本科毕业论文

论文题目：**基于大语言模型的复杂陈述事实检验算法研究**

姓　　名：宋宇霖　　学　　号：19307130139

院　　系：计算机科学技术学院

专　　业：计算机科学与技术

指导教师：肖仰华　　职　　称：　教授

单　　位：计算机科学技术学院

完成日期：2023 年 5 月 8 日

目录

摘要

事实检验任务作为自然语言处理的一个子领域备受关注。而复杂陈述性事实检验任何的难度更高，目前最为先进的大语言模型直接执行复杂陈述性事实检验任务效果依然较为一般。而传统方法虽效果稍好，但通常在模型中使用多个组件来完成识别，分解，检索，推理和聚合等任务，存在着模型结构复杂，鲁棒性差，可解释性差等问题。本文：1)提出了一套基于大模型的复杂陈述性事实检验方法 ChatVerifier，利用思维链技术和上下文学习技术，充分挖掘大模型强大的理解，推理和生成能力，并结合大模型自我反思和迭代的能力，引导模型自我发现问题之后进行如修改事实分解、进一步进行补充检索等操作。2)对于每一项所使用技术的改进的影响进行了定量实验和定性分析，并结合例子进行了阐释。实验证明，本文精心设计的思维链提示词让大语言模型在多跳复杂推理验证任务上实现了超过 20%的性能提升，并超过了在该任务上微调过的其他强劲基线模型的性能。更为重要的是，ChatVerifier 还具有很强的可解释性，可以向用户展示事实验证的逻辑推理链条并提供依据，使用户可以自行判定是否接受这样的推理。这样的设计具有被应用在真实场景的事实检验任务中的潜力，可以阻止社交媒体上的谣言传播，帮助人们获取和辨别真实的信息。

关键词：复杂陈述，事实检验，大语言模型，思维链，陈述分解

ABSTRACT

Fact-checking, as a subfield of natural language processing, has received significant attention. However, verifying complex declarative facts presents even greater challenges. While state-of-the-art large language models can directly perform fact-checking tasks, their performance on complex declarative fact-checking is still limited. Traditional methods achieve slightly better results but often use multiple components to complete tasks such as entity linking and identification, claim decomposition, retrieval, machine reading comprehension and reasoning, and so on, resulting in complex model structure, poor robustness, and low interpretability.

This paper proposes ChatVerifier, a large-model-based method for complex declarative fact-checking that fully leverages the powerful understanding, reasoning, and generation abilities of large models using the techniques like chain of thoughts (CoT) and in-context learning. The model also uses self-reflection and iteration to guide itself in discovering and addressing problems, such as modifying fact decomposition or asking for further supplementing retrieval. The effects of each improvement made in the technology are quantitatively tested and qualitatively analyzed with examples for detailed clarification. The results show that the carefully designed chain of thoughts prompts in this paper enabled the large language model to achieve more than a 20% performance improvement on complex declarative fact verification tasks. Meanwhile, it outperforms other strong baseline models fine-tuned on this task. More importantly, ChatVerifier is highly interpretable and can show users the logical reasoning chain of fact verification and provide evidence, allowing users to judge whether to accept the conclusion.

I hope this design can be applied to fact-checking tasks in real-world scenarios, preventing the spread of rumors on social media, and helping people obtain and identify real information.

Key words: complex claim, fact verification, large language model, chain of thought, claim decomposition

第一章 引言

1.1 研究背景与意义

社交媒体在互联网迅猛普及的势头下高速发展，让信息的传播更加频繁且迅速。但是，发声门槛的降低也意味着不真实、不准确的言论能在社区里迅速传播，带来不好的社会影响。因此，建立一套易于使用、准确率高的事实检验算法将帮助人们便捷地判断所看到的信息是否真实，这对于阻止谣言的传播，避免虚假信息的泛滥有着重要作用。本论文选题聚焦于复杂陈述事实检验算法的研究，力求对逻辑复杂的陈述性事实进行分析和验证，并给出判定依据与判定结果。

陈述性事实检验任务，即给予一句自然语言陈述的事实性语句，检索相关证据并判定其可靠性。早期的大多数工作多采用两阶段式的流水线进行判定：首先利用检索模型从知识库中检索与陈述语句相关的证据，再将证据与陈述性事实输入到阅读理解模型中，让模型输出最终的预测标签[17]。

复杂陈述性事实检验任务，即要求模型对于包含多跳推理，亦或是多个子问题组合而成的复杂事实陈述，检索相关证据并检验陈述的可靠性。对于简单的陈述性实施检验任务,现有工作 KILT 在主流数据集 Fever 上已经实现了高达 93.5%的预测准确度[18]，基本接近了人类专家的水平。但是在复杂陈述性事实检验任务上，现有工作的效果依然不尽如人意。因此许多工作提出了将复杂陈述分解为简单的陈述亦或是问题，再进行检索和检验的方法。这样的做法在效果上确实有所增益，但是子问题的分解也存在着粒度过细，难以判定和聚合结果；粒度过粗，模型处理效果不好的问题，依然是相关领域的一大难点。

在一句相对复杂的陈述性事实中，往往有不止一个可能出错的可质疑点。为了寻找并检验这些可质疑点，目前已有的方法[2]利用了诸如实体链接器，问题分解模型，检索模型，推理模型和逻辑聚合模块等多个模块来实现事实检验。但是这样的方法具有比较强的局限性。首先，这样模型的结构十分复杂，各个模块的来源不同，各自需要在特定任务上进行训练，这导致模型的通用性较差，部署成本较高。其次，基于实体链接的方法找出的可质疑点的分解方法在实践中并不够准确，进一步影响事实检验的效果。再次，模型的鲁棒性不强。由于模块之间缺乏交互和检验，任何一个模块如果在工作时出错，都可能将错误传递至模型的整条流水线，最终显著影响模型的性能。最后，事实检验的可解释性一般。尽管各个模块所输出的中间结果可作为解释溯源的依据，但这样的溯

源需要人工一步一步回推，效率不高。除此之外，大多数其他工作的方法的推理模块在输出时仅仅输出最终结果，而不会展示整条推理链和参考依据，这对人工溯源也产生了难度，降低了模型整体的可解释性。

近期随着大语言模型（Large Language Model）的快速发展，诸如 PaLM、LLaMA、GPT-3 等千亿参数级的大模型展现出了很强的能力。已有相关工作证明大模型具有存储知识[9][10]和进行逻辑推理的能力[11]。随着模型规模的不断增大，大语言模型的生成能力不断增强，且涌现出了根据上下文学习(in-context-learning)[12]，甚至不经训练，直接理解任务描述就能完成任务的能力（zero-shot-learning）[13]。大模型的上述能力也为事实验证推理提供了一条新的道路：利用强大的推理能力和模型参数所存储的世界知识，利用上下文学习或零样本学习技术直接生成对陈述性事实进行检验的结论和依据。

尽管大模型展现出很强的能力，在不少已有工作以及我进行的实验中，使用大模型直接进行复杂陈述性事实检验，效果依然远不如传统的两阶段流水线方法。大模型存在着忽略复杂陈述中的重要错误点，对整体陈述把握不够好，陈述证据时出现幻觉(hallucination)的问题,并不能直接被作为很好的事实检验工具。

近期，不少研究逐渐发现了大模型在长逻辑链条的逻辑推理方面的涌现能力。已有工作将代码加入到大模型的训练中，显著提升了大模型的逻辑推理能力[14]。更重要的是，代码训练的加入也让模型具有了复杂思维链(Chain of Thought)推理的能力[11][15]，可以将复杂的任务分解为多个相对简单的子任务，并逐步进行分析和推理，最终给出论证结果。这些工作为利用大模型解决复杂陈述性事实验证任务提供了思路。

顺着这样的想法,本文对利用大模型进行复杂陈述性事实检验进行了探索，提出了一个新的复杂事实检验算法，基于大语言模型，利用思维链技术，将原本繁琐的复杂事实检验流水线融合入提示词（prompt）中，仅仅依赖大语言模型和外部知识检索器，即可完成复杂事实推理检验的全过程。具体而言，本文提出了名为 ChatVerifier 的复杂事实检验算法,利用思维链技术指导模型将复杂事实陈述分解为合理粒度的子陈述，并引导模型思考子陈述是否被原本的陈述逻辑蕴含，从而保证分解可信度。接下来，对分解后的子陈述并分别进行证据检索，并将证据和子陈述一起提供给模型，引导模型从证据中抽取和每一句子陈述相关的信息，并对子陈述是否被证据支持进行判断和解释。在这个过程中，同时引导模型判定证据是否充分，在不充分的情况下进行补充检索和重新验证。最后利用规则方法给出最终的判定结果，并将检验过程呈现给用户。

1.2 研究贡献与创新

本文的创新点在于，结合生成式大语言模型强大的多任务能力，将传统复杂事实检验的检验过程进行简化，达到了不错的效果。具体而言，本文的贡献如下：

1. 对复杂事实分解进行探索,利用大语言模型对复杂陈述性事实实现了高质量分解。

2. 创新性地将大模型应用于复杂事实检验任务,结合最新研究中对于思维链方法和 in-context-learning 方法的研究和结论，将问题分解、分解检验，推理验证等多个步骤融合入提示词(prompt)中，从而简化已有工作中冗长的流水线，减少中间错误的产生，并且在复杂事实检验数据集上验证得到良好的效果，超越微调过的强劲基线模型。

3. 对于所采用的各项大模型技术的加入对于进行复杂事实检验效果的影响一一进行了定量和定性的对比分析，并总结了优点与不足。

4. 实现了具有很强的可解释性，用户体验友好的 ChatVerifier 事实检验器 demo。

1.3 文章结构

本文共有 8 个章节，其中第一章对于我研究的任务进行了定义，并对相关任务已有工作的研究情况进行了简要分析，并梳理了研究背景和思维脉络，最后对我提出的方法进行了概述，点明了研究贡献与创新。第二章分为四个小节，介绍了与本文工作相关的各个领域内的技术，让读者对于该领域有一定的广度的了解。第三章以描述和形式化的语言介绍了本文方法中核心模型的架构与推理和生成的细节，有利于读者深入了解模型内部的原理，增强对全文思路的认识，为进行进一步的研究打下基础。第四章详细阐述了本文所提出的 ChatVerifier 方法各个部分的设计细节和实现原理。第五章阐述了实验数据集的获取和处理，实验设定和实验设计，各项实验的结果，并对于实验中可以观察到的现象进行了深入的分析。第六章是 ChatVerifier 的 demo 的简要阐述和演示，第七章是对于全文的总结，最后还附上参考文献、致谢和附录。

第二章 相关工作综述

2.1 复杂陈述事实分解

对于可能有多个质疑点需要验证的复杂陈述事实，将其分解为细粒度的要点一一进行判定是当前主流的做法。Pruthvi Patel 等人的工作[21]对于复杂问题进行人工分解和分析，证明了利用思维链进行问题分解和处理对于大模型无论是在事实检验，做数学题，还是回答问题等任务上都有不小的帮助。这篇工作也对自动分解技术进行了探索，发现目前无论是基于规则，基于 Bert 模型还是基于 GPT-3 模型的的自动分解技术都存在着明显的不足。此外，分解粒度的选取也是值得探讨的话题，John Glover 等人[22]的工作分析了把文段分解成不同的粒度进行验证对于推理正确率的影响，发现在不同情境下，最佳分解粒度不尽相同。

在 LOREN[2]这篇工作中,作者利用实体链接技术识别出陈述性事实中的关键短语，并针对这些短语和原本的陈述构造相应的问题来进行询问，再利用阅读理解模型进行验证。这是一种关键短语级别的验证。

还有工作另辟蹊径，在分解是不局限于陈述文本所包含的关键词，而加入了对于文本理解和推断。Ben Zhou 等人的工作[8]训练 T5 模型学习陈述中的表面含义和可能包含的潜在引申义，用于生成陈述中提及或相关的一些方面(aspects)。这样做的好处是可以深度挖掘陈述性事实背后的含义，在检索证据时匹配到更加丰富的知识，对于处理多跳问题，亦或是有引申含义的陈述（如政治相关的陈述）有比较大的帮助。

2.2 相关证据检索

在知识检索与处理方面，由于知识库中检索到的证据大多以文本的形式存在，因此大多事实验证工作[1][2]主要对文本知识进行检索和分析，常见的检索方法包括基于统计概率的 BM25 或者 TF-IDF 方法，典型应用如 ElasticSearch 框架。另一种常见的检索方法是基于语言模型编码器，利用文段向量的相似度进行检索的密集向量检索方法(Dense Retrieval)[3]。前者对文本库和用于检索的文本的特性，诸如词频，逆文档频率，文本长度等特征进行了统计分析，从而寻找相关度较高的文本。而后者则采用预训练语言模型，例如 Bert 模型，对于文本进行编码，并以文本编码向量的点积作为文本之间的匹配度，从而检索相

关的文本。在实践中，传统统计方法的响应速度较快，对于大范围文本检索的支持更好，而密集向量检索方法的检索效果会更强一些，且对于模糊匹配和搜索更加擅长，缺点是需要对文本库进行预编码，检索速度较慢且成本较高。

除了文本知识之外，维基百科知识库中也有不少知识存储在表格之中[4]。对于此类信息，有工作[5]提出了对于表格和文本信息进行双通道融合建模的方法，将表格的表格中的信息利用启发式规则转化为文本，同时将文本中的信息根据规则转化为表格，再分别利用这两类信息训练一个表格编码器和一个文本编码器，对文本信息与表格信息进行编码，最终整合多渠道的证据进行事实验证。

本文主要聚焦于复杂陈述性事实的验证，因此对于检索方面并未做过多研究和讨论。出于成本和响应速度的考量，本文选取了基于 ElasticSearch 框架开发的 Wikipedia Search Api 来对维基百科文档进行检索，提供事实验证的证据信息支持。

2.3 事实推理验证

事实推理验证任务其实可以被看成是一个文本蕴含识别任务(Recognizing Textual Entailment, RTE)，即判定给定的证据是否支持或者反对给定陈述。如果只有简单的证据,那么推理检验任务可以被直接建模为文本蕴含识别任务。但实际情况下，由于检索时难以确保匹配到直接相关的证据，且复杂陈述可能需要多来源的证据才能验证，因此在事实检验前往往需要对证据进行进一步的处理。目前最直接也比较常见的做法是将检索到的多份证据和陈述连接在一起作为模型的输入，在预训练模型的基础上微调阅读理解模型理解证据和陈述的关系，从而进行判定。也有文章提出了更多的方法来对证据做进一步处理，如利用大模型对证据进行归纳总结。

在事实验证的实际应用上，用户可能不仅仅希望得到结果，也希望得到解释。不可解释性是当前利用神经网络进行事实检验的一大问题。因此也有工作尝试探讨事实检验任务的可解释性。LOREN 这篇工作通过将陈述性事实分解为短语，并生成探针问题来一一验证，最后结合基于常识的逻辑组合推断来给出验证判断结果的依据，让问题具体的错误缘由可以通过模型对探针问题的回答得以体现，提供可以溯源和展示推断过程的一条路径。[2] 而 Nedjma Ousidhoum 的工作[7]中也提出了类似的思路，使用 SpaCy dependency parser 来提取陈述中的焦点(focal point)，进而检索和生成相应问题用于事实检验。

2.4 大模型的涌现能力

随着大模型的发展，参数量达到数百亿级，尤其是650亿以上参数规模的大语言模型自然出现了多种小模型所不具备的能力(emergent ability)[15]，这些能力没有经过特别的目标函数训练而自然而然地产生，因此被称为是涌现能力。与本文工作相关的涌现能力包括思维链推理，上下文学习和零样本学习。

思维链是指使用大语言模型在处理某一复杂任务的时候，不直接要求模型输出最终结果，而是引导模型一步一步思考，每一步进行相对简单直接的推理，形成类似于人思考时一步一步推演的思维链。这样的方法可以显著提高模型在复杂推理任务上的表现。Jason Wei 等人的工作[15]总结了思维链技术的好处在于：①通过将问题分解为多步逐一解决，在每一步上可以进行的运算更多，提供了更广阔的处理空间。②思维链展现了模型推理具体过程，为探究模型能力和研究模型的局限性提供了外部观察的窗口，也为模型给出的结果增添了可解释性。③这样的方法可以被广泛应用于包括数学运算，复杂推理，符号推理等众多任务上。④思维链可以简单地通过在提示词(prompt)中说明或者提供例子来实现。

上下文学习，即给定任务之后，提供几个任务的示例而不训练模型，模型就可以为新的测试用例准确地生成解决方案[20]。这样的特性让大语言模型具备了很强的泛化能力，向通用人工智能模型迈进了一步。这样的能力也为低数据训练的情境下完成复杂问题分解和推理提供了可能。

零样本学习指仅向模型提供任务的描述和要求,让模型在不训练(即不改动参数)的情况下完成任务。这样的能力是通过指令微调[21]所得来的，通过向模型提供包含多种多样的各类任务指令的数据来提高模型对于没有见过的任务的适应性和理解能力，从而在不训练的情况下达到好的效果。

第三章 前导知识概述

3.1 模型架构和原理介绍

3.1.1 Transformer 模型架构

2017 年，Google 团队提出了 Transformer 模型架构，以其高度的可并行性和良好的性能获得了学界和工业界的认可，并在后续的研究中得到非常广泛的使用。Transformer 模型架构包含两个部分：编码器(Encoder)和解码器(decoder)。模型结构如图 3-1[16]所示.

编码器由多个编码器层组成，在原始的设计里，编码器包含六个编码器层，每个编码器层包括一个多头注意力层和一个前向全连接层。六个编码器被串行在一起，前一层的输出作为下一层的输入，最终输出便是最后一个编码器的输出。

解码器和编码器类似，也是由六个解码器层组成。每个解码层包含三个子层，分别是：①带掩码的多头自注意力层，对解码器的输入向量；②多头注意力层，同时对编码的输出向量和①的输出进行变换；③前向全连接层。

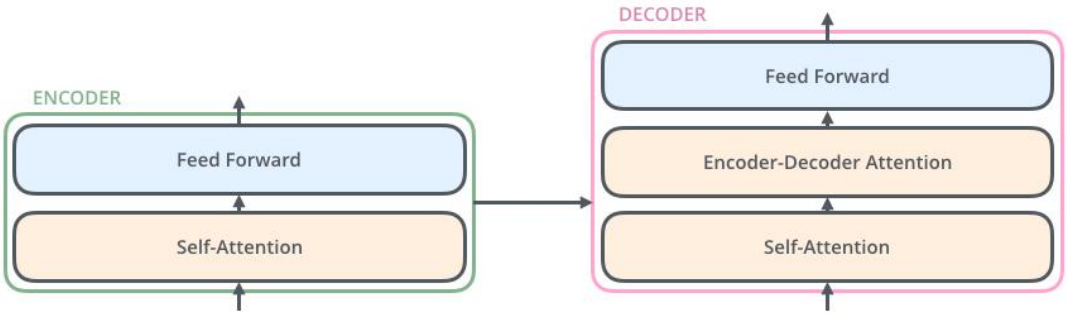

图 3-1 Transformer 模型架构简易示意图

Transformer 相比起早先如 RNN，CNN 架构的改进之处在于首次完全利用自注意力机制来计算输入的特征表示，取代了过去常用的序列对齐 RNN 等模型。自注意力机制的原理简要概述如下：

1. 首先利用可学习的参数矩阵，为每个输入矩阵的词向量计算三个向量：Query 向量(Q)，Key 向量(K)和 Value 向量(V)。
2. 将一个词的 Query 向量和其他词语的 Key 向量进行点积，从而计算输入中每个词向量对于输入中所有词的“注意力”。

3. 将注意力进行规范化后与 key 相乘，再累加，得到这个词所在位置的编码向量表示。
4. 1-3 步表示了单个注意力机制的计算方法。Transformer 采用了多头注意力，包含有 8 套独立的注意力参数矩阵，每套都利用 1-3 步计算出一个表示编码向量表示，最终连接在一起，经过一个全连接层映射得到最终编码向量表示。

形式化而言，多头注意力对输入进行编码的过程可以这样表示：

$$\mathrm{MultiHead(Q,K,V)} = \mathrm{Concat(head_1, \ldots, head_h)W^O}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

其中，映射矩阵的参数类型和维度如下：$W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_k}, W_i^O \in \mathbb{R}^{hd_v \times d_{model}}$。在实践中，Attention 可以用这样计算：

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})$$

3.1.2 GPT 模型架构与训练方法

GPT 模型是基于 Transformer 架构，通过预训练技术在大量文本上训练得到的通用生成模型。不同于常见的 BERT 模型[6]所采用的 Transformer 编码器（encoder）架构，GPT 只采用了 Transformer 的解码器（decoder）架构，通过以自回归的方式自左向右预测和生成训练文本序列下一个词的作为训练目标，学习语言的结构和规律，从而积累语言知识。具体架构的描述，在上一节中已有阐述。

3.1.3 GPT 模型的生成原理

GPT 模型作为典型的 Transformer decoder 模型，采用自回归的方式生成字符。具体而言，GPT 模型在每一步生成时，将当前已生成的字符转化为词向量输入模型，模型的输出是下一个词的分布，即一个维度为词表长度的一维向量。对于词的选取，已有多篇工作提出了不同的算法。较为常见的有贪心搜索(Greedy Search)，束搜索(Beam Search)，Top-k 采样，Top-P 采样等。实验中采用的 GPT-3.5、GPT-4 模型采用的是 Top-P 采样(Nucleus Sampling)的方法。

Top-P 采样方法基于 greedy search 的思想，每次选取一个词，也只维护一个生成窗口：首先设定一个概率阈值 P，从生成词分布中选取概率最大的一组词，使其概率之和大于 P。接下来，对这一组词重新进行一次分布映射，并根据每个词的概率进行采样，进行下一步的生成。这样的方法在生成速度，生成多样性和是否是最优解之间做了一个平衡，整体效果较佳。

第四章 系统与算法设计

4.1 问题表征

在复杂事实检验的任务场景中，给定输入$X = \{claim_i | i \in [1, |X|]\}$,其中 X 表示一个复杂陈述性事实集合，$claim_i$表示一条复杂陈述性事实。

模型需要首先将输入 X 中的每一句复杂陈述分解为多句子陈述 $subclaim$ 的集合，即对于某一条$claim$，进行如下映射：

$$① f_{decompose}(claim) \rightarrow (subclaim_1, subclaim_2, ..., subclaim_j)$$

接下来，并根据对其中的每一条子陈述$subclaim$，检索对应的证据集合：

$$② f_{retrieve}(subclaim) \rightarrow (evidence_1, evidence_2, ..., evidence_k)$$

这样一来，得到由输入集合X可以得到集合X_{Decomp_Evi}:

$$X_{Decomp\_Evi} = \left\{\{(subclaim_j, (evidence_1, ..., evidence_k)) |\ j \in N^+\}_i \middle| i \in [1, |X|]\right\}$$

最后，利用对于X_{Decomp_Evi}中的每一个元素X_d进行事实检验：

$$③ f_{multistep-verify}(X_d) \rightarrow (label, explanation)$$

综合①②③，最终得到标签集合 Y:

$$Y = \{(label_i, explanation_i) | i \in [1, |X|],\ label_i \in \{Supported, Refuted, NEI\}\}$$

整合起来，整个任务可以被形式化的表示为:

$$f_{verify}(X) = Y$$

4.2 模型总览

图 4-2 展示了本文提出的复杂事实检验方法 ChatVerifier 的框架总览。该模型受到 John Glover 等人的工作[22]的启发，采用先进行事实分解，再让模型反思和修改分解，进一步利用每一条子陈述进行事实检索，最后进行事实检验的流程。在这个过程中，如果模型发现证据信息不充分，还可以调用其他检索手段进行进一步检索，提高检索内容的质量。

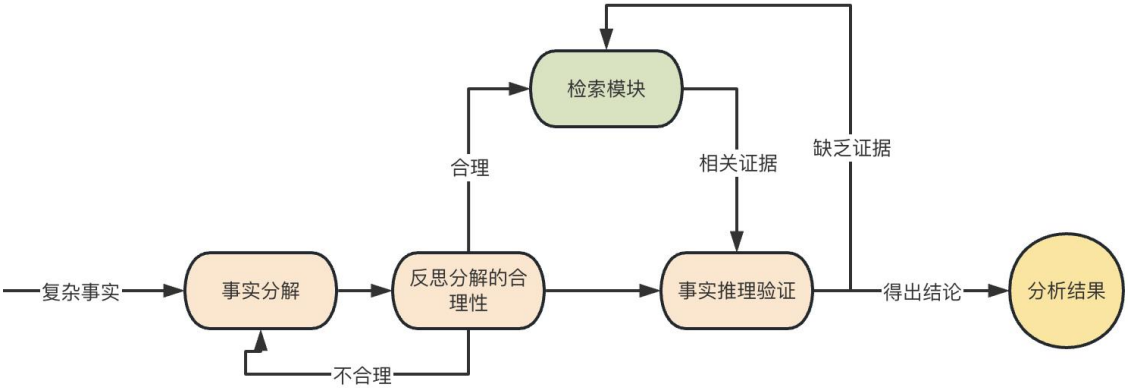

图 4-2 ChatVerifier 模型框架总览

4.3 算法设计

4.3.1 事实分解

事实分解是复杂事实检验的重要一步，这一步分解的好坏关系到知识检索的效果，以及事实验证的准确性。如果分解粒度过细，可能使检验过程繁琐，出现错误的概率增加。如果分解粒度不够细，也可能导致检索时难以充分检索每一个关键点的证据，亦或者是模型在判定时出错。在设计分解方法时，我主要考虑了两个分解原则：

1. 分解粒度相对较细。例如，
   a) 对于完全独立的两个句子，应该分开。
   b) 对于出现从句的两个句子，应该分别陈述。
   c) 对于出现连词、同位语等可以拆分的语法机构时，应该进行拆分，保证单句长度较短。
2. 与原本的陈述性事实等价。等价意味着，分解后的一组子陈述和原本的陈述应该相互逻辑蕴含。

在以这两条原则作为目标的驱动下，我参考了 Yao Fu 等人的工作[23]，对于提示词进行了多轮迭代。这其中主要考虑的想法有：

1. 分解提示词前后指代一致，不出现歧义
2. 分解提示词和证据之间分为多轮独立处理，避免分解时融入证据信息
3. 格式规范，便于自动化分析和检验

经过调整之后的提示词能够比较好的保证以上分解原则，但是出现了普遍的过度分解的现象。

原复杂陈述性事实：

Fred Mace played for the Stalybridge Celtic Football Club, an English football club, in 1919.

分解后的子陈述：

1. Fred Mace is a person.

2. Stalybridge Celtic Football Club is an English football club.

3. The Stalybridge Celtic Football Club existed in 1919.

4. Fred Mace played for the Stalybridge Celtic Football Club.

例 4-3-1.1 过度分解的例子

如例 4-3-1.1 所示，其中第一条和第三条子陈述虽然正确，但却是相对冗余的。如果任由模型这样子分解，会增加检索和验证的步骤，引入更多的误差和错误，降低效率和准确度。其次，这样的设计对于用户体验也不友好——分解逻辑过于冗长会影响用户体验。合理而简洁的分解应如例 4-3-1.2 所示：

合理的子陈述分解：

1.Stalybridge Celtic Football Club is an English football club.

2.Fred Mace played for the Stalybridge Celtic Football Club in 1919.

例 4-3-1.2 合理的分解

根据 Yao fu 等人的工作[23]指出，逻辑复杂的样例更能够提升模型进行思维链思考的能力，并让模型学习例子的逻辑。因此，我在观察大量分解样例之后，总结出几处模型容易犯错的情况，并在分解步骤的提示词之中添加了人工修正后的样例用于指导模型的分解。这样的思路加入之后，基本解决了过度分解的问题。

我对于每一套迭代的提示词算法设计进行了实验（详见 5-2 实验设计）。最终多版本的分解次分解提示词设计详见附录 1-1，最终在 ChatVerifier demo 中应用的分解提示词详见附录 1-2.

4.3.2 分解反思循环

在复杂陈述分解时，可能出现分解后的子陈述和分解前的复杂陈述性事实逻辑不一致的情况。如例 4-3-2 所示：

原复杂陈述性事实：

Terry Alan Kath obtained fame before writer and producer of the song "Sow into You".

分解后的子陈述：

1. Terry Alan Kath was a person.
2. Terry Alan Kath obtained fame.
3. Terry Alan Kath was a writer and producer of the song "Sow into You".
4. Terry Alan Kath obtained fame before he wrote and produced the song "Sow into You".

例 4-3-2 错误的事实分解

例子中，子陈述 3 是错误的，Terry Alan Kath 的父亲才是歌曲“Sow into You”的作者。为了解决这样的问题，我在分解与事实检验之间设置了一个反思循环，在得到分解结果之后要求模型检验：这一组子陈述是否和原本的陈述相互逻辑蕴含（entailment），如果不是则重新修改，是则输出特定标签 SURE。通过这样的做法，模型在实验中的多数情况下可以将犯下的错误修改正确，避免将分解错误带入到后期的事实验证。

最终在ChatVerifier demo中使用的提示词可见附录1-2分解prompt一栏。

4.3.3 事实检索

关联性强，信息充分的参考证据是模型进行复杂陈述性事实检验的重要支撑。为了提高检索的准确性，我在事实分解的时候，尽量调整提示词，让模型给出的子陈述更加完整，主谓宾语尽可能补充齐全，从而提高检索的准确性。

在检索引擎的选取上，维基百科是当下规模较大，覆盖面较广，真实性较强的文本库。其运营组织 Wikipedia 提供了基于 ElasticSearch 实现的 search API，可以自动对子陈述中的实体进行识别，并在维基百科中对相关实体对应的文本进行检索。检索的原理在相关工作综述 2.2 一节已有叙述。我在经过实验中，选择采用 WikiSearchApi 对于分解之后的子陈述进行一一检索，并将检索到的证据以{实体：信息}的形式存储下来，汇总后提供给模型进行事实验证。

在面向用户的事实检验应用中，我额外引入了更加强大的检索模块：SerpApi，该 API 基于谷歌搜索，可以提供来源更加丰富的信息，但是也引入了

一定的不真实性。具体而言，我引导模型使用 Wikipedia 的证据进行验证时，如果发现某一子陈述缺乏足够的证据信息，则输出 NOT_ENOUGH_INFO 标签。我的算法检测到该标签之后，将提取出对应的子陈述，并调用 SerpApi 进行进一步的检索，然后重新要求模型结合新检索到的信息进行第二次检验。

4.3.4 事实验证

由于大模型在长文本阅读理解方面展现了很强的能力，在信息繁多，文本分散的情况下也能把握关键知识，因此我在推断结果时。并未对多个子陈述的检索证据做分步处理，而是直接将信息合并作为第二轮对话的内容，和第一轮对话所做的分解一起输入模型，进行事实检验。在检验时，我在提示词中设计了以下环节：

1. 以强烈的用词要求模型仅仅根据我提供的事实,对于每一条子陈述进行分析，并说明支持、反对亦或是信息不足的理由。这样的要求避免了模型根据不可靠的世界知识而产生带幻觉的检验回答。
2. 要求模型先进行分析，再输出最后的标签。这样的设计充分利用了模型自回归生成的特性，让模型在生成时完成推理，提高推理准确性。
3. 说明每一个标签的具体含义，避免模型输出与分析不匹配的答案。

在设计 prompt 时，也有一些要点需要考虑。首先，大模型的推理非常严格，因此撰写提示词时，需要保证上下文的对应关系明确，表达名词一致。否则，在大量实验时，模型生成的输出可能各异，会造成不稳定性。

在研究过程中，我设计和迭代的每一轮的提示词都在附录 1-1 中得以完整体现，最终效果最好的，被应用于 ChatVerifier 的提示词可参考附录 1-3。

4.3.5 补充检索

本文以维基百科提供的检索接口作为主要引擎，并在模型认定某一条子陈述的验证出现信息不足的情况时，额外加入了 google search(SerpApi)进行补充检索。没有将后者应用于默认搜索方法主要有以下几个考量之处：

1. 维基百科作为公信度高，常年开源给公众编辑，有来自开源社区大量审查者的百科全书，内容具有较强的可信度和真实性。相比之下，google search 上的内容来源更加丰富，但是可信度的保证会相对弱一些，需要用户自行甄别内容。

2. 在实验中，维基百科已经可以解决大部分的复杂陈述检验问题。因此没有必要对所有的子陈述 SerpApi 进行检索，只对某些缺乏证据的子陈述进行额外的检索即可。

3. SerpApi 需要付费使用，受限于经费十分有限，我无法使用其进行大规模实验，因此作为备选项提供给用户。

第五章 实验验证和效果分析

5.1 数据集准备

本文以 HoVer 数据集[19]作为主要测试数据集。HoVer 是一个开放领域的多跳数据集，支持事实抽取和陈述检验两个任务。本文主要聚焦于复杂事实检验，因此在事实检验这个任务上进行验证。HoVer 数据集中的每一个样例包含：

1. 复杂陈述性事实：多个陈述句的集合
2. 参考证据：一系列主题-信息对的集合
3. 标签：支持(SUPPORTED)或者不支持(NOT_SUPPORTED)，表明参考证据是否能支持检验陈述性事实的准确性。

值得注意的是，在数据集中，无论参考证据是反驳了陈述性事实，还是没有提供相关的信息，都被归为不支持(NOT_SUPPORTED)标签。

由于HoVer数据集是从HotpotQA数据集通过启发式规则方法自动生成而来，在初步试验中，经过我的人工评估，发现其中存在相当多的逻辑瑕疵，即模型犯下的错误很大一部分是因为数据集信息缺漏/标注不正确而导致。为了更加准确地评估大模型的事实验证性能，同时受限于人力和经费有限，我从测试集人工查看和筛选了 120 个标注正确的问题，作为本文中出现的主要实验的测试集。这 120 个问题从原本 HoVer 的测试集中随机抽取，标签的分布如表 5-1 所示：

表 5-1 整理后的测试集分布

标签	样例数	占比
SUPPORTED	80	66.7%
NOT_SUPPORTED	40	33.3%

5.2 实验设计

为了更为全面的评估本文中采用的各项技术改进对于复杂陈述事实评估性能的影响，本文做了大量对比实验，具体实验设定如下。使用的提示词模板和系统输入(system input)可以参考附录 1-1。

实验 1 直接检验：给定复杂陈述性事实，直接要求模型利用自己的知识进行检验。

实验 2 直接检验+外部知识：和实验 1 的设定一致，再额外提供包含全部信息的证据集，直接要求模型进行检验。

实验 3 直接检验+magic words: 和②的设定一致，再要求中加入著名的 magic words：Let’s think step by step.[11]，引导模型进行思维链推理，但不给定具体的推理过程和逻辑。

实验 4 直接 + magic words + 外部知识检验：在③的设定上加入了参考证据集。

实验 5 显式思维链检验：在提示词中显式地加入思维链的思想，引导模型先判定事实是否复杂，并将复杂事实分解为简单的事实陈述，再根据提供的事实进行事实检验。

实验 6 显式思维链+外部知识检验：和④的设定类似，加入了数据集提供的参考证据集。

实验 7 多轮显式思维链+外部知识检验：将分解，检索和检验的任务分成多轮对话完成，每一轮对话的提示词更加有针对性，也避免了两个任务混淆在一起。

实验 8 多轮显式思维连+单分解示例+外部知识检验：加入了一个简单的人工写的分解样例，用于规范模型分解和输出的格式。

实验 9 多轮显式思维链+多分解示例+外部知识检验：加入了人工改写后的四个由大量案例分析所总结出来的模型易错点典型案例，进一步指导模型的分解。

实验 10 微调过的 FlanT5-large 模型：FlanT5-large 是 google 在 2022 年最新发布的强大多任务模型，在多个任务上有非常出众的效果。我将 FlanT5-large 在 HoVer 的训练集上微调了 3 个 epoch，并在我的测试集上进行测试。

在实验设计时，为了控制变量，降低提示词表述修改对于结果的影响，我尽量仅对实验设定改动部分的提示词进行修改，其他部分的提示词尽可能保持一致。由于本工作主要聚焦于思维链和上下文学习技术对于模型效果的提升，为了排除知识检索质量对于实验的影响，这一系列实验直接采用了 HoVer 数据集中所提供的正确参考证据集进行检验。

5.3 实验参数与实验结果

实验中使用的大语言模型为 OPENAI 官方提供的 gpt-3.5-turbo 模型。超参数设定如表 5-3-1 所示，未提及的超参数为默认参数设定。

表 5-3-1 超参数设定

参数名称	参数取值
temperature	0.3

top_p	1
frequency_penalty	0
presence_penalty	0

值得注意的是，随着模型的迭代，gpt-3.5-turbo 模型在不同时间的参数可能发生改变。本文中所有实验均在 2023 年 4 月完成，所使用模型版本为对应时期版本。

实验设计章节中列举出的所有实验的结果如表 5-3-2 所示。

表 5-3-2 各项实验结果

实验方法	准确率（%）
直接检验	75.8
直接 + magic words 检验	67.5
直接 + 外部知识检验	71.7
直接 + magic words + 外部知识检验	77.5
显式思维链检验	64.2
显式思维链+外部知识检验	79.2
多轮显式思维链+外部知识检验	82.5
多轮显式思维链+单分解示例+外部知识检验	86.7
多轮显式思维链+多分解示例+外部知识检验	88.3
微调过的 FlanT5-large	86.7

5.4 实验结果分析和消融研究

5.4.1 为何直接检验正确率这么高？

如表 5-3-2 可见，将复杂陈述性事实直接输入模型，不提供任何参考证据的情况下，模型的准确率异乎寻常的好，达到了 75.8%。但事实上，这样高的准确率并不代表着模型表现很好。通过具体的样例分析，我从两个方面进行解释：

1. 通过对不同类型的数据样例分别研究发现，模型出现错误的标签分布十分不均。在 80 个标签为 SUPPORTED 的标签中，模型仅仅预测错误了 8 个，正确率达到 90%。而在 40 个标签为 NOT_SUPPORTED 的样例中，模型预测错误了 21 个，正确率仅有 47.5%。可见，模型并非预测很准确，而倾向于将大量的样例预测为了 SUPPORTED。如果按照这样的正确率进行推演，在数据集标签均匀分布的情况下，模型的预测正确率在加权修正过仅有 68.8%，排在各项实验倒数第三的位置。这样的表现比较符合常理了。

2. 通过进行人工样例分析，我发现由于没有提供证据，模型在输出时产生了明显的幻觉(Hallucination)，对于将大部分的样例，给出了不正确且不真实的解释，并将标签预测为了“SUPPORTED”。如果将解释的正确率纳入考量，而不仅仅是查看标签的正确率，模型预测正确的比例不足 20%。因此看似高的标签预测结果是建立在不正确的幻觉的基础，并不能被认定为模型的表现出色。

5.4.2 加入外部知识带来的影响

如 Chatgpt 这样的大模型对于世界知识的记忆和存储十分丰富，在很多日常情境下被当做知识库使用。因此本文测试了在不提供参考证据，直接利用大模型参数存储的知识进行事实检验的结果。从实验结果来看，Chatgpt 在面对复杂事实检验的知识密集型任务时，自我存储的世界知识并不是很可靠。在使用显式思维链检验时，提供外部检索的证据可以让检验准确率从 64.2%显著提高到 79.2%。此外，通过进行人工案例分析发现，在不提供知识时，模型很擅长信口胡诌，出现幻觉(hallucination)。我对于直接检验设定下的 10 个用例进行人工检验，其中 7 个样例是模型输出了或者错误的标签，并给出的错误的解释；1 个样例是模型自知缺乏证据支持，无法判断；而仅有两个是模型给出了正确的解释并做出了正确的判定。由此可见，对于知识密集型的复杂陈述性事实检查任务，模型的世界知识并不可靠。当前阶段下大模型进行复杂事实检验，依然十分依赖外部显式知识的补充。

5.4.3 加入思维链带来的影响

许多研究已经证明了思维链技术在复杂推理任务上可以有效提高大模型的表现。实验结果的数据也证明了这一点。在仅仅加入简单显式思维链的设定下，模型进行事实检验的准确率从 71.7%提升到了 79.2%，并在进一步改进思维链，并加入分解示例的情况下，可以将准确率提升至 88.3%。可见，思维链可以显著提升模型进行复杂事实验证的效果。通过案例分析可以发现，模型更好地关注到了复杂陈述中出现错误的一些小细节，并明确而有依据地给出了分析。在下面这个例子中，如果直接输入事实性陈述和证据，模型会无法发现出 SMERSH 和 Pinewood Studios 这两个地点是不同的，进而给出 SUPPORTED 的错误答案。

而利用 CoT 进行分解之后，模型就可以发现 SMERSH 在证据中并没有提及，并正确地指出错误之处。

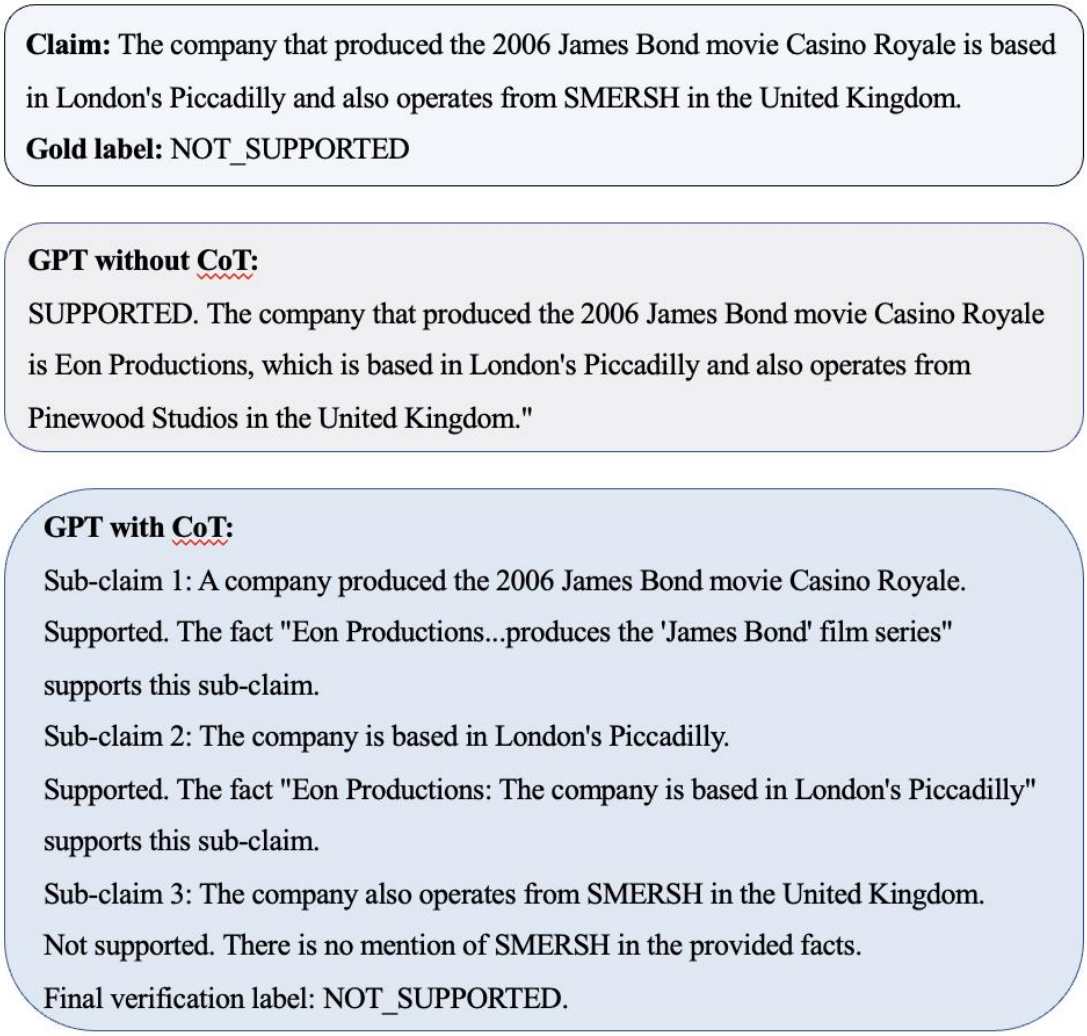

图 5-2 CoT 带来的影响

在实验结果中，我们可以看到加入了在不提供外部证据的情况下，模型在加入了显式思维链后的正确率（64.2%）远低于直接检验（75.8%），也低于采用加入 magic words 直接检验(67.5%)。尽管直接检验的正确率并不能很明显的说明问题，这在 5.4.1 小节中已有讨论，但加入思维链后反而导致效果下降，其中的原因可以通过具体案例分析来做出可能的解释。思维链将复杂陈述性事实拆解，让模型一步一步分析其中的每一个小点，因此模型对于每个问题点的审视更为仔细和慎重。在这个过程中，模型直接出现幻觉，编造证据的概率会变小，模型会更多地意识到自己缺乏证据信息，无法做出判定。因此，模型不会像在 5.4.1 小节中分析的那样，把大量的陈述性事实预测为 SUPPORTED。因

此，再标签为 SUPPORTED 的数据上，采用了思维链的模型相比直接检验，准确率大幅下滑，而在标签为 NOT_SUPPORTED 的数据上，采用了思维链的模型准确度虽有提升，但整体来看，准确度依然下降。

此外，实验还探究了隐式思维链和显式思维链的效果差别。隐式思维链，即不明确在提示词中指明模型应该如何做，在直接检验的简短提示词之后，加入已经在许多论文中被广泛应用的“magic words”: Let’s thinks step by step。从具体的样例分析来看，隐式思维链下的模型的逻辑思路更加多变，有时简洁，有时复杂，有时特别详细，有时会偏离正确的方向。单纯从准确率来看，隐式思维链和简单的显式思维链的差距不大，仅为 1.6%，但是当对于思维链进一步优化，并显式加入示例时，两者的差距可以拉大至 9.1%。可以见，精心设计的带示例的显式思维链对于模型的复杂推理能力依然有不小提升。对于复杂事实检验这样的任务较为具体，思路明确的任务，明确地设计和指明一条思维链，对于指导大模型完成任务更有帮助。

5.4.4 加入分解示例带来的影响

从实验结果可以看到，在加入单个分解示例后，模型的准确率由 82.5%提升到了 86.7%。在加入针对模型易犯的错误设计的示例之后，模型的准确率又稍有提升到了 88.3%。可见，大语言模型的上下文学习能力很强，仅需一个例子就可以很好地学习到分解和格式，思路和规范，同时在下游任务中展现出效果。而根据观察模型在直接输出时常犯的错误，将其修正后作为样例提供给模型，可以进一步提升模型的性能。

但是在实验中也观察到，当提供更多的示例时，模型的分解会稍微更为死板，会被例子所过多的约束。因此两到三个多样化的，精心选取的，可以纠正模型错误的示例是比较好的选择。

5.5 错误分析

这一章节选取实验中效果最好的，也是最终 demo 中采用的多轮思维链+多示例设定下的模型，作为错误案例分析对象，以探讨当前方法的局限性。

对于标签为 SUPPORTED 的复杂事实陈述和标签为 NOT_SUPPORTED 的复杂事实陈述，我各抽取了检验时模型出错的五个案例进行人工核查，总结了出现错误的理由如表 5-5 所示：

标签类型	错误类型	出现次数
SUPPORTED	非直接回答	2
	阅读理解有误	2
	分解逻辑关系理解有误	1
NOT_SUPPORTED	阅读理解有误	3
	对提示词执行不到位	2

表 5-5 错误案例类型统计

非直接回答的含义是模型没有很好地执行提示词中描述的任务，而是对于任务和输入本身进行了其他解读亦或是反问。例如，面对一段由多句陈述组成的复杂事实，模型并未按照提示词要求进行分解和检验，而是反问道“我很抱歉，但是你提供的陈述似乎是两句毫无关系的独立陈述，请问你是否可以提供单句的陈述让我来帮你解答？”这样的情况尽管非常少见，但是依然存在并影响了模型表现。这样的现象存在可能是由于 gpt-3.5-turbo 模型为了满足作为对话 AI 的需求，在进行人类反馈指导的强化学习（RLHF）时引入了一些对话的技巧，如进行反问来明确用户的意图。这样的情况其实只需要继续明确需求，就能正常完成检验。后续工作可以考虑加入对于这种情况进行检测并进一步明确需求的设计。通过更换非面向对话设计的模型，或者对模型进行适量的微调，也可能解决这样的情况。

分解逻辑关系理解有误的情况在一些涉及逻辑关系的陈述中存在。例如一句陈述表示：“Robert Cunninghan Humphreys 所在的大学和科罗拉多大学不都是私立学校。”模型成功分解并验证了 Robert Cunninghan Humphreys 所在的大学是哥伦比亚大学，并且正确指出了“哥伦比亚大学不是私立大学”是错误的，而“科罗拉多大学不是私立大学”是正确的。但是模型不理解“不都是(are not both)”这样的逻辑词只要求至少二者其一不是，错误地将“不都是”理解为了“都不是”，所以给出了错误的判定。从这一点看出，模型对于逻辑词的理解有待加强，现阶段识别逻辑词并利用规则进行约束或许可以提高模型的推理的准确率。

阅读理解有误错误是一类相对常见的错误，在不同标签的错误样例中都会出现，即没有很好地理解参考证据对于文本的支持亦或是反对。在实际情况中，可能出现在文本表达不够接近自然语言（如用文本直接表示表格化数据，产生

大量的逗号连接的词语集合，缺乏文本化的表达）、需要更多领域专业知识来理解综合理解、理解出现偏差的情况。

对提示词执行不到位是一类目前仅在标签为 NOT_SUPPORTED 的案例中发现的错误。尽管提示词中已经明确说明，仅当证据支持每一句子陈述时，标签为 SUPPORTED,模型依然可能错误地认为只要证据中没有反驳陈述的信息即可认定为 SUPPORTED。这样的现象和模型对于 SUPPORTED 一词的固有理解有关，尽管发生的几率很小，但其出现意味着模型对于 prompt 的理解并不完全可靠，存在一定的不稳定性。

第六章 Demo 展示

除了实验场景下的测试，我也搭建了一个真实用户场景下的事实检验 demo。在模型设计方面，我采用了实验中效果最好的多轮显式思维链+多分解示例+外部知识检验设定，对提示词做了少许修改，使目标标签变为三类：

1. 支持(SUPPORTED)：检索到的证据支持陈述性事实。
2. 反对(REFUTED)：检索到的证据反驳了该陈述性事实。
3. 信息不足(NOT_ENOUGH_INFO)：当前检索到的信息不足以进行检验。

这样改进之后，我可以根据标签对返回结果进行解析，并在信息不足时进行补充检索，并重新要求模型检验。

前端搭建上，我选用了 gradio 框架进行搭建，效果图如图 6-1 所示：

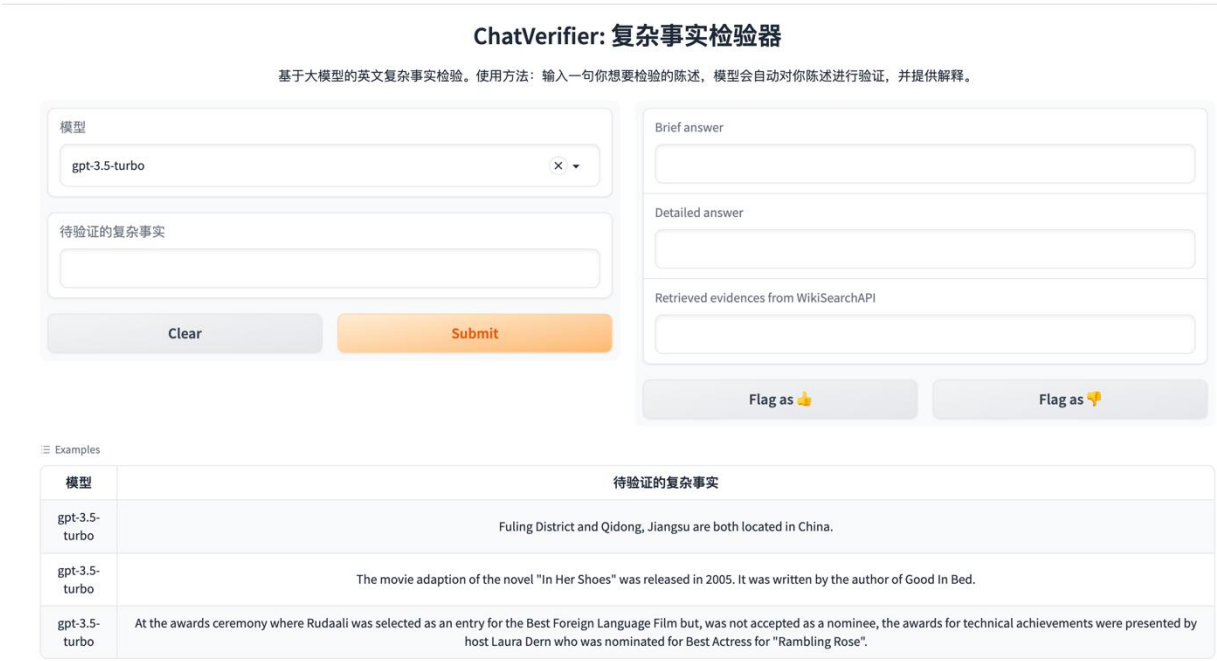

图 6-1 ChatVerifier demo 展示图

如图 6-2 所示的是一个复杂事实检验示例。ChatVerifier 模型返回了复杂事实检验的分析，对每一句分解后的子陈述都进行了阐释，并给出了最终的检验结果。在检验过程中检索到的证据也被列在了界面上，供用户参考。

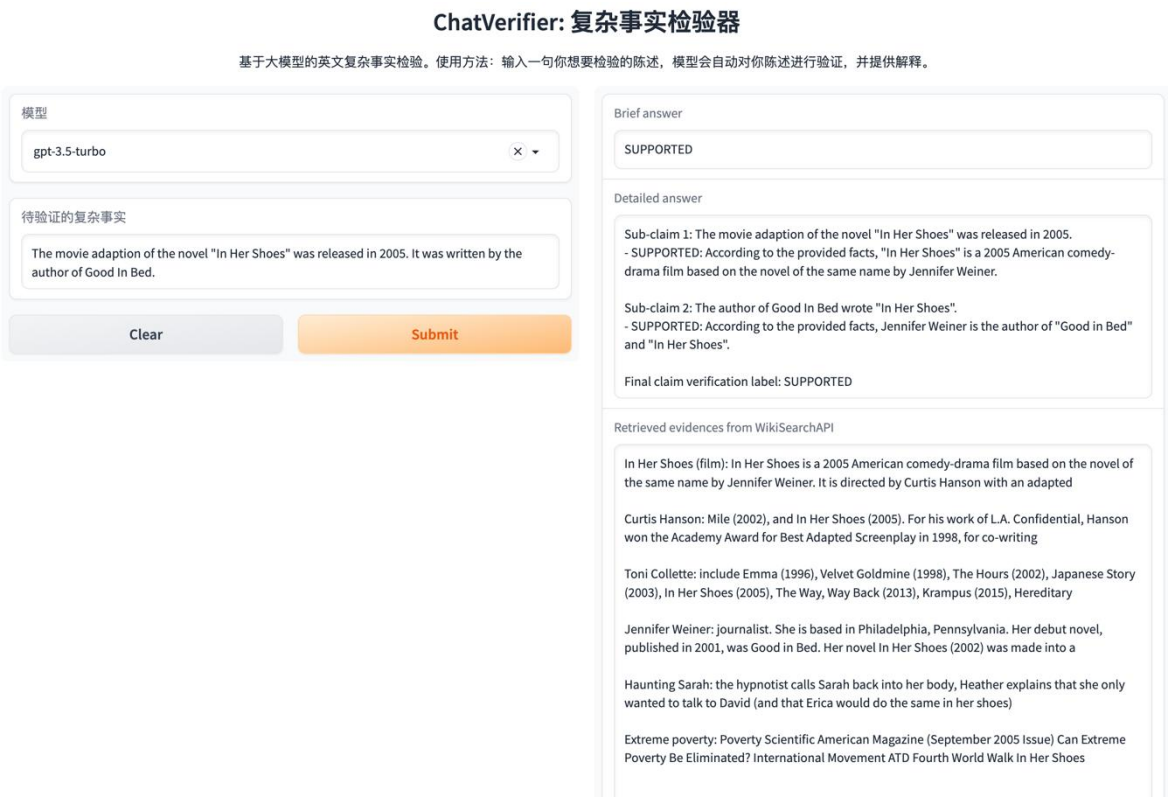

图 6-2 复杂事实检验结果示例

第七章 结论

在这篇文章里，我提出了 ChatVerifier，一套基于大模型的复杂陈述性事实检验方法，利用思维链技术和上下文学习技术，充分挖掘大模型强大的理解，推理和生成能力，对于开放领域的复杂陈述性事实进行检验。实验证明，本文精心设计的思维链提示词不仅让大模型在复杂推理验证任务上实现了超过 20%的性能提升，准确率达到 88.3%,也超过了该任务上的微调过后的强劲基线模型 FlanT5-large。更为重要的是，ChatVerifier 还具有很强的可解释性，可以向用户展示事实验证的逻辑推理链条，并提供详细的解释。这样的特性让 ChatVerifier 具备应用于现实场景的强大潜力。

参考文献

[1] THORNE J, VLACHOS A, CHRISTODOULOPOULOS C, et al. FEVER: a large-scale dataset for Fact Extraction and VERification[C]. arXiv: Computation and Language. 2018.

[2] CHEN J, BAO Q, SUN C, et al. LOREN: Logic-Regularized Reasoning for Interpretable Fact Verification[C]. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 10, pp. 10482-10491. 2022.

[3] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense Passage Retrieval for Open-Domain Question Answering[C]. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6769–6781, Online. Association for Computational Linguistics.

[4] ALY R, GUO Z, SCHLICHTKRULL M, et al. FEVEROUS: Fact extraction and VERification over unstructured and structured information[C]. Neural Information Processing Systems. 2021.

[5] Hu, Nan, Zirui Wu, Yuxuan Lai, Xiao Liu, and Yansong Feng. Dual-Channel Evidence Fusion for Fact Verification over Texts and Tables[C]. In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5232-5242. 2022.

[6] DEVLIN J, CHANG M W, LEE K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[C/OL]. Proceedings of the 2019 Conference of the North, Minneapolis, Minnesota. 2019. http://dx.doi.org/10.18653/v1/n19-1423. DOI:10.18653/v1/n19-1423.

[7] Ousidhoum, Nedjma, Zhangdie Yuan, and Andreas Vlachos. Varifocal Question Generation for Fact-checking[Z]. arXiv preprint arXiv:2210.12400 (2022).

[8] Zhou, Ben, Kyle Richardson, Xiaodong Yu, and Dan Roth. Learning to Decompose: Hypothetical Question Decomposition Based on Comparable Texts[Z]. arXiv preprint arXiv:2210.16865 (2022).

[9] PETRONI F, ROCKTÄSCHEL T, RIEDEL S, et al. Language Models as Knowledge Bases?[C/OL]. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China. 2019. http://dx.doi.org/10.18653/v1/d19-1250. DOI:10.18653/v1/d19-1250.
[10] Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How Much Knowledge Can You Pack Into the Parameters of a Language Model?[M]. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5418–5426, Online. Association for Computational Linguistics.
[11] ROBERTS A, RAFFEL C, SHAZEER N. How Much Knowledge Can You Pack Into the Parameters of a Language Model[Z]. arXiv: Computation and Language. 2020.
[12] BROWN TomB, MANN B, RYDER N, et al. Language Models are Few-Shot Learners[Z]. arXiv: Computation and Language. 2020.
[13] Wei, Jason, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners[Z]. arXiv preprint arXiv:2109.01652 (2021).
[14] CHEN M C, TWOREK J, JUN H, et al. Evaluating Large Language Models Trained on Code[Z]. Cornell University - arXiv. 2021.
[15] Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. "Chain of thought prompting elicits reasoning in large language models[Z]. arXiv preprint arXiv:2201.11903 (2022).
[16] illustrated-transformer[EB/OL]. https://jalammar.github.io/illustrated-transformer/
[17] Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler et al. Retrieval-augmented generation for knowledge-intensive nlp tasks[C]. Advances in Neural Information Processing Systems 33 (2020): 9459-9474.
[18] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel.

KILT: a Benchmark for Knowledge Intensive Language Tasks[C]. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2523–2544, Online. Association for Computational Linguistics.
[19] Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. HoVer: A Dataset for Many-Hop Fact Extraction And Claim Verification[C]. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 3441–3460, Online. Association for Computational Linguistics.
[20] Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan et al. Language models are few-shot learners[C]. Advances in neural information processing systems 33 (2020): 1877-1901.
[21] Pruthvi Patel, Swaroop Mishra, Mihir Parmar, and Chitta Baral. Is a Question Decomposition Unit All We Need?[C]. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 4553–4569, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
[22] John Glover, Federico Fancellu, Vasudevan Jagannathan, Matthew R. Gormley, and Thomas Schaaf. Revisiting text decomposition methods for NLI-based factuality scoring of summaries[C]. In Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM), pages 97–105, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.
[23] Fu, Yao, Hao-Chun Peng, Ashish Sabharwal, Peter Clark and Tushar Khot. Complexity-Based Prompting for Multi-Step Reasoning[Z]. ArXiv abs/2210.00720 (2022): n. pag.

致谢

本文是在肖仰华导师和陈江捷学长的指导下完成的，从开题到构思，包括方法设计实现和论文撰写与修缮，都得到了两位前辈的悉心指点。江捷学长对于我的毕业设计十分关心，在每周的周会上与我讨论实现的思路和细节，指导我完成算法的实现和设计。他作为博士生的视野与学术能力让我十分钦佩，他的悉心帮助也让我的毕业设计，和我个人的发展都受益匪浅。没有他的帮助，我无法在短短两个月内完成这样的工作。肖老师的意见和指导深入而具有洞察力，他指引我不要拘泥于单一的评测指标，重视实际使用场景的用户体验，对我的设计方向有着很积极的影响。

同时也要感谢与我同住 12 号楼的室友伯锐，祁昊，源泽，少文等小伙伴们，我们一起交流毕业设计的思路和想法，并督促、鼓励彼此推进进度，完成实验，撰写论文。这段日子的陪伴让我们的友谊更加坚固，这段共同的经历也成为了我们人生宝贵的财富。

在探索毕业设计的这段日子，我正处于本科生涯的末期，面临进入社会，决定生涯走向的难题。这段时间里，我和许多朋友探讨过自己的未来，也和许多朋友分享了自己的迷茫与苦恼。我深深地记得在凌晨的南区踱步思索，在太湖沿岸与朋友们彻夜畅谈，在交叉二号楼的讨论室里与学长学姐们探讨人生道路的选择。经过这段时间的思考，我对前路的方向逐渐明晰。踏上了走向社会的第一步。

由衷地感谢这段日子里与我交流，给予我建议和帮助的师长与朋友们，你们的智慧与经验让我看到了人生的丰富的可能性，为我扫去了不安和迟疑，注入了踏实与信心。

以上各位师长，朋友给予了我莫大的帮助，仅在此献上我最真挚的感激。

同时也要在此感谢帮助和支持我的父母，他们开明的育人方式让我获得了独自思考和做决定的机会，在这个过程中，我磨练了自己的性格，也明确了自己的方向。他们的支持和鼓励让我坚定地走自己理想中的道路，不被形形色色的纷扰迷失了脚步。

附录

附录 1-1 各项实验使用的 prompt

System_input 在未提及时统一设定为：
You are a helpful assistant that verifies complex claims.

以下为各种实验设定下使用的提示词：
1. 直接检验：
Please verify this claim using all information you know: {claim}. The final answer of verification for the claim should be either SUPPORTED or NOT_SUPPORTED. Please give the final answer in the last line.
2. 直接+证据检验：
Please verify this claim: {claim}. The final answer of verification for the claim should be either SUPPORTED or NOT_SUPPORTED. Please give the final answer in the last line. Here are some facts for your reference: {evidence}.
3. 直接+magic words 检验:
Please verify this claim using all information you know: {claim}. The final answer of verification for the claim should be either SUPPORTED or NOT_SUPPORTED. Please give the final answer in the last line. Let's think step by step.

4. 显式思维链检验：
First, you need to decompose the claim into several sub-claims. Then, try to verify these sub-claims using all information you know. Finally, give the final answer of verification for the claim. The final answer should be either SUPPORTED or NOT_SUPPORTED. Please give the final answer in the last line.

5. 显式思维链+外部知识检验：
Please verify this claim: {claim}. First, you need to decompose the claim into several sub-claims. Then, try to verify these sub-claims using only the facts I provide. Finally, give the final verification of the claim. The final verification should be either SUPPORTED or NOT_SUPPORTED. Please give the final verification in the last line. Here are some facts I provide: {evidence}.

6. 多轮思维链+外部知识检验

Round1:分解

System input: You are a helpful assistant that decomposes complex claims.

Prompt:参考附录 1-2，删除所有例子

Round 2: 检索和验证

System input: You are a helpful assistant that verifies complex claims.

Prompt:参考附录 1-3

7. 多轮思维链+单分解示例+外部知识检验

请参考附录 1-2 的格式，只保留了 4 个样例中的第 1 个。

8. 多轮思维链+多分解示例+外部知识检验

和附录 1-2 一致。

附录 1-2 复杂陈述分解一步使用的 prompt

Please decompose the claim into several sub-claims, no matter if it is true.
If the claim can not be decomposed, just say BRIEF.
Here are some examples:

claim: "Arnold is currently the publisher and editorial director of Media Play News, one of five Hollywood trades and the only one dedicated to the home entertainment sector."
sub-claims:
1. Arnold is currently the publisher and editorial director of Media Play News.
2. Media Play News is one of five Hollywood trades.
3. Media Play Newsis the only one dedicated to the home entertainment sector.' \

claim: "Trump won the 2020 US Presidential Election."
sub-claims:
1. Trump won the 2020 US Presidential Election.'

claim: "Tazza (TV series) is a 2008 South Korean television series starring the actor who played Prince Yeonsan in a film that runs 119 minutes."
sub-claims:
1. Tazza (TV series) is a 2008 South Korean television series.
2. One actor starred in Tazza also played Prince Yeonsan in a film.
3. That film runs 119 minutes.

claim: "Adam McKay co-wrote the film that Cassandra Lang made her cinematic debut in and served as head writer for "Saturday Night Live".
sub-claims:
1. Adam McKay co-wrote the film that Cassandra Lang made her cinematic debut in.
2. Adam McKay served as head writer for "Saturday Night Live".

claim: {claim}

附录 1-3 事实检验一步使用的 prompt

Then, try to verify these sub-claims with explanations using only the facts I provide. The verification label should only be either SUPPORTED or REFUTED or NOT_ENOUGH_EVIDENCE. Finally, give the final verification label of the claim according to sub-claims.
If all sub-claims are supported, the final verification label should be SUPPORTED.
If any one of sub-claims is verified as REFUTED, the final verification label should be REFUTED.
In other cases, the final verification label should be NOT_ENOUGH_EVIDENCE.
Here are some facts I provide:
{fact_list}