star

Abstract

生成循序渐进的“思维链”原理可以提高语言模型在复杂推理任务(如数学或常识性问答)中的表现

- 背景

诱导语言模型的基本原理生成。

- 要么需要构建大量的基本原理（思考过程）数据集
- 要么需要使用牺牲准确性的few-shot

- 问题

提出了一种技术

- 迭代利用少量的基本原理示例
- 和没有基本原理（思考过程）的大型数据集
- 以执行更复杂的推理的能力

- 解决方法

“Self-Taught Reasoner” (STaR)
依赖于一个简单的循环

- 通过给几个带有思考过程的示例进行提示，生成回答许多问题的思考过程
- 如果思考过程有误，尝试先给出答案再让模型思考答案生成的基本原理
- 提取最终产生正确答案的所有思考数据微调模型

- 重复

与微调直接预测最终答案的模型相比，STaR 显著提高了多个数据集的性能

- 效果

Introduction

思维过程

- 将一个问题逐个分解为多个子问题
- 按照链式的方式串行思考

思维链（chain-of-thoughts）

中间步骤的“暂存器（scratchpads）”

- 可以在算术问题上获得完美的分布内（in-distribution）性能
- 以及强大的训练任务数据分布外（out-of-distribution）泛化能力
- 直接回答答案方法训练的模型　无法很好地应对训练任务数据分布外泛化能力

进行rationale generation有两种主要方法

fine-tune based on rationale dataset

- 构建rationale generation的的微调数据集
- 微调训练出一个具备rationale generation能力的fine-tune model

微调模式

few-shot examples promot method

- 通过手写或者自动模板生成技术，生成出一套rationale generation examples prompt template
- 引导LLM遵循examples的实例，进行内部rationale generation推理，得到最终答案

prompt模式

缺点

微调模式

- 人工构造微调数据的方法很昂贵
- 人工构造方法很难为每个问题任务或都构建一个微调数据集

prompt模式

- 基于prompt模板的方法（人工 or 自动生成）只能解决已知的问题任务域
- 对于未知任务的泛化能力无法保证效果

本文方法

- 利用LLM自身包含的推理能力，通过提示词，引导LLM产生高质量的rationales
- 微调那些导致正确答案的基本原理来进一步完善模型的能力
- 重复此过程，每次使用改进的模型生成下一个训练集

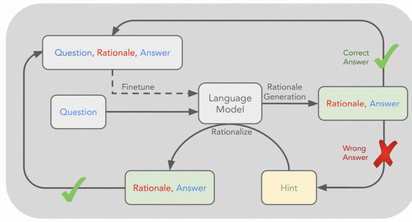
协同进化过程

- rationale generation提升了微调数据集的质量
- 微调数据集通过增强sft-model的能力，进一步也提升了rationale generation的效果

问题　原本无法解决的问题没有得到新的训练数据

改进方法　rationalization

- 于模型未能正确回答的每个问题
- 通过为模型提供正确的答案来生成新的理由
- 让模型可以向后推理——给出正确的答案，模型可以更容易地生成有用的基本原理
- 然后将这些基本原理收集为训练数据的一部分




```
Q: What can be used
to carry a small dog?
Answer Choices:
(a) swimming pool
(b) basket
(c) dog show
(d) backyard
(e) own home
A: The answer must be
something that can be
used to carry a small
dog. Baskets are
designed to hold things.
Therefore, the answer
is basket (b).
```

开发了自学推理器（Self-Taught Reasoner，STaR，）

- 在每次迭代中，首先通过尝试使用当前模型来构建微调数据集
- 使用合理化来扩充该数据集，证明模型未能解决的问题的真实答案是合理的
- 最后，在组合数据集上微调大型语言模型

效果　CommonsenseQA

- STaR improves over both a few-shot baseline (+35.9%)
- a baseline fine-tuned to directly predict answers (+12.5%)

贡献

提出了一种引导机制

- 可以从一些带有基本原理的初始示例中迭代生成基本原理数据集

rationale生成　合理化可以改善之前未答对的问题

消融实验　通过数学和常识推理领域的各种消融来评估这些技术

模型自我改进　提出了第一种允许预先训练的大型语言模型迭代地使用其语言建模能力来改进自身的技术

Method

Rationale Generation Bootstrapping（STaR Without Rationalization）

首先，我们有一个预训练 LLM

以及一个关于问题 x 的初始数据集D，并包含正确的最终答案 y　$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$

迭代优化从一个小prompt数据集（包含中间推理过程）P开始　$\mathcal{P} = \{(x_1^p, r_1^p, y_1^p)\}_{i=1}^{P}$

其中 P ≪ D（例如P = 10）

- 这里表示prompt示例集远小于初始数据集数量，
- 完整的Rationale需要在后续的迭代中逐步补全

接下来，与标准的few-shot prompting一样，我们将prompt示例集连接到 D 中的每个示例　$x_i = (x_1^p, r_1^p, y_1^p, \ldots, x_P^p, r_P^p, y_P^p, x_i)$

将拼接后的数据集x_i输入LLM　基于概率预测原理，LLM生成对应的$\hat{y}_i$以及与之对应的$\hat{r}_i$

接下来是专家修正过程（Rationalize）　过滤出能够产生正确答案的Rationale

我们基于过滤后的数据集（x_i，y_i，r_i），微调LLM，得到一个新的$sft-model$

基于新微调的sft-model，继续从prompt开始重复整个流程

不断上述重复这个过程，直到性能达到稳定水平

从强化学习的角度看，STaR 可以看作是 RL 风格的策略梯度目标（RL-style policy gradient objective）算法的近似

M 可以被视为离散潜变量模型（discrete latent variable model）　$p_M(y \mid x) = \sum_r p(r \mid x) p(y \mid x, r)$

M 在预测 y 之前首先对潜在推理原因 r 进行采样

$\mathbb{1}(\hat{y} = y)$

$$J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot \mid x_i)} \mathbb{1}(\hat{y}_i = y_i),$$

$$\nabla J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot \mid x_i)} \left[\mathbb{1}(\hat{y}_i = y_i) \cdot \nabla \log p_M(\hat{y}_i, \hat{r}_i \mid x_i) \right],$$

奖励函数来自ground truth反馈　整个数据集的总奖励期望为

- 奖励函数会丢弃所有无法得出正确答案的采样

STaR 会采用贪婪模式

- 不断缩小当前值和估计值之间的损失
- 以此完成 J 的近似优化

Rationalization合理化

对于导致失败的rationales，算法无法获得任何训练信号

合理化（rationalizonb）

- 通过输入一个hint（合理推理提示词）
- 引导LLM生成显而易见地推理过程以及正确答案

```
Q: Where do you put your grapes just
before checking out?
Answer Choices:
(a) mouth
(b) grocery cart (CORRECT)
(c) super market
(d) fruit basket
(e) fruit market
A: The answer should be the place
where grocery items are placed before
 checking out. Of the above choices,
grocery cart makes the most sense for
 holding grocery items. Therefore,
the answer is grocery cart (b).
```

hint示例

- 用于合理化（而不是生成基本原理）的few-shot提示
- 其提示包含在绿色中　生成理由的提示中提供了“(b) 杂货车”是正确答案的提示
- 后面是基本原理（思维过程）和模型生成的答案

当向我们的数据集添加合理化生成的rationales时，我们不会在其数据集中包含 hint（合理推理提示词），就好像模型在没有提示的情况下就得出了基本原理

将先前生成的数据集与合理化生成的数据集进行整合，并进行微调训练

算法流程

Algorithm 1 STaR

Input M: a pretrained LLM; dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$ (w/ few-shot prompts)

$M_0 \leftarrow M$ # Copy the original model
for n **in** $1 \ldots N$ **do** # Outer loop
　$(\hat{r}_i, \hat{y}_i) \leftarrow M_{n-1}(x_i) \quad \forall i \in [1, D]$ # Perform rationale generation
　$\{x_i^{p'}, \hat{y}_i^{p'}\} \leftarrow M_{n-1}(\text{add\_hint}(x_i, y_i)) \quad \forall i \in [1, D]$ # Perform rationalization
　$\mathcal{D}_n \leftarrow \{(x_i, \hat{r}_i, y_i) \mid i \in [1, D] \wedge \hat{y}_i = y_i\}$ # Filter rationales using ground truth answers
　$\mathcal{D}_n^{ra} \leftarrow \{(x_i, \hat{r}_i^{p'}, y_i) \mid i \in [1, D] \wedge \hat{y}_i \neq y_i \wedge \hat{y}_i^{p'} = y_i\}$ # Filter rationalized rationales
　$M_n \leftarrow \text{train}(M, \mathcal{D}_n \cup \mathcal{D}_n^{ra})$ # Finetune the original model on correct solutions - inner loop
end for

准备一个pre-train LLM及数据集

- 基模型在整个迭代算法中保持不变
- 一个包含初始few-shot prompts的数据集

拷贝一份pre-train LLM作为当前最新sft-model容器

- 该容器在后续的迭代轮次中不断更新为最新的sft-model
- 用于生成中间推理过程

通过few-shot技术，引导当前最新sft-model容器自我生成rationales推理过程，得到一个rationales dataset

专家介入进行数据蒸馏

- 从本轮rationales dataset中筛选出能够得到正确结果的rationales dataset
- 得到一个ground truth rationales dataset

基于ground truth rationales dataset

- 对pre-train LLM进行微调，并将微调训练得到的模型更新到当前最新sft-model容器中
- 每次得到一份新的ground truth rationales dataset后，不能直接基于上一轮的sft-model进行微调训练，那样容易导致过拟合
- 正确地做法是使用新的ground truth rationales dataset对pre-train LLM重新进行微调训练

why

- 新的数据是sft-model自己产生的
- 在它基础上微调会使得梯度朝着一个方向不停地更新

重复以上步骤

reference

论文学习

STaR: Bootstrapping Reasoning With Reasoning