的Prompt，以引导模型生成你所期望的回答或输出。

Prompt Engineering(提示工程)

Prompt Engineering，即是针对特定任务构造能充分发挥大模型能力的 Prompt 的技巧。

大模型 Prompt Engineering 的关键在于设计合适的提示，以引导模型生成期望的结果。
合理的提示设计应该具备以下特点：

- 清晰明确：提示应该清楚地描述任务要求，避免歧义和模糊性。
- 信息完整：提示应该提供足够的信息，使模型能够理解任务的上下文和约束条件。
- 一致性：提示应该与生成结果的期望一致，避免引导模型生成不符合预期的输出。
- 多样性：通过设计多样的提示，可以引导模型生成多样化的结果，提高生成的创造性和多样性。

prompt的结构

大模型prompt结构示例：

1. Context（可选）
   1.1 角色：描述参与对话或任务的角色，例如学生、教师、工程师等。
   1.2 任务：概述对话或任务的目标和背景。
   1.3 知识：提供与任务相关的知识或背景信息。

2. Instruction（必选）
   2.1 步骤：明确描述完成任务的步骤或要求。
   2.2 思维链：列出解决问题的思维过程或步骤。
   2.3 示例：提供一个示例来解释或说明任务的要求。

3. Input Data（必选）
   3.1 句子：提供一个或多个句子作为模型输入。
   3.2 文章：提供一个文章段落或完整的文章作为模型输入。
   3.3 问题：提供一个或多个问题作为模型输入。

4. Output Indicator（可选）：
定义模型输出的期望形式或指示，可以是预期的答案格式、输出类型或其他指标。

一个示例：

1. Context:
   1.1 角色：学生
   1.2 任务：计算一个数的平方
   1.3 知识：学生已经学习了平方的概念和计算方法。

2. Instruction:
   2.1 步骤：请计算给定数的平方。
   2.2 思维链：了解平方的定义 -> 应用平方计算公式 -> 进行乘法运算 -> 得出平方结果。
   2.3 示例：例如，计算2的平方，即 2*2，结果为4。

3. Input Data:
   3.1 句子：计算3的平方。
   3.2 文章：请计算给定数的平方：5的平方是多少?
   3.3 问题：求5的平方是多少?

4. Output Indicator:
   输出结果应该是平方的数值，例如对于输入"计算3的平方"，期望输出为9。

prompt的结构

你是文字提取器**（角色）**，你要结构化的提取用户描述中动作和条件。 **（任务）**
条件是指：某个事件或者行动所需要满足的一些前提条件或要求，如：10分钟后，车内温度降至15度后、到达目的地后、无。条件可以有多个。没有写“无”。 **（明确定义字段）**
动作是指：对某个动作进行命令或描述。如：打开空调、关闭窗户、播放音乐、提醒我买菜、导航到目的地、保持空调保持26度、无。动作可以有多个。
输出完毕后结束，不要生成新的用户输入，不要新增内容。 **（明确定义字段）**

示例： **（给出示例）**
用户输入：到公司附近后，提醒我买杯美式咖啡
动作1:提醒我买杯美式咖啡
条件1:到公司附近后

用户输入：放学后，播放动画片，若太累了，播放摇篮曲
动作1:播放动画片
条件1:放学后
动作2:播放摇篮曲
条件2:太累了

用户输入：打开车窗，关闭空调，5分钟后关闭车窗，打开空调
动作1:打开车窗
条件1:无
动作2:关闭空调
条件2:无
动作3:关闭车窗
条件3:5分钟后
动作4:打开空调
条件4:无

请根据以下文本，按照模版输出内容。 **（按格式输出）**
用户输入：{用户query}

例子

你是一个智能助手，帮我记录或者查询生日信息。请从以下句子中抽取信息：意图、时间、人物、关系
意图只能是记录信息、查询信息、修改信息、删除信息**（枚举）**,当用户陈述生日时，意图是记录信息**（信息解释）**
关系只能是亲人、朋友、未知**（枚举）**

示例"""**（样例）**
输入: 妈妈生日是哪天
输出:
意图:查询信息
时间:待查询
人物: 妈妈
关系:亲人
"""

输入：我儿子的生日是三月初七
输出：

设计原则 1 - 清晰明确

在与语言模型交互时，您需要牢记一点:以**清晰、具体**的方式表达您的需求。假设您面前坐 着一位来自外星球的新朋友，其对人类语言和常识都一无所知。在这种情况下，您需要把想表达的意图 讲得非常明确，不要有任何歧义。同样的，在提供 Prompt 的时候，也要以足够详细和容易理解的方 式，把您的需求与上下文说清楚。

并不是说 Prompt 就必须非常短小简洁。事实上，在许多情况下，更长、更复杂的 Prompt 反而会让语言模型更容易抓住关键点，给出符合预期的回复。原因在于，复杂的 Prompt 提供了更丰富的上下文和细节，让模型可以更准确地把握所需的操作和响应方式。

设计原则 1 - 清晰明确 - 使用分隔符清晰地表示输入的不同部分

你可以选择用 ```，"""，< >，<tag> </tag>，：等做分隔符，只要能明确起到隔断作用即可。

```
from tool import get_completion
text = f"""
您应该提供尽可能清晰、具体的指示，以表达您希望模型执行的任务。\
这将引导模型朝向所需的输出，并降低收到无关或不正确响应的可能性。\
不要将写清晰的提示词与写简短的提示词混淆。\
 在许多情况下，更长的提示词可以为模型提供更多的清晰度和上下文信息，从而导致更详细和相关的输出。
"""
# 需要总结的文本内容
prompt = f"""
把用三个反引号括起来的文本总结成一句话。
```{text}```
"""
# 指令内容，使用 ``` 来分隔指令和待总结的内容
response = get_completion(prompt)
print(response)
```
```

设计原则 1 - 清晰明确 - 寻求结构化的输出

有时候我们需要语言模型给我们一些**结构化的输出**，而不仅仅是连续的文本。例如JSON、HTML等。这种输出非常适合在代码 中进一步解析和处理。例如，您可以在 Python 中将其读入字典或列表中。

在以下示例中，我们要求 GPT 生成三本书的标题、作者和类别，并要求 GPT 以 JSON 的格式返回给我 们，为便于解析，我们指定了 Json 的键。

```
prompt = f"""
请生成包括书名、作者和类别的三本虚构的、非真实存在的中文书籍清单，\
并以 JSON 格式提供，其中包含以下键:book_id、title、author、genre。
"""
response = get_completion(prompt)
print(response)
```

```
{
        "books": [{
                "book_id": 1,
                "title": "迷失的时光",
                "author": "张三",
                "genre": "科幻"
        }, {
                "book_id": 2,
                "title": "幻境之门",
                "author": "李四",
                "genre": "奇幻"
        }, {
                "book_id": 3,
                "title": "虚拟现实",
                "author": "王五",
                "genre": "科幻"
        }]
}
```

设计原则 1 - 清晰明确 - 要求模型检查是否满足条件

在如下示例中，我们将分别给模型两段文本，分别是制作茶的步骤以及一段没有明确步骤的文本。我们将要求模型判断其是否包含一系列指令，如果包含则按照给定格式重新编写指令，不包含则回答“未提供 步骤”。

```
text_1 = f"""
泡一杯茶很容易。首先，需要把水烧开。在等待期间，拿一个杯子并把茶包放进去。\
一旦水足够热，就把它倒在茶包上。等待一会儿，让茶叶浸泡。几分钟后，取出茶包。\
如果您愿意，可以加一些糖或牛奶调味。就这样，您可以享受一杯美味的茶了。
"""
prompt = f"""
您将获得由三个引号括起来的文本。如果它包含一系列的指令，则需要按照以下格式重新编写这些指令:
第一步 - ...
第二步 - ...
第N步 - ...
如果文本中不包含一系列的指令，则直接写“未提供步骤”。"
\"\"\"{text_1}\"\"\"
"""
response = get_completion(prompt)
print("Text 1 的总结:")
print(response)
```

```
Text 1 的总结:
第一步 - 把水烧开。
第二步 - 拿一个杯子并把茶包放进去。
第三步 - 把烧开的水倒在茶包上。
第四步 - 等待几分钟，让茶叶浸泡。
第五步 - 取出茶包。
第六步 - 如果需要，加入糖或牛奶调味。
第七步 - 就这样，您可以享受一杯美味的茶了。
```

设计原则 1 - 清晰明确 - 提供少量示例

"Few-shot" prompting，即在要求模型执行实际任务之前，给模型一两个已完成的样例，让模型了解我们的要求和期望的输出样式。 利用少样本样例，我们可以轻松“预热”语言模型，让它为新的任务做好准备。这是一个让模型快速上手新任务的有效策略。

```
prompt = f"""
您的任务是以一致的风格回答问题。
<孩子>: 请教我何为耐心。
<祖父母>: 挖出最深峡谷的河流源于一处不起眼的泉眼;最宏伟的交响乐从单一的音符开始;最复杂的挂毯以一根孤独的线开始编织。
<孩子>: 请教我何为韧性。
"""
response = get_completion(prompt) print(response)
```

```
<祖父母>: 韧性是一种坚持不懈的品质，就像一棵顽强的树在风雨中屹立不倒。它是面对困难和挑战时不屈不 挠的精神，能够适应变化和克服逆境。韧性是一种内在的力量，让我们能够坚持追求目标，即使面临困难和挫折 也能坚持不懈地努力。
```

设计原则2 - 给模型步骤提示

在设计 Prompt 时，给予语言模型**充足的推理步骤**非常重要。能让语言模型充分“思考”，输出结果也将更可靠准确。

接下来我们将通过给定一个复杂任务，给出完成该任务的一系列步骤，来展示这一策略的效果。

首先我们描述了杰克和吉尔的故事，并给出提示词执行以下操作:
首先，用一句话概括三个反引号限定 的文本。
第二，将摘要翻译成英语。
第三，在英语摘要中列出每个名称。
第四，输出包含以下键的 JSON 对象:英语摘要和人名个数。要求输出以换行符分隔。

```
prompt_2 = f"""
1-用一句话概括下面用<>括起来的文本。
2-将摘要翻译成英语。
3-在英语摘要中列出每个名称。
4-输出一个 JSON 对象，其中包含以下键:English_summary，num_names。

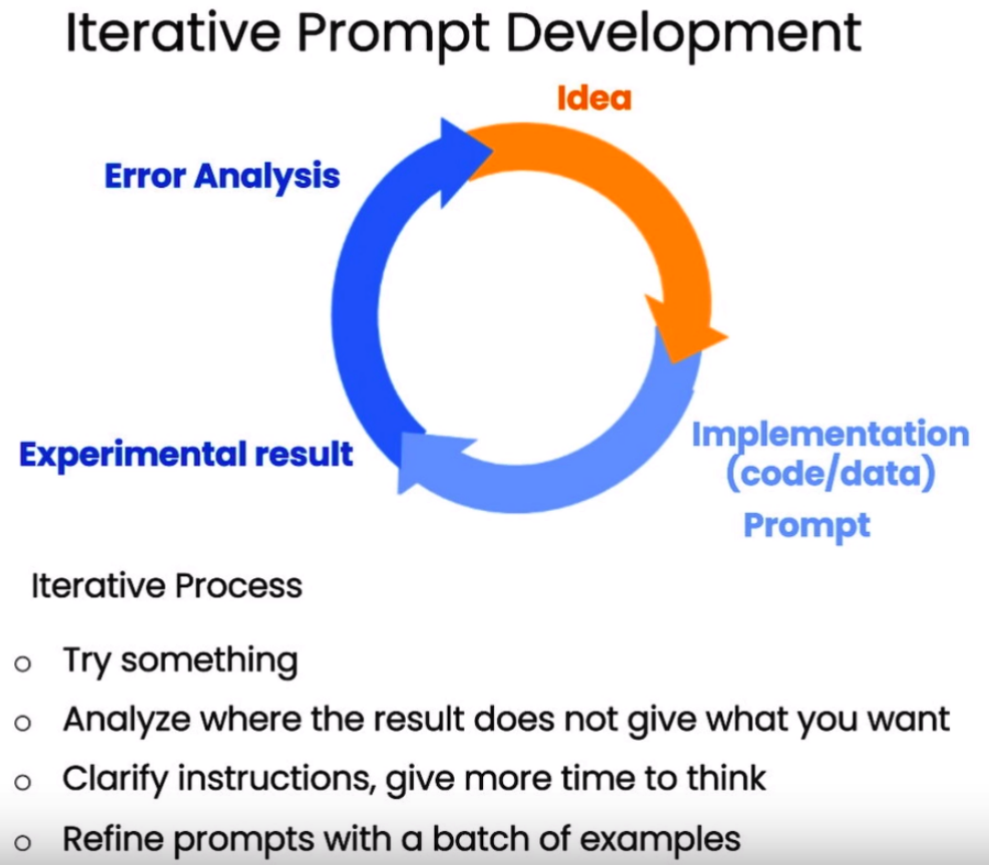
请使用以下格式:
文本:<要总结的文本>
摘要:<摘要>
翻译:<摘要的翻译>
名称:<英语摘要中的名称列表>
输出 JSON:<带有 English_summary 和 num_names 的 JSON>

Text: <{text}>
"""
response = get_completion(prompt_2)
print("\nprompt 2:")
print(response)
```

提示迭代优化

在开发大语言模型应用时，很难通过第一次尝试就得到完美适用的 Prompt。但关键是要有一个**良好的迭代优化过程**，以不断改进 Prompt。相比训练机器学习模型，Prompt 的一次成功率可能更高，但仍需要通过多次迭代找到最适合应用的形式。有了任务想法后，可以先编写初版 Prompt，注意清晰明确并给模型充足思考时间。

运行后检查结果，如果不理想，则分析 Prompt 不够清楚或思考时间不够等原因，做出改进，再次运行。如此循环多次，终将找到适合应用的 Prompt。



提示优化1 - 解决生成文本太长

在 Prompt 中添加长度限制，要求生成更简洁的文案。提取回答并根据空格拆分，中文答案为97个字，较好地完成了设计要求。

- 当在 Prompt 中设置长度限制要求时，语言模型生成的输出长度不总能精确符合要求，但基本能控制在 可接受的误差范围内。比如要求生成50词的文本，语言模型有时会生成60词左右的输出，但总体接近预 定长度。
- 这是因为**语言模型在计算和判断文本长度时依赖于分词器**，而分词器在字符统计方面不具备完美精度。目前存在多种方法可以尝试控制语言模型生成输出的长度，比如指定语句数、词数、汉字数等。

```
# 优化后的 Prompt，要求生成描述不多于 50 词
prompt = f"""
您的任务是帮助营销团队基于技术说明书创建一个产品的零售网站描述。

根据```标记的技术说明书中提供的信息，编写一个产品描述。

使用最多50个词。

技术规格:```{fact_sheet_chair}```

"""
response = get_completion(prompt)
print(response)
```

提示优化2 - 添加表格描述

继续添加指引，要求提取产品尺寸信息并组织成表格，并指定表格的列、表名和格式;再将所有内容格 式化为可以在网页使用的 HTML。

```
# 更进一步
prompt = f"""
您的任务是帮助营销团队基于技术说明书创建一个产品的零售网站描述。
根据```标记的技术说明书中提供的信息，编写一个产品描述。
该描述面向家具零售商，因此应具有技术性质，并侧重于产品的材料构造。
在描述末尾，包括技术规格中每个7个字符的产品ID。

在描述之后，包括一个表格，提供产品的尺寸。表格应该有两列。第一列包括尺寸的名称。第二列只包括英寸的测量值。
给表格命名为“产品尺寸”。
将所有内容格式化为可用于网站的HTML格式。将描述放在<div>元素中。

技术规格: ```{fact_sheet_chair}```
"""

response = get_completion(prompt)
print(response)
```

场景1 - 文本概括

以商品评论的总结任务为例:对于电商平台来说，网站上往往存在着海量的商品评论，这些评论反映了所有客户的想法。如果我们拥有一个工具去概括这些海量、冗长的评论，便能够快速地浏览更多评论， 洞悉客户的偏好，从而指导平台与商家提供更优质的服务。

接下来我们提供一段在线商品评价作为示例，可能来自于一个在线购物平台，例如亚马逊、淘宝、京东 等。评价者为一款熊猫公仔进行了点评，评价内容包括商品的质量、大小、价格和物流速度等因素，以 及他的女儿对该商品的喜爱程度。

```
prod_review = """
这个熊猫公仔是我给女儿的生日礼物，她很喜欢，去哪都带着。
公仔很软，超级可爱，面部表情也很和善。但是相比于价钱来说，
它有点小，我感觉在别的地方用同样的价钱能买到更大的。 快递
比预期提前了一天到货，所以在送给女儿之前，我自己玩了会。
 """
```

场景1 - 文本概括

1.1 限制输出文本长度

我们首先尝试将文本的长度限制在30个字以内。

```
from tool import get_completion
prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。 请对三个反
引号之间的评论文本进行概括，最多30个字。
评论: ```{prod_review}```

 """
response = get_completion(prompt)
print(response)
```

```
熊猫公仔软可爱，女儿喜欢，但有点小。快递提前一天到货。
```

场景1 - 文本概括

1.2 设置关键角度侧重

在某些情况下，我们会针对不同的业务场景对文本的侧重会有所不同。例如，在商品评论文本中，物流部门可能更专注于运输的时效性，商家则更关注价格和商品质量，而平台则更看重整体的用户体验。 我们可以通过增强输入提示(Prompt)，来强调我们对某一特定视角的重视。

1.2.1 侧重于快递服务

```
prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。
请对三个反引号之间的评论文本进行概括，最多30个字，并且侧重在快递服务上。

评论: ```{prod_review}```
"""

response = get_completion(prompt)
print(response)
```

```
快递提前到货，公仔可爱但有点小。
```

1.2.2 侧重于价格与质量

```
prompt = f"""
您的任务是从电子商务网站上生成一个产品评论的简短摘要。
请对三个反引号之间的评论文本进行概括，最多30个词汇，并且侧重在产品价格和质量上。

评论: ```{prod_review}```
"""

response = get_completion(prompt)
print(response)
```

```
可爱的熊猫公仔，质量好但有点小，价格稍高。快递提前到货。
```

场景1 - 文本概括

1.3 关键信息提取

虽然我们通过添加关键角度侧重的 Prompt ，确实让文本摘要更侧重于某一特定方面，然 而，我们可以发现，在结果中也会保留一些其他信息，比如偏重价格与质量角度的概括中仍保留了“快递 提前到货”的信息。如果我们只想要提取某一角度的信息，并过滤掉其他所有信息，则可以要求 LLM 进行 **文本提取(Extract)** 而非概括(Summarize)。

```
prompt = f"""
您的任务是从电子商务网站上的产品评论中提取相关信息。
请从以下三个反引号之间的评论文本中提取产品运输相关的信息，
最多30个词汇。

评论: ```{prod_review}```
"""
response = get_completion(prompt)
print(response)
```

```
产品运输相关的信息:快递提前一天到货。
```

场景2 - 推断 - 情感推断

1.1 情感倾向分析

让我们以一则电商平台上的台灯评论为例，通过此例，我们将学习如何对评论进行情感二分类(正面/负面)。

```
lamp_review = """
我需要一盏漂亮的卧室灯，这款灯具有额外的储物功能，价格也不算太高。\
我很快就收到了它。在运输过程中，我们的灯绳断了，但是公司很乐意寄送了一个新的。\
几天后就收到了。这款灯很容易组装。我发现少了一个零件，于是联系了他们的客服，他们很快就给我寄来了缺 失的零件!\
在我看来，Lumina 是一家非常关心顾客和产品的优秀公司!
"""
```

```
prompt = f"""
以下用三个反引号分隔的产品评论的情感是什么?

用一个单词回答:「正面」或「负面」。

评论文本: ```{lamp_review}```
"""
response = get_completion(prompt) print(response)
```

```
正面
```

场景2 - 推断 - 情感推断

1.2 识别情感类型

我们将继续使用之前的台灯评论，但这次我们会试用一个新的 Prompt 。我们希望模型能够识别出评论作者所表达的情感，并且将这些情感整理为一个不超过五项的列表。

```
# 中文
prompt = f"""
识别以下评论的作者表达的情感。包含不超过五个项目。将答案格式化为以逗号分隔的单词列表。

评论文本: ```{lamp_review}```
"""
response = get_completion(prompt) print(response)
```

```
满意,感激,赞赏,信任,满足
```

场景2 - 推断 - 情感推断

1.3 识别愤怒

对于许多企业来说，洞察到顾客的愤怒情绪是至关重要的。这就引出了一个分类问题:下述的评论作者 是否流露出了愤怒?因为如果有人真的情绪激动，那可能就意味着需要给予额外的关注，因为每一个愤 怒的顾客都是一个改进服务的机会，也是一个提升公司口碑的机会。这时，客户支持或者客服团队就应 该介入，与客户接触，了解具体情况，然后解决他们的问题。

```
# 中文
prompt = f"""
以下评论的作者是否表达了愤怒? 评论用三个反引号分隔。给出是或否的答案。

评论文本: ```{lamp_review}```
"""
response = get_completion(prompt) print(response)
```

```
否
```

场景2 - 推断 - 信息提取

2.1 商品信息提取

信息提取是自然语言处理(NLP)的重要组成部分，它帮助我们从文本中抽取特定的、我们关心的信 息。我们将深入挖掘客户评论中的丰富信息。在接下来的示例中，我们将要求模型识别两个关键元素: 购买的商品和商品的制造商。 我们会要求模型将回应以一个 JSON 对象的形式呈现，其中的 key 就是商品和品牌。

```
prompt = f"""
从评论文本中识别以下项目: - 评论者购买的物品
- 制造该物品的公司
评论文本用三个反引号分隔。将你的响应格式化为以 “物品” 和 “品牌” 为键的 JSON 对象。
如果信息不存在，请使用 “未知” 作为值。

让你的回应尽可能简短。
评论文本: ```{lamp_review}```
"""
response = get_completion(prompt)
print(response)
```

```
{
"物品": "卧室灯",
"品牌": "Lumina"
}
```

场景2 - 推断 - 信息提取

2.2 综合情感推断和信息提取

在上面小节中，我们采用了三至四个 Prompt 来提取评论中的“情绪倾向”、“是否生气”、“物品类型”和“品 牌”等信息。然而，事实上，我们可以设计一个单一的 Prompt ，来同时提取所有这些信息。

```
prompt = f"""
从评论文本中识别以下项目:
- 情绪(正面或负面)
- 审稿人是否表达了愤怒?(是或否)
- 评论者购买的物品
- 制造该物品的公司

评论用三个反引号分隔。将你的响应格式化为 JSON 对象，以 “情感倾向”、“是否生气”、“物品类型” 和 “品牌” 作为键。
如果信息不存在，请使用 “未知” 作为值。
让你的回应尽可能简短。
将 “是否生气” 值格式化为布尔值。
评论文本: ```{lamp_review}```
"""
response = get_completion(prompt) print(response)
```

```
{
"情感倾向": "正面",
"是否生气": false,
"物品类型": "卧室灯",
"品牌": "Lumina"
}
```

场景2 - 推断 - 主题推断

大型语言模型的另一个很酷的应用是推断主题。假设我们有一段长文本，我们如何判断这段文本的主旨 是什么?它涉及了哪些主题?让我们通过以下一段虚构的报纸报道来具体了解一下。

```
story = """
在政府最近进行的一项调查中，要求公共部门的员工对他们所在部门的满意度进行评分。
调查结果显示，NASA 是最受欢迎的部门，满意度为 95%。

一位 NASA 员工 John Smith 对这一发现发表了评论，他表示:
“我对 NASA 排名第一并不感到惊讶。这是一个与了不起的人们和令人难以置信的机会共事的好地方。我为成为这样一个创新组织的一员感到自豪。”

NASA 的管理团队也对这一结果表示欢迎，主管 Tom Johnson 表示:
“我们很高兴听到我们的员工对 NASA 的工作感到满意。 我们拥有一支才华横溢、忠诚敬业的团队，他们为实现我们的目标不懈努力，看到他们的辛勤工作得到回报是太棒了。”

调查还显示，社会保障管理局的满意度最低，只有 45%的员工表示他们对工作满意。 政府承诺解决调查中员工提出的问题，并努力提高所有部门的工作满意度。
"""
```

场景2 - 推断 - 主题推断

3.1 推断讨论主题

以上是一篇关于政府员工对其工作单位感受的虚构报纸文章。我们可以要求大语言模型确定其中讨论的五个主题，并用一两个词语概括每个主题。输出结果将会以逗号分隔的Python列表形式呈现。

```
prompt = f"""
确定以下给定文本中讨论的五个主题。
每个主题用1-2个词概括。 请输出一个可解析的Python列表，每个元素是一个字符串，展示了一个主题。

给定文本: ```{story}```
"""
response = get_completion(prompt)
print(response)
```

```
['NASA', '满意度', '评论', '管理团队', '社会保障管理局']
```

场景2 - 推断 - 主题推断

3.2 为特定主题制作新闻提醒

假设我们有一个新闻网站或类似的平台，这是我们感兴趣的主题:美国航空航天局、当地政府、工程、 员工满意度、联邦政府等。我们想要分析一篇新闻文章，理解其包含了哪些主题。可以使用这样的 Prompt:确定以下主题列表中的每个项目是否是以下文本中的主题。以 0 或 1 的形式给出答案列表。

```
prompt = f"""
判断主题列表中的每一项是否是给定文本中的一个话题，
以列表的形式给出答案，每个元素是一个Json对象，键为对应主题，值为对应的 0 或 1。
主题列表:美国航空航天局、当地政府、工程、员工满意度、联邦政府
给定文本: ```{story}```
"""
response = get_completion(prompt)
print(response)
```

```
[
  {"美国航空航天局": 1},
  {"当地政府": 1},
  {"工程": 0},
  {"员工满意度": 1},
  {"联邦政府": 1}
]
```

从输出结果来看，这个 story 与关于“美国航空航天局”、“员工满意度”、“联邦政府”、“当地政府”有关， 而与“工程”无关。

这种能力在机器学习领域被称为零样本(Zero-Shot)学习。这是因为我们并没有提供 任何带标签的训练数据，仅凭 Prompt ， 它便能判定哪些主题在新闻文章中被包含。

场景3 - 文本转换 - 文本翻译

1.1 翻译为西班牙语

```
from tool import get_completion
prompt = f"""
将以下中文翻译成西班牙语: \
```您好，我想订购一个搅拌机。```
"""
response = get_completion(prompt)
print(response)
```

```
Hola, me gustaría ordenar una batidora.
```

1.2 识别语种

```
prompt = f"""
请告诉我以下文本是什么语种:
```Combien coûte le lampadaire?``` ""  "

response = get_completion(prompt)
print(response)
```

```
这段文本是法语。
```

场景3 - 文本转换 - 文本翻译

1.3 多语种翻译

```
prompt = f"""
请将以下文本分别翻译成中文、英文、法语和西班牙语:
```I want to order a basketball.```
"""
response = get_completion(prompt)
print(response)
```

```
中文:我想订购一个篮球。
英文:I want to order a basketball.
法语:Je veux commander un ballon de basket.
西班牙语:Quiero pedir una pelota de baloncesto.
```

1.4 同时进行语气转换

```
prompt = f"""
请将以下文本翻译成中文，分别展示成正式与非正式两种语气: ```Would you like to order a pillow?```
"""
response = get_completion(prompt)
print(response)
```

```
正式语气:您是否需要订购一个枕头?
非正式语气:你想要订购一个枕头吗?
```
```

场景3 - 文本转换 - 语气与写作风格调整

在写作中，语言语气的选择与受众对象息息相关。比如工作邮件需要使用正式、礼貌的语气和书面词 汇;而与朋友的聊天可以使用更轻松、口语化的语气。 选择恰当的语言风格，让内容更容易被特定受众群体所接受和理解，是技巧娴熟的写作者必备的能力。 随着受众群体的变化调整语气也是大语言模型在不同场景中展现智能的一个重要方面。

```
prompt = f"""
将以下文本翻译成商务信函的格式:
```小老弟，我小羊，上回你说咱部门要采购的显示器是多少寸来着?```
"""
response = get_completion(prompt)
print(response)
```

```
尊敬的先生/女士，
我是小羊，我希望能够向您确认一下我们部门需要采购的显示器尺寸是多少寸。上次我们交谈时，您提到了这个问题。

期待您的回复。
谢谢!
此致，
小羊
```
```

场景3 - 文本转换 - 文件格式转换

大语言模型如 ChatGPT 在不同数据格式之间转换方面表现出色。它可以轻松实现 JSON 到 HTML、XML、Markdown 等格式的相互转化。下面是一个示例,展示如何使用大语言模型**将 JSON 数据转换为 HTML 格式 。**

```
data_json = { "resturant employees" :[
    {"name":"Shyam", "email":"shyamjaiswal@gmail.com"},
    {"name":"Bob", "email":"bob32@gmail.com"},
    {"name":"Jai", "email":"jai87@gmail.com"}
]}
```

```
prompt = f"""
将以下Python字典从JSON转换为HTML表格，保留表格标题和列名:{data_json}
"""
response = get_completion(prompt)
print(response)
```

name	email
Shyam	shyamjaiswal@gmail.com
Bob	bob32@gmail.com
Jai	jai87@gmail.com

场景3 - 文本转换 - 拼写及语法纠正

在使用非母语撰写时，拼写和语法错误比较常见，进行校对尤为重要。例如在论坛发帖或撰写英语论文 时，校对文本可以大大提高内容质量。 **利用大语言模型进行自动校对可以极大地降低人工校对的工作量。**下面是一个示例，展示如何使用大语 言模型检查句子的拼写和语法错误。

```
text = [
  "The girl with the black and white puppies have a ball.",  # The girl has a
ball.
  "Yolanda has her notebook.", # ok
  "Its going to be a long day. Does the car need it’ s oil changed?",  # Homonyms
  "Their goes my freedom. There going to bring they’ re suitcases.",  # Homonyms
  "Your going to need you’ re notebook.",  # Homonyms
  "That medicine effects my ability to sleep. Have you heard of the butterfly
affect?", # Homonyms
  "This phrase is to cherck chatGPT for spelling abilitty"  # spelling
]
```

场景3 - 文本转换 - 拼写及语法纠正

在使用非母语撰写时，拼写和语法错误比较常见，进行校对尤为重要。例如在论坛发帖或撰写英语论文 时，校对文本可以大大提高内容质量。 **利用大语言模型进行自动校对可以极大地降低人工校对的工作量。**下面是一个示例，展示如何使用大语 言模型检查句子的拼写和语法错误。

```
for i in range(len(text)):
    time.sleep(20)
    prompt = f"""
  请校对并更正以下文本，注意纠正文本保持原始语种，无需输出原始文本。
  如果您没有发现任何错误，请说“未发现错误”。
  例如:
      输入:I are happy.
      输出:I am happy.
  ```{text[i]}```
    """    "
  response = get_completion(prompt)
  print(i, response)
```

```
0 The girl with the black and white puppies has a ball.
1 Yolanda has her notebook.
2 It's going to be a long day. Does the car need its oil changed?
3 Their goes my freedom. There going to bring their suitcases.
4 You're going to need your notebook.
5 That medicine affects my ability to sleep. Have you heard of the butterfly effect?
6 This phrase is to check chatGPT for spelling ability.
```
```

场景4 - 文本扩展 - 定制客户邮件

在这个客户邮件自动生成的示例中，我们**将根据客户的评价和其中的情感倾向，使用大语言模型针对性地生成回复邮件。** 具体来说，我们先输入客户的评论文本和对应的情感分析结果(正面或者负面)。然后构造一个 Prompt，要求大语言模型基于这些信息来生成一封定制的回复电子邮件。 下面先给出一个实例，包括一条客户评价和这个评价表达的情感。这为后续的语言模型生成回复邮件提 供了关键输入信息。通过输入客户反馈的具体内容和情感态度，语言模型可以生成针对这个特定客户、 考虑其具体情感因素的个性化回复。这种**针对个体客户特点的邮件生成方式，将大大提升客户满意度。**

```
# 我们可以在推理那章学习到如何对一个评论判断其情感倾向
sentiment = "消极的"
# 一个产品的评价
review = f"""
他们在11月份的季节性销售期间以约49美元的价格出售17件套装，折扣约为一半。\
 但由于某些原因(可能是价格欺诈)，到了12月第二周，同样的套装价格全都涨到了70美元到89美元不等。\
 11件套装的价格也上涨了大约10美元左右。\
虽然外观看起来还可以，但基座上锁定刀片的部分看起来不如几年前的早期版本那么好。\ 不过我打算非常温柔地使用它，例如，\
我会先在搅拌机中将像豆子、冰、米饭等硬物研磨，然后再制成所需的份量，\
切换到打蛋器制作更细的面粉，或者在制作冰沙时先使用交叉切割刀片，然后使用平面刀片制作更细/不粘的效 果。\
制作冰沙时，特别提示:\
将水果和蔬菜切碎并冷冻(如果使用菠菜，则轻轻煮软菠菜，然后冷冻直到使用;\
如果制作果酱，则使用小到中号的食品处理器)，这样可以避免在制作冰沙时添加太多冰块。\
大约一年后，电机发出奇怪的噪音，我打电话给客服，但保修已经过期了，所以我不得不再买一个。\
总的来说，这些产品的总体质量已经下降，因此它们依靠品牌认可和消费者忠诚度来维持销售。\
货物在两天内到达。
"""
```

场景4 - 文本扩展 - 定制客户邮件

```
from tool import get_completion
prompt = f"""
你是一位客户服务的AI助手。
你的任务是给一位重要客户发送邮件回复。
根据客户通过“```”分隔的评价，生成回复以感谢客户的评价。提醒模型使用评价中的具体细节
用简明而专业的语气写信。
作为“AI客户代理”签署电子邮件。
客户评论:
```{review}```
评论情感:{sentiment}
"""
response = get_completion(prompt)
print(response)
```

尊敬的客户，
非常感谢您对我们产品的评价。我们非常抱歉您在购买过程中遇到了价格上涨的问题。我们一直致力于为客户提供最优惠的价格，但由于市场波动，价格可能会有所变化。我们深表歉意，如果您需要任何帮助，请随时联系我们的客户服务团队。

我们非常感谢您对我们产品的详细评价和使用技巧。我们将会把您的反馈传达给我们的产品团队，以便改进我们的产品质量和性能。

再次感谢您对我们的支持和反馈。如果您需要任何帮助或有任何疑问，请随时联系我们的客户服务团队。

祝您一切顺利!

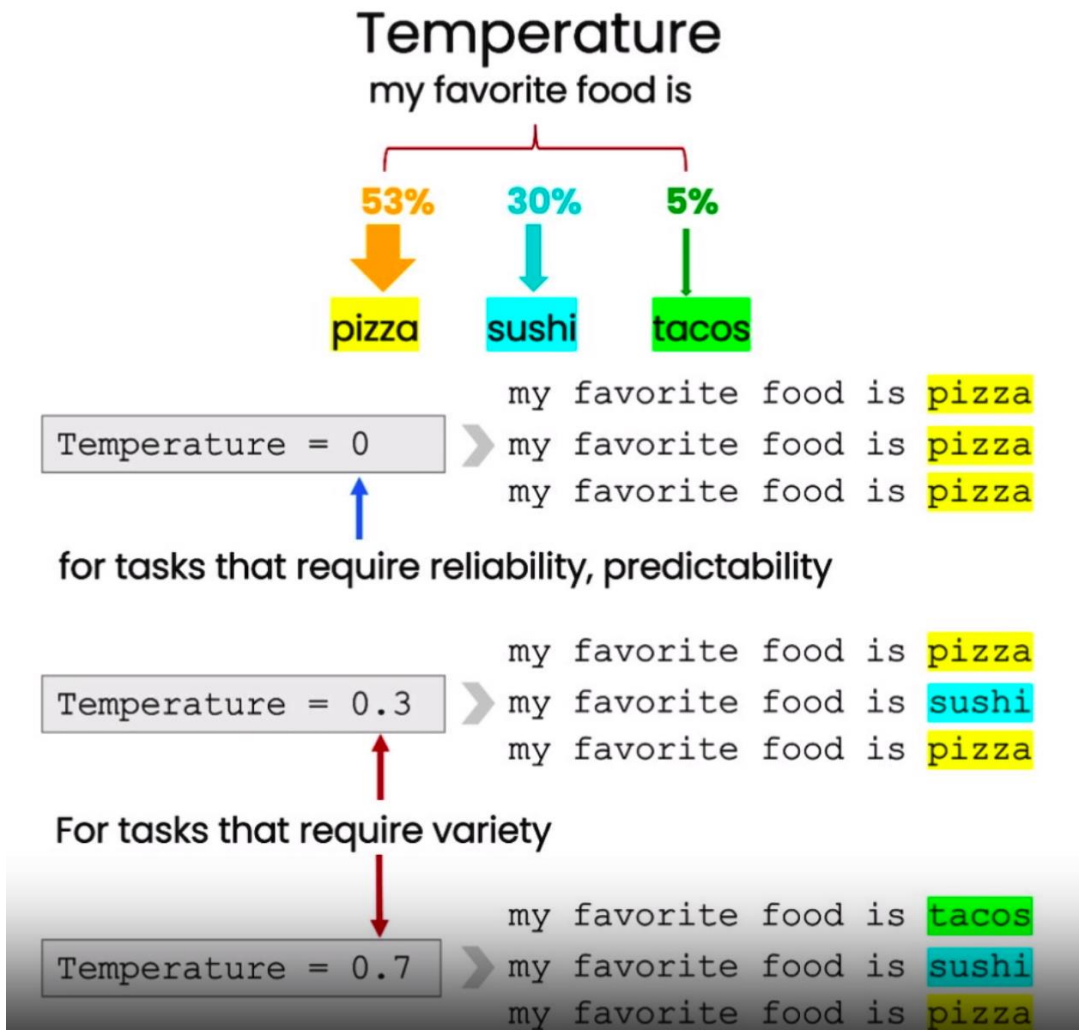
AI客户代理
```

场景4 - 文本扩展 - 引入温度系数

语言模型中的 “温度”(temperature) 参数可以控制生成文本的随机性和多样性。temperature 的值越 大，语言模型输出的多样性越大;temperature 的值越小，输出越倾向高概率的文本。

一般来说，如果需要可预测、可靠的输出，则将 temperature 设置为0 。
温度(temperature)参数可以控制语言模型生成文本的随机性。温度为0时，每次使用同样的 Prompt，得到的结果总是一致的。而在上面的样例中，当温度设为0.7时，则每次执行都会生成不同的 文本。

举个例子，在某一上下文中，语言模型可能认为“比萨”是接下来最可能的词，其次是“寿司”和“塔可”。若 temperature 为0，则每次都会生成“比萨”;而当 temperature 越接近 1 时，生成结果是“寿司”或“塔可”的可能性越大，使文本更加多样。



场景5 - 意图理解

用大模型代替传统自然语言处理模型，可以从用户输入内容中提取出结构化提取信息

prompt设计的结构

- 用户输入：将用户输入的内容拼接到prompt中，提交给大模型
- 身份定义：定义大模型扮演角色，帮助大模型理解指令
- 背景说明：明确对话发生的背景信息，帮助大模型理解指令
- 字段说明：说明要提取的字段的含义，以及字段存在的枚举值
- 输出示例：输出内容示例

场景5 - 意图理解 - prompt设计的结构

你是一个智能助理，你需要帮用户结构化记录生日信息、物品存放信息、月经信息
用户输入是一句非常口语化的指令，你需要记录用户指令，并从用户的指令中结构化的输出提取出信息
输出完毕后结束，不要生成新的用户输入，不要新增内容

1. 提取话题，话题只能是：生日、纪念日、月经、物品存放。
2. 提取目的，目的只能是：记录、预测、查询、庆祝、设置、记录物品、拿到物品、寻找、删除、修改。
3. 提取人物，人物指：过生日的人物、过纪念日的人物、来月经的人物、放物品的人物。输出只能是：我，爸爸、妈妈、孩子、爱人、恋人、朋友、哥哥、姐姐。没有写“无”
4. 提取人关系，关系指人物与用户的关系，关系只能是：本人、亲人、配偶、朋友、未知、待查询。没有写“无”
5. 提取时间，比如：今天、3月1日、上个月、农历二月初六、待查询。没有写“无”
6. 提取时间类型，时间类型只能是：过生日的时间、过纪念日的时间、月经开始时间、月经结束时间。 没有写“无”
7. 提取物品，比如：衣服、鞋子、书、电子产品、其它。
8. 提取物品对应位置，比如：衣柜、书柜、鞋柜、电子产品柜、待查询。
9. 按示例结构输出内容，结束

场景5 - 意图理解 - prompt设计的结构

用户：今天我过生日
话题：生日
目的：记录
人物：我
关系：本人
时间：今天
时间类型：生日时间
物品：无
位置：无

用户：每年的七月二十二日张派生日记录一下
话题：生日
目的：记录
人物：张派
关系：未知
时间：七月二十二日
时间类型：生日时间
物品：无
位置：无

用户：恋爱纪念日
话题：纪念日
目的：查询
人物：我
关系：本人
时间：待查询
时间类型：纪念日时间
物品：无
位置：无

用户：明天提醒我今天来过生日
话题：生日
目的：提醒
人物：我
关系：本人
时间：今天
时间类型：生日时间
物品：无
位置：无

场景5 - 意图理解 - 提示词设计总结

根据您的总结，以下是对提示词设计的改进建议：

1. 简化流程：确保提示词的流程尽可能简单明了，避免过于复杂的步骤和要求。简化流程有助于减少大模型出错的概率。

2. 强调语义理解：确保提示词能够帮助大模型理解语义，并输出合适的枚举。避免过于具体或限制性的要求，而是提供足够的上下文和信息，让大模型能够根据语义进行输出。

3. 多使用肯定句：在提示词中多使用肯定句，明确告诉大模型要做什么，而不是限制它不做什么。肯定句能够更清晰地传达您的意图，提高大模型的理解和回答准确性。

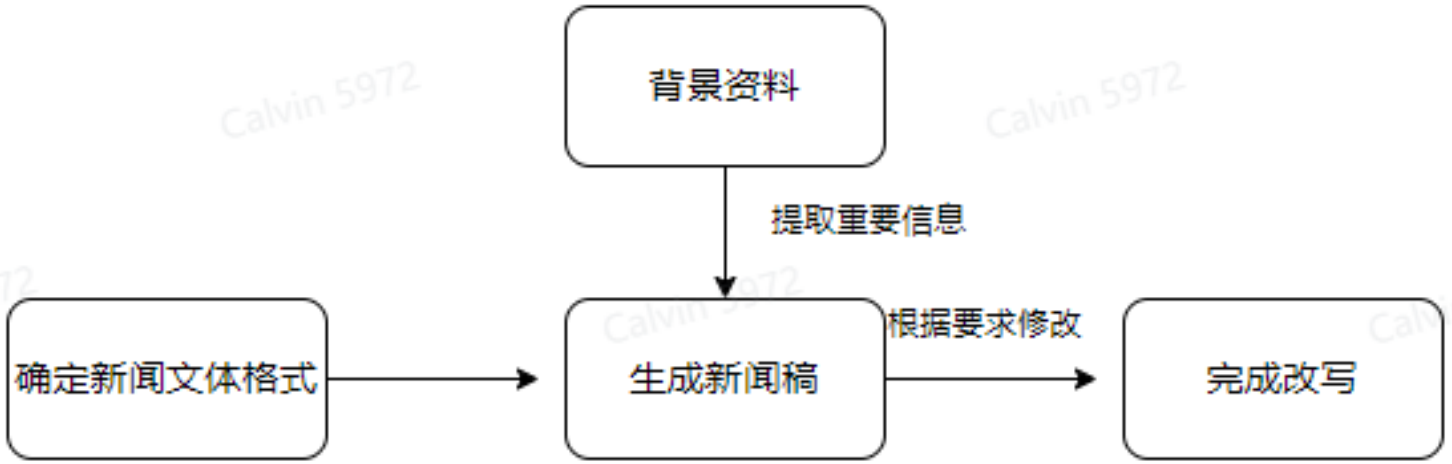
4. 结合功能设计prompt：在与大模型的交互中，结合具体的功能需求来设计提示词。不要期望一次交互就能解决所有问题，而是根据需要分阶段进行交互，每次交互都针对特定的功能或问题。

这些改进建议可以帮助您更有效地与大模型进行交互，提高交互的效果和准确性。根据具体的应用场景和需求，您可以进一步调整和优化提示词的设计。

场景6 - 新闻写作

要求实现改变，并保留重要信息。其中改写要求不构成对原稿件的抄袭；重要信息包括：企业、融资情况和企业情况。实现难点如下：

A. 新闻文体格式、语体特殊，需特别处理。
B. 大模型改写程度较小，难以达标。
C. 重要信息构成复杂，提取保留有难度。



场景6 - 新闻写作

背景资料：
"""
{背景资料}
"""

你是一个新闻改写人，需要根据背景资料改写新闻。新闻包括"标题"、"正文"，改写发生在正文部分。
"标题"应该是一个完整的句子，能够概括新闻内容。标题内容由企业和融资情况、融资用途构成。
"正文"应该第一段说明企业和融资情况（包括融资进度、融资方、融资金额、融资目的）；在之后的段落中，从名称、业务、团队、发言、核心技术、技术方向、产品进展等角度叙述企业情况。若背景资料中没有提到，则忽略。
改写可以改变调整信息顺序，必须改变背景资料段落中句子的语法结构、句子长度、词语搭配。

你的新闻需要遵守以下规定：
1. 使用的语言简短精炼。
2. 涉及的信息真实，不允许杜撰。
3. 在正文中重新表达背景资料的信息，但是人名、物名、数字、日期、行为不能改动。

输出格式：
"""
今日新闻：
标题：
正文：
"""

场景6 - 新闻写作

背景资料

神顶科技（南京）有限公司（以下简称“神顶科技”）近日完成数千万元新一轮战略融资，投资方为深圳高新投集团和昆山台商发展基金。
本轮资金将用于进一步扩大在智能感知融合芯片的方案开发和量产推广。
神顶科技拥有完全自主研发的3D智能感知融合技术，目前，公司研发团队凭借丰富的产品化经验和供应链整合能力，研发出基于Smart Fusion技术的3D智能感知融合SoC芯片平台。该平台不仅符合移动机器人特有的场景需求、保留了移动机器人硬件的全面可扩展性，而且在性价比方面同样具有优势。
谈及未来发展，神顶科技相关负责人表示：“目前，神顶科技已经获得多项核心技术和专利成果，并在感知融合、图像算法、人工智能等方面积累了技术优势。此次融资后，神顶科技将继续坚持创新驱动，推进智能感知融合领域的发展，为移动机器人、AR/VR增强/虚拟现实及自动驾驶等相关领域的感知系统智能化提供更优质的方案和服务。”

生成新闻稿

标题：神顶科技获数千万元融资，推进智能感知融合领域发展

正文:

近日，神顶科技 (南京) 有限公司宣布获得数千万元新一轮战略融资，投资方为深圳高新投集团和昆山台商发展基金。本轮资金将用于进一步扩大在智能感知融合芯片的方案开发和量产推广。

神顶科技是一家完全自主研发 3D 智能感知融合技术的公司，其研发团队凭借丰富的产品化经验和供应链整合能力，研发出基于 Smart Fusion 技术的 3D 智能感知融合 SoC 芯片平台。该平台符合移动机器人特有的场景需求，保留了移动机器人硬件的全面可扩展性，而且在性价比方面同样具有优势。

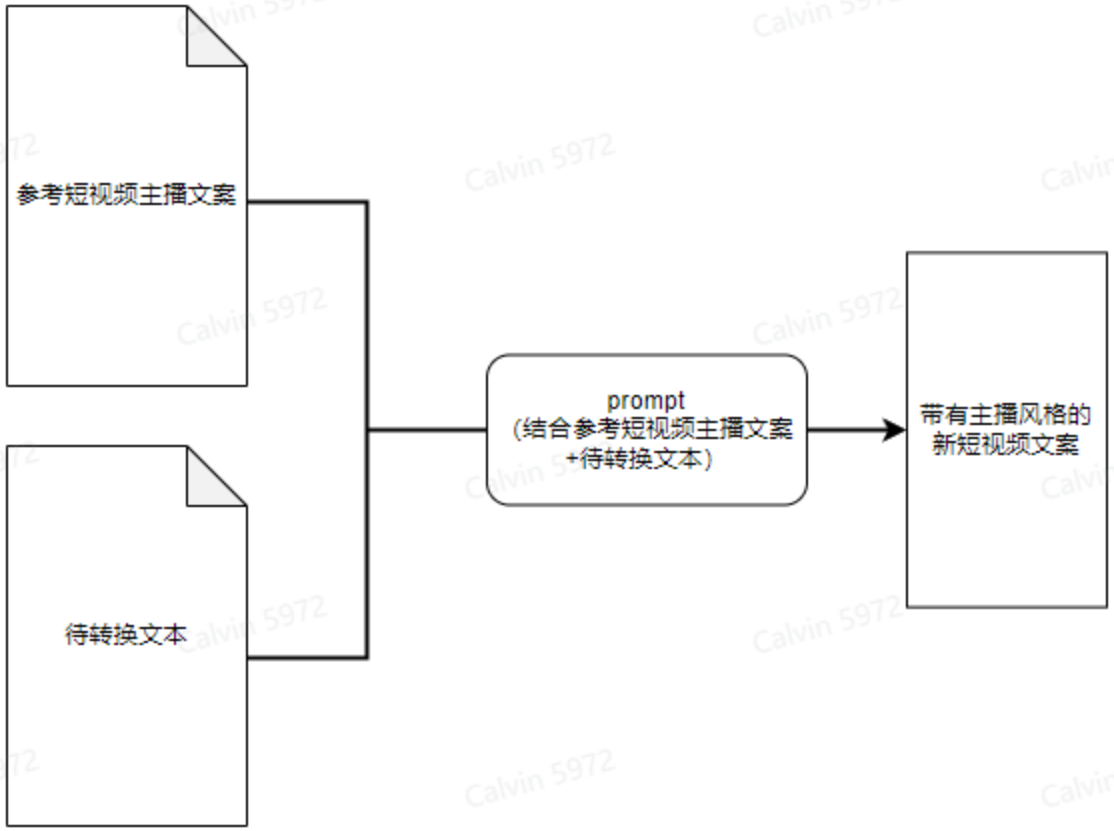
神顶科技的未来发展充满信心。公司相关负责人表示，“目前，我们已经获得多项核心技术和专利成果，并在感知融合、图像算法、人工智能等方面积累了技术优势。此次融资后，我们将继续坚持创新驱动，推进智能感知融合领域的发展，为移动机器人、AR/VR 增强/虚拟现实及自动驾驶等相关领域的感知系统智能化提供更优质的方案和服务。”

场景7 - 风格写作

给主播撰写短视频文案：根据主播风格，将指定文本转换成指定短视频主播风格的短视频文案。

- 不同主播风格不同，难以准确有文字描述不同短视频主播的风格
- 待转换文本与短视频文案风格差距较大

- 风格改写类尽量使用few shot：对于风格改写类场景，风格较难用文字总结，并且大模型也较难通过文字get到具体的风格。
- few shot示例要有明显的风格：提供给大模型的few shot示例，要人工摘选出有明显风格的文本，大模型才能够更好的学习其具体的风格。



场景7 - 风格写作

- Part1 - Few shot：Few shot示例：通过短视频范文1、2、3，将few shot参考示例传递给大模型；给大模型一些含有主播风格的短视频文案示例，让大模型参考提供的短视频主播文案示例进行改写，效果优秀；

- Part2 - 任务描述：通过简洁的描述，将任务完整的描述给大模型。

- Part3-待改写原文

短视频范文1：
"""
{示例1}
"""

短视频范文2：
"""
{示例2}
"""

短视频范文3：
"""
{示例3}
"""

你是一个短视频主播，请你参考上面的短视频范文1、2、3，将下面的原文改写成新的短视频文案。

原文：
"""
{需要改写的原文}
"""

场景8 - 研发效能 - 写代码

- 编写一个[语言]脚本来连接到数据库并执行[database operation]。
- 提供一个[语言]代码来执行类似于[file operations list]的文件操作。
- 编写一个[语言]脚本来连接到[database]并执行[operation]。
- 编写一个[语言]脚本来处理[data type]并满足以下要求：[requirements list]。
- 开发一个[语言]函数，使用[methodology或library]执行[task]，输入为：[input variables]。
- 你能帮我编写一个[语言]算法来解决[problem]，给定约束条件：[constraints list]吗?
- 你能创建一个[language]函数来[task]吗？它接受[input variables]并返回[output]，在这些约束条件下：[constraints list]。
- 编写一个[语言]脚本来解析[file format]，提取[information]，并按照以下要求将数据存储在[data structure]中：[requirements list]。
- 编写一个[语言]函数来使用[algorithm]计算[mathematical concept]。该函数应接受以下输入：[input variables]，并返回[expected output]。
- 创建一个[语言]程序，读取[input file type]，执行[operations]，然后按照[format description]的格式将结果写入[output file type]。
- 使用[library/API]实现一个[语言]脚本，检索[data type]，执行[operation]，然后按照[database schema]的结构将其存储在[database]中。
- 为[task]实现一个[语言]算法，给定这些输入参数[input parameters]，它应输出[expected output]并考虑这些约束条件[constraints list]。
- 请编写一个名为[function name]的[语言]函数，以[task]，[input variables]作为输入，并返回[output]，在这些约束条件下[constraints list]。
- 创建一个[语言]脚本来解析[file format]，提取[information]，并按照以下要求将数据存储在[data structure]中：[requirements list]。

场景8 - 研发效能 - 代码优化

- 建议优化这个[语言]函数：[代码片段]。
- 你能提供一个更高效的版本这个[语言]算法吗：[代码片段]?
- 我该如何提高这个[语言]脚本的性能：[代码片段]?
- 这个[语言]函数：[代码片段]运行得比我想象的要慢。有什么优化建议吗?
- 我需要提高这个[语言]算法的速度：[代码片段]。你有什么改进建议吗?
- 我该如何使这个[语言]数据处理代码更高效：[代码片段]?
- 下面的[语言]函数：[代码片段]在处理[输入类型]时运行得比预期慢。有什么优化建议吗?
- 当处理[大型数据集]时，我该如何提高这个[语言]函数：[代码片段]的性能?
- 提供以下用于处理[数据类型]的[语言]代码的优化建议：[代码片段]。
- 我该如何优化这个[语言]函数：[代码片段]以便在处理[大数据量]时更快地执行[任务]并保持[精确度要求]?
- 我有一个[语言]函数：[代码片段]。它的功能符合预期，但在处理[特定数据]时运行得比我想象的要慢。有什么性能改进的建议吗?
- 下面的[语言]代码：[代码片段]执行[任务]。然而，对于大小为[数据大小]的[数据类型]，它似乎效率低下。我该如何优化它?

场景8 - 研发效能 - 代码重构

- 建议对这个[语言]函数进行重构：[代码片段]。
- 如何使这段[语言]代码更易读：[代码片段]？
- 有哪些方法可以重构这个[语言]脚本以提高性能：[代码片段]？
- 我想重构这段[语言]代码以更符合面向对象的原则：[代码片段]。有什么建议吗？
- 您能展示一下如何重构这个[语言]函数以更符合惯用方式吗：[代码片段]？
- 我正在考虑重构这个[语言]脚本以使用[概念或特性]。您会如何处理？
- 如何将以下[语言]代码重构为符合[特定编码原则或模式]的形式：[代码片段]？
- 您能展示一下如何重构这个[语言]函数以使用更多现代特性，比如[特定特性]吗？
- 建议一种重构以下[语言]代码的方式：[代码片段]，以改善[特定方面]。
- 如何重构这段[语言]代码以提高[方面]并符合[特定编码标准或原则]？
- 我想重构这个[语言]函数：[代码片段]，使其更符合惯用方式和可维护性。此外，它还应处理[特定边界情况]。有什么建议吗？
- 建议一种重构以下[语言]代码的方式：[代码片段]，以符合[特定设计模式]并改善[特定方面]。

场景8 - 研发效能 - 代码评审

- 请审查我的代码并提出改进或优化的建议：[粘贴你的代码在这里]
- 指出我代码中可能存在的错误或漏洞：[粘贴你的代码在这里]
- 审查我的代码以查找安全漏洞并提出修复建议：[粘贴你的代码在这里]
- 你能找到这个[语言]代码中的性能问题吗？[代码片段]？
- 这个[语言]代码中是否存在安全漏洞？[代码片段]？
- 请审查这个[语言]代码的风格和最佳实践：[代码片段]。
- 你能在这个[语言]代码中找到任何内存泄漏吗？[代码片段]？
- 我对这个[语言]代码中的安全问题感到担忧：[代码片段]。你有什么想法？
- 帮我理解为什么这个[语言]函数的工作结果与预期不符：[代码片段]。
- 你能帮我调试来自我的[语言]程序的错误消息吗：[错误消息]？
- 在这个[语言]代码中处理[数据类型]的过程中，能找出任何潜在问题吗？[代码片段]？
- 你能找出这个[语言]函数中处理[任务]的错误吗？[代码片段]？
- 这个[语言]方法在[任务]方面有什么问题？[代码片段]？

场景8 - 研发效能 - Debug

- 帮我调试处理 [数据类型] 的 [语言] 脚本，并提出可能的修复建议：[代码片段]。
- 找出以下 [语言] 代码中的内存泄漏，并提出可能的优化建议：[代码片段]。
- 请审查这段 [语言] 代码，它应该在给定输入 [输入变量] 的情况下完成 [任务]，并返回 [输出]：[代码片段]。
- 找出处理 [数据类型] 并输出 [输出类型] 的 [语言] 脚本中的潜在错误：[代码片段]。
- 识别这个 [语言] 函数中的逻辑错误，它旨在使用这些输入参数 [输入参数] 并期望输出 [输出描述]：[代码片段]。
- 请审查以下 [语言] 代码，它应该在给定输入 [输入变量] 的情况下完成 [任务]，并返回 [输出]，并遵循这些编码准则：[编码准则]：[代码片段]。
- 在处理 [数据类型]、使用这些资源 [资源列表] 并输出 [输出类型] 的 [语言] 脚本中，找出并修复潜在的错误：[代码片段]。
- 找出这个 [语言] 函数中的逻辑错误，它旨在使用这些输入参数 [输入参数] 并期望产生 [输出描述]，但目前却给出了 [错误的输出]。
- 调试给定的 [语言] 代码：[代码片段]。它应该执行 [期望的行为]，但却产生了 [当前的行为]。
- 审查以下名为 [函数名] 的 [语言] 函数：[代码片段]。请识别任何潜在的错误、性能问题和与 [编码标准] 不符的地方。
- 调试以下 [语言] 代码：[代码片段]。它在给定输入示例：[输入示例] 时，应该执行 [期望的行为]，但实际上却产生了 [当前的行为]。
- 请审查处理大小为 [数据大小] 的 [数据类型] 时的 [语言] 函数：[代码片段]，以查找任何潜在的内存泄漏或性能问题。

场景8 - 研发效能 – 文档生成

- 为[编程语言]代码库生成API文档。
- 描述从[编程语言]代码库生成文档的过程。
- 提供关于编写有效文档的指导，针对[软件解决方案]。
- 有哪些工具可用于从[编程语言]代码库自动生成文档?
- 提供生成用户手册的说明，针对[编程语言]。
- 我希望你扮演文档生成器的角色，并为[代码或项目]创建一个[类型的文档]文档。

参考资料

1. **Prompt 工程实践教程**
https://lslfd0slxc.feishu.cn/docx/Nqm9dX81hotVYUxFQuxcVR82n2g

2. **吴恩达 - 开发者的提示工程**
https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/


Thank
You