

提示学习

周睿宁

武汉大学

zhouruining5720@foxmail.com

2024 年 12 月 13 日



① 提示设计

② 上下文学习

③ 思维链提示

本章讨论如何通过提示学习方法来有效使用大语言模型解决实际任务

- 设计合适的提示 (Prompting)
 - 书 10.1 节
- 将任务描述与示例以文本形式加入到提示中 (In-context Learning)
 - 书 10.2 节
- 将一系列中间推理步骤加入到示例中 (Chain-of-Thought)
 - 书 10.3 节

① 提示设计

人工提示设计
自动提示优化

② 上下文学习

③ 思维链提示

因为较高的微调代价，基于自然语言的提示方法已经成为了使用大语言模型解决下游任务的主要途径。

- 人工设计
- 自动优化

可以通过以上两种方法来生成合适的任务提示。

① 提示设计

人工提示设计
自动提示优化

② 上下文学习

③ 思维链提示

四个关键要素

提示工程 (Prompting Engineering):

针对特定任务设计合适的任务提示，需要考虑以下四要素：

- 任务描述
- 输入数据
- 上下文信息
- 提示策略

任务描述

任务描述

- 使用清晰且具体的表达来描述任务目标；
- 可以对于输入或输出的格式进行更详细的说明；
- 可以使用关键词或者特殊符号来强调特殊设置。

知识问答的任务描述：请使用所提供的以三个井号 (###) 分隔的文章回答问题。
如果在文章中找不到答案，请回答“无法找到答案。”

代码补全的任务描述：你是一名程序员。给你一个代码片段，你的目标是完成这段代码，确保它能实现描述的功能。

对话推荐的任务描述：推荐 10 个符合用户偏好的商品。推荐列表可以包含对话框之前提到的商品。推荐列表的格式为：商品 ID 标题（年份）。请勿在推荐列表中提及商品标题以外的任何内容。

图 1: 提示设计中任务描述的例子

输入数据

- 通常情况下，直接使用自然语言描述输入数据的内容即可。
- 对于结构化数据，
 - * 使用线性化方法转化为文本序列；eg. 知识图谱、表格
 - * 转化为基于数据结构的代码表示。eg. 图上的节点

线性化后的表格数据：

```
### 学号 姓名 成绩 # 01278 张三 89.0 # 03813 李四 92.5 # 06714 王五 79.0 ###
```

代码格式的图数据：

```
Graph[name="Knowledge-Graph"]{  
entity_list=[“James Cameron”, “Ontario”, ...],  
triple_list=[(“James Cameron” -> “Ontario”)[relation=“born in”], ...],  
}
```

图 2: 提示设计中输入数据的例子

上下文信息

上下文信息

在引入上下文信息的时候，需要对于这些信息进行合适的格式化，以加强大语言模型对它们的利用，从而模仿这些示例数据的任务目标、输出格式、输出和输入之间的映射。具体见第 2 章上下文学习板块。

基于任务示例回答问题：

示例问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

示例回答：罗杰一开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。答案是 11。

用户输入：袋子里有 16 个球，其中一半是高尔夫球，并且所有高尔夫球中有一半是蓝色的球。请问有多少个蓝色高尔夫球？

模型输出：袋子里有 16 个球，其中一半是高尔夫球。因此有 $16 / 2 = 8$ 个高尔夫球。一半的高尔夫球是蓝色的。因此有 $8 / 2 = 4$ 个蓝色高尔夫球。答案是 4。

图 3: 提示设计中上下文信息的例子（灰色文字表示模型生成结果）

提示策略

提示策略

添加特定的前缀或后缀有助于激发模型解决特定任务的能力。

eg.

- * “让我们一步一步地思考”
- * “你是这项任务（或这个领域）的专家”

解答数学问题：

用户输入：袋子里有 16 个球，其中一半是高尔夫球，并且所有高尔夫球中有一半是蓝色的球。请问有多少个蓝色高尔夫球？

模型输出：让我们一步一步地思考。

袋子里一共有 16 个球，其中一半是高尔夫球，也就是说，一共有 8 个高尔夫球。所有高尔夫球中有一半是蓝色的，也就是说有 4 个蓝色的高尔夫球。

推荐旅游景点：

用户输入：你是一名旅行向导。我会把我的位置告诉你，你需要向我推荐附近的旅游景点。我的第一个请求是“我在伊斯坦布尔，我只想参观博物馆”。

模型输出：当然，以下是一些您可以考虑参观的博物馆：

图 4: 提示设计中提示策略的例子

进一步总结了四项提示设计的基本原则：

- **清晰地表达任务目标:** 任务描述中应当包含任务的各种要素信息，如任务目标、输入/输出数据和回复限制。
eg. “给定一份长文档，我希望你生成一份简明摘要，长度不能超过 50 个单词”
- **分解为简单且详细的子任务:** 将一个复杂任务分解为若干个相对独立但又相互关联的子任务，可以显式地将子任务按编号列出。
eg. “通过依次执行以下任务形成一段连贯的叙述：1. ... ; 2. ... ; 3. ...”

- 提供少样本示例：可以根据目标任务专门设计若干高质量的示例。
- 采用模型友好的提示格式：
 - * 对于提示中需要重点强调的部分，OpenAI 官方文档中建议用户可以使用特殊符号（例如、三引号 “““和”””、XML 标签等）进行分隔；
 - * 在执行任务时使用英语指令可能会获得更好的执行效果。

① 提示设计

人工提示设计
自动提示优化

② 上下文学习

③ 思维链提示

为什么要用自动提示优化？

- ① 需要耗费较多的人工成本；
- ② 要求设计人员具有丰富的提示工程经验；
- ③ 大语言模型对于提示设计比较敏感，人工设计的提示有时很难获得最优的效果，还可能导致任务性能的下降。

离散提示通常是由一系列自然语言词元组成的完整句子表达。
eg. “请根据提供的检索信息回答下列问题”

实现思路大致是在离散的词元空间中进行组合搜索。

基于梯度的方法

基于梯度的方法

- ① 将提示初始化为一系列 “[MASK]” 标记。
- ② 迭代地将提示中的词元替换为词典中的其他词元。
- ③ 通过词元替换产生的对数似然变化来近似梯度。
- ④ 为提示的每个位置贪心搜索出最佳的词元。

缺点：对提示的每个位置都进行所有候选词元的替换和梯度评估，导致搜索过程的效率较低。

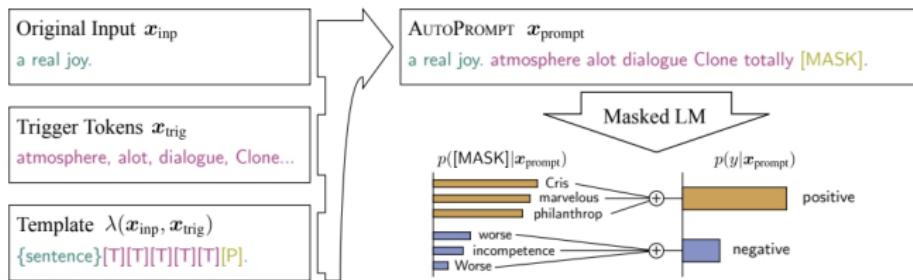


图 5：离散词元优化-基于梯度的方法，以 AutoPrompt 为例

基于强化学习的方法

- ① 利用一个预训练语言模型作为强化学习中的策略网络 $\pi(a|s)$ 并依次生成提示中的词元。
- ② 设计特定的奖励信号 R , 该信号可通过强化学习算法用于策略网络参数的训练。eg.
 - * 真实标签与基于提示的预测标签是否一致
 - * 生成文本与给定条件的匹配程度
- ③ 基于训练好的策略网络, 采用贪心搜索策略来生成任务提示中的每个词元。

基于编辑的方法

该方法关注如何通过编辑现有的提示来改进模型的性能，基于模型在目标任务上的表现来判断提示的好坏。适用于无法直接访问模型内部状态（如梯度）的情况。eg. 只能通过 API 调用的模型

- ① 预先定义好编辑操作，常用的提示编辑操作有修改任务描述、添加或删除上下文任务示例、调整输入到输出的标签映射。
- ② 在现有提示的基础上迭代地对提示进行修改。
- ③ 直至达到最大迭代轮次或者模型最佳性能。

基于大语言模型的方法

直接将提示优化看作一个待求解的任务，进而直接使用大语言模型作为提示生成器来生成或改进提示。

- ① 利用提示生成模型(用于生成提示的大语言模型) 基于少量上下文示例生成一批候选的任务指令。
- ② 使用目标模型(用于下游测试的大语言模型) 对这些候选指令在目标任务的表现进行评估。
- ③ 可采用模型困惑度或任务准确率作为衡量指令质量的指标。
- ④ 在每一轮迭代中，得到高评分指令，然后利用大语言模型生成与高评分指令相似的新指令，以扩展候选指令集。
- ⑤ 迭代完成后，选择模型表现最佳的候选指令作为最终提示。

连续提示由一组连续空间中的嵌入向量组成，可以根据下游任务的损失直接通过梯度更新进行优化。

已有的连续提示优化研究通常依赖于有监督学习方法，但受大语言模型巨大参数量的限制，连续提示受到的关注较为有限。

监督学习方法

这类方法将连续提示向量视为可训练的模型参数，基于下游任务，通过最小化交叉熵损失来优化连续提示。

- **Prefix-tuning**: 在语言模型的每个 Transformer 层预置一串前缀用于训练。
- **Prompt-tuning**: 在输入层加入可训练的提示向量。

缺点: 与输入无关，缺乏对于输入语义的充分考虑。

迁移学习方法

为了在数据稀缺场景下获得较好的模型性能：

- 可以为若干个具有代表性的源任务学习一个所有任务共享的连续提示，然后使用该提示初始化下游任务的提示。
- 可以为每个源任务独自学习任务特定的连续提示。在目标任务的实例时，可以采用注意力机制等方式学习目标实例与每个源任务提示的相关性权重系数。

① 提示设计

② 上下文学习

示例选择

示例格式

示例顺序

底层机制

③ 思维链提示

什么是上下文学习

在 GPT-3 中，OPENAI 团队首次提出上下文学习（In-context learning, ICL）这种提示形式，目前其已经成为调用大语言模型解决下游任务的一种主流途径。

上下文学习使用由任务描述和（或）示例所组成的自然语言文本作为提示。

$$\underbrace{\text{LLM}}_{\text{大语言模型}} \left(\underbrace{I}_{\text{任务描述}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, f(\underbrace{x_{k+1}}_{\text{输入}}, \underbrace{___}) \right) \rightarrow \hat{y}_{k+1}.$$

图 6: 上下文学习的提示构建过程

上下文学习的提示构建过程

$$\underbrace{\text{LLM}}_{\text{大语言模型}} \left(\underbrace{I}_{\text{任务描述}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, f(\underbrace{x_{k+1}}_{\text{输入}}, \underbrace{___}) \right) \rightarrow \hat{y}_{k+1}.$$

基本构建过程：

- 通过自然语言描述任务，并从任务数据集中选择一些样本作为示例；
- 根据特定的模板，将这些示例按照特定顺序组合成提示内容；
- 将测试样本添加到提示后面，整体输入到大语言模型以生成输出。

上下文学习的提示构建过程

$$\underbrace{\text{LLM}}_{\text{大语言模型}} \left(\underbrace{I}_{\text{任务描述}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, f(\underbrace{x_{k+1}}_{\text{输入}}, \underbrace{_____}_{\text{答案}}) \right) \rightarrow \hat{y}_{k+1}.$$

符号表示：

- 由 k 个样本构成一组示例数据 $D_k = \{f(x_1, y_1), \dots, f(x_k, y_k)\}$
- 函数 $f(x_k, y_k)$ 负责将第 k 个任务样本转换为自然语言提示
- 任务描述 I ; 测试输入 x_{k+1} ; 测试输出 \hat{y}_{k+1}

① 提示设计

② 上下文学习

示例选择

示例格式

示例顺序

底层机制

③ 思维链提示

示例样本选择

目的: 从含有大量样本的集合中选取最有价值的示例。

几种常见的示例选择方法：

- 基于相关度排序的方法；
- 基于集合多样性的方法；
- 基于大语言模型的方法。

基于相关度排序的方法

典型实现就是基于 k 近邻 (kNN) 的相似度检索算法。

- ① 使用文本嵌入模型 (BERT) 将候选项映射到低维嵌入空间中；
- ② 根据这些候选项与测试样本的嵌入语义相似度进行排序；
- ③ 选择出最相关的 k 个示例作为示例集合。

缺点：独立地评估每个示例的相关性，而忽略了示例集合的整体效果。

基于集合多样性的方法

旨在针对特定任务选择出具有代表性的、信息覆盖性好的示例集合，以确保所选示例能够反映尽可能多的任务信息。

- 启发式的 MMR 算法 (Maximum Margin Ranking)

$$\text{MMR} = \arg \max_{D_i \in R \setminus S} \left[\lambda \cdot \text{sim}_1(D_i, Q) - (1 - \lambda) \max_{D_j \in S} \text{sim}_2(D_i, D_j) \right]$$

- * 所有候选示例 R
- * 已选择的示例集合 S
- * 候选示例 R
- * 目标任务 Q

基于大语言模型的方法

将 LLM 作为评分器对候选样本进行评估并选择出优质的示例。

- 计算在加入当前示例后模型性能的增益来评估示例的有效性；
缺点：需遍历候选示例进行重复多次计算。
- 根据大语言模型的评分结果选择出少量的正负示例用于训练一个分类器，该分类器通过正负示例学习到如何筛选出高质量示例，从而指导后续的示例选择过程。

① 提示设计

② 上下文学习

示例选择

示例格式

示例顺序

底层机制

③ 思维链提示

示例格式构造

示例格式

即为公式中的示例格式化函数 $f(\cdot)$ 。

$$\underbrace{\text{LLM}}_{\substack{\text{大语言模型} \\ \text{任务描述}}} \left(\underbrace{I}_{\text{示例}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, f(\underbrace{x_{k+1}, \quad}_{\substack{\text{输入} \\ \text{答案}}}) \right) \rightarrow \hat{y}_{k+1}.$$

两种主流的方法包括：

- 人工标注的格式
- 自动生成的格式

人工标注的格式

- 显式标识出输入与输出，让大语言模型自动学习到输入与输出之间的映射关系。

(1) 包含输入与输出的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：11。I

示例模板：问题：{输入} 答案：{输出}

具体示例：问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：11。

- 进一步，在提示内部加入相关的任务描述有助于模型更精准地理解任务的要求。

(2) 增加任务描述的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：11。1

示例模板：下面是一个小学数学问题。问题：{输入} 答案：{输出}

具体示例：下面是一个小学数学问题。问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：11。

- 最后，可以在输出中加入思维链，展示模型的逐步推理过程。

(3) 增加思维链的示例格式：

输入：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？

输出：让我们一步一步地思考。罗杰一开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。因此答案是 11。

示例模板：下面是一个小学数学问题。问题：{输入} 答案：{输出}

具体示例：下面是一个小学数学问题。问题：罗杰有 5 个网球，他又买了 2 罐网球，每罐有 3 个网球。他现在有多少个网球？答案：III 让我们一步一步地思考。罗杰开始有 5 个球，2 罐每罐 3 个网球就是 6 个网球。 $5 + 6 = 11$ 。因此答案是 11。

自动生成的格式

核心在于借助大语言模型来扩充新任务的示例模版。

- ① 人工标注一部分的示例模板作为种子集合加入到大语言模型的输入中；
- ② 利用大语言模型为新任务生成相应的示例模版；
- ③ 对这些生成的模版进行筛选和处理，使之符合任务要求。

自动生成的格式

输入: 示例输入 & 输出 & 指令 + 输入 & 输出 输出: 指令

示例输入:

Application Form:

Name:____ Age:____ Sex:____

示例输出: Name: John Doe. Age: 25. Sex: Male

示例指令: I am looking for a job and I need to fill out an application form. Can you please help me complete it?

示例输入: [10, 92, 2, 5, -4, 92, 5, 101]

示例输出: [-4, 2, 5, 5, 10, 92, 92, 101]

示例指令: Sort the given list ascendingly.

输入: Address: 123 Main Street, City: San Francisco

输出: 94105

指令: Given an address and city, come up with the zip code.

图 7: 自动生成的示例格式 (灰色文字表示模型生成指令)

① 提示设计

② 上下文学习

示例选择

示例格式

示例顺序

底层机制

③ 思维链提示

示例顺序

大语言模型往往会影响到位置偏置的影响，表现为对示例顺序具有一定的敏感性。

目的：为所选择的示例找到最有效的排序方式。

$$\underbrace{\text{LLM}}_{\text{大语言模型}} \left(\underbrace{I}_{\text{任务描述}}, \underbrace{f(x_1, y_1), \dots, f(x_k, y_k)}_{\text{示例}}, f(\underbrace{x_{k+1}, \underbrace{_\!__\!}_{\text{答案}}}) \right) \rightarrow \hat{y}_{k+1}.$$

分为两个步骤：

- 产生候选示例顺序
- 评估示例顺序质量

产生候选示例顺序

- 枚举给定示例的所有可能排列组合，然后从中随机选一种。
缺点：产生的结果具有较大的方差，可能导致模型性能的不稳定。
- ¹根据示例与测试样本之间的语义相似度进行排序，然后将相似度更高的示例放在更靠近测试样本的位置。

¹大语言模型在做出预测时，倾向于依赖于提示末端的信息。



评估示例顺序质量

- 可以直接测试大语言模型基于该示例顺序的性能，以此作为当前示例顺序的评分；
 - * 无法获得测试样本时，需要人工创建独立的验证集进行示例顺序的评估。
- 另一种不依赖测试样本的评估方法是基于该示例顺序大语言模型预测分布的熵值，选择熵值较低的作为有效的顺序。²

$$H(S) = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^k P(w_j|s_1, \dots, s_{i-1}) \log_2(P(w_j|s_1, \dots, s_{i-1}))$$

序列长度 n ; 词表大小 k

²熵值越低，意味着模型预测更加集中，则模型预测的置信度更高

① 提示设计

② 上下文学习

示例选择

示例格式

示例顺序

底层机制

③ 思维链提示

预训练阶段对上下文学习能力的影响

预训练任务和预训练数据都会对上下文学习能力产生影响。

- 预训练任务部分，即关注如何设计训练任务。MetaCL 使用了数十个不同领域的 NLP 任务作为元训练任务。对于每一个元训练任务，令大语言模型采用上下文学习的方式进行训练，即根据示例和待预测样本的输入预测对应的输出（无需任务描述）。

结果显示，元训练任务越多且越多样，模型的上下文学习能力就越强。

	Meta-training	Inference
Task	C meta-training tasks	An unseen target task
Data given	Training examples $\mathcal{T}_i = \{(x_j^i, y_j^i)\}_{j=1}^{N_i}, \forall i \in [1, C] \quad (N_i \gg k)$	Training examples $(x_1, y_1), \dots, (x_k, y_k)$, Test input x
Objective	For each iteration, 1. Sample task $i \in [1, C]$ 2. Sample $k + 1$ examples from \mathcal{T}_i : $(x_1, y_1), \dots, (x_{k+1}, y_{k+1})$ 3. Maximize $P(y_{k+1} x_1, y_1, \dots, x_k, y_k, x_{k+1})$	$\text{argmax}_{c \in \mathcal{C}} P(c x_1, y_1, \dots, x_k, y_k, x)$

图 8: MetaCL 的元训练流程

预训练阶段对上下文学习能力的影响

预训练任务和预训练数据都会对上下文学习能力产生影响。

- 预训练数据部分，即关注如何如何选择训练数据。
 - * 混合不同领域的预训练数据，增强预训练语料的多样性；
 - * 将前后相关的文本直接拼接进行训练，让模型能够更好地理解文本之间的关联性；
 - * 通过计算预训练数据和上下文学习的测试数据的梯度之间的相似度，可以筛选出具有较高相似度的训练数据子集。

推理阶段大语言模型执行上下文学习的方式

由于上下文学习不涉及显式的学习过程或参数更新，大语言模型使用示例数据的方式主要分为两种范式，包括：

- **任务识别**: 通过分析示例来理解并识别需要执行的任务。
- **任务学习**: 尝试从示例中提取正确的信息来完成任务。

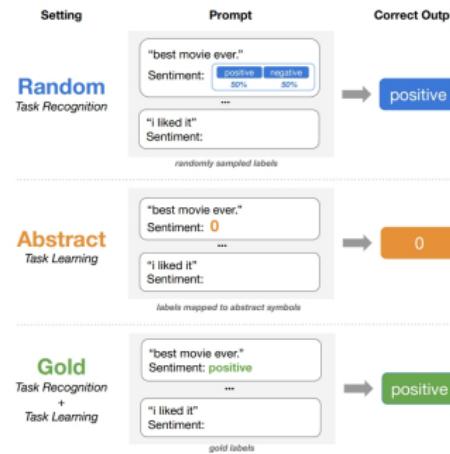


图 9: 在三种设置下进行实验以得到模型的任务识别 & 学习能力

推理阶段大语言模型执行上下文学习的方式

研究认为大语言模型能够同时展现出任务识别和任务学习两种能力，但是这两种能力的强弱与模型的规模紧密相关，其中

- * 规模较小（如 350M 参数）的模型已经能展现出较强的任务识别能力；
- * 规模较大（如 66B 参数）的模型能展现出更强的任务学习能力。

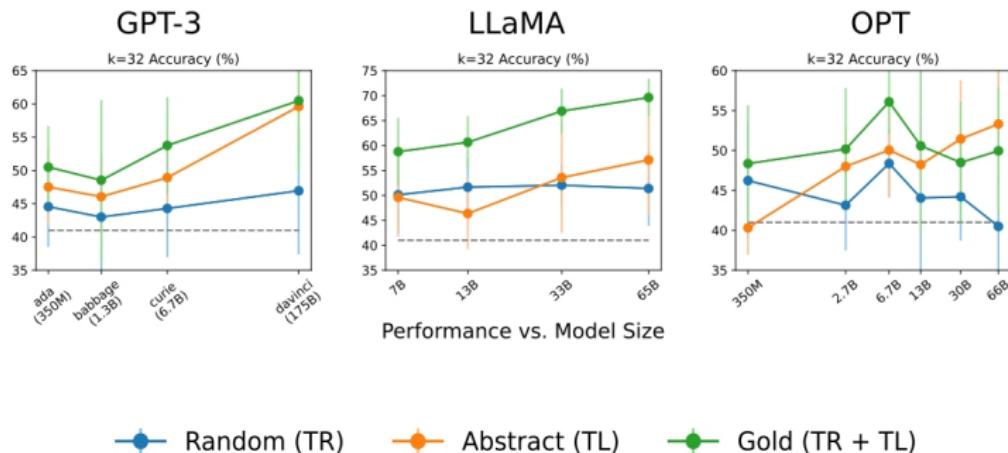


图 10: 任务识别 & 学习能力与模型大小的关系 (示例样本数 k=32)

① 提示设计

② 上下文学习

③ 思维链提示

思维链提示的优化策略

什么是思维链提示

思维链提示旨在增强大语言模型在各类复杂推理任务上的表现。
eg. 算术推理、常识推理以及符号推理

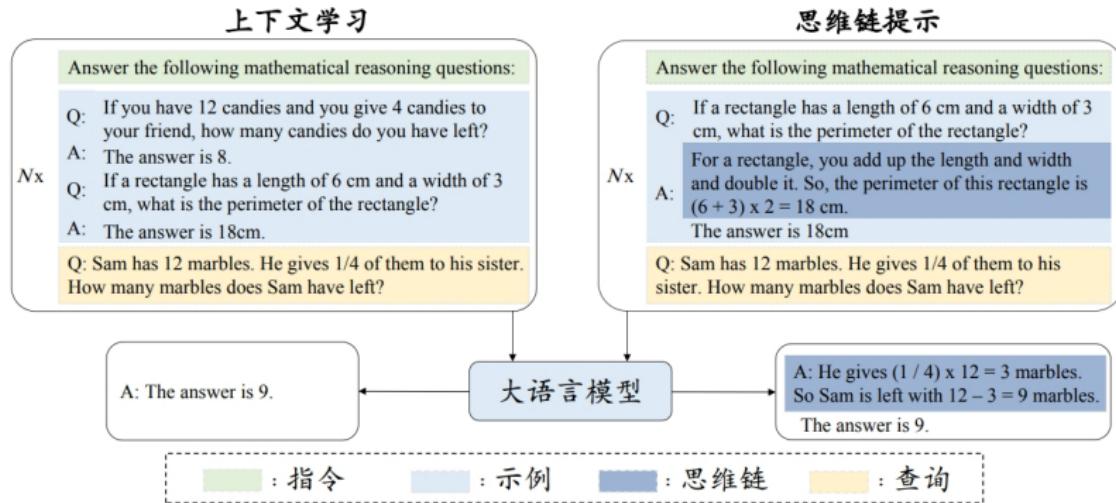


图 11: 上下文学习和思维链提示的比较

思维链提示的基本形式

思维链提示作为上下文学习的一种扩展形式

〈输入，输出〉 → 〈输入，**思维链**，输出〉

思维链提供了一系列语义连贯且具有逻辑性的中间推理步骤，有效地建立起输入与输出之间的桥接关系。

此外，提供**诱导性指令**可以让大模型生成思考过程（无需提供思维链示例），eg.

- * “Let's think step by step.”
- * “Take a deep breath and work on this problem step-by-step.”

① 提示设计

② 上下文学习

③ 思维链提示

思维链提示的优化策略

目的:解决推理过程错误、生成答案不稳定等问题。

- **思维链示例设计:**针对输入端对思维链示例进行增强。
- **思维链生成方法:**针对大模型的思维链生成过程进行改进。
- **拓展的推理结构:**针对整个思维链结构进行优化。

使用思维链提示时通常采用上下文学习的设定，即思维链提示通过示例的形式输入给大语言模型。以下为两种设计方法：

- 复杂化的思维链：一种简单有效的方法是基于复杂度指标设计思维链示例，体现在推理步骤的增多。具体而言，就是将示例问题拆解为足够多的子问题，对这些子问题的解答便构成了推理过程。
- 多样化的思维链：利用聚类算法将训练集中的问题划分为 k 个簇，从每个簇中选择距离质心最近且满足规则的问题作为该簇的代表性问题，然后再生成对应的思维链和答案作为示例。

高级的思维链生成方法

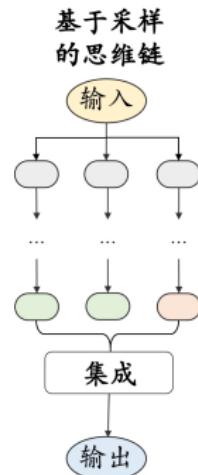
为了避免中间推理步骤出错，导致最终生成错误的答案，需要对生成思维链的过程进行改进。

- 基于采样的方法。

目标：通过采样多条推理路径来缓解单一推理路径的不稳定问题。

代表方法：*Self-consistency*

- ① 首先使用大语言模型生成多个推理路径和对应的答案；
- ② 对于这些候选答案进行集成并获得最终输出，集成方法可以选择：
 - * 投票法（最常用）
 - * 进行某种形式的加权



高级的思维链生成方法

- 基于验证的方法。

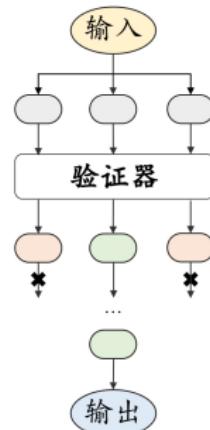
目标：防止推理过程中出现错误传递或累积的现象。

代表方法：*DIVERSE*

- ① 分别训练了针对整个推理路径和中间推理步骤的验证器。

- 如何构造整个推理路径的训练数据：通过思维链提示的方法让大语言模型生成推理路径和最终答案，然后对比是否与真实答案一致，以此来标注为正例/反例。最终得到有正例/反例标注的〈输入，思维链，输出〉三元组。
- 如何构造中间推理步骤的训练数据：对每个训练集中的问题，采样多次得到多个推理路径，对于得出正确答案的路径，中间步骤我们都认为是正确的，作为正例；反例同理。

基于验证
的思维链



拓展的推理结构

目标：链式推理结构在处理较为复杂的任务时（eg. 需要进行前瞻和回溯探索）存在一定的局限性。需要对结构进一步拓展，从而获得更强的推理性能。

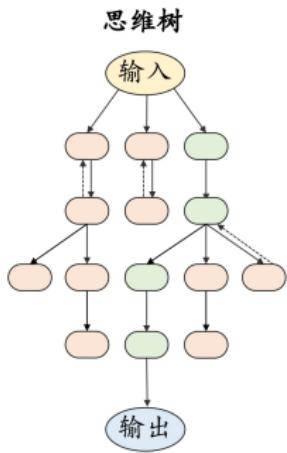


图 12: 树形结构的推理

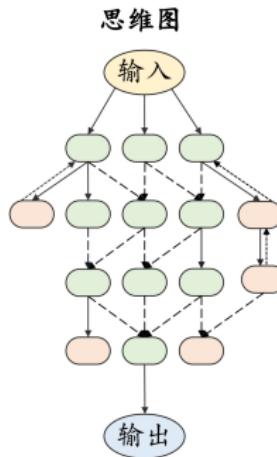


图 13: 图形结构的推理

树形结构的推理

优势：思维树从一个节点出发可以生成多个节点。

- 当某一个思考步骤无法得到正确答案时，可以回溯到它的父节点，选择另一个子节点继续推理；
- 通过前瞻性判断，能够预估当前节点得到最终答案的可能性并给出一个评分，然后优先选择那些评分更高的推理路径进行下一步的推理。

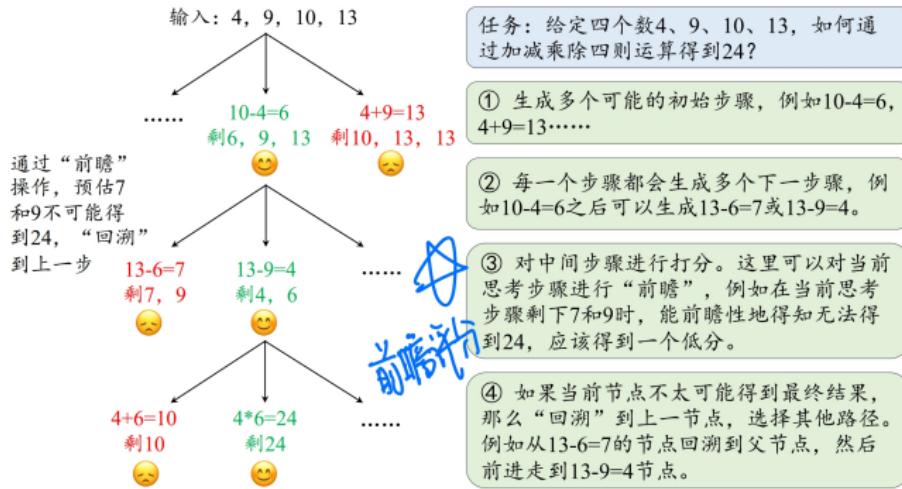


图 14: 使用思维树方法解决 24 点游戏

图形结构的推理

优势：图形结构允许图上的任意节点相连。

- 可以在生成新的中间步骤的时候考虑其他推理路径。即子节点可以和其他子节点进行汇聚，得到新的中间步骤，然后进行下一步的推理。

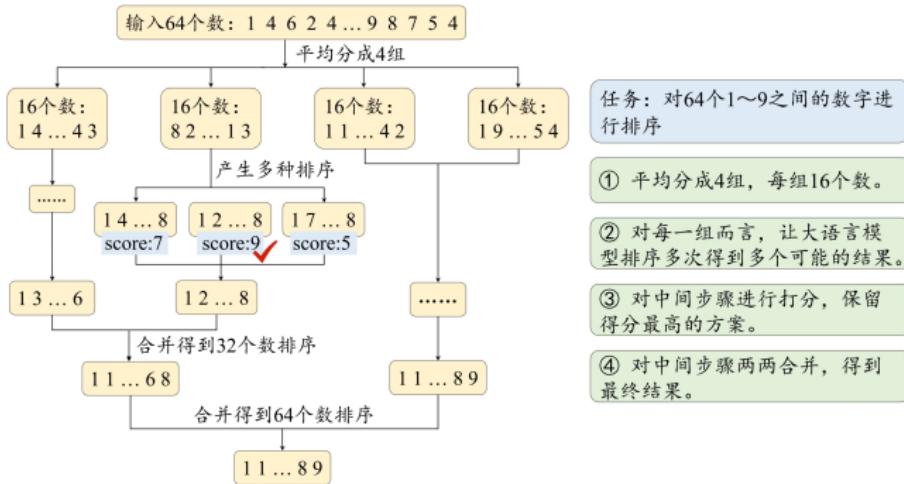


图 15: 使用思维图方法解决数组排序

思维链提示对模型推理的影响

现有的研究有以下发现：

- 对思维链进行扰动后，学者发现思维链中具体的内容并不重要，重要的是它们与问题的**相关性**以及推理过程的**逻辑性**。
- 认为思维链是一种增强的上下文学习：因为任务过于复杂，基础提示已经无法准确表达任务意图，因此需要思维链提示来**增强对于任务意图的表达**。
- 不对语言模型使用思维链提示，让其解码时生成 k 条可能的文本序列，发现具有推理路径的文本序列产生的正确答案概率显著高于文本序列。这表明，思维链提示只是**通过激发模型生成中间推理步骤**来提高其生成正确答案的概率。

Thanks!