Computer Science and Application 计算机科学与应用, 2024, 14(7), 18-26
Published Online July 2024 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2024.147159

Hans汉斯

ProofsNavigator：基于引导的可解释知识推理方法

韦泽杨[1]，贾旭东[2]，陈　涛[1*]，钟甫广[1]

[1]五邑大学电子信息工程学院，广东 江门
[2]加州州立大学北岭分校计算机科学与工程学院，美国 洛杉矶

收稿日期：2024年6月2日；录用日期：2024年7月4日；发布日期：2024年7月12日

摘　要

让大规模语言模型生成推理步骤有助于构建可解释的知识推理系统。现有的知识推理方法可能生成不可靠并且与目标无关的推理步骤。为解决这一问题，该文提出一种基于引导的逐步推理方法ProofsNavigator。首先，使用通过Beam搜索生成多个候选推理步骤；然后，通过分别对候选推理步骤的有效性以及跟假设的相关性进行验证，挑选高质量的推理步骤；最后，将所选择的推理结论加入知识集中以进行下一轮循环。实验结果显示，该方法在三个难度依次递增的任务上的准确率分别为40.0%、35.6%和7.1%，比先前最优对比方法分别高1.1%、2.3%和0.2%。此外，该方法在标注数据较少的情况下仍能保持较好的性能。

关键词

知识推理，证明生成，逐步推理

ProofsNavigator: A Bootstrap Base Method for Explainable Knowledge Reasoning

Zeyang Wei[1], Xudong Jia[2], Tao Chen[1*], Fuguang Zhong[1]

[1]School of Electronics and Information Engineering, Wuyi University, Jiangmen Guangdong
[2]College of Engineering and Computer Science, California State University, Los Angeles USA

Received: Jun. 2nd, 2024; accepted: Jul. 4th, 2024; published: Jul. 12th, 2024

Abstract

Generating reasoning steps with large-scale language models aids in constructing explainable

*通讯作者。

文章引用: 韦泽杨, 贾旭东, 陈涛, 钟甫广. ProofsNavigator: 基于引导的可解释知识推理方法[J]. 计算机科学与应用, 2024, 14(7): 18-26. DOI: 10.12677/csa.2024.147159

knowledge reasoning systems. Existing methods for knowledge reasoning might produce unreliable and irrelevant reasoning steps. To address this issue, this article introduces a guided, step-by-step reasoning approach named ProofsNavigator. Initially, it generates multiple candidate reasoning steps through Beam search. Then, it selects high-quality reasoning steps by validating the validity of each candidate step and its relevance to the hypothesis. Finally, the selected reasoning conclusions are added to the knowledge set for the next iteration cycle. Experimental results show that this method achieves accuracies of 40.0%, 35.6%, and 7.1% on three tasks of increasing difficulty, respectively, outperforming the previous best methods by 1.1%, 2.3%, and 0.2%. Moreover, this method maintains good performance even with less annotated data.

Keywords

Knowledge Reasoning, Proofs Generation, Stepwise Reasoning

1. 引言

赋予机器自动推理的能力是人工智能领域长期以来追求的目标之一[1]。现有的生成式语言模型在多跳推理和逻辑推理等任务上取得了较好的表现，但这些任务往往只关注最终的答案，忽视了模型的推理过程[2] [3]。为了解决这个问题，Clark 等人[4]提出一种知识推理任务的新范式(如图 1 所示)，给定自然语言形式的一个假设以及一组知识集，要求模型为假设提供推理步骤，从而解释模型如何从知识集的子集中通过推理生成假设。推理步骤被组织为树状结构，其中每一个非叶子节点都代表一个推理步骤。

已有方法可分为一步推理方法[5] [6]和逐步推理方法[7]-[10]。逐步推理方法选择相关的知识逐步结合，利用推理的组合性，使模型更容易学习并推广到长推理[8]。然而，由于自然语言固有的模糊性，每个推理步骤的搜索空间比简单的合成任务要大得多，模型需要对新的例子进行大量的组合泛化[11]，这是现有最先进的大型语言模型都难以胜任的能力[12]。

为了应对这一挑战，本文提出一种逐步推理方法 ProofsNavigator。首先，给定目标假设以及一组已知的知识集，ProofsNavigator 使用 Entailer 模块通过 Beam 搜索生成多个候选推理步骤；然后，ProofsNavigator 通过 Verifier 和 Navigator 模块分别对候选推理步骤的有效性以及跟假设的相关性进行验证，挑选高质量的推理步骤；最后，将所选择的推理结论加入知识集中以进行下一轮循环。通过这种迭代的方法，ProofsNavigator 逐步生成完整的推理树。该方法能指引模型学习在庞大的搜索空间之中朝着假设的方向进行推理，最终只生成与假设相关的推理步骤，避免了模型在推理过程中陷入无效或不相关的推理。通过 ProofsNavigator 的引导，模型能够更加精准地判断推理的方向，并以此生成更加精确和有针对性的推理步骤，从而提高了推理的效率和准确性。

在 EntailmentBank 数据集上的实验结果显示，ProofsNavigator 在三个难度依次递增的任务分别达到了 40%、35.6%和 7.1%的准确率，比之前最好的对比方法分别高出 1.1%、2.3%和 0.2%。此外，在仅使用 10%的训练数据与 21%的模型参数的情况下达到了 26.5%的准确率，比参数量达 110 亿的大型模型高 0.9%，可见该方法在标注数据稀缺的情况下仍有较好的表现。

本文主要贡献总结如下：

1) 本文提出了 ProofsNavigator，一种逐步知识推理方法，该方法可以指引模型学习在庞大的搜索空

间之中朝着假设的方向进行推理。实验结果显示 ProofsNavigator 的性能优于目前最优的对比模型。

2) ProofsNavigator 在小样本数据上表现优秀，在数据有限的情况下表现良好，仅使用 10%训练数据与 21%的模型参数，性能超过了参数量达 110 亿大模型。

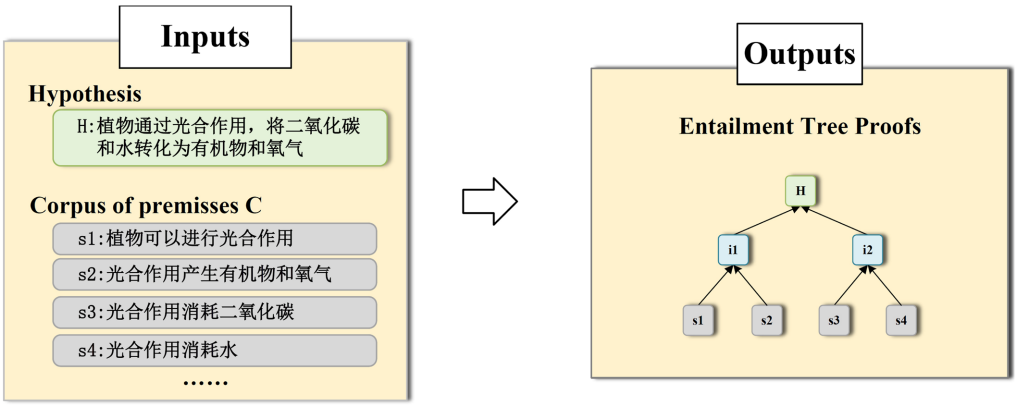

Figure 1. A Example of generating an entailment tree based on assumptions and knowledge set
图 1. 根据假设和知识集生成一颗推理树的例子

2. 方法

本文提出的基于 ProofsNavigator 的文本知识推理方法的框架图如图 2 所示，从图中可以看出，给定假设 H 和知识集 C，ProofsNavigator 在每次迭代中，首先基于一个状态 $State_i$，使用知识推理模块 Entailer 执行单步前向推理，并通过 Beam 搜索方法生成 n 条中间知识 i，其中每个结论都对应着一个推理步骤。其次，使用推理验证模块 Verifier 来验证推理步骤的有效性以及通过推理引导模块 Navigator 检验当前推理的方向。最后，将推理过程有效且推理方向正确的中间结论添加进知识集 C 中，得到新的状态 $State_{i+1}$，依次进入下一轮的迭代推理。

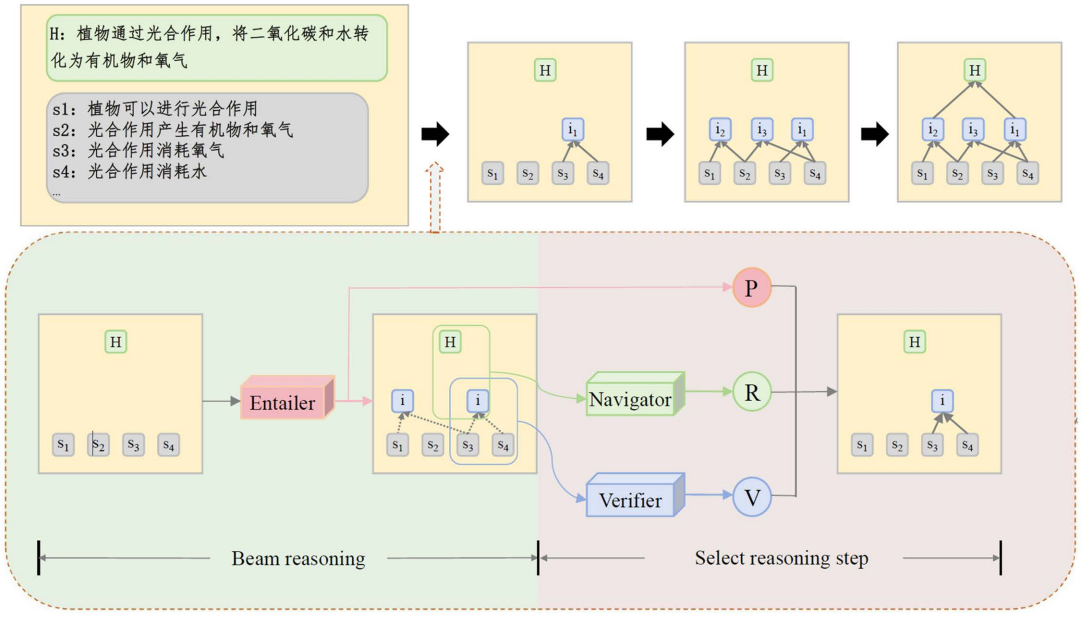

Figure 2. Framework diagram of a bootstrap interpretable knowledge reasoning method proposed in this article
图 2. 本文提出的一种基于引导的可解释知识推理方法的框架图

2.1. 任务定义

如图 1 所示，该任务的输入包括假设 H 以及一系列知识的集合 $C=\{s_1,s_2,s_3,\cdots,s_n\}$，$H$ 和 s_i 是自然文本形式的知识。根据任务难度的不同，s_i 可以分为与 H 相关的知识和与 H 不相关的知识。该任务期望模型的输出是以蕴涵树 T 的形式所组织而成的推理过程，该推理过程可以表明模型如何根据 C 中的知识推理得到 H。T 的根节点是 H，叶子节点 s_i 是模型在 C 中选择的相关知识，中间节点 i 是模型在推理过程中生成的新知识。

每个非叶子结点都代表着一个推理步骤 *Step*，该节点由其孩子节点推理而得。如果模型所输出的推理树的根节点是 H 且所有推理步骤都是正确的，则 T 被认为是有效的。本文将数据集中人工注释的蕴涵树标记为 T_{gold}，将其叶子节点集表示为 C_{gold}。

2.2. 算法模型

2.2.1. 知识推理模块

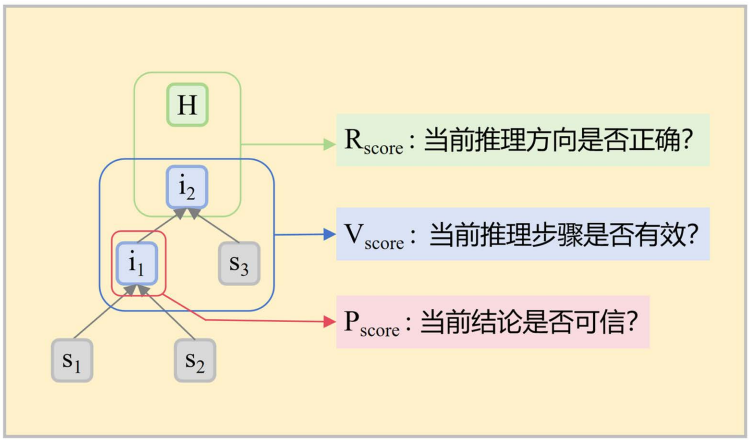

Figure 3. Three inspection mechanisms and their representative meanings
图 3. 三种检验机制及其代表的含义

在本文中，我们提出了 Entailer 是一种序列到序列的生成式模型，负责根据给定的假设和知识集合生成新的知识。模型将假设、知识集以及迭代过程生成的中间结论串联作为输入，输出则是新的中间结论，与输入的知识一起构成新的推理步骤。

Entailer 是一个序列到序列的生成模型，其输入是线性化的状态 $State_i$，输出是中间结论以及代表该其置信的分数 P_{score}(如图 3 所示)。T 中的任意一个非叶结点都代表着一个推理步骤，每个推理步骤都可以通过以下形式抽取出对应的训练数据(以图 1 为例)：首先，从非叶子节点的孩子节点推理得到该节点的过程，例如(i_1, i_2)→H、(s_1, s_2)→i_1 和(s_3, s_4)→i_2。其次，在此基础上，通过将前提中的节点替换成对应的孩子，得到新的训练数据。例如将(i_1, i_2)→H 可分别替换 i_1、i_2 得到(s_1, s_2, i_2)→H、(i_1, s_3, s_4)→H 和(s_1, s_2, s_3, s_4)→H 等。

在推理过程中，Entailer 可能会生成语法上错误的推理步骤。以图 1 作为示例，首先，模型可能会超出所提供的知识集，使用模型内部的知识进行推理，例如(s_{new}, s_1)→i_3。其次，模型的推理步骤可能不完整，例如(s_1)→i_1。此外，模型可能仅通过复制前提作为结论，而非进行推理。例如(s_i, i_2)→s_1。本文通过 Beam 搜索在每次迭代中生成多个推理步骤，然后筛选掉有明显错误的推理步骤，如语法错误或格式错误等。

2.2.2. 推理验证模块

尽管本文根据一定的规则筛选掉了部分明显错误的推理步骤，但是模型的输出仍存在着无效或者低质量的推理步骤。为了进一步提高推理步骤的有效性和质量，参考 NLProofS [9]的设置，本文通过微调预训练语言模型来实现推理验证模块 Verifier，该模块输出一个处于[0, 1]之间的分数 V_{score}(如图 3 所示)，代表着推理步骤的有效性。

2.2.3. 推理引导模块

随着输入知识数量的增加，整个推理搜索空间将迅速增长，并且树中也会出现复杂的分支。为了确保每一步推理都逻辑正确且与假设紧密相关，本文引入了推理引导模块 Navigator。该模块的主要作用是评估模型推理过程的中间结论与假设的相关程度，实现在庞大的搜索空间中指引模型的推理方向，确保模型的推理方向与假设保持一致。

本文通过微调预训练模型来实现 Navigator。模型以(中间结论，假设)的句子对作为输入，输出一个属于[0, 1]之间的分数 R_{score}(如图 3 所示)，从而指示中间结论与 H 的相似程度，即代表着模型推理的方向。训练数据的构建方法如下：设 T_{gold} 是人工标注正确答案的推理树，当前节点 *node* 的深度为 d_{node}，*node* 的祖先节点的深度记作 d_{ance}。本文计算该节点与其祖先结点(包括根节点)的深度差 D_{Δ}：

$$D_{\Delta} = d_{node} - d_{ance} \tag{1}$$

然后采用 D_{Δ} 的倒数作为数据的相关性分数 R_{score}(如图 3 所示)：

$$R_{\text{score}} = \frac{1}{D_{\Delta}} \tag{2}$$

其中，R_{score} 反映了当前节点与其祖先节点(或假设)的相关性，它与两个节点间的距离呈现单调非递增的关系。

3. 实验设置

3.1. 数据集

Table 1. EntailmentBank datasets
表 1. EntailmentBank 数据集

Split	训练集	开发集	测试集	总计
推理树数量	1131	187	340	1840
单树最大推理步骤	17	15	11	17
总推理步数	4175	597	1109	5881

为了评估 Navigator 的证明生成能力，本文参考 NLProofS [9]的设置在 EntailmentBank[1] 数据集上进行实验。EntailmentBank 是第一个支持以推理树形式进行知识推理的复杂数据集，由专家注释构建而得，更具挑战性。如表 1 所示，EntailmentBank 包含有 1,840 棵蕴涵树，其中 1,313 棵用于训练，187 棵用于验证，340 棵用于测试，每棵树都对应于 ARC 数据集的一个问题，平均包含 7.6 个节点和 3.2 个推理步骤。根据 C 范围的不同，每棵树可分为三个越来越困难的任务：

Task1：$C = C_{gold}$

Task2：$C = C_{gold}$ + 15~20 条干扰知识

Task3：$C = C_{corpus}$

[1]https://github.com/allenai/entailment_bank.

在 Task1 中，*C* 没有任何干扰知识，即 *C* 恰好由基本知识(即黄金树的叶子结点)组成。在 Task2 中，*C* 由基本知识以及干扰知识组成，始终保持为 25 条句子。在 Task3 中，*C* 是一个源自 WorldTree V2 的大型语料库，拥有 12,000 条知识。本文对这三个 Task 都进行评估。本文的方法直接适用于任务 1 和任务 2。对于任务 3，由于模型输入长度的限制，无法将整个语料库作为模型输入。为了解决这个问题，EntailmentWriter 方法为每个假设检索 25 个相关的知识，本文使用与 EntailmentWriter 方法同样的知识集进行实验，并使用 Task2 中训练得到模型评估其在 Task3 上的性能。

3.2. 评价指标

本文使用 EntailmentBank 开发的官方指标对 EntailmentBank 进行评估。该算法先通过 Jaccard 相似度将预测树 T_{pred} 中的节点与黄金树 T_{gold} 的节点进行对齐，然后通过以下几个指标对预测树的质量进行评估：叶子结点、推理步骤和中间结论。

1) Leaves：通过计算叶子节点 s_{pred} 和 s_{gold} 之间的 F1 分数从而评估 T_{pre} 是否使用了正确的叶子节点。

2) Steps：通过比较推理步骤 $step_{pre}$ 和 $step_{gold}$ 之间的 F1 分数从而评估蕴涵树结构中的推理步骤是否正确。

3) Intermediates：通过对比中间结论 i_{pred} 和 i_{gold} 间的 F1 分数从而评估模型生成的中间结论的准确性。如果 BLEURT-Large-512 分数(衡量模型生成的中间结论 i_{pre} 与相应的黄金标准 i_{gold} 之间的一致性)超过阈值 0.281，中间结论被认为是正确的。

对于上述三个指标，如果 F1 得分等于 1，则对应的 AllCorrect 得分被赋予 1；否则，AllCorrect 得分为 0。当且仅当所有叶子、步骤和中间结论都正确时，Overall-AllCorrect 指标得分为 1。这是一个严格的指标，因为 T_{pred} 结果中的任何错误都将导致 Overall-AllCorrect 得分为 0。

3.3. 实验环境

本文通过微调 flan-T5-large 预训练模型[13] (7.83 亿参数)来实现 Entailer 模块。Verifier 模块和 Navigator 模块通过微调预训练的 RoBERTa 模型[14] (3.55 亿)来实现。

对于 Task1 和 Task2，本文为不同的实验分别训练了单独的 Entailer、Verifier 和 Navigator 模型块。而对于 Task3，本文重新使用 Task2 中训练好的模型，无需额外训练。

P_{score}、V_{score} 和 R_{score} 均采取 0.4:0.3:0.3 的比例进行加权求和。

ProofsNavigator 的所有实验均在配备 2 个 NVIDIA GeForce RTX 3090 (24G 显存)，128G 内存的机器上进行。本文使用 AdamW 优化模型作为优化器，学习率从 0 线性升温到最大值，然后按照 cosine schedule 衰减。不同 Task 的超参数分别根据验证数据进行调整。最优的模型根据验证集上最佳的“Overall-All-Correct”指标所选择。

4. 实验结果与分析

4.1. 知识推理实验

如表 2 所示，对于所有三项 Task，ProofsNavigator 在最严格的 Overall-AllCorrect 指标上均取得优秀的成绩。以 Task2 为例，首先 ProofsNavigator 总体上生成了更正确的推理，将 Overall-AllCorrect 从 20.9% 提升至 35.6%。其次，ProofsNavigator 更忠实于推理步骤进行推理，而不是借助外部知识，将 Steps-AllCorrect 从 22.9%提升至 37.7%。第三，ProofsNavigator 能生成更正确的中间结论，将 Intermediates-AllCorrect 从 28.5%提升至 38.5%。

与拥有 110 亿参数的 EntailmentWriter 模型相比，ProofsNavigator 仅使用 21%的模型参数便在 Task1、Task2、Task3 的 Overall-AllCorrect 指标上以绝对值高出 4.7%、10.0%、4.2%，这表明了 ProofsNavigator

具有高效的性能以及优秀的泛化能力，能够在较少的参数限制下，更好地捕捉和理解数据中的模式，在多个 Task 上实现优异的表现。

本文注意到所有方法在 Task3 的表现上都不佳，主要原因是官方数据集中所检索的知识可能不包含 C_{gold} 所需的所有知识(68%的情况)。

Table 2. Knowledge reasoning experiment results
表 2. 知识推理实验结果

Task	Method	Leaves		Steps		Intermediates		Overall
		F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
Task1	EntailmentWriter	98.7	86.2	50.5	37.7	67.6	36.2	33.5
	EntailmentWriter (11B)	99.0	89.4	51.5	38.2	71.2	38.5	35.3
	NLProofS	97.8	90.1	55.6	42.3	72.4	40.6	38.9
	ProofsNavigator (ours)	98.1	90.3	57.7	43.24	73.2	42.1	40.0
Task2	EntailmentWriter	84.3	35.6	35.5	22.9	61.8	28.5	20.9
	EntailmentWriter (11B)	89.1	48.8	41.4	27.7	66.2	31.5	25.6
	NLProofS	90.3	58.8	47.2	34.4	70.2	37.8	33.3
	ProofsNavigator (ours)	90.0	58.2	49.3	37.7	69.9	38.5	35.6
Task3	EntailmentWriter	35.7	2.9	6.1	2.4	33.4	7.7	2.4
	EntailmentWriter (11B)	39.9	3.8	7.4	2.9	35.9	7.1	2.9
	NLProofS	43.2	8.2	11.2	6.9	42.9	17.3	6.9
	ProofsNavigator (ours)	43.0	9.1	12.6	7.1	41.9	18.2	7.1

4.2. 消融实验

Table 3. Ablation experiment results
表 3. 消融实验结果

Index	Method	Leaves		Steps		Intermediates		Overall
		F1	AllCorrect	F1	AllCorrect	F1	AllCorrect	AllCorrect
(a)	full model	90.0	58.2	49.3	37.7	69.9	38.5	35.6
(b)	w/o Entailer score	88.4	52.4	43.0	32.1	79.4	42.7	32.1
(c)	w/o Verifier score	89.4	56.2	47.8	36.2	68.4	37.4	34.1
(d)	w/o Navigator score	90.5	58.2	48.4	36.2	69.4	36.8	34.1
(e)	flan-t5 -> t5	90.0	57.4	48.3	35.6	70.2	38.8	35.0

ProofsNavigator 需要 Entailer score (P_{score})、Verifier score (V_{score})和 Navigator score (R_{score})三个组件共同发挥作用。为了研究每个模块在知识推理结果所发挥的作用，本文在 EntailmentBank 数据集的 Task2 任务上进行消融实验，实验结果如表 3 所示。从中可以看出：Navigator 所提供精准的相关性分数有助于模型朝着正确的方向进行推理。比较(a)和(d)，当屏蔽 Navigator 模块时，模型在 Steps 和 Intermediates 这两个与模型推理方向相关的指标上表现均有所下降，这代表 Navigator 对推理方向的选择发挥着重要作用。另外，更强的生成模型可以实现更高的生成性能(比较(a)和(e))，这意味着 ProofsNavigator 方法可以通过

使用更强的生成模型进一步改进。此外，在使用同样的 T5 模型下，ProofsNavigator 方法的性能同样超过了先前的最优对比方法。

4.3. 数据稀缺性实验

Table 4. Experimental results in a data-scarce environment
表 4. 数据稀缺环境下的实验结果

方法	训练数据采样比例				
	100%	50%	20%	10%	1%
EntialmentWriter	20.9	20.5	18.0	12.9	8.0
MetGen	28.0	24.1	22.0	19.1	14.7
NLProofS	33.3	31.5	25.0	23.0	16.8
Ours	35.6	31.8	27.4	26.5	20.0

人工标注的数据是非常昂贵的。为了研究模型在少量标注数据情况下的表现，本文在不同的采样比例下随机读取 EntailmentBank 的训练集数据，然后在 Task2 上进行训练。而挑选超参数以及测试模型性能则在完整的开发集和测试集上运行，实验结果如表 4 所示。

首先，在训练数据数量相同，采样比例依次为 100%、50%、20%、10%和 1%和情况下，ProofsNavigator 分别实现 35.6%、31.8%、27.4%、25.3%和 20.0%的准确率，比对比方法分别高出 2.3%、0.3%、2.4%、2.3%和 3.2%。这说明 ProofsNavigator 具有较强的鲁棒性和泛化能力，能够在面对数据采样比例降低的情况下仍能保持准确率。

此外，在仅使用 10%训练数据的情况下，ProofsNavigator 实现了 26.5%的准确率，比参数量达 110 亿的大模型高出 0.9%。这说明 ProofsNavigator 具有较强的特征提取能力，能够在训练数据稀缺的场景下有效地利用有限的标注数据，从少量数据中学习到足够的信息，实现了性能上的提升。

5. 结论

本文提出一种基于引导的可解释知识推理方法 ProofsNavigator，该方法通过 Navigator 模块引导模型朝着假设的方向进行推理。在 EntailmentBank 数据集上的三个任务中的性能均超过了先前的最优对比模型，此外该方法在小规模数据亦有较好的性能。

参考文献

[1] Newell, A. and Simon, H. (1956) The Logic Theory Machine—A Complex Information Processing System. *IEEE Transactions on Information Theory*, **2**, 61-79. https://doi.org/10.1109/tit.1956.1056797

[2] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., *et al*. (2018) HotpotQA: A Dataset for Diverse, Explainable Multi-Hop Question Answering. *Proceedings of the* 2018 *Conference on Empirical Methods in Natural Language Processing*, Brussels, 31 October-4 November 2018, 2369-2380. https://doi.org/10.18653/v1/d18-1259

[3] Dua, D., Wang, Y., Dasigi, P., *et al*. (2019) DROP: A Reading Comprehension Benchmark Requiring Discrete Reasoning over Paragraphs. *Proceedings of the* 2019 *Conference of the North American Chapter of the Association for Computational Linguistics*: *Human Language Technologies*, *Volume* 1 (*Long and Short Papers*), Minneapolis, 2-7 June 2019, 2368-2378. https://doi.org/10.18653/v1/N19-1246

[4] Dalvi, B., Jansen, P., Tafjord, O., Xie, Z., Smith, H., Pipatanangkura, L., *et al*. (2021) Explaining Answers with Entailment Trees. *Proceedings of the* 2021 *Conference on Empirical Methods in Natural Language Processing*, Punta Cana, 7-11 November 2021, 7358-7370. https://doi.org/10.18653/v1/2021.emnlp-main.585

[5] Saha, S., Ghosh, S., Srivastava, S. and Bansal, M. (2020) PRover: Proof Generation for Interpretable Reasoning over Rules. *Proceedings of the* 2020 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Online, 16-20 November 2020, 122-136. https://doi.org/10.18653/v1/2020.emnlp-main.9

[6] Sun, C., Zhang, X., Chen, J., Gan, C., Wu, Y., Chen, J., *et al*. (2021) Probabilistic Graph Reasoning for Natural Proof Generation. *Findings of the Association for Computational Linguistics*: *ACL-IJCNLP* 2021, Online, 1-6 August 2021, 3140-3151. https://doi.org/10.18653/v1/2021.findings-acl.277

[7] Liang, Z., Bethard, S. and Surdeanu, M. (2021) Explainable Multi-Hop Verbal Reasoning through Internal Monologue. *Proceedings of the* 2021 *Conference of the North American Chapter of the Association for Computational Linguistics*: *Human Language Technologies*, Online, 6-11 June 2021, 1225-1250. https://doi.org/10.18653/v1/2021.naacl-main.97

[8] Tafjord, O., Dalvi, B. and Clark, P. (2021) ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language. *Findings of the Association for Computational Linguistics*: *ACL-IJCNLP* 2021, Online, 1-6 August 2021, 3621-3634. https://doi.org/10.18653/v1/2021.findings-acl.317

[9] Yang, K., Deng, J. and Chen, D. (2022) Generating Natural Language Proofs with Verifier-Guided Search. *Proceedings of the* 2022 *Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, 7-11 December 2022, 89-105. https://doi.org/10.18653/v1/2022.emnlp-main.7

[10] Qu, H., Cao, Y., Gao, J., Ding, L. and Xu, R. (2022) Interpretable Proof Generation via Iterative Backward Reasoning. *Proceedings of the* 2022 *Conference of the North American Chapter of the Association for Computational Linguistics*: *Human Language Technologies*, Seattle, 10-15 July 2022, 2968-2981. https://doi.org/10.18653/v1/2022.naacl-main.216

[11] Ruis, L., Andreas, J., Baroni, M., *et al*. (2020) A Benchmark for Systematic Generalization in Grounded Language Understanding. *Proceedings of the* 34*th International Conference on Neural Information Processing Systems*, Vancouver, 6-12 December 2020, 19861-19872.

[12] Rae, J.W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., *et al*. (2021) Scaling Language Models: Methods, Analysis & Insights from Training Gopher. arXiv: 2112.11446. https://doi.org/10.48550/arXiv.2112.11446

[13] Chung, H.W., Hou, L., Longpre, S., *et al*. (2022) Scaling Instruction-Finetuned Language Models. arXiv: 2210.11416. https://doi.org/10.48550/arXiv.2210.11416

[14] Liu, Y., Ott, M., Goyal, N., *et al*. (2019) RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv: 1907.11692. https://doi.org/10.48550/arXiv.1907.11692