**Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions**

## 交叉使用检索和思维链（CoT）解决知识密集型多步骤问题

**太长不看版：**

为解决知识密集型多步骤问题，作者提出IRCOT方法，将信息检索和思维链（CoT）结合，令二者相互提供支持，以增强检索和CoT两个过程。基于Retriever-Reader范式，他们使用BM25作为基础检索器加上Flan-T5和GPT3作为0/少样本的CoT生成器共同组成IRCOT检索器，用Flan-T5和GPT3作为QA Reader(对Flan-T5-\*的QA使用直接提示，对GPT3的实验使用CoT提示)来实现IRCOT QA。除此之外，作者通过设置演示集（提示演示）和超参数，提升了泛化性，并证明了模型在OOD设置下仍具有较好的优化效果。作者在HotpotQA，2WikiMultihopQA，MuSiQue和IIRC 4个需要多步骤推理数据集的开放领域设置下，评估了IRCOT的有效性。在与最近的相关领域其它模型比较后，认为IRCOT是少样本多步骤ODQA的最先进技术。

**摘要**

基于提示的大型语言模型（LLMs）在生成自然语言推理步骤或用于多步骤问题回答（QA）的思想链（CoT）方面具有惊人的强大功能。然而，当LLM无法获得必要的知识或在其参数范围内没有最新的知识时，它们就会遇到困难。虽然使用问题检索外部知识源的相关文本有助于 LLMs，但作者观察到这种一步式检索和读取方法对多步骤 QA 来说是不够的。在这里，需要检索的内容(what to retrieve) 取决于已经导出的内容(what has already been derived)，反过来（已经导出的内容）也可能取决于之前检索的内容（二者相交互）。出于该考虑，作者提出了IRCoT，一种用于多步骤 QA 的新方法，它将检索与 CoT 中的推理步骤（句子）交错，用 CoT 指导检索，反过来使用检索结果来改进 CoT。在四个数据集上：HotpotQA，2WikiMultihopQA，MuSiQue 和 IIRC，将IRCoT与GPT3 结合使用可以显著提升检索（高达21 点）以及下游 QA（最高15 点）。同时作者观察到，在分布外OOD（out-of-distribution）设置以及更小的模型（如未经过额外训练的Flan-T5-large)中，也获得了类似的实质性收益。IRCoT 可以减少模型幻觉，带来更符合事实的 CoT 推理。

**引言**

大型语言模型通过思维链（CoT）能够回答复杂的问题，特别当回答问题所需的所有信息都**被提供作为上下文**（代数问题），或者**被假定出现在模型的参数中**（常识推理）时，思维链（CoT）是极有效的。

然而，对于许多开放领域的问题，所需的全部知识并非总是存在或是存在于最新的模型的参数中，从外部来源检索知识会更有益。

为此，作者提出一个问题：**我们如何增强对需要复杂、多步骤推理的开放领域、知识密集型任务的思维链提示？**

虽然基于问题一次性从知识源检索成功地为许多基于事实的任务增强了语言模型（LM）的相关知识，但这种策略对于更复杂的多步骤推理问题有明显的局限性。

作者认为**检索和推理步骤必须互相通知（相互提供信息）**。如果没有检索，模型可能因为幻觉而生成错误的推理步骤。此外，如果没有生成第一个推理步骤，鉴于相关文本可能与问题本身没有词汇上（lexical）或语义上（semantic）的重叠或联系，支持第二步的文本就不容易确定。也就是说，**我们需要检索到的事实来生成更符合事实的正确推理步骤，以及需要推理步骤来检索相关事实。**

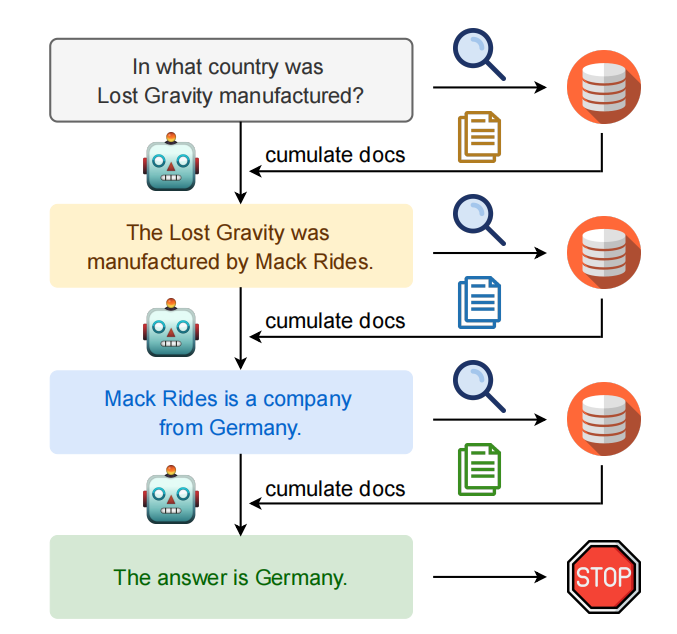
**为此，作者**提出了一种名为**IRCoT（Interleaved Retrieval guided by Chain-of-Thought）的**方法来解决这个问题。首先使用问题作为查询检索到一组基本段落。

随后，交替执行以下两个步骤：

(i) 扩展CoT：使用**问题**、**到目前为止收集的段落**和**到目前为止生成的CoT句子**生成下一个CoT句子；

(ii) 扩展检索到的信息：使用**最后一个CoT句子作为查询**检索**更多的段落加入到收集**集中。

重复这些步骤，**直到CoT报告答案**或**达到允许的最大推理步骤数**。完成后，所有收集的段落作为检索结果返回。最后，使用这些作为上下文通过直接QA提示或CoT提示来回答问题。



作者在HotpotQA，2WikiMultihopQA，MuSiQue和IIRC 4个需要多步骤推理数据集的开放领域设置下，评估了IRCOT的有效性。

· 在固定预算最优回忆设置下，使用OpenAI GPT3（code-davinci-002）时，与基线（单步问题为基础的检索）相比，IRCoT的检索召回率提高了11-21个百分点。

· 当IRCoT与基于提示的阅读器一起使用时，在下游少数几次QA性能上它还带来了显著的改善（高达15个F1点），并且在生成的CoT中减少了50%的事实错误。

· 该方法也适用于更小的Flan-T5模型（11B，3B和0.7B），显示出类似的趋势。

· 作者发现使用具有IRCoT的Flan-T5-XL（3B）时的QA性能甚至超过了58倍大的使用基于问题一步检索的GPT3。

· 这些改进也适用于分布外（OOD）设置，即在测试另一个数据集时使用来自一个数据集的演示。

· IRCOT的QA得分超过了最近一些关于开放领域QA（ODQA）的研究报告的得分，虽然与他们进行公平的对比是不可能的（参见附录C）。

主要贡献了一种新颖的检索方法IRCoT，它利用LM的思维链生成能力来引导检索，并反过来利用检索来改善CoT推理。展示了IRCoT：

1. 在IID和OOD设置下，提高了几个多步开放域QA数据集的检索和少镜头QA性能

2. 减少了生成CoT中的事实错误

3. 在没有任何训练的情况下，使用大规模（175B模型）和小规模模型（Flan-T5-\*，≤11B）都可以提高性能。

**相关工作**

**面向开放域问答的提示 (Prompting)**

· 大型语言模型（LLMs）通过利用少数几个示例（prompting）来学习各种任务。

· LLMs 需要展示连贯的推理过程（即思维链条，Chain-of-Thoughts ），即便是在只有少数几个或零个示例的情况下。

· Prompting技术已应用于开放域QA，但改善检索和QA去回答多步骤开放域问题的价值仍相对较少被探索。

近期提出的三种多步骤开放域QA方法

· SelfAsk (Press et al., 2022)引导LLMs将问题分解成子问题，并通过调用谷歌搜索API来回答这些子问题。

· DecomP (Khot et al., 2023) 是一个通用框架，它将任务分解并将子任务分派给适当的子模型。它也是将问题分解，但是将检索任务委托给基于BM25的检索器。

· ReAct (Yao et al., 2022) 系统将问题框定义为生成一系列的推理和行动步骤。这些步骤更复杂，依赖于更大的模型（例如PaLM-540B），并且需要进行微调以便在多步骤ODQA中胜过CoT方法。

这些工作都没有被证明对于未经训练的较小模型有效。

**监督的多步开放域 QA**

· 先前的工作探索了在完全监督设置下，为开放域QA进行迭代检索。Das et al. (2019) 提出了一个迭代检索模型，该模型使用神经查询表示进行检索，然后根据阅读理解模型的输出更新。

· Feldman and El-Yaniv (2019) 用类似的神经查询重构思想应用于多跳开放域QA。

· Xiong et al. (2021) 将广泛使用的密集段落检索（DPR）（Karpukhin et al., 2020）扩展到多跳设置。

· Asai et al. (2020) 利用维基百科段落中实体链接产生的图结构进行迭代多步骤检索。

· GoldEn (Gold Entity) 检索器 (Qi et al., 2019) 基于现成检索器检索的段落迭代生成文本查询，但这需要为下一个查询生成器提供训练数据。

· Nakano et al. (2021) 使用了GPT-3通过与浏览器的交互来回答长形式问题，但依赖于这些交互的人工注释。

所有这些方法都依赖于在大规模数据集上的监督训练，不能轻易地扩展到少样本的环境中。

**3链式思维引导检索和开放域问答**

作者想要在少数样本设置中，使用一个超大知识源来回答一个知识密集型的多步推理问题Q。为此他们遵循检索和阅读的范式，在检索步骤做出一些贡献（§3.1），而对阅读步骤使用标准的提示策略（§3.2）。

· 检索和阅读：即检索器先从知识源中检索文档，然后QA模型阅读理解检索到的文档和问题，生成最终答案。arXiv:2101.00774

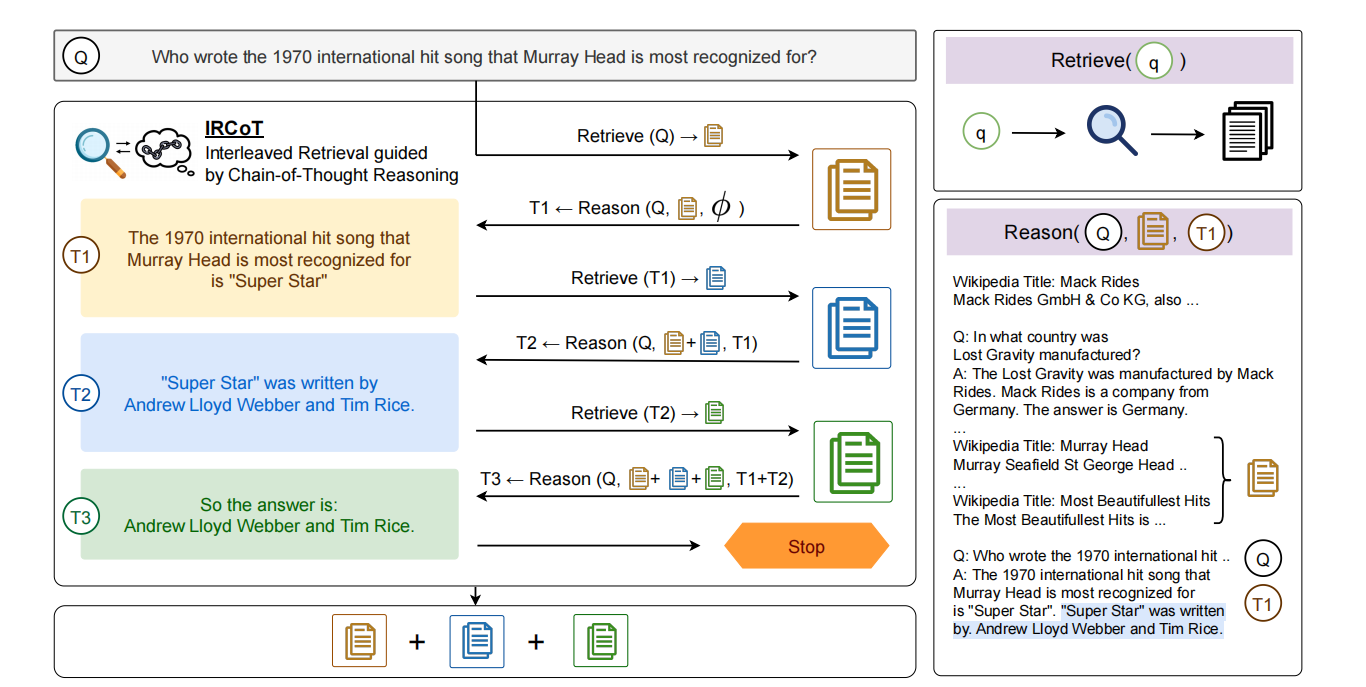
**3.1 与链式思维推理交错检索**

检索器IRCoT，可以从以下三个成分实例化：

(i)一个基础检索器，可以接受一个查询并从一个语料库或知识源返回给定数量的段落

(ii)一个具有零/少数样本链式思维（CoT）生成能力的语言模型

(iii)一小部分已注释的问题，有解释如何用自然语言（思维链）到达答案的推理步骤，和一组从知识源中集合出的支持推理链和答案的段落。



首先通过使用问题Q作为查询检索K个段落，来收集一组基础段落。然后，交错两个步骤（推理和检索）迭代，直到满足终止条件（生成的CoT句子包含“答案是：”字符串，或者已达到最大步数8）。

**检索引导的CoT步骤（“推理”）**使用**问题**、**已收集的段落**和**已生成的CoT句子**生成下一个CoT句子。

**CoT引导的检索步骤（“检索”）**使用最后生成的CoT句子作为查询，以检索更多段落并将它们添加到收集的段落中。（考虑到模型上下文限制，**收集的段落数量设为15）**

**在上下文演示中**，使用完整的链式推理（CoT）过程。**检索段落时**，使用真实的支持性段落（ground-truth supporting paragraphs）和随机采样的M个段落，将它们混合后按照模版格式连接起来。

对于**测试实例**，模型仅展示到目前为止产生的CoT句子，让模型完成剩下的部分。

对于每个推理步骤，仅采用第一个生成的句子，并丢弃其他的。**检索段落时，**展示到目前为止所有之前检索步骤中收集的段落。

任务的提示模板如下：

维基百科标题：<页面标题>

<段落文本>

...

维基百科标题：<页面标题>

<段落文本>

Q: <问题>

A: <CoT-Sent-1> ... <CoT-Sent-n>

**3.2 问答解答器**

问答解答器使用检索器检索到的段落来回答问题。

问答解答器有两个版本，通过两种提示策略实现：

**连续训练（CoT）提示**：在测试时作者要求模型从头开始生成完整的CoT。CoT的最后一句预期是形式为“回答是：...”，以便可以程序性地提取答案。否则模型生成的所有文本将被当作答案返回。

**直接提示：**使用与CoT提示相同的模板，但答案字段（“A: ”）仅包含最终答案而不是CoT。

**4 实验设置**

对于HotpotQA，使用它自带的维基百科语料库来进行开放域设置。对于其他三个数据集，作者使用相关的上下文为开放域设置构建了一个语料库。对于每个数据集，使用原始开发集中随机采样的100个问题来调整超参数，并将另外500个随机采样的问题作为测试集。

**4.1 模型**

**检索器**

使用BM25作为基础检索器。

两个检索系统：

（i）**一步检索器（OneR）**：使用问题作为查询来检索K个段落。作者选择在开发集上表现最佳的K ∈ {5, 7, 9, 11, 13, 15}。

（ii）**IRCoT检索器：**使用BM25作为它的底层检索器，并用不同大小的OpenAI GPT3（code-davinci-002）和Flan-T5作为其CoT生成器。

为了向这些LM展示上下文示例，作者

· 为所有数据集编写了20个问题的链式推理（CoT）。

· 创建了由每个数据集的15个问题组成的3套演示（“训练”）集。

· 在每次实验中，通过使用第一套演示集寻找开发集（dev set）的最佳超参数，并使用选定的超参数在测试集上评估每个演示集合。

· 对于每个实验，报告这3次结果的平均值和标准差。

在**测试时**：

· 尽可能多地将演示示例打包进模型的上下文长度限制内。

· GPT-3 (code-davinci-002) 的上下文限制为 8000 个词片。

· Flan-T5-\* 没有硬性限制，因为它使用相对位置嵌入。但是在80G A100 GPU的内存中，将Flan-T5的上下文限制在6000个词片，这是可以适应的最大数量。

IRCoT 检索器有一个关键**超参数**：

· K ∈ {2, 4, 6, 8}，表示每个步骤检索的段落数量。

· 此外，在为 IRCoT 的推理器（Reasoner）模块创建“训练”演示时，使用金标准（gold）段落和少量 M ∈ {1, 2, 3} 的分散段落。

**检索指标**

作者设置最大检索段落数量为15个，并通过在检索到的段落集合中测量金标准（gold）段落的召回率来衡量性能。

为使报告所得的指标可以被视为**在固定预算下的最优召回率（fixed-budget optimal recall）**。作者寻找能够最大化开发集上召回率的超参数K（以及IRCoT的M），并用于测试集上。

**QA阅读器（QA Reader）**

作者使用与IRCoT检索器推理步骤中使用的相同LM。对Flan-T5-\*的QA使用直接提示策略，对GPT3的实验使用CoT。

QA阅读器有一个超参数M（上下文演示中的干扰物段落的数量），作者设置M ∈{1，2，3}当与IRCoT一起使用时，检索器M被绑定为CoT生成器和阅读器。

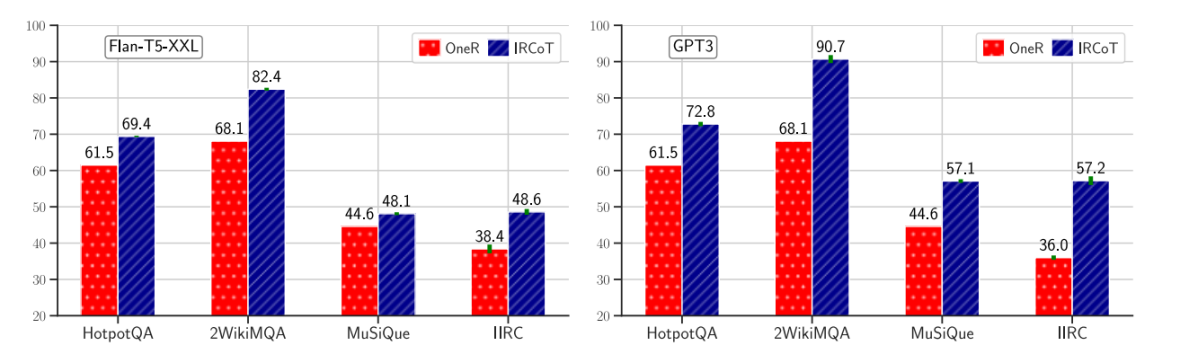
**开放域问答（ODQA）模型**

作者结合了检索器(retrievers)和阅读器(readers)，并实验了由不同语言模型LM构建的 ODQA 模型——**OneR QA**和**IRCoT QA**（CoT生成器和阅读器选择相同的LM）。并寻找可以最大化开发集上的答案 F1 得分的超参数 K和M，来选择 ODQA 模型的最佳超参数。

**5 结果**

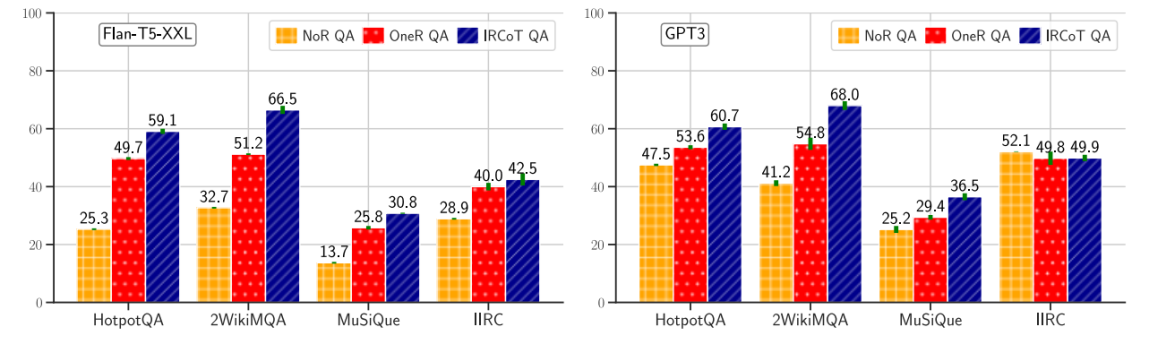
**IRCOT检索优于oneR（单步检索）**

对于Flan-T5和GPT3，IRCoT的检索召回率在所有数据集上均显著优于oneR。



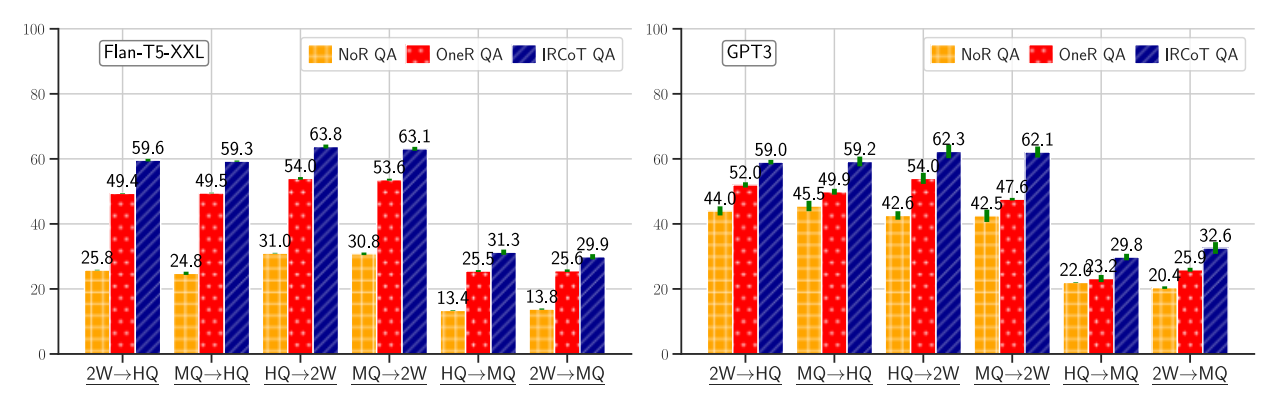
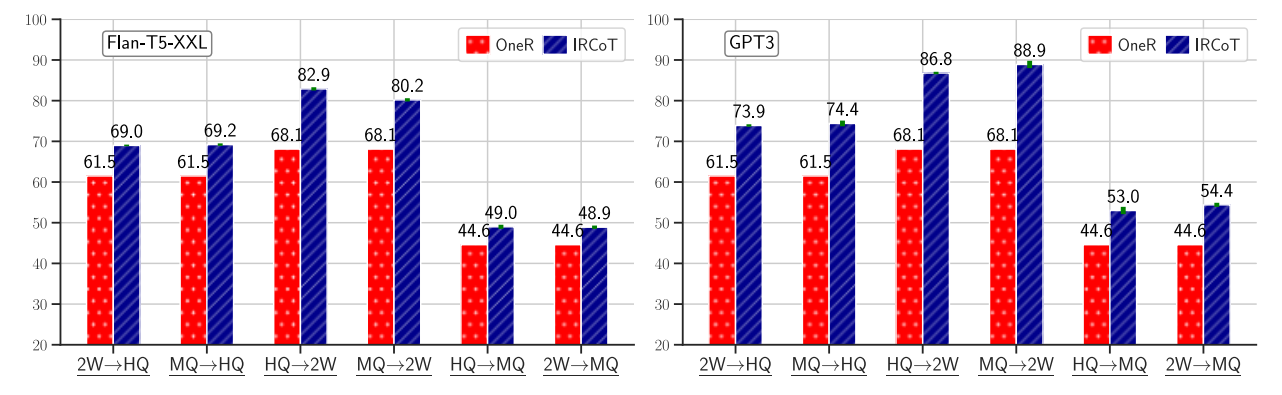
**IRCOT QA>NoR,OneR QA**

对于GPT3，尽管检索能力显著提高，IRCoT并没有提高IIRC的QA得分。作者认为这可能是因为IIRC的相关知识可能已经存在于GPT3中，其NoR QA评分相似也证明了这一点。



**IRCOT在OOD环境中有效**

由于对新数据集编写CoT是存在困难的，作者评估了NoR、OneR和IRCoT泛化到新数据集的能力。他们使用来自一个数据集的提示演示（使用特定的prompt和对应的示例来指导或者"启发”模型如何应答问题）来对另一个数据集进行评估。对于所有的数据集以及Flan-T5-XXL和GPT3的表现，依然是IRCoT检索优于OneR，IRCoT QA优于OneR QA和NoRQA。



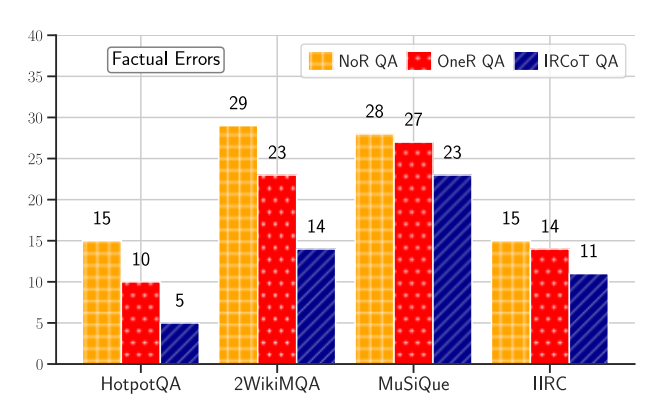
结果X→Y表明提示演示来自数据集X，并且评估位于数据集Y上

**IRCOT生成的CoT事实错误更少**

NoR完全依赖于参数知识，它经常在第一句话中出现错误，从而使整个CoT脱轨。

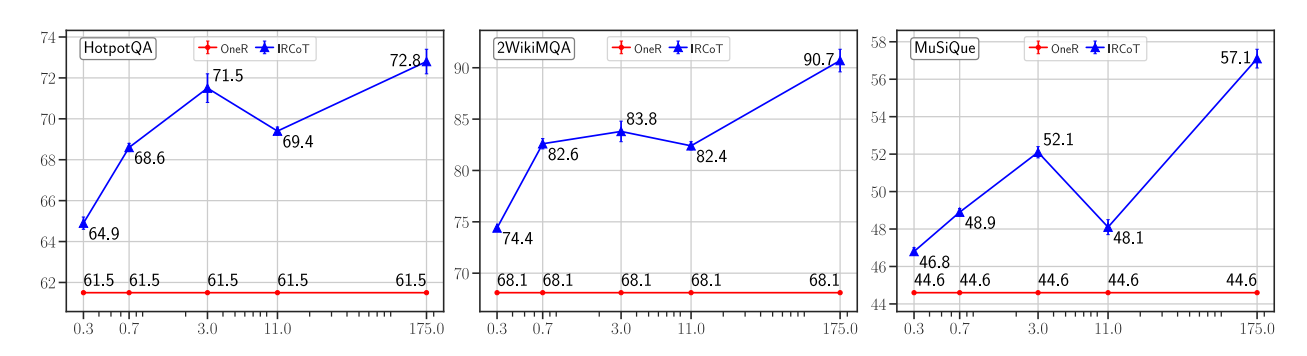
OneR可以检索到最接近问题的相关信息，不太可能在早期出现这样的错误，但在CoT之后仍然会出现错误。

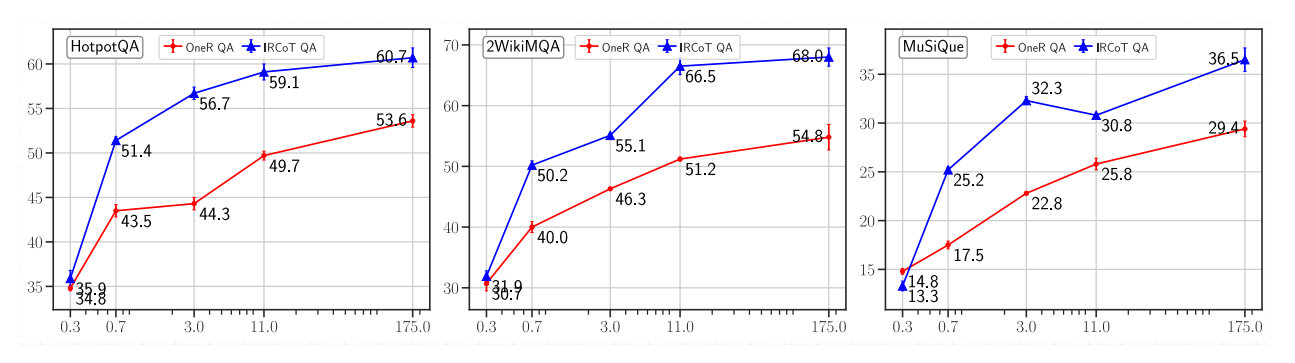
IRCoT通常能够在每个步骤中防止此类错误。



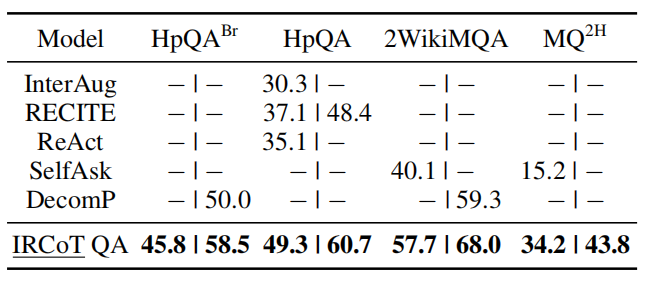
**IRCOT对小模型也很有效 3B表现大于175B GPT3 oneR/NoR**

CoT其实并不适合在小模型上使用，由于CoT需要大模型身备一些方面“最基础”的知识，如果模型过小则会导致大模型无法理解最基本的“原子知识”，从而也无从谈起进行推理。但作者比较了Flan-T5（0.2B、0.7B、3B、11B）和GPT3（175B）的OneR和IRCoT的召回率，发现即使是小型模型的CoT生成能力也可以被IRCOT改善。3B模型的IRCoT甚至优于58倍大的GPT3模型的OneR





**IRCOT是少样本多步骤ODQA的SOTA（State Of The Art，最先进技术）**



尽管这些不是面对面的比较（不同的方法使用不同的api，知识来源，甚至llm），IRCoT QA的性能显著优于所有最近的系统（InternetAugmented QA，RECITE，ReAct，SelfAsk和DecomP），在通过检索增强的llm（没有监督训练）可以实现的目标方面创造了一种新的技术状态。

**6 结论**

连锁思维提示（Chain-of-thought prompting）显著提高了大型语言模型（LLMs）进行多步推理的能力。作者利用这一能力改进了信息检索，引入了IRCoT，采用交错的CoT推理和检索步骤，彼此相互指导,以此提高了复杂知识密集型开放域任务在少数样本设定中的问答（QA）表现。在四个数据集上，无论是在大型还是相对小型的语言模型中,与一步检索相比，IRCoT显著改善了检索和QA表现。此外，IRCoT生成的连锁思维包含较少的事实错误。

**局限性**

CoT方法对于小模型的限制仍然存在，故在一定程度上限制了IRCoT的适用性。

IRCoT还依赖于基本LM来支持长输入（会存在上下文限制），因为多个检索到的段落需要适合LM的输入，以及至少一些带有段落的QA或CoT演示。未来的工作可以探索重新对检索到的段落进行排序和选择的策略，而不是将所有段落传递给LM，以减轻LM支持长时间输入的需要。

因为IRCoT对CoT的每个句子单独调用一个(L)LM，这带来了额外的计算成本。

部分实验是使用来自OpenAI（代码-davinci-002）的商业LLM API进行的，提交后，这个模型被OpenAI弃用，故这些实验的复制具有挑战性。