



中国科学院大学
University of Chinese Academy of Sciences

硕士学位论文

基于大语言模型的自然语言到谓词逻辑翻译方法研究

作者姓名：黄宇威

指导教师：王石 正高级工程师

中国科学院计算技术研究所

学位类别：电子信息硕士

学科专业：计算机技术

培养单位：中国科学院计算技术研究所

2026 年 6 月

A Study of Natural Language to First Order Logic Translation
Based on Large Language Models

A thesis submitted to
University of Chinese Academy of Sciences
in partial fulfillment of the requirement
for the degree of
Master of Electronic and Information Engineering
in Computer Technology

By

Yuwei Huang

Supervisor: Professor Shi Wang

Institute of Computing Technology, Chinese Academy of Sciences

June, 2026

中国科学院大学 学位论文原创性声明

本人郑重声明：所呈交的学位论文是本人在导师的指导下独立进行研究工作所取得的成果。承诺除文中已经注明引用的内容外，本论文不包含任何其他个人或集体享有著作权的研究成果，未在以往任何学位申请中全部或部分提交。对本论文所涉及的研究工作做出贡献的其他个人或集体，均已在文中以明确方式标明或致谢。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日 期：

中国科学院大学 学位论文授权使用声明

本人完全了解并同意遵守中国科学院大学有关收集、保存和使用学位论文的规定，即中国科学院大学有权按照学术研究公开原则和保护知识产权的原则，保留并向国家指定或中国科学院指定机构送交学位论文的电子版和印刷版文件，且电子版与印刷版内容应完全相同，允许该论文被检索、查阅和借阅，公布本学位论文的全部或部分内容，可以采用扫描、影印、缩印等复制手段以及其他法律许可的方式保存、汇编本学位论文。

涉密及延迟公开的学位论文在解密或延迟期后适用本声明。

作者签名：

日 期：

导师签名：

日 期：

摘要

随着大语言模型在自然语言理解与生成任务中的不断发展，如何提升其在复杂逻辑场景下的语义精确建模能力，已成为当前研究中的重要问题。自然语言到一阶谓词逻辑翻译（Natural Language to First Order Logic, NL2FOL）任务旨在将自然语言中的语义内容准确映射为形式化逻辑表达，为后续自动推理、知识验证与结果解释提供可检查的表示基础。然而，现有方法在复杂逻辑场景下仍面临若干突出困难，包括显性复杂逻辑结构下的语义偏移、句级隐性逻辑关系的恢复困难，以及篇章级跨句上下文中的逻辑依赖建模不足。

围绕上述问题，本文系统研究了基于大语言模型的自然语言到一阶谓词逻辑翻译方法，并从显性复杂逻辑结构、句级隐性逻辑关系和篇章级隐性逻辑关系三个层面展开分析与方法设计。

针对显性复杂逻辑结构文本中存在的逻辑关系层次深、语义约束强、推理路径易偏移等问题，本文提出一种面向推理阶段的增强方法（Logic Structure and Entropy Guided Inference Controlling, LSEG）。该方法通过构造逻辑保持扰动样本估计模型内部的逻辑稳定区域，进一步提取分层逻辑校正方向，并结合层级熵变化识别推理过程中的不稳定状态，对相应隐状态实施自适应干预，从而缓解复杂逻辑翻译中的语义漂移问题。相关实验表明，该方法能够在不更新模型参数的条件下，提高模型在复杂逻辑翻译任务中的语义一致性与结果稳定性。

针对句级隐性逻辑关系难以直接显式建模的问题，本文提出一种基于伪代码中间表示的多阶段翻译方法。该方法首先将自然语言句子转换为更具结构性的伪代码中间表示，用于显式化原句中的条件、约束与依赖关系；随后引入中间结果验证与校正机制，以保证中间表示的结构完整性、语义一致性和逻辑可映射性；最后将校正后的伪代码映射为一阶谓词逻辑表达式。实验结果表明，该方法能够有效降低自然语言与形式逻辑之间的表示跨度，增强模型对句级隐性逻辑关系的恢复能力。

针对篇章级文本中逻辑关系跨句分布、上下文依赖较强的问题，本文进一步研究了面向篇章上下文的自然语言到一阶谓词逻辑翻译方法。通过构造篇章级隐性逻辑翻译数据，并设计相应的监督微调框架，本文考察了不同上下文输入方式对跨句逻辑恢复的影响。实验结果表明，结合更充分的篇章上下文信息，能够有效提升模型对篇章级隐性逻辑关系的整体建模能力。

综上，本文围绕自然语言到一阶谓词逻辑翻译中的关键难点，分别从推理阶段控制、中间表示显式化和篇章上下文建模三个方向提出了相应方法。研究结果表明，提升 NL2FOL 能力不能仅依赖单一步骤的直接生成，而需要结合逻辑结构保持、隐性关系显式化以及上下文整体建模等思路，才能更有效地提高逻辑翻译结果的准确性、鲁棒性与可验证性。本文的研究可为后续构建更可靠、更可解释的神经符号推理系统提供一定参考。

关键词：大语言模型；自然语言到一阶谓词逻辑翻译；逻辑结构建模；深度神经网络；可解释性；神经符号推理

Abstract

With the continuous development of large language models in natural language understanding and generation, how to improve their ability to model semantics precisely in complex logical scenarios has become an important research problem. Natural Language to First Order Logic translation (Natural Language to First Order Logic, NL2FOL) aims to accurately map semantic content in natural language into formal logical expressions, thereby providing a verifiable representational basis for subsequent automatic reasoning, knowledge verification, and result interpretation. However, existing methods still face several prominent challenges in complex logical scenarios, including semantic drift under explicit complex logical structures, difficulties in recovering sentence level implicit logical relations, and insufficient modeling of logical dependencies across sentences in discourse level contexts.

To address these issues, this thesis systematically investigates NL2FOL methods based on large language models, and develops corresponding approaches from three perspectives, namely explicit complex logical structures, sentence level implicit logical relations, and discourse level implicit logical relations.

To address the challenges in explicit complex logical structure texts, including deep hierarchical logical relations, strong semantic constraints, and unstable reasoning paths, this paper proposes an inference stage enhancement method, namely Logic Structure and Entropy Guided Inference Controlling (LSEG). The proposed method constructs logic preserving perturbation samples to estimate the logical stability regions within the model, and further extracts hierarchical logic correction directions. By leveraging layer wise entropy variations to identify unstable states during the reasoning process, the method applies adaptive interventions to the corresponding hidden states, thereby mitigating semantic drift in complex logic translation. Experimental results demonstrate that, without updating model parameters, the proposed method improves both semantic consistency and output stability in complex logic translation tasks.

For the problem that sentence level implicit logical relations are difficult to model explicitly, this thesis proposes a multi stage translation method based on pseudocode intermediate representations. The method first converts natural language sentences into more structured pseudocode representations, which are used to make explicit the conditions, constraints, and dependency relations contained in the original sentences. Then, an intermediate result verification and correction mechanism is introduced to ensure the structural completeness, semantic consistency, and logical mappability of the intermediate representations. Finally, the corrected pseudocode representations are mapped into first order logic expressions. Experimental results demonstrate that this method

effectively reduces the representational gap between natural language and formal logic, and enhances the model’s ability to recover sentence level implicit logical relations.

For discourse level texts where logical relations are distributed across sentences and strongly depend on context, this thesis further studies a context aware NL2FOL method for discourse level translation. By constructing discourse level implicit logic translation data and designing a corresponding supervised fine tuning framework, this thesis examines the influence of different contextual input settings on cross sentence logical recovery. Experimental results show that incorporating richer discourse context effectively improves the model’s overall ability to model discourse level implicit logical relations.

In summary, this thesis focuses on several key difficulties in NL2FOL and proposes corresponding methods from three directions, namely inference stage control, intermediate representation explicitization, and discourse context modeling. The results indicate that improving NL2FOL cannot rely solely on one step direct generation, but instead requires the combination of logical structure preservation, implicit relation explicitization, and holistic context modeling, so as to improve the accuracy, robustness, and verifiability of logical translation more effectively. The study in this thesis may provide useful support for future research on building more reliable and interpretable neuro symbolic reasoning systems.

Key Words: Natural Language to First Order Logic Translation; Large Language Models; Logical Structure Modeling; Implicit Logical Relations; Intermediate Representation; Neuro Symbolic Reasoning

目 录

第 1 章 绪论	1
1.1 研究背景	1
1.2 研究意义	2
1.3 主要工作	3
1.4 论文组织架构	4
第 2 章 相关工作 and 研究现状	7
2.1 引言	7
2.2 传统方法	7
2.2.1 符号主义方法	7
2.2.2 统计学习方法	8
2.2.3 早期神经网络方法	9
2.3 基于大语言模型的翻译方法	11
2.3.1 直接翻译	11
2.3.2 微调大语言模型	13
2.4 基于神经-符号解耦架构的翻译方法	15
2.4.1 双系统视角下的任务解耦	16
2.4.2 翻译与验证的主流实现范式	16
2.4.3 高级交互模式与局限性	18
2.5 本章小结	19
第 3 章 面向显性逻辑结构文本的 NL2FOL 技术	21
3.1 引言	21
3.2 方法介绍	22
3.2.1 问题定义与方法动机	22
3.2.2 LSEG 方法总体框架	24
3.2.3 逻辑校正方向构建方法	26
3.2.4 基于熵信号的自适应推理干预机制	28
3.2.5 方法分析与讨论	31
3.3 实验结果与分析	32
3.3.1 实验设置	32
3.3.2 实验结果	33
3.3.3 跨模型验证结果	34

3.3.4 消融实验结果	36
3.3.5 层级熵分析	37
3.4 本章小结	39
第 4 章 面向隐性逻辑承载句级的 NL2FOL 技术	41
4.1 引言	41
4.2 方法介绍	43
4.2.1 问题定义与研究动机	43
4.2.2 整体框架设计	44
4.2.3 伪代码中间表示构建方法	45
4.2.4 基于验证的中间结果校正机制	47
4.2.5 从伪代码到一阶谓词逻辑的映射方法	48
4.2.6 方法分析与讨论	50
4.3 实验结果与分析	51
4.3.1 实验数据	51
4.3.2 实验设置	52
4.3.3 整体实验结果	53
4.3.4 消融实验	54
4.3.5 案例分析	55
4.3.6 方法讨论	57
4.4 本章小结	57
第 5 章 面向隐性逻辑承载篇章级的 NL2FOL 技术	59
5.1 引言	59
5.2 方法介绍	60
5.2.1 问题定义与研究动机	60
5.2.2 篇章级隐性逻辑数据构造方法	61
5.2.3 监督微调框架设计	63
5.2.4 训练目标与实现细节	63
5.2.5 方法分析与讨论	64
5.3 实验结果与分析	65
5.3.1 实验设置	65
5.3.2 整体实验结果	65
5.3.3 案例分析	67
5.3.4 方法讨论	67
5.4 本章小结	69

第 6 章 总结与展望	71
6.1 全文总结	71
6.2 未来展望	72
参考文献	75
致谢	79
作者简历及攻读学位期间发表的学术论文与其他相关学术成果 ..	81

图目录

图 1-1	NL2FOL 任务示意图	1
图 1-2	论文组织架构图	4
图 2-1	CCG 方法示意图 ^[1]	9
图 2-2	Seq2Tree 模型示意图 ^[2]	10
图 2-3	LogicLLaMA 模型 ^[3]	14
图 2-4	TALMs 方法示意图 ^[4]	17
图 3-1	显性逻辑结构中的典型语义偏移示例	23
图 3-2	LSEG 方法整体框架	25
图 3-3	FOLIO 与 MALLS 上不同层区间的 LSEG 干预效果	38
图 3-4	模型各层平均熵分布	38
图 4-1	基于伪代码中间表示的句级隐性逻辑翻译框架	42
图 4-2	消融实验结果	55
图 5-1	基于监督微调的篇章级隐性逻辑翻译方法框架	60

表目录

表 3-1	LSEG 与现有方法在 FOLIO 和 MALLS 上的逻辑等价性结果	34
表 3-2	LSEG 在 LogicLLaMA 上的跨模型验证结果	35
表 3-3	熵引导自适应调度机制的消融结果	36
表 3-4	不同层区间上的 LSEG 干预效果	37
表 4-1	不同模型上的句级隐性逻辑翻译结果	54
表 4-2	句级隐性逻辑翻译案例分析	56
表 5-1	本章方法在不同训练设置下的整体实验结果	66
表 5-2	与主流模型的对比结果	67
表 5-3	篇章级隐性逻辑翻译案例分析	68

第 1 章 绪论

1.1 研究背景

近年来，以大语言模型 (Large Language Models, LLM) 为代表的人工智能技术 (Artificial Intelligence, AI) 发展迅速，推动了自然语言处理 (Natural Language Processing, NLP) 领域的持续发展^[5]。以 DeepSeek 系列^[6,7]、Qwen 系列^[8,9] 和 GPT 系列^[10] 为代表的模型，通过在大规模数据上进行预训练，在文本生成、知识理解和复杂任务处理等方面表现出较强能力。

但从现有研究和应用情况来看，大语言模型在处理需要多步推理的复杂逻辑任务时仍存在明显不足^[11]。逻辑推理本质上是一个需要严格证据评估、关系建构和逻辑推导的认知过程^[12]。这一过程对语义表达的准确性和结构的严密性都有较高要求，而自然语言本身具有一定的灵活性，往往难以直接支撑如此精确的逻辑计算。因此，在处理逻辑推理问题时，采用一阶谓词逻辑 (First Order Logic, FOL)，通常比完全依赖自然语言的推理过程更稳定，也更便于验证^[13]。

要利用 FOL 提升大语言模型的推理能力，一个关键前提是将自然语言准确翻译为 FOL 表达式，即自然语言到一阶谓词逻辑翻译 (Natural Language to First-Order Logic, NL2FOL)。然而，这一过程对准确性要求很高，哪怕只是局部翻译出现偏差，也可能破坏整个逻辑表达式的语义正确性，并进一步影响后续推理结果^[14]。因此，如何保证 NL2FOL 的准确性，成为利用符号逻辑增强大语言模型推理能力时必须解决的基础问题。如图 1-1 所示，NL2FOL 任务旨在将自然语言准确映射为对应的一阶谓词逻辑表达式。

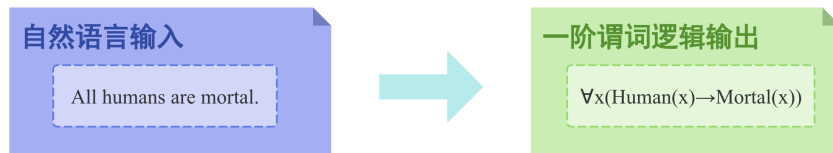


图 1-1 NL2FOL 任务示意图

Figure 1-1 Illustration of the NL2FOL Task

目前，已经有一些研究尝试直接利用大语言模型进行 NL2FOL 翻译^[15,16]。这些方法在逻辑结构较为简单的文本上取得了一定效果，但当输入文本包含更复杂的逻辑关系时，模型的翻译质量往往明显下降^[17]。这表明，现有方法在复杂逻辑场景下仍然存在较大的改进空间，也说明 NL2FOL 仍然是一个值得进一步研究的问题。

此外，这一问题也与大语言模型的可解释性密切相关。当前大语言模型的决策过程通常缺乏透明性，这种“黑箱”特征在高风险应用场景中会带来明显限制^[18]。如果能够将自然语言输入转化为形式化的逻辑表达式，就可以为模型的

推理过程提供一条更清晰、可检查的分析路径。例如，当模型给出某个结论时，研究者或用户可以结合相应的逻辑表达式回溯其推导过程，分析结论是否由前提合理推出。相比仅依赖自然语言生成的解释，这种基于逻辑表示的分析方式更明确，也更便于验证。因此，研究 NL2FOL 的转化方法，不仅关系到大语言模型逻辑推理能力的提升，也对增强模型的可解释性和可靠性具有重要意义。

1.2 研究意义

自然语言到一阶谓词逻辑翻译问题具有较为明确的理论意义。尽管近年来大语言模型在多类任务中表现出较强能力，但其推理过程仍然主要建立在统计生成基础之上，在涉及复杂逻辑结构、隐含语义关系和严格形式约束的场景中，仍然容易出现语义偏移和逻辑不一致等问题^[19]。因此，研究如何将自然语言更准确地转化为形式化逻辑表达，不仅有助于进一步理解大语言模型在逻辑推理任务中的能力边界，也有助于推动自然语言理解从表层生成走向更严格的语义建模。

从理论层面看，NL2FOL 研究的一个重要价值在于，它为连接自然语言处理与符号推理提供了具体路径。自然语言强调表达的灵活性和上下文适应性，而形式逻辑强调语义的明确性和推导的可验证性，两者在表示方式和处理机制上存在明显差异。如何在二者之间建立较为稳定的映射关系，本身就是一个具有挑战性的研究问题。围绕这一问题展开研究，有助于深化对自然语言中逻辑结构、语义组合方式以及形式化表达机制的认识，也有助于为神经符号融合研究提供更扎实的任务基础^[20,21]。

此外，针对自然语言中不同类型的逻辑承载方式开展研究，也具有方法层面的意义。在真实文本中，逻辑信息并不总是以显式、规则的方式出现。有些逻辑关系直接体现在句子的表层结构中，有些则隐含在词语组合、语义指向或上下文关系之中。这意味着，NL2FOL 不是单一条件下的简单翻译问题，而是涉及显性逻辑结构、句级隐性逻辑以及篇章级语义依赖等多种情形的复杂任务。对这些不同情形进行区分和研究，有助于更细致地刻画任务难点，也有助于推动面向复杂语义场景的逻辑翻译方法发展。

除了理论意义之外，NL2FOL 研究也具有较强的应用意义。形式逻辑表达能够为自动推理、知识验证、规则执行和事实分析提供统一且可检查的表示基础。如果自然语言能够被稳定地转化为逻辑表达式，那么原本依赖自然语言直接处理的复杂推理过程，就可以在更明确的形式框架中展开。这对于提升推理系统的正确性、稳定性和可控性具有现实价值^[13,15]。尤其在需要较高可靠性的场景中，逻辑表示不仅能够帮助系统进行后续演绎和验证，也能够为错误定位和结果分析提供依据。

与此同时，这项研究对于提升大语言模型的可解释性和可信性也具有实际意义。当前大语言模型的内部决策过程通常不透明，用户往往只能看到输入与输出，而难以判断中间推理是否合理^[18]。将自然语言转化为形式化逻辑表达后，可

以在一定程度上把模型处理问题的结果放到一个更清晰的结构中进行分析,使结论与前提之间的关系更容易被检查。这种形式化表示虽然不能完全揭示模型内部机制,但能够为模型输出提供一种更明确的外部分析方式,从而增强系统结果的可理解性和可验证性。

进一步看,随着大语言模型逐步进入问答、辅助决策、专业分析和智能交互等更复杂的应用场景,系统对逻辑一致性和结果可靠性的要求也在不断提高。在此背景下,研究自然语言到一阶谓词逻辑翻译,不仅有助于改进当前模型在复杂逻辑任务中的表现,也有助于为后续构建更加可靠的智能推理系统提供支持。因此,无论从自然语言理解与符号逻辑结合的研究发展来看,还是从高可靠人工智能系统的实际需求来看,围绕 NL2FOL 展开系统研究都具有较为突出的理论价值和现实意义。

1.3 主要工作

本文围绕基于大语言模型的自然语言到一阶谓词逻辑翻译方法展开研究,重点关注当前方法在复杂逻辑场景下面临的准确性不足、隐含逻辑关系识别困难以及篇章级上下文建模不足等问题。针对这些问题,本文从显性复杂逻辑结构、句级隐性逻辑和篇章级隐性逻辑三个层面开展研究,力求提升大语言模型在不同类型文本上的逻辑翻译能力,并增强翻译结果的稳定性与可验证性。

首先,针对显性复杂逻辑结构文本翻译过程中容易出现语义偏移和逻辑错误的问题,提出了一种训练无关的推理阶段校正方法。该方法主要面向包含多重量词、否定、条件、蕴含以及嵌套结构的复杂文本,研究如何在不更新模型参数的条件下,通过对推理阶段内部表示的调节,减轻复杂逻辑结构下的表示偏移问题,从而提高模型在复杂逻辑翻译任务中的语义一致性和翻译稳定性。

其次,针对句级文本中隐性逻辑关系难以直接识别和建模的问题,研究了一种基于中间表示的自然语言到一阶谓词逻辑翻译方法。对于表层缺乏显式逻辑连接词、但内部蕴含清晰逻辑关系的文本,直接从自然语言生成逻辑表达式往往容易出现关系遗漏和结构错配。为此,本文引入更便于表达和检查的中间表示,将自然语言中的隐含逻辑关系先转化为结构更清晰的中间形式,再进一步映射为一阶谓词逻辑表达式,并结合相应的验证机制提高翻译结果的准确性和可控性。

最后,针对篇章级文本中逻辑关系跨句分布、上下文依赖较强的问题,研究了面向篇章上下文的自然语言到一阶谓词逻辑翻译方法。现有研究大多集中于句子级翻译任务,而在真实文本中,逻辑关系往往分布于多个句子之间,需要结合上下文语义进行整体理解。针对这一问题,本文面向篇章级场景,研究能够结合上下文信息进行整体建模的逻辑翻译方法,以提升模型对跨句逻辑关系和篇章层面隐含语义的处理能力。

综上,本文围绕自然语言到一阶谓词逻辑翻译任务中的关键问题,分别从显性复杂逻辑结构、句级隐性逻辑和篇章级隐性逻辑三个方面展开研究,希望为提

升大语言模型在逻辑翻译任务中的准确性、鲁棒性和可验证性提供一定参考。

1.4 论文组织架构

本文共分为六章，本章为绪论，旨在介绍本文的研究背景及研究意义、主要工作和文章组织架构，如图 1-2所示。

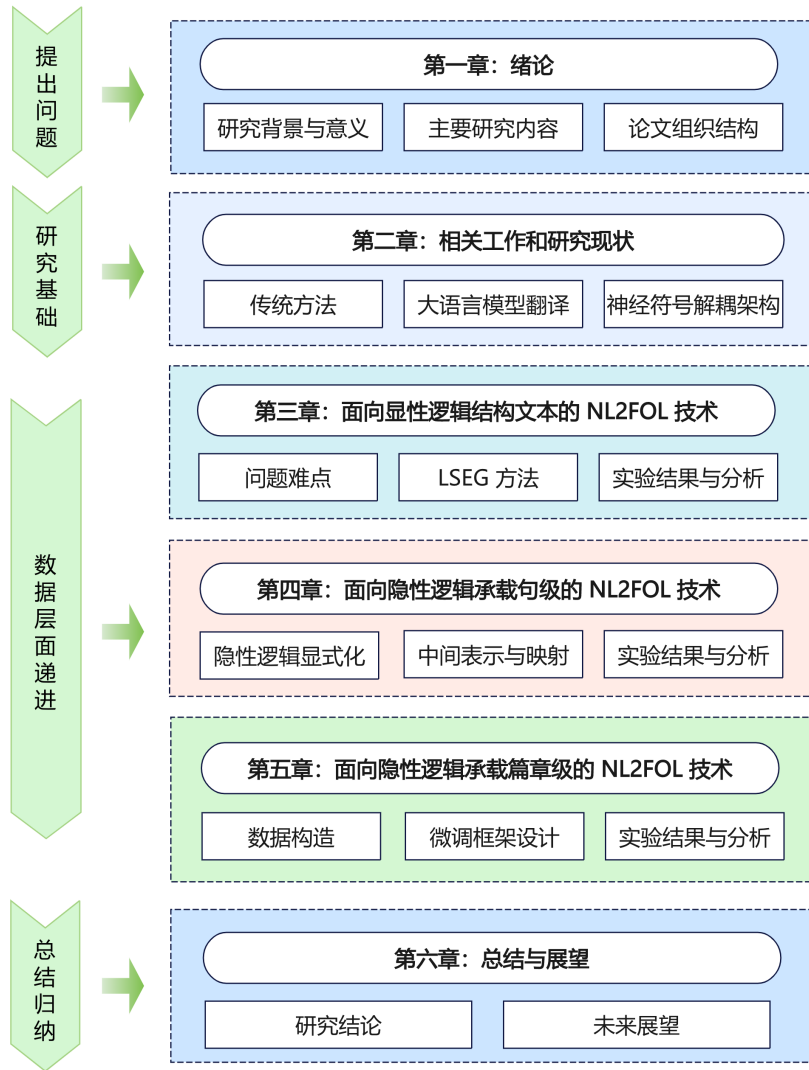


图 1-2 论文组织架构图

Figure 1-2 Organizational Structure of the Thesis

后续章节内容如下：第二章为相关工作，主要梳理自然语言到逻辑表达的相关研究进展、大语言模型推理能力研究现状，以及神经符号推理等与本文相关的研究基础，为后续章节的展开提供背景支撑。第三章针对显性复杂逻辑结构文本的翻译问题展开研究，重点分析大语言模型在复杂逻辑结构下出现语义偏移和逻辑错误的原因，并介绍相应的推理阶段增强方法及实验结果。第四章针对句级文本中的隐性逻辑关系翻译问题展开研究，介绍基于中间表示的自然语言到一

阶谓词逻辑翻译方法，并对其有效性进行实验分析。第五章针对篇章级文本中的隐性逻辑关系翻译问题展开研究，介绍结合上下文信息的自然语言到一阶谓词逻辑翻译方法，并分析该方法在篇章级场景下的表现。第六章为总结与展望，对全文工作进行总结，归纳本文的主要研究结论，并对未来值得进一步研究的问题进行展望。

第2章 相关工作和研究现状

2.1 引言

对于 NL2FOL 任务而言，不同阶段的方法在建模能力、泛化性能和结果可靠性方面各有特点。早期方法强调语言规则与逻辑规则之间的精确对应，具有较强的可解释性，但适用范围有限；数据驱动方法提升了模型对复杂语言现象的适应能力，但通常依赖较多标注数据；近年来的大语言模型方法增强了开放场景下的迁移能力和推理潜力，但在复杂逻辑结构、隐含逻辑关系以及语义保真度方面仍然存在明显不足^[11,14,19,22]。与此同时，神经符号方法试图结合大语言模型的语义理解能力与符号系统的可验证推理能力，为提升逻辑翻译的准确性和可信性提供新的思路^[13,15,20]。

近年来，围绕 NL2FOL 的研究开始逐渐增多。已有工作尝试直接利用大语言模型完成自然语言到一阶谓词逻辑翻译，并通过数据构造、任务适配和中间表示设计等方式提升翻译效果^[3,16,17,23]。与此同时，相关研究也开始关注逻辑翻译过程中的可验证性、组合泛化能力以及翻译结果与原始语义之间的等价关系。这些进展表明，NL2FOL 已经不再只是传统意义上的语义解析问题，而逐步发展为大语言模型推理、形式化语义表示与神经符号推理交叉背景下的重要研究方向。

基于上述背景，本章将围绕与本文密切相关的研究展开综述。首先介绍基于传统范式的自然语言到逻辑翻译方法，其次讨论基于大语言模型的翻译方法，随后介绍基于神经符号解耦架构的相关研究，最后总结现有研究的主要进展与不足，为后续章节的展开奠定基础。

2.2 传统方法

2.2.1 符号主义方法

早期的自然语言语义分析系统主要由符号主义方法主导。这类方法的基本思想是，语言意义可以通过一套明确、可书写的规则体系来表达和计算。研究者通常通过人工构建词典、文法规则和逻辑映射规则，将自然语言句子逐步分析为结构化语义表示，再进一步转换为对应的逻辑形式。在这一范式下，语言理解的关键不在于从数据中学习概率模式，而在于通过规则系统精确刻画语义组合过程。

这类方法的代表性成果可以追溯到二十世纪六七十年代。其中，Winograd 开发的 SHRDLU 系统^[24]是早期自然语言理解研究中的里程碑。该系统能够在模拟的“积木世界”中理解用户的自然语言指令、执行动作并回答问题，将语言理解、知识表示、推理和行动规划整合到同一框架之中。虽然其应用环境较为封闭，但它清楚地展示了：如果领域知识和语言规则被充分建模，自然语言可以被系统地映射为可操作的内部表示。

与此同时, Montague 提出的蒙塔古文法^[25] 为自然语言形式化研究提供了坚实的理论基础。蒙塔古文法强调组合性原则, 即复杂表达的意义应由其组成部分及其组合方式决定。这一思想对后续自然语言到逻辑形式的研究影响深远, 因为它为如何将自然语言中的词汇、短语和句法结构逐步组合为完整逻辑表达提供了严格的数学框架。从某种意义上说, 后续大量语义解析方法都不同程度继承了这一思路。

在具体系统层面, Woods 等人开发的 LUNAR 系统^[26] 以及 Warren 与 Pereira 提出的 CHAT-80^[27] 是这一时期具有代表性的应用成果。LUNAR 系统面向阿波罗计划带回的月岩样本数据, 能够将特定形式的英文问题翻译为数据库查询。CHAT-80 则借助 Definite Clause Grammar (DCG) 将自然语言问句转化为 Prolog 查询, 用于处理世界地理数据库中的检索问题。这些系统在预设的狭窄领域中都表现出较高精度, 也说明规则驱动方法在领域封闭、任务边界清晰的情况下具有明显优势。

不过, 符号主义方法的局限也十分明显。首先, 这类方法高度依赖专家手工设计规则和知识库, 系统开发与维护成本很高。其次, 其迁移能力较弱, 一旦输入句子偏离预设模式, 或者涉及领域外知识、未登录词和复杂修饰结构, 系统性能就容易显著下降。Androutsopoulos 等人^[28] 在综述中将这一问题概括为“知识获取瓶颈”, 即系统性能在很大程度上取决于人工规则覆盖的完备性, 而这种覆盖在开放场景中几乎难以实现。也正因如此, 符号主义方法虽然奠定了自然语言到逻辑翻译研究的理论基础, 却难以适应真实世界中开放、动态且充满歧义的语言环境。

2.2.2 统计学习方法

为了克服符号主义方法在鲁棒性和可扩展性方面的不足, 研究者在二十世纪九十年代后期开始逐步转向统计学习方法。这一转变的核心在于, 系统不再主要依赖人工编写的规则, 而是通过从标注语料中学习自然语言与逻辑形式之间的映射关系来完成语义分析。与符号主义方法相比, 统计学习方法更强调从数据中获取语言规律, 因此在处理语言变体、表达歧义和一定程度的输入噪声时通常更具灵活性。

这一时期最具代表性的路线之一, 是基于组合范畴语法 (Combinatory Categorical Grammar, CCG) 的统计语义分析。Zettlemoyer 与 Collins^[1] 提出的方法将句法范畴与基于 λ 演算的语义表示结合起来, 通过在带有逻辑形式标注的语料上训练模型, 自动学习词汇项和组合规则。CCG 的一个重要特点, 是能够把句法组合与语义组合统一到同一框架之中, 因此在自然语言到逻辑形式任务中表现出较强的表达能力和结构清晰性。

如图 2-1 所示, CCG 方法并不是简单地把句子看作线性序列, 而是通过带有语义信息的句法范畴逐步组合, 最终得到完整的逻辑表示。相比早期完全手工构造的规则系统, 这种做法保留了较强的结构解释能力, 同时通过概率模型增强

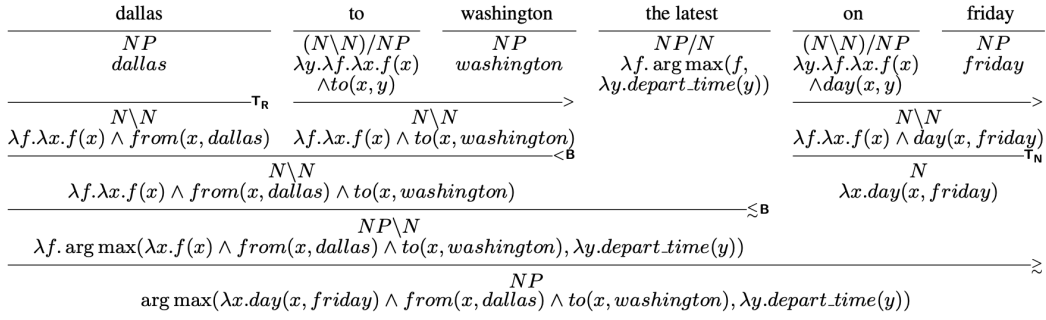
图 2-1 CCG 方法示意图^[1]

Figure 2-1 CCG Method

了对歧义的处理能力。围绕这一方向，Zettlemoyer 与 Collins 后续又进一步完善了从句到逻辑形式的统计学习框架^[29]，推动了统计语义解析的发展。

除了 CCG 路线之外，Wong 与 Mooney^[30,31] 借鉴统计机器翻译思想，提出了基于同步上下文无关文法（Synchronous Context-Free Grammar, SCFG）的方法，将自然语言与逻辑形式视为同步生成的结构。Liang 等人^[32] 则进一步提出基于依存组合语义的方法，希望在更灵活的句法依存结构上完成语义组合。与此同时，Clarke 等人^[33] 还探索了通过环境反馈学习语义表示的方法，即不直接依赖完整的逻辑形式标注，而是利用最终执行结果或答案信号对模型进行训练。

总体来看，统计学习方法相较于符号主义方法的一个重要进步，在于它不再完全依赖人工规则，而是能够通过数据自动学习部分语言模式。这使得模型在面对更丰富的表达形式时具有更好的适应能力，也使自然语言到逻辑翻译研究逐步从纯符号范式过渡到数据驱动范式。然而，这一阶段的方法并没有真正摆脱资源依赖问题。相反，它们对高质量标注数据提出了新的强需求，而自然语言到逻辑形式的配对标注往往需要具备语言学和逻辑学知识的专家参与，构造成本较高。

为缓解这一问题，研究者尝试使用弱监督方法减少对完整逻辑表达标注的依赖。例如，Liang 等人^[32] 和 Clarke 等人^[33] 的工作都体现了这种思路，即通过答案监督、环境反馈或执行结果间接学习语义表示。尽管这类方法降低了标注要求，但同时也引入了更复杂的学习问题，例如仿真问题和搜索空间扩张问题。因此，数据依赖始终是统计学习方法面临的根本限制之一。

此外，这类方法在处理复杂逻辑结构时仍存在明显不足。当句子中出现多重量词、嵌套条件、长距离依赖或更复杂的逻辑约束时，模型往往难以稳定保持逻辑表达的完整性和一致性。换言之，统计方法虽然在鲁棒性和自动化方面优于纯规则系统，但并未根本解决复杂语义组合和高精度逻辑翻译的问题。

2.2.3 早期神经网络方法

随着深度学习的发展，尤其是序列到序列（Sequence-to-Sequence, Seq2Seq）模型在机器翻译等任务中的成功，自然语言到逻辑翻译研究进一步进入神经网络

络阶段。与前一阶段相比，这类方法更强调端到端建模，即直接从输入句子学习到目标逻辑表达式之间的映射关系，而不再显式依赖复杂的句法语义模块划分。

Sutskever 等人^[34]提出的 Seq2Seq 框架为这一方向提供了基础模型形式。随后，Dong 与 Lapata^[2]将该框架引入自然语言到逻辑形式任务中，使用编码器读取自然语言输入序列，再由解码器逐步生成对应的逻辑表达式。与传统统计方法相比，这种方法减少了对手工特征和显式规则设计的依赖，理论上能够自动学习输入与输出之间更复杂的对齐关系。

在此基础上，Bahdanau 等人^[35]提出的注意力机制进一步提升了模型处理长句和复杂对应关系的能力。注意力机制使解码器在生成每一个目标符号时，都能够动态关注输入中的不同部分，从而缓解固定长度表示带来的信息瓶颈。对于自然语言到逻辑翻译任务而言，这意味着模型可以更细粒度地学习语言成分与逻辑成分之间的局部对齐关系。

为了进一步增强模型对逻辑结构的建模能力，后续研究开始尝试在神经网络框架中显式引入结构信息。Dong 与 Lapata^[2]提出了 Seq2Tree 方法，直接生成逻辑表达式的树结构，而不是简单生成线性 token 序列。相比普通的 Seq2Seq，Seq2Tree 更符合逻辑表达式本身的层次结构特征，因此在一定程度上改善了结构生成质量。之后，Yin 与 Neubig^[36]提出的语法神经网络进一步将目标语法规则显式纳入解码过程，以降低输出非法结构的风险。

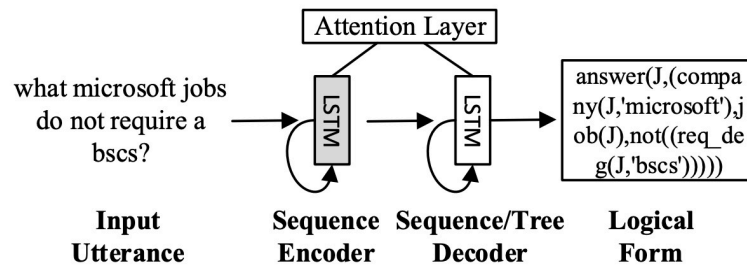


图 2-2 Seq2Tree 模型示意图^[2]

Figure 2-2 Seq2Tree Model

如图 2-2 所示，Seq2Tree 方法试图直接利用目标逻辑表达的树形结构进行解码，而不是将其视作纯线性序列。这种思路表明，研究者已经意识到自然语言到逻辑翻译并不是一般意义上的文本生成问题，而是一个带有显著结构约束的语义映射问题。

尽管如此，早期神经网络方法同样存在明显局限。第一，这类方法仍然依赖较大规模的高质量配对数据，尚未从根本上摆脱数据瓶颈。Jia 与 Liang^[37]的研究表明，即使在神经语义解析框架下，训练数据规模依然会显著影响模型性能。第二，这类方法的内部决策过程通常较难解释，当模型生成错误逻辑表达式时，研究者往往难以像分析规则系统那样准确定位错误来源。第三，基础的神经生成模型缺乏内在的逻辑约束，容易出现括号不匹配、量词作用域混乱或结构不自洽

等问题。虽然 Seq2Tree 和语法约束方法在一定程度上缓解了这一问题，但逻辑翻译结果的可靠性仍然难以得到充分保证。

总的来看，从符号主义到统计学习，再到早期神经网络，传统范式在提升自然语言处理复杂性和系统鲁棒性方面取得了持续进展。符号主义方法强调规则精确性和结果可解释性，统计学习方法增强了数据驱动建模能力，早期神经网络方法则进一步提升了端到端学习和表示建模能力。然而，这些方法始终没有完全解决知识获取成本高、标注数据稀缺以及逻辑健全性不足等问题。也正是在这些长期困难的推动下，研究才进一步转向大语言模型及其所带来的新一轮方法变革。

2.3 基于大语言模型的翻译方法

随着以 GPT 系列^[38]为代表的超大规模预训练语言模型的出现，自然语言处理领域的研究范式发生了明显变化。与早期主要依赖任务特定结构设计和监督数据驱动训练的方法不同，大语言模型通过在海量无标注文本上进行预训练，能够学习较为丰富的语言知识、世界知识以及一定程度上的推理能力，并表现出较强的任务迁移潜力。尤其是在上下文学习（In-Context Learning, ICL）能力被系统观察和总结之后^[38,39]，研究者开始意识到，模型可以在不更新参数的情况下，仅凭任务说明和少量示例快速适应新任务。这一能力为自然语言到一阶谓词逻辑翻译提供了新的技术路径，也使 NL2FOL 不再完全依赖传统语义解析框架中的规则设计和显式结构建模。

从方法形态上看，基于大语言模型的 NL2FOL 研究大致可以分为两条主要路线。第一条路线将大语言模型作为黑箱，通过提示工程直接引导模型生成逻辑表达式，其优势在于部署便捷、无需训练且迁移成本较低。第二条路线则通过监督微调、参数高效微调等方式，将通用语言模型进一步专门化，使其更好适应自然语言到逻辑表达式的映射任务。总体来看，大语言模型方法显著提高了自然语言到逻辑翻译的可达性与开放场景适应能力，但其可靠性问题也变得更加突出。

2.3.1 直接翻译

将大语言模型作为黑箱翻译器直接使用，是 NL2FOL 场景下最直接的一类方法。在这一范式中，研究的重点不再是重新设计网络结构，而是通过提示的构造方式来影响模型的行为。换言之，模型的参数保持不变，研究者主要通过任务描述、输入输出格式、上下文示例和中间推理线索来引导其完成从自然语言到逻辑表达式的映射。由于不需要额外训练，这类方法在实际应用中具有明显的便利性，也常被用作评估大语言模型零样本或少样本逻辑翻译能力的基线。

最简单的形式是零样本（Zero-Shot）翻译。用户只需在提示中提供明确的任务指令，例如要求模型将给定英文句子翻译成一阶谓词逻辑表达式，模型便可以直接给出输出。这种方式几乎没有额外准备成本，因此在新任务探索、模型能力测试以及快速原型构建中十分常见。然而，零样本方法虽然便捷，其性能通常不

够稳定。Kojima 等人^[40]指出,即便大语言模型已经具备一定推理潜力,单纯依赖直接指令往往仍难以稳定激发其复杂推理能力。对于 NL2FOL 这类高精度任务而言,这种不稳定性会被进一步放大,因为逻辑表达式中任意一个局部错误都可能破坏整体语义。

为了更精确地控制模型行为,研究者更常采用少样本(Few-Shot)翻译。Brown 等人^[38]在 GPT-3 中系统展示了少样本学习的潜力,即通过在上下文中提供若干输入输出示例,让模型通过类比方式推断当前任务的模式。在 NL2FOL 场景下,这些示例不仅告诉模型“该做什么”,还隐含规定了逻辑谓词的命名习惯、量词和逻辑连接词的写法、常量或变量的表示方式,以及复杂句子在形式化时的组织方式。因此,范例的选择与组织往往会显著影响最终输出质量。

围绕少样本提示,已有大量研究讨论了提示设计的关键因素。Liu 等人^[41]在关于提示方法的综述中指出,示例的质量、多样性和任务匹配程度是影响结果的重要因素。Lu 等人^[42]则进一步发现,示例顺序和整体格式都会对模型表现产生显著影响,即使任务本身并未变化,单纯调换范例排列方式也可能造成结果波动。这说明,模型在提示场景下并不是严格按照逻辑规则进行演绎,而是在较大程度上受上下文表面模式驱动。Rubin 等人^[43]因此提出从检索角度选择更合适的上下文示例,以减少提示构造的随机性。对于 NL2FOL 来说,这一问题尤为关键,因为不同示例中谓词命名、逻辑结构风格和表达细节的细微差异,都可能影响模型对当前句子的形式化方式。

当输入句子本身包含更复杂的逻辑关系时,单纯依赖输入输出示例往往仍然不足。很多自然语言句子在转换为逻辑表达式之前,通常需要经历多个隐含步骤,例如识别实体与属性、确定关系方向、分析量化范围、判断否定附着位置、分辨条件结构与蕴含关系等。如果这些中间步骤没有被显式揭示,模型很容易在最终输出时发生逻辑偏移。正是在这一背景下,Wei 等人^[44]提出的思维链(Chain-of-Thought, CoT)提示方法开始被广泛关注。该方法的核心思想是,在示例中不仅给出最终答案,还给出中间推理过程,使模型学会先分解问题、再逐步推导。在 NL2FOL 场景中,这种做法可以引导模型先进行语义分析,再完成符号组合,从而在一定程度上降低一步到位生成复杂逻辑表达式的难度。

此后,围绕 CoT 的研究进一步扩展。Kojima 等人^[40]发现,即使在没有推理范例的情况下,仅通过在提示中加入“Let’s think step by step”之类的语句,也可能诱导模型生成更完整的中间分析过程。Wang 等人^[45]则提出了自洽性(Self-Consistency)方法,通过对同一问题采样多条不同推理路径,并对最终结果进行聚合,从而减弱单一路径推理错误的影响。对于逻辑翻译任务而言,这种方法的启示在于:如果模型可以从不同角度分析同一句子,那么对多个候选表达式进行一致性判断,可能比依赖单次生成更稳健。Zhou 等人^[46]的由少到多提示(Least-to-Most Prompting)则进一步提出先解决较简单的子问题,再利用子问题结果处理更复杂的整体问题。这一思路也与复杂自然语言逻辑翻译中“先拆解、后组合”的需求具有较高一致性。

更进一步的工作开始尝试突破线性推理链的限制。Yao 等人^[47]提出的思维树 (Tree-of-Thoughts, ToT) 允许模型在推理过程中保留多个候选分支, 并结合搜索或自我评估机制决定后续展开方向。相比简单的线性 CoT, ToT 的思想更接近复杂问题求解中的试探、比较与回溯过程。虽然这类方法主要提出于一般推理任务场景, 但其对 NL2FOL 同样具有启发意义。因为逻辑翻译本身并不总是一个“从左到右、唯一确定”的过程, 尤其是当输入中出现嵌套结构、隐含约束或作用域歧义时, 允许模型探索多个分析分支再进行比较, 理论上更有利于获得更可靠的形式化结果。

不过, 基于提示的直接翻译方法也暴露出较强的脆弱性。首先, 这类方法的表现高度依赖提示模板、示例内容和输入表述方式, 即使语义等价但表面形式不同的提示, 也可能导致结果出现明显波动^[42]。其次, 模型生成的中间推理过程未必真实反映其内部判断依据, 推理步骤与最终输出之间不一定保持严格一致^[48]。这意味着, 即便模型给出了看似完整的分析过程, 也不能简单地将其视为严格可信的逻辑推导。再次, 提示方法本质上仍然依赖大模型已有能力边界, 当输入中包含多重量词、长距离依赖或更复杂的逻辑组合时, 模型往往仍然容易出现量词作用域混乱、逻辑连接关系错配和局部正确但整体不等价等问题。因此, 提示工程虽然显著降低了 NL2FOL 的进入门槛, 但难以从根本上保证逻辑翻译结果的稳定性和可验证性。

2.3.2 微调大语言模型

与仅依赖提示工程的方法相比, 另一条更具针对性的路线是对大语言模型进行微调。其核心思想是利用自然语言与逻辑表达式的配对数据, 进一步调整预训练模型的参数, 使其更好地适应 NL2FOL 任务的输入分布、输出格式以及逻辑结构要求。对于逻辑翻译这类高精度任务而言, 微调的吸引力在于它不再完全依赖大模型已有的泛化能力, 而是尝试让模型在专门数据上形成更稳定的任务偏好。

从技术上看, 微调可以分为全参数微调和参数高效微调两类。全参数微调能够充分适配任务, 但计算成本较高。相比之下, LoRA^[49]等参数高效微调 (PEFT) 方法通过只训练少量附加参数, 在较低资源消耗下实现对基础模型的任务适配, 因此在开源大语言模型的下游任务中被广泛采用。对于 NL2FOL 任务, 这类方法尤其具有现实意义, 因为逻辑翻译通常不是高频工业任务, 研究者更需要在有限算力条件下探索高质量适配方案。

在这一方向上, Yang 等人^[3]提出的 LogicLLaMA 是具有代表性的工作。该研究以 LLaMA2^[50]为基础模型, 通过构建专门面向 NL2FOL 的高质量训练数据, 并结合参数高效微调和针对性训练策略, 使模型能够更稳定地将自然语言翻译为一阶谓词逻辑表达式。与单纯依赖少样本提示的方法相比, LogicLLaMA 的优势在于其输出风格更统一、逻辑结构更稳定, 对复杂逻辑句子的适应能力也有所提升。

LogicLLaMA 的关键贡献之一在于 MALLS 数据集的构建。该数据集并非简单人工收集，而是借助更强模型和自动化流程生成大规模高质量的自然语言与 FOL 配对数据，再结合解析验证、人工校对和自动过滤等步骤进行清洗与筛选^[3]。这一做法体现出当前 NL2FOL 微调研究中的一个重要特点：由于高质量逻辑标注成本很高，研究者往往需要借助更强模型辅助构造训练数据，从而形成“强模型生成数据、弱模型接受训练”的路径。对该方向而言，数据构造本身已经不再只是准备环节，而是直接影响模型性能的重要组成部分。

在训练策略上，LogicLLaMA 不仅进行了基础的监督微调，还探索了更复杂的校正训练与反馈式优化思路。例如，让模型学习如何修正其他模型生成的错误逻辑表达，或让模型在多种候选形式中学会更合理的输出偏好。这类做法的本质，是希望模型不仅会“生成”，还能够在一定程度上学习“判断何为更好的生成结果”。从任务角度看，这种训练方式比单纯监督模仿更接近逻辑翻译的真实需求，因为 NL2FOL 不只是语法重写，而是要求输出结果在语义上与输入保持严格对应。

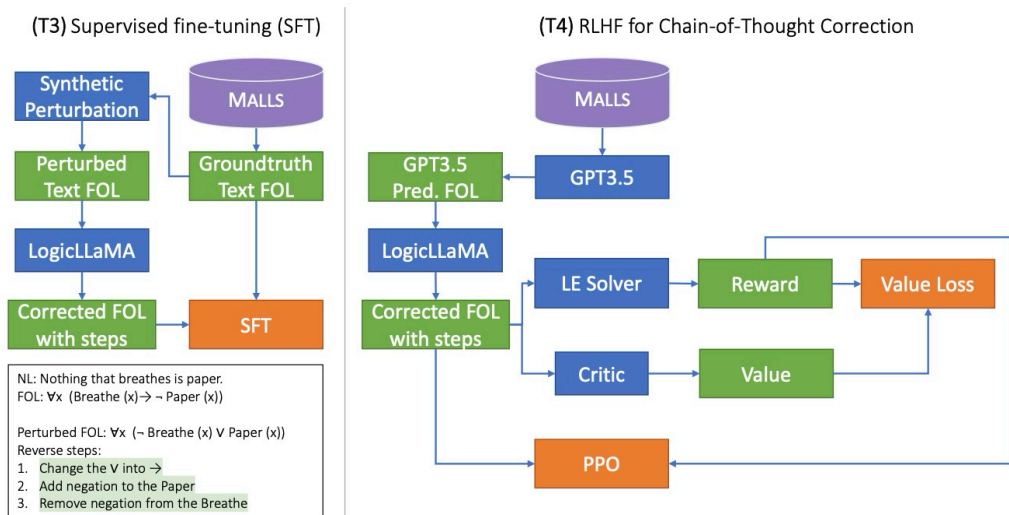


图 2-3 LogicLLaMA 模型^[3]
 Figure 2-3 LogicLLaMA Model

如图 2-3 所示，LogicLLaMA 并不是简单地把通用语言模型直接应用于逻辑翻译，而是在数据构建、模型适配和训练流程上都进行了专门设计。实验结果表明，经过任务特定微调的中小型开源模型，在某些 NL2FOL 基准上能够显著优于未微调的大模型少样本提示结果，甚至在部分场景下接近更强闭源模型的表现^[3]。这一结果说明，对于逻辑翻译这类结构化生成任务，专门的数据和适配训练仍然具有重要价值。

除了 LogicLLaMA 之外，近年来也出现了一些与微调路线相关的扩展探索。例如，Liu^[23] 提出的 CODE4LOGIC 虽然以少样本代码生成为主，但其核心思想同样说明，NL2FOL 并不一定适合被视为“直接从自然语言到逻辑公式”的单步

生成问题。通过在自然语言与逻辑表达式之间引入代码式中间结构,可以显著降低模型直接操作形式逻辑符号的难度。类似地,Ryu 等人^[17]的工作进一步强调,对于复杂逻辑翻译任务,仅依赖单步生成往往不够稳健,分解、验证和重组是提升性能的重要方向。这些工作虽然不完全等同于传统意义上的监督微调,但与 LogicLLaMA 一样,都体现了一个共同认识,即通用大语言模型若想在 NL2FOL 上取得稳定表现,往往需要通过任务特定设计进行增强。

不过,微调路线虽然通常能够在性能上优于纯提示方法,却并未从根本上消除大语言模型在逻辑翻译中的深层局限。首先,这类方法依赖质量较高、规模较大的自然语言与逻辑表达式配对数据,而这类数据获取成本高、构造周期长。其次,模型即使经过专门训练,其内部机制仍然是基于概率分布的 token 预测,而不是内置了形式逻辑规则的推理系统。因此,当输入逻辑结构明显超出训练分布、组合形式更复杂或语义歧义更强时,模型仍然容易出现组合泛化不足的问题^[51]。此外,微调后的模型虽然输出格式更稳定,但其内部决策过程并不会因此变得透明,研究者仍然较难准确定位翻译错误的来源。在高可靠应用场景中,这意味着微调改善了任务适配性,却未必真正解决了结果可验证性和推理可信性问题。

综上所述,基于大语言模型的方法无论是在灵活性还是在开放场景适应能力方面都带来了明显提升。提示工程降低了任务使用门槛,微调方法则提升了特定场景下的表现稳定性。然而,这些方法仍然不同程度地受到概率生成机制的限制,在逻辑健全性、语义保真度、组合泛化和结果可解释性方面都存在明显不足。也正是在这一背景下,研究者开始进一步寻求更稳健的解决路径,例如将语言模型与外部验证模块、符号推理器或中间表示机制结合起来,从而推动研究重心逐渐转向神经符号混合架构。

2.4 基于神经-符号解耦架构的翻译方法

由于将大语言模型作为黑箱直接用于 NL2FOL 时存在逻辑健全性不足等问题,研究的重心逐渐转向神经-符号混合架构。与其试图将大语言模型本身训练成一个完全可靠的逻辑推理器,这一路线更强调不同计算范式之间的分工协作。其基本思想是,充分利用大语言模型在自然语言理解、语义联想和开放场景适应方面的优势,同时引入符号系统在规则执行、形式验证和精确推理方面的能力,通过明确的任务解耦构建更稳健的翻译系统。

从整体上看,神经-符号方法并不只是对大语言模型输出结果增加一个后处理步骤,而是在系统层面重新划分任务边界。自然语言中的模糊性、歧义性和上下文依赖性,更适合由数据驱动的神经模型进行处理;而逻辑表达式的语法合法性、类型一致性和推理正确性,则更适合由符号模块负责。这种架构思路为提升 NL2FOL 的准确性、可解释性和可验证性提供了新的可能,也使逻辑翻译问题逐渐从单模型生成任务转向多模块协同任务。

2.4.1 双系统视角下的任务解耦

神经-符号方法的核心思想，可以从双系统视角来理解。相关思想一方面可以追溯到人工智能早期关于模块化认知架构的讨论^[52]，另一方面也与认知科学中的双过程理论具有较强一致性^[53]。在这一视角下，系统通常被划分为两个功能不同但彼此互补的部分^[54]。

第一部分是以大语言模型为代表的神经模块。该模块更接近双过程理论中的“系统 1”，负责处理自然语言输入中的语义内容。它擅长从上下文中捕捉语义线索，识别实体、关系、事件和隐含约束，并根据任务提示生成初步的结构化表示。在 NL2FOL 任务中，这一部分的主要职责，是将原始自然语言句子转化为候选逻辑表达式，或者转化为进一步形式化所需的中间表示。由于大语言模型在开放语境中具有较强适应能力，因此它非常适合作为系统的语义理解前端。

第二部分是形式逻辑工具、求解器、验证器和知识库为代表的符号模块。该模块更接近“系统 2”，负责对神经模块生成的中间结果进行约束、检查和执行。相较于大语言模型偏概率式、生成式的处理方式，符号模块的优势在于其行为规则明确、结果可重复，并能够对逻辑表达式进行确定性的形式分析。对于 NL2FOL 来说，符号模块不仅可以检查逻辑表达式的语法和类型是否正确，还可以进一步分析其是否满足某些一致性约束，或能否支持后续推理过程。

Sarker 等人^[20]在神经-符号人工智能综述中指出，这类分工合作的模式可以在一定程度上结合神经网络和符号系统各自的优点。对于逻辑翻译任务而言，这意味着系统不必要求大语言模型独立完成全部步骤，而是允许模型专注于其更擅长的语义理解和结构生成部分，再由符号工具承担严格验证和规则执行任务。这样做不仅有助于提高翻译结果的可靠性，也为后续分析错误来源、解释系统行为提供了更清晰的切入点。

从方法论角度看，任务解耦的价值还体现在它改变了问题求解的方式。在纯大语言模型方法中，翻译、判断和纠错往往被压缩到一次生成之中，系统内部缺乏明确的中间边界。而在神经-符号框架下，这些功能可以被分散到不同模块中完成。例如，大语言模型负责提出候选解释，验证器负责指出结构问题，求解器负责检查语义一致性，重排序模块负责在多个候选中选择更合理结果。这种显式模块化设计，使系统具备了传统端到端生成模型所不具备的可检查性和可干预性。

2.4.2 翻译与验证的主流实现范式

在 NL2FOL 任务中，神经-符号解耦最常见的实现方式是“翻译-验证”流水线。这类方法通常先由大语言模型完成从自然语言到候选逻辑表达式的初步翻译，再由外部符号工具对翻译结果进行形式检查和后续处理。其核心不在于让符号系统替代大语言模型完成全部翻译，而在于通过外部验证机制弥补大语言模型在逻辑健全性方面的不足。

在这一流程中，第一阶段通常是翻译阶段。系统首先输入原始自然语言句

子,再由大语言模型生成一个或多个候选逻辑表达式。这个阶段的关键在于语义映射,也就是如何把自然语言中的实体、关系、属性、量化结构和逻辑约束尽可能完整地表达出来。由于大语言模型具备较强的语言理解和模式归纳能力,它在这一阶段通常能够较快给出形式上看似合理的候选结果。

第二阶段则是验证阶段。与翻译阶段不同,验证阶段强调形式正确性和规则一致性,因而更依赖符号工具而非生成模型。一个较常见的验证流程通常包括以下几层。首先是语法检查,即检查逻辑表达式是否满足既定文法,例如括号是否匹配、谓词符号是否合法、逻辑连接结构是否完整。其次是类型检查,即检查谓词的元数和参数类型是否符合约束。例如,一个表示二元关系的谓词是否错误地接收了三个参数。再次是与背景知识或知识库的一致性检验,即检查候选逻辑表达式是否与现有事实、预定义本体或已知规则存在明显冲突。对于更复杂的系统,还可能进一步结合 SAT、SMT 或模型查找工具,对逻辑表达式进行可满足性分析和语义约束检查。

从思想上看,这类方法与工具增强语言模型 (Tool-Augmented Language Models, TALMs) 的研究方向是相一致的^[4,55]。其关键并不在于让语言模型学会内部模拟所有外部工具,而在于允许模型在必要时调用外部工具,以获得比纯语言生成更可靠的中间结果和最终结果。在 NL2FOL 场景中,这种工具调用尤其有价值,因为逻辑表达式一旦出现局部错误,后续推理往往会整体失效,而外部工具恰好能够在这些关键位置提供确定性的形式约束。

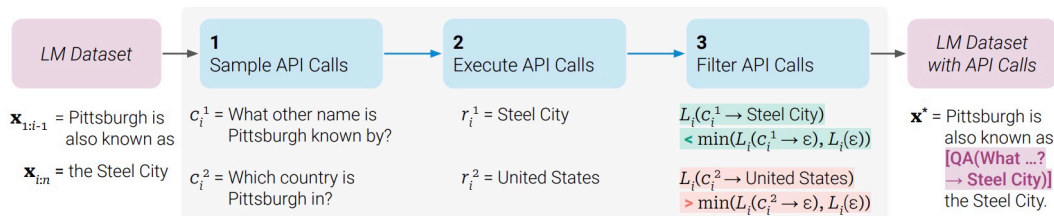


图 2-4 TALMs 方法示意图^[4]

Figure 2-4 TALMs Method

如图 2-4 所示,工具增强型方法的关键在于将语言模型与外部工具连接起来,使模型在需要时不再单独完成全部任务,而是将部分步骤交由更合适的系统处理。对于 NL2FOL 而言,这一思路意味着大语言模型可以负责提出逻辑翻译候选,而语法检查器、求解器或验证模块负责判断这些候选是否满足形式约束。由此,翻译系统不再完全依赖单次生成结果,而是具备了基于外部反馈进行质量控制的可能。

在现有研究中,这种思路已经在逻辑推理和自然语言形式化任务中得到不同程度体现。Olausson 等人提出的 LINC^[15] 将语言模型与一阶逻辑证明器结合起来,通过形式推理工具增强系统在逻辑任务中的稳定性。Xu 等人提出的 Symbol-LLM^[56] 则试图从数据和模型接口层面加强语言模型对符号关系的建模能力。这些工作虽然任务设置并不完全相同,但都说明一个共同趋势:当任务需要精确处

理逻辑结构时，仅依赖大语言模型本身通常不够，而外部符号组件可以在系统中发挥关键作用。

2.4.3 高级交互模式与局限性

在基础的“翻译-验证”流水线之上，研究者进一步提出了更复杂的交互模式，以提升系统在复杂逻辑场景中的适应能力。最直接的一类方法是生成与重排序（generate and rerank）。与其只保留模型生成的第一个候选逻辑表达式，系统可以要求模型生成多个候选结果，再由验证模块根据语法合法性、类型一致性、知识约束满足情况等标准进行重排序，从中选出最合理的表达式。这种做法的优势在于，它不再把一次生成的结果视为唯一答案，而是把翻译过程视作一个候选搜索和选择过程，从而在一定程度上缓解单次生成错误带来的风险。

另一类重要方法是迭代式修正。与简单的“一次翻译，一次验证”不同，这类系统允许验证模块将错误信息反馈给大语言模型，再由模型根据反馈重新生成结果。例如，当验证器发现某个谓词元数不匹配、变量绑定不完整或逻辑结构存在缺口时，系统并不直接终止，而是把这些错误转化为结构化提示，要求模型进行修正。这种方式使系统从线性的单向流水线，转变为一个“生成—检查—反馈—修正”的闭环过程。Yao 等人提出的 ReAct 框架^[57]虽然并非专门针对 NL2FOL，但其“推理与行动交替进行”的思想对这一类方法具有重要启发意义。

除此之外，还有一类方法强调对复杂问题进行显式分解，即先把难句拆成若干更小的语义片段，再分别翻译和验证，最后组合为完整结果。这种分解式思路尤其适合处理长句、嵌套结构或具有多重逻辑连接关系的输入。相关研究表明，对于复杂逻辑翻译任务，单步端到端生成往往容易在中间结构上发生偏差，而通过对子句、成分或逻辑依赖关系进行显式拆解，可以降低每一步的难度，并使系统更容易发现局部错误^[17,58]。从某种意义上说，这类方法体现了神经-符号路线中的另一种重要思想，即复杂逻辑结构不一定要一次性生成完成，而可以通过更细粒度的中间过程逐步构建。

不过，尽管上述方法显著增强了系统的可验证性和结果控制能力，它们仍然面临一个难以回避的核心瓶颈，即整个系统的性能上限在很大程度上仍受制于翻译阶段的语义保真度。验证器、求解器和符号工具虽然能够有效判断一个逻辑表达式在形式上是否合规，却通常无法直接判断该表达式是否忠实反映了原始自然语言的真实含义。换言之，外部符号工具擅长判断“表达式是否成立”，但未必能够判断“这个表达式是不是原句真正想表达的那个意思”。

这一问题在 NL2FOL 中尤为突出。例如，对于含有量词辖域歧义的句子，即便模型生成的两个候选逻辑表达式在语法和类型上都完全正确，验证模块也未必能够区分哪一个与原文语义更匹配。类似地，当自然语言表达本身带有词义歧义、隐含约束或上下文依赖时，如果大语言模型在翻译的第一步就误解了原意，那么后续再强的符号验证也只能建立在一个错误前提之上。也就是说，神经-符号架构虽然能显著提升结果的形式可靠性，却并不能自动消除翻译阶段的语义

偏移问题。

因此,从现有研究来看,神经-符号解耦方法的重要价值并不在于它已经彻底解决了 NL2FOL 的全部困难,而在于它更清晰地暴露了问题真正的瓶颈所在:系统是否可靠,最终仍然取决于翻译阶段能否给出足够准确、足够忠实的逻辑表达。验证模块可以筛掉形式上明显错误的结果,也可以帮助系统进行修正和重排序,但如果前端生成本身缺乏足够高的语义保真度,整个系统的性能依然会受到显著限制。也正因如此,如何在翻译阶段进一步提升大语言模型的结构保持能力、复杂逻辑处理能力和语义对应精度,仍然是该方向的核心问题。

综上,基于神经-符号解耦架构的翻译方法为 NL2FOL 提供了一条比纯生成方法更稳健的研究路径。它通过明确划分语义理解与形式验证的职责边界,提高了系统的可检查性、可解释性和可控制性,也为复杂逻辑翻译任务提供了更具工程可行性的实现框架。但与此同时,这一路线也表明,仅靠后端验证并不足以彻底解决问题,前端翻译的准确性仍然是决定整个系统表现的关键因素。这一认识也为后续研究进一步聚焦翻译阶段的能力增强提供了直接动因。

2.5 本章小结

本章围绕自然语言到一阶谓词逻辑翻译任务的相关研究进行了综述,并从传统范式、大语言模型方法以及神经-符号解耦架构三个方面梳理了该领域的发展脉络。

首先,在传统范式方面,相关研究经历了从符号主义方法、统计学习方法到早期神经网络方法的演进过程。符号主义方法强调规则精确性和结果可解释性,在受限领域中取得了较好效果,但存在知识获取成本高、迁移能力弱等问题。统计学习方法在一定程度上缓解了规则依赖问题,增强了对语言变体的适应能力,但仍然受到高质量标注数据稀缺和复杂逻辑建模能力不足的限制。早期神经网络方法进一步推动了端到端语义映射的发展,但其在逻辑约束建模、结果可解释性和逻辑健全性方面仍然存在明显不足。

其次,基于大语言模型的方法为 NL2FOL 任务带来了新的研究契机。无论是基于提示的直接翻译方法,还是通过监督微调与参数高效微调进行任务适配的方法,都显著提高了自然语言到逻辑表达翻译的可达性和开放场景适应能力。相关研究表明,大语言模型在简单逻辑结构和常见表达模式下已经能够取得较好效果。然而,这类方法的表现仍然容易受到提示方式、示例构造、训练数据分布以及模型组合泛化能力的影响,在复杂逻辑结构、隐含语义关系和高精度语义保真要求下,仍然存在逻辑不稳定、结果脆弱和可验证性不足等问题。

再次,基于神经-符号解耦架构的方法进一步从系统层面回应了上述不足。通过将自然语言理解与形式验证分离,这类方法充分利用了大语言模型在语义理解上的优势以及符号系统在规则执行和逻辑验证上的优势,从而提高了系统的可检查性和可控制性。翻译-验证流水线、候选重排序、迭代修正以及分解式翻译等方法,都在一定程度上增强了复杂逻辑翻译任务中的结果可靠性。不过,

现有神经-符号方法同样表明，后端验证模块虽然能够缓解形式错误，却难以从根本上弥补前端翻译阶段的语义偏移问题。换言之，逻辑翻译系统的整体性能在很大程度上仍然受制于自然语言到逻辑表达式这一核心映射过程本身的准确性。

综合来看，现有研究已经为 NL2FOL 任务提供了较为丰富的方法基础，但仍存在若干值得进一步研究的问题。其一，面对显性复杂逻辑结构时，现有方法仍然容易出现语法正确但语义不等价的翻译结果，复杂逻辑结构下的翻译稳定性仍有待提升。其二，对于句级文本中的隐性逻辑关系，现有方法往往难以通过表层语言直接恢复完整逻辑结构，缺乏更适合作为桥梁的中间表示与验证机制。其三，当前研究大多集中于句子级翻译任务，对于篇章级上下文中的潜在逻辑关系建模仍然相对不足。

基于上述分析，本文后续研究将围绕自然语言到一阶谓词逻辑翻译中的关键困难展开。首先，针对显性复杂逻辑结构文本，研究面向推理阶段的增强方法，以提高复杂逻辑翻译中的语义一致性和结果稳定性。其次，针对句级隐性逻辑关系，研究基于中间表示的翻译方法，以增强隐含逻辑关系的显式建模能力。最后，针对篇章级文本中的跨句逻辑关系，研究结合上下文语义信息的翻译方法，以推动 NL2FOL 从句子级场景向篇章级场景拓展。

第3章 面向显性逻辑结构文本的 NL2FOL 技术

3.1 引言

NL2FOL 任务的核心要求，在于将自然语言中的语义内容准确映射为形式化逻辑表达式，并尽可能保持与原始语义的严格对应。前一章的综述表明，尽管现有方法已经在规则驱动、统计学习、神经网络以及大语言模型等方向上取得了一定进展，但在复杂逻辑场景下，模型仍然容易出现语义偏移、结构错配和逻辑不一致等问题。尤其当输入文本中包含多重量词、否定、条件、蕴含以及嵌套结构时，这种问题会更加突出。对于这类显性复杂逻辑结构文本而言，翻译错误往往不再只是局部符号选择失误，而更可能导致整个逻辑表达式的含义发生变化。

从现有研究来看，大语言模型虽然已经展现出较强的语言理解和生成能力，但其在 NL2FOL 任务中的表现仍然存在明显不稳定性。一方面，模型生成的逻辑表达式在形式上可能是合法的，但在语义上却未必与原始句子等价；另一方面，随着逻辑结构复杂度提升，模型更容易在量词辖域、逻辑连接关系和否定附着位置等关键细节上产生偏差。已有研究表明，这类错误并不完全来源于解码阶段的随机性，也可能与模型内部表示在前向传播过程中逐步偏离原始逻辑结构有关。对于显性复杂逻辑结构而言，由于逻辑成分之间存在更加严格的组合关系，这种内部表示偏移往往会被进一步放大，并最终表现为语义上不一致的逻辑翻译结果。

针对上述问题，本章聚焦于显性复杂逻辑结构文本的自然语言到一阶谓词逻辑翻译任务，研究如何在不更新模型参数的前提下，提高大语言模型在复杂逻辑翻译中的语义一致性和结果稳定性。为此，本章提出一种面向推理阶段的增强方法 LSEG。该方法的基本思想是，从表示层面分析并干预模型在翻译过程中的逻辑偏移，而不是单纯依赖额外监督或重新训练。具体而言，LSEG 通过构造逻辑保持扰动样本来估计逻辑稳定中心，并进一步提取分层逻辑校正方向；在推理阶段，再结合层级熵变化识别模型内部的不稳定状态，并对相应层的隐状态进行自适应干预，从而减轻复杂逻辑结构下的语义漂移问题。

与现有依赖监督微调或额外训练数据的方法相比，本章方法具有两个较为突出的特点。其一，LSEG 无需修改模型参数，能够以推理阶段插件的形式直接作用于已有大语言模型，具有较好的灵活性和可迁移性。其二，该方法并非在输出端对逻辑表达式进行后处理，而是尝试直接从内部表示层面对逻辑偏移进行校正，因此在方法设计上更加贴近复杂逻辑翻译错误的形成机制。通过这种方式，本章希望为提升大语言模型在显性复杂逻辑结构下的 NL2FOL 翻译能力提供一种新的思路。

本章其余内容安排如下：第 3.2 节介绍本章方法的整体设计，包括问题定义、方法动机以及 LSEG 的总体框架；第 3.3 节给出实验结果与分析，重点考察该方法在基准数据集上的表现、跨模型适用性以及各组成部分的作用；第 3.4 节对本

章内容进行总结。

3.2 方法介绍

3.2.1 问题定义与方法动机

自然语言到一阶谓词逻辑翻译任务的目标，是将自然语言句子转换为与其语义严格对应的一阶谓词逻辑表达式。设输入自然语言句子为 x ，输出逻辑表达式为 y ，则该任务可以形式化为一个条件生成问题，即在给定输入 x 的条件下，生成最可能且语义上与原句等价的逻辑表达式 y 。若将大语言模型记为参数为 θ 的条件生成模型，则该过程可以表示为：

$$y = \arg \max_{y'} p_{\theta}(y' | x) \quad (3-1)$$

如式 (3-1) 所示，NL2FOL 任务本质上是一个以语义等价性为目标的条件生成问题。对于一般自然语言生成任务而言，模型只需保证输出在语法和语义上基本合理即可；但对于 NL2FOL 任务而言，这种要求远远不够。逻辑表达式不仅需要形式上满足语法约束，更重要的是要在语义上与原始自然语言保持严格对应。换言之，模型生成的结果不仅要像一个逻辑表达式，还必须是原句真正对应的那个逻辑表达式。

这一任务在显性复杂逻辑结构场景下尤其困难。所谓显性复杂逻辑结构，是指输入句子中包含较为明确的逻辑成分，如全称量词、存在量词、否定、条件、蕴含、合取、析取以及多层嵌套结构等。这些逻辑成分并非隐含在篇章推断之中，而是直接体现在句子的表层表达里。然而，尽管逻辑关系是显性的，模型在翻译时仍然很容易出错。其原因在于，这些逻辑成分之间并不是简单并列的，而是受到严格的组合约束。例如，量词的辖域位置会直接决定命题的含义，否定的附着对象会改变整个句子的真假条件，条件结构与蕴含关系的结合顺序也会影响逻辑公式的整体解释。一旦模型在这些细节上出现偏差，即使生成结果在形式上看起来合理，其语义也可能已经偏离原句。

以量词辖域为例，自然语言句子 “Every student reads a book” 同时涉及全称量词与存在量词。如果模型错误地处理了两者之间的辖域关系，就可能把 “每个学生都读了某一本书” 翻译成 “存在一本书被每个学生都读过”。这两种表达式在形式上都合法，但表达的语义显然不同。类似地，在含有否定、条件或多重连接词嵌套的句子中，即便模型只是在局部连接关系的选择上产生细微偏差，也可能导致整个逻辑结构的意义发生改变。因此，显性复杂逻辑结构并不是只要出现逻辑词就容易翻译的场景，相反，它恰恰是最考验模型语义精确映射能力的场景之一。

为了更直观地说明显性逻辑结构中的语义偏移问题，图 3-1 给出了一个典型示例。对于自然语言句子中带有排他性含义的结构 (either ... or ...)，如果模型仅将其翻译为普通析取 (or)，而没有保持更强的约束关系，那么即便输出结果在

形式上与正确答案十分接近，也会在语义层面产生明显偏移。由此可见，显性复杂逻辑结构翻译的困难并不只是符号层面的生成问题，更关键的是逻辑关系是否被精确保留。

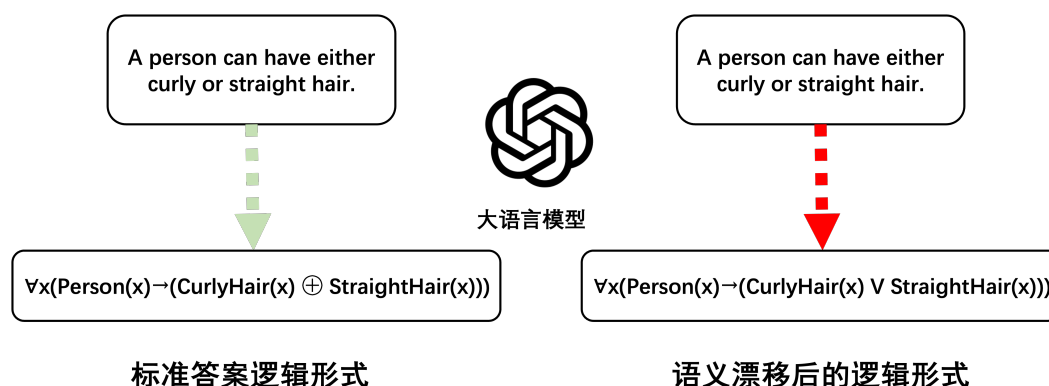


图 3-1 显性逻辑结构中的典型语义偏移示例

Figure 3-1 A Typical Example of Semantic Drift in Explicit Logical Structure Translation

已有研究表明^[59]，这类错误并不完全来自最终解码阶段，而可能与模型内部表示在前向传播过程中的逐步偏移有关。对于 NL2FOL 而言，由于逻辑结构对局部偏差高度敏感，这种表示层面的细微漂移往往会在层间传播过程中被不断放大，最终导致语法正确但逻辑错误的结果。也就是说，模型在输出端暴露出的逻辑错误，往往只是更深层内部表示失稳的外在表现。尤其在显性复杂逻辑结构场景下，由于多个逻辑成分之间存在明确而严格的依赖关系，任何局部表示的偏移都更容易沿着推理链条持续累积，并最终破坏整体逻辑语义。

基于上述分析，我们认为，显性复杂逻辑结构下的翻译困难，本质上并不只是输出端的符号选择问题，而是一个内部表示稳定性问题。现有方法大多通过监督微调、提示设计或输出后处理来提升翻译效果，但这些方法要么依赖较高的数据和训练成本，要么只能在输出层面对错误进行纠正，难以直接作用于逻辑偏移形成的内部过程。具体而言，监督微调虽然能够增强模型对任务模式的适应能力，但其效果在很大程度上依赖标注数据规模与分布质量；提示设计能够在一定程度上改善模型的显式推理表现，但往往缺乏对内部表征演化过程的直接约束；输出后处理则更多是在生成结束之后进行修补，难以改变错误已经在表示层中形成这一事实。因此，这些方法虽然在一定场景下有效，但对于显性复杂逻辑结构翻译中普遍存在的表示漂移问题，仍然缺乏针对性更强的解决路径。

基于这一认识，本章尝试从表示层面入手，研究一种无需微调、能够在推理阶段直接校正模型内部逻辑偏移的方法，以提升复杂逻辑翻译中的语义一致性和结果稳定性。本章的基本出发点是：当模型面对逻辑结构较为复杂的输入时，其内部表征并非始终沿着稳定的逻辑轨迹演化，而可能在某些关键层中偏离原本应保持的语义方向。如果能够在推理过程中识别这种偏移，并及时施加适度的方向校正，那么模型就有可能在不修改参数的前提下，生成更符合原句语义的逻辑表达式。也正是在这一动机下，本章提出了 LSEG 方法，希望通过对模型内部

表示的分层干预，为显性复杂逻辑结构下的 NL2FOL 翻译提供一种更具针对性、也更符合错误形成机制的增强思路。

3.2.2 LSEG 方法总体框架

针对显性复杂逻辑结构翻译中存在的表示漂移问题，本章提出 LSEG 方法。该方法的核心思想是，不直接修改模型参数，也不单纯依赖输出端重写，而是在推理阶段对模型内部隐状态进行有针对性的校正，使其尽可能保持与原始逻辑结构一致的表示方向。与传统监督微调方法相比，LSEG 更关注模型在推理过程中如何保持逻辑稳定，而不是如何通过额外训练让模型记住更多样本模式。换言之，本章并不试图重新塑造模型的全部参数空间，而是尝试在已有模型能力的基础上，通过更细粒度的表示干预来缓解复杂逻辑翻译中的语义偏移问题。

从整体上看，LSEG 可以分为两个相互配合的阶段。第一阶段是逻辑校正方向构建阶段。该阶段的目标，是为模型估计一个能够反映逻辑稳定区域的方向向量，用来描述当前输入的表示应当朝哪个方向移动，才能更接近逻辑一致的内部状态。第二阶段是熵引导推理干预阶段。该阶段的目标，是在实际生成过程中，根据模型不同层的内部不稳定程度，决定何时、何处以及以多大强度注入校正方向，从而避免对所有层一视同仁地进行干预。

这种设计体现了 LSEG 的两个基本判断。第一，复杂逻辑翻译中的错误往往不是因为模型完全不知道如何翻译，而是因为模型在某些层、某些时刻偏离了本应维持的逻辑轨迹。因此，方法的关键不在于重新教会模型，而在于帮助模型在推理过程中维持正确方向。第二，推理阶段的干预不能是静态和固定的。如果不考虑模型内部状态变化，直接在所有层施加统一强度的修正，很容易导致过度校正，反而破坏原本已经稳定的语义表示。因此，LSEG 并不是简单地把一个固定向量加到模型内部，而是把逻辑方向构建和动态稳定性检测结合起来，形成一种更具针对性的推理控制机制。

从框架结构来看，LSEG 的总体流程如下。首先，针对一组逻辑保持的输入变体，提取模型在不同层的表示，并通过平均操作估计某一逻辑结构在该层的稳定中心。然后，将原始输入在该层的表示与这个逻辑稳定中心进行比较，得到逻辑位移向量。进一步地，在跨样本层面上对这些位移向量进行统计分析，提取更具有普适性的分层校正方向。最后，在实际推理过程中，根据不同层的熵变化识别模型内部的不稳定层，仅在這些层上注入相应方向的干预向量，从而在尽量不影响语言流畅性的前提下，减轻逻辑漂移问题。

这一框架有几个较为明显的特点。首先，它不依赖模型参数更新，因此具有较低的部署成本。对于已经具备一定语言理解和生成能力的大语言模型而言，这种不改参数的接入方式具有较强的工程便利性。其次，它把干预位置从输出端前移到中间表示层，更接近错误形成的源头。相比于生成完成后再做逻辑修补，LSEG 更强调在错误尚未固化为输出之前，提前对潜在偏移进行抑制。再次，它通过熵信号控制干预强度，避免了统一修正带来的副作用。这样一来，只有在模

型内部确实出现不稳定迹象时，系统才会触发更明显的逻辑校正，从而减少对正常语义表示的干扰。最后，由于方向向量的构建依赖的是逻辑保持扰动和表示统计，而非特定任务头或特定训练目标，因此该方法具有一定的模型无关性和迁移潜力。

LSEG 的整体框架如图 3-2 所示。可以看到，该方法由两个相互衔接的部分组成。首先，在离线阶段，方法通过逻辑保持扰动样本提取分层表示，并据此构建逻辑稳定中心与分层校正方向。随后，在在线推理阶段，方法依据各层熵变化识别当前生成过程中的不稳定层，并仅对这些层实施方向注入与自适应控制。这样的两阶段设计使 LSEG 既能够提供稳定的逻辑校正依据，又能够避免对原本稳定的层施加不必要的干预，从而在逻辑一致性与语言流畅性之间取得更好的平衡。

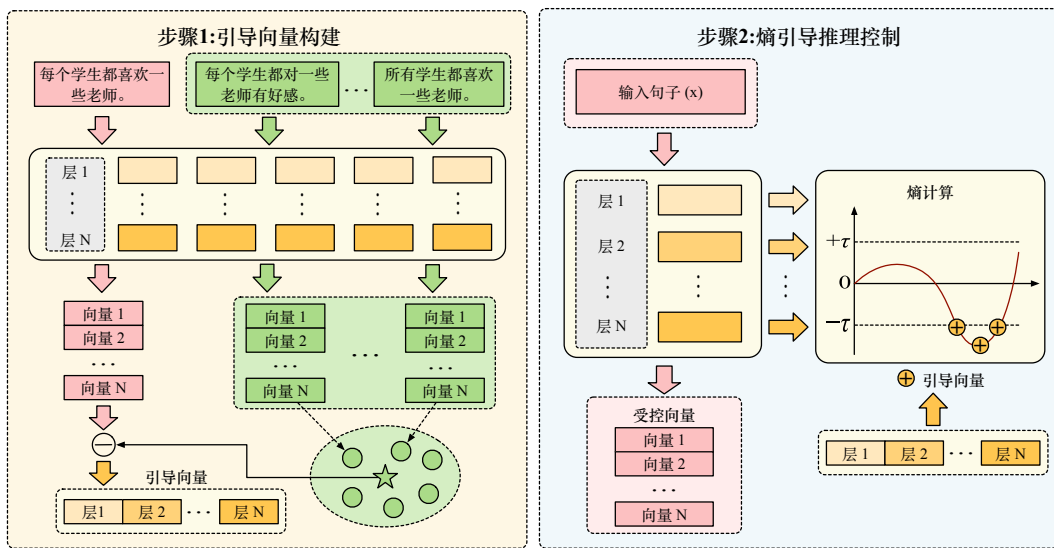


图 3-2 LSEG 方法整体框架

Figure 3-2 Overall Framework of the LSEG Method

进一步来看，LSEG 的整体设计并不是简单地将两个模块机械拼接在一起，而是形成了一种协同结构。离线阶段负责提供应当朝哪里校正的信息，在线阶段负责决定何时进行校正、在哪些层进行校正以及校正到什么程度。前者为后者提供方向依据，后者则为前者提供动态调用机制。也就是说，逻辑校正方向构建阶段解决的是校正目标问题，而熵引导推理干预阶段解决的是校正时机与方式问题。只有将二者结合起来，才能使推理阶段的逻辑增强既具有明确方向性，又具有较好的状态适应性。

从方法定位上看，LSEG 既不同于纯监督微调方法，也不同于单纯的输出后处理方法。监督微调的主要优势在于可以通过任务数据让模型学习更稳定的模式，但这种方式往往需要额外训练成本，而且模型参数一旦固定，其推理过程中的瞬时偏移仍未必能够被及时抑制。输出后处理虽然可以对生成结果进行一定修正，但它更多是在错误已经形成之后再行补救，难以直接干预错误形成的内

部过程。相比之下，LSEG 将干预位置前移到中间表示层，并通过动态状态检测控制干预强度，因此在方法逻辑上更贴近复杂逻辑错误的形成机制。

此外，这种框架还为后续分析提供了较好的可解释性基础。由于 LSEG 明确区分了逻辑方向构建和动态状态检测两个部分，因此本章不仅能够分析方法是否有效，还能够进一步分析其有效性究竟来自哪里。例如，若某些层区间上的干预效果更明显，则可以从表示层语义整合位置的角度加以解释；若熵波动与逻辑错误之间存在较强相关性，则说明熵信号可以作为内部稳定性的重要代理指标。这种模块化设计使得 LSEG 不仅是一个可以直接部署的推理增强方法，也为理解大语言模型在复杂逻辑翻译任务中的内部行为提供了更清晰的分析框架。

总体而言，LSEG 方法并不试图通过额外训练彻底重塑模型的逻辑能力，而是希望在模型已有能力的基础上，通过表示层面的方向引导与状态感知干预，尽可能减少显性复杂逻辑结构翻译中的语义漂移问题。正因为如此，该方法既保留了较低的部署成本，又在理论上更接近复杂逻辑错误的形成源头，为后续的分层校正方向构建与熵引导推理机制奠定了整体框架基础。

3.2.3 逻辑校正方向构建方法

LSEG 的第一步，是构建能够反映逻辑稳定性的校正方向。为了做到这一点，本章并不直接把原始输入的隐状态当作唯一参考，而是引入一组与原句逻辑等价、但表层表达存在差异的输入变体。这样做的原因在于，自然语言表层可能包含大量与逻辑骨架无关的词汇、句法和风格差异，如果仅使用单一句子的表示来估计逻辑结构，很容易把这些表层噪声也一并纳入。因此，需要通过一组逻辑保持的扰动样本，尽可能保留不变的逻辑骨架，同时削弱表层差异带来的扰动。

所谓逻辑保持扰动，是指在不改变目标逻辑形式的前提下，对输入句子的表层表达进行轻量改写。例如，可以进行同义替换、轻微句法重组或措辞调整，但不能改变量词辖域、逻辑连接方式、谓词和论元之间的语义关系。举例来说，对于“Every student read a book”，可以将其改写为“Each student has read at least one book”，因为两者表达的逻辑关系保持不变；但不能改写为“There is a book that every student read”，因为这会改变量词的作用范围。由此可见，逻辑保持扰动的核心要求，并不是表面文字尽量相似，而是在表达形式发生变化的同时，逻辑骨架仍然保持不变。

设原始输入为 x ，其逻辑保持变体集合为 $\{x'_1, x'_2, \dots, x'_n\}$ 。对于模型第 l 层，原始输入的隐状态记为 $h^{(l)}(x)$ ，而各个扰动变体在该层的隐状态分别记为 $h^{(l)}(x'_1), h^{(l)}(x'_2), \dots, h^{(l)}(x'_n)$ 。在此基础上，可以定义该层的逻辑稳定中心为这些扰动表示的平均值，即

$$h_{\text{logic}}^{(l)} = \frac{1}{n} \sum_{i=1}^n h^{(l)}(x'_i) \quad (3-2)$$

如式 (3-2) 所示，这个平均表示并不是某个具体样本的表示，而是多个逻辑

等价样本在该层上共享的稳定逻辑区域。由于不同变体在词汇和句法上存在差异，平均操作可以在一定程度上削弱表层因素的影响，保留更具有共性的逻辑结构信息。因此， $h_{\text{logic}}^{(l)}$ 可以被视为该逻辑结构在第 l 层上的稳定锚点。换言之，它所代表的并不是单个句子的特定表征，而是该类逻辑结构在当前层上的相对稳定中心。

在得到逻辑稳定中心之后，首先定义原始输入在该层上的表示为

$$h_{\text{orig}}^{(l)} = h^{(l)}(x) \quad (3-3)$$

随后，可以进一步定义原始输入在该层上的逻辑位移向量为

$$\Delta^{(l)} = h_{\text{logic}}^{(l)} - h_{\text{orig}}^{(l)} \quad (3-4)$$

如式 (3-4) 所示，位移向量 $\Delta^{(l)}$ 描述了原始输入表示应当朝哪个方向移动，才能更接近由逻辑保持变体共同刻画出的稳定逻辑区域。也就是说， $\Delta^{(l)}$ 刻画了从当前表示状态到逻辑稳定状态之间的修正方向。在直观上，如果把模型某一层的表现空间看作一个高维语义空间，那么原始输入表示与逻辑稳定中心之间的差值，就可以理解为模型在这一层上相对于稳定逻辑状态的偏移程度与偏移方向。

不过，单个样本对应的位移向量仍然可能受到个体噪声影响，因此还需要进一步在样本层面上提取更稳健的分层主方向。为此，LSEG 在每一层收集多个样本的位移向量集合 $\{\Delta_1^{(l)}, \Delta_2^{(l)}, \dots, \Delta_N^{(l)}\}$ ，并对其进行主成分分析。记第一主成分为

$$v^{(l)} = \text{PCA}_1 \left(\{\Delta_n^{(l)}\}_{n=1}^N \right) \quad (3-5)$$

其中， $v^{(l)}$ 就是第 l 层的主校正方向。采用主成分分析而不是直接使用单一样本位移向量，主要有两个原因。第一，主成分能够提取跨样本共享的主要偏移模式，从而减弱个别句子中的偶然噪声。第二，第一主成分提供的是一个统计意义上最稳定的方向，更适合在推理阶段被重复复用。相比之下，若直接使用某一个样本的位移向量，则这一方向中可能混入特定句子的措辞、长度或局部结构差异，难以作为通用校正依据。

从表示学习的角度看，这一过程实际上是在估计“逻辑一致表示”与“当前原始表示”之间的系统性偏移模式。通过逻辑保持扰动，本章首先为同一逻辑结构构造出一组表层不同但逻辑一致的观测样本；通过平均表示，又将这些观测样本压缩为一个相对稳定的逻辑中心；进一步地，通过主成分分析，本章从多个样本中提取出最具代表性的偏移方向。这样得到的 $v^{(l)}$ 不再是某个单独样本的偶然结果，而更接近于跨样本共享的逻辑修正方向。

这一设计还有一个重要含义，即本章所构建的校正方向并不是基于标签监督或任务损失显式学习得到的，而是通过表示统计方式从逻辑保持样本中归纳

出来的。也正因为如此，该方法不依赖额外参数训练，也不要求模型具备专门的任务头结构。只要模型能够输出分层隐状态，并且输入变体在逻辑上保持一致，就可以按照同样思路构建对应的分层校正方向。这使得该方法天然具有较强的可迁移性和模型适配性。

为了更直观地展示这一阶段的作用过程，图 3-2 中左半部分给出了逻辑校正方向构建的整体示意。可以看到，原始输入首先被扩展为一组逻辑保持扰动样本，这些样本经过模型前向传播后，在各层得到对应的表示向量。随后，系统在层内对这些表示进行聚合，得到逻辑稳定中心，并进一步据此计算逻辑位移方向。该过程的关键不在于获得某一个绝对正确的隐藏表示，而在于利用多个逻辑等价样本之间的共性，估计出更接近逻辑不变结构的表示参考点。

总体而言，逻辑校正方向构建阶段的目标，是通过逻辑保持扰动、稳定中心估计和分层主方向提取，建立一个面向推理阶段的逻辑校正坐标系。在这一坐标系中，不同层上的主校正方向分别刻画了模型表示应当如何偏移，才能更接近逻辑一致的内部状态。后续推理阶段的干预并不是凭空注入一个任意向量，而是以这里得到的分层主方向为基础，在模型内部不稳定时刻施加有依据的方向性修正。正因为如此，逻辑校正方向构建不仅是 LSEG 的前置步骤，也是整个方法能够具备方向感和可解释性的基础。

3.2.4 基于熵信号的自适应推理干预机制

在获得各层的逻辑校正方向之后，接下来的问题是：这些方向应当在什么时刻、什么层以及以多大强度注入模型内部。一个最直接的做法，是在所有层上持续施加固定强度干预。但这种方式往往过于粗糙。因为并不是模型在每一层、每一个生成步骤都处于不稳定状态，如果对本来已经较为稳定的层继续施加强干预，反而可能破坏其已有的语义表示。因此，LSEG 并不采用固定式干预，而是引入基于熵变化的自适应控制机制，以实现推理过程更细粒度的状态感知与动态校正。

从直观上看，模型在生成某一 token 的过程中，不同层所对应的内部状态并不是完全等价的。有些层仍处于较为分散的候选竞争状态，而有些层则已经形成相对集中的决策倾向。如果在后一种情况下继续强行注入统一强度的校正，便可能打断原本已经较为稳定的语义整合过程。相反，如果能够识别模型在哪些层上出现了异常波动，并仅在这些位置实施方向性修正，那么干预就会更有针对性，也更有可能在控制逻辑漂移的同时保留语言生成的自然性。因此，本章将层级熵变化视为内部稳定性的重要代理信号，用来判断模型当前是否偏离了应有的逻辑演化轨迹。

具体来说，在解码步骤 t 和第 l 层，首先通过共享输出投影矩阵将该层隐状态 $h_t^{(l)}$ 映射到词表空间，从而得到该层对应的 token 分布，即

$$p_t^{(l)} = \text{softmax}(W_{\text{out}} h_t^{(l)}) \quad (3-6)$$

其中, W_{out} 表示模型输出投影矩阵。如式 (3-6) 所示, 这一步的目的是将中间层的表示显式投射到可观测的概率分布空间中, 以便进一步分析不同层在当前生成步骤上的不确定程度。

在得到词表分布之后, 可以进一步计算该层在当前步骤上的信息熵, 即

$$H_t^{(l)} = - \sum_k p_{t,k}^{(l)} \log p_{t,k}^{(l)} \quad (3-7)$$

如式 (3-7) 所示, $H_t^{(l)}$ 用来刻画模型在该层对下一个 token 预测的不确定程度。若熵值较高, 说明当前层给出的分布较为分散, 模型尚未形成明确偏好; 若熵值较低, 则说明预测分布更集中, 模型在该层上的决策倾向更强。需要强调的是, 这里的熵并不直接等同于逻辑正确性, 但它可以作为内部状态稳定程度的重要表征。尤其对于显性复杂逻辑结构翻译而言, 当模型在某些关键层发生异常快速收缩或波动时, 往往意味着其内部语义整合过程可能已经偏离正确逻辑方向。

为了进一步捕捉层与层之间的状态变化, 本章定义相邻层之间的熵差为

$$\Delta H_t^{(l)} = H_t^{(l)} - H_t^{(l-1)} \quad (3-8)$$

如式 (3-8) 所示, $\Delta H_t^{(l)}$ 描述了模型从第 $l-1$ 层过渡到第 l 层时, 不确定性发生了怎样的变化。如果某一层相对于前一层出现了明显的熵骤降, 通常意味着模型在这一层快速收缩了候选空间; 而若出现异常升高, 则可能意味着原有语义判断受到了扰动, 内部表示重新变得不稳定。因此, 相比单独观察每一层的绝对熵值, 层间熵差更能反映表示演化过程中的动态变化。

由于不同样本、不同层以及不同解码位置上的绝对熵差尺度可能并不一致, 因此仅依赖原始熵差数值并不足以稳定识别异常波动。为此, 本章进一步对层间熵差进行标准化处理, 得到

$$z_t^{(l)} = \frac{\Delta H_t^{(l)} - \mu_{\Delta H}}{\sigma_{\Delta H}} \quad (3-9)$$

其中, $\mu_{\Delta H}$ 和 $\sigma_{\Delta H}$ 分别表示熵差的均值和标准差。如式 (3-9) 所示, 标准化后的 $z_t^{(l)}$ 能够更好地反映当前层熵波动相对于整体分布所处的位置。当其绝对值较大时, 说明该层在当前步骤上的状态变化明显偏离常规范围, 可以被视为潜在的不稳定层。

基于这一判断, 本章将那些波动幅度超过阈值的层定义为当前解码步骤中的不稳定层集合, 即

$$L_t = \{ l \mid |z_t^{(l)}| > \tau \} \quad (3-10)$$

其中, τ 为预设阈值。如式 (3-10) 所示, 集合 L_t 表示在当前生成步骤中需要重点关注并可能施加逻辑校正的层。与对所有层统一干预的方式不同, 这种做

法使系统能够只在模型内部状态出现明显异常时才触发修正，从而降低不必要干预所带来的副作用。

在确定不稳定层之后，还需要决定每一层应当以多大强度接受校正。LSEG 并不为所有不稳定层赋予相同强度，而是让干预强度与该层的不稳定程度成比例。其定义为

$$\alpha_l = \alpha_{\max} \frac{|\Delta H_t^{(l)}|}{\max_{j \in L_t} |\Delta H_t^{(j)}|} \quad (3-11)$$

其中， $\alpha_{\max}$ 表示允许的最大干预强度。如式 (3-11) 所示，熵变化越剧烈的层，其干预强度越接近上限；而波动相对较弱的层，则只接受较小幅度的修正。这样做的目的，在于使方向注入不再是“是否干预”的二元决策，而是一个随内部状态动态变化而连续调整的过程，从而避免因统一强干预而造成的过度校正问题。

最终，在第 l 层和步骤 t 上，LSEG 将离线阶段获得的逻辑校正方向 $v^{(l)}$ 注入当前隐状态，得到干预后的表示

$$h_t'^{(l)} = h_t^{(l)} + \alpha_l v^{(l)} \quad (3-12)$$

如式 (3-12) 所示， $h_t'^{(l)}$ 表示经过逻辑方向修正后的隐状态。其基本含义是，在保留原有表示主体结构的基础上，将当前层表示沿着逻辑稳定方向进行适度偏移，以抵消模型在该层可能出现的语义漂移。需要指出的是，这种注入并不是对原有表示的完全替代，而是一次幅度受控的局部修正，因此能够在尽量保持生成流畅性的同时，提高模型对复杂逻辑结构的对齐能力。

算法 1 LSEG：基于逻辑结构与熵引导的 NL2FOL 推理控制算法

- 1: **输入：** 语言模型 f_θ ，校准数据集 $\{x_i\}_{i=1}^N$
 - 2: 对每个 x_i 生成逻辑保持扰动样本 $\{x'_{i,j}\}_{j=1}^n$
 - 3: 基于扰动均值表示计算逐层位移向量 $\Delta_i^{(l)}$
 - 4: 在每一层执行 PCA，得到干预方向向量 $\{v^{(l)}\}_{l=1}^L$
 - 5: **for** 每个解码时间步 t **do**
 - 6: 计算各层 token 熵 $H_t^{(l)}$
 - 7: 计算标准化层间熵变化 $z_t^{(l)}$
 - 8: $\mathcal{L}_t \leftarrow \{l \mid |z_t^{(l)}| > \tau\}$
 - 9: **for** 每个 $l \in \mathcal{L}_t$ **do**
 - 10: 计算自适应注入强度 α_l
 - 11: 更新隐藏状态 $h_t^{(l)} \leftarrow h_t^{(l)} + \alpha_l v^{(l)}$
 - 12: **end for**
 - 13: **end for**
-

为了更清晰地展示 LSEG 在推理阶段的整体执行过程，本章将上述机制概括为算法 1。该算法包括两个相互衔接的部分：一是离线阶段的分层逻辑校正方

向构建，二是在线阶段基于熵信号的自适应推理干预。前者负责提供各层的逻辑修正依据，后者则根据模型在实际生成过程中的内部状态动态决定干预位置与干预强度。二者共同作用，使得推理阶段的逻辑增强既具有明确方向性，也具有较好的状态适应性。

从方法设计上看，这一机制有两个重要意义。第一，它使推理干预不再是静态和全局的，而是根据模型当下状态进行动态调整。这样一来，系统不必在所有层上都持续施加强修正，而是能够把干预集中在真正出现异常波动的位置。第二，它把“是否干预”与“如何干预”统一到同一套内部诊断框架中，使得整个校正过程更具针对性和可解释性。对于显性复杂逻辑结构翻译而言，这种自适应控制尤为重要，因为复杂逻辑错误往往集中出现在少数关键层和关键时刻，如果能够在这些位置及时施加修正，就更有可能在不明显破坏语言生成流畅性的前提下，提高整体逻辑一致性。

总体而言，基于熵信号的自适应推理干预机制，是 LSEG 从离线方向构建走向在线实际应用的关键桥梁。它不仅为逻辑校正方向提供了具体的使用时机与使用方式，也使整个方法从静态方向估计上升为动态状态感知的推理增强系统。正因为有了这一机制，LSEG 才能够在不修改模型参数的前提下，对显性复杂逻辑结构翻译中的内部表示漂移进行更有针对性的控制。

3.2.5 方法分析与讨论

从整体上看，LSEG 的有效性主要来自两个方面。一方面，它通过逻辑保持扰动构造出更接近逻辑不变结构的参考区域，并据此估计逻辑稳定中心与分层校正方向，从而为模型内部表示提供可参照的校正坐标。对于显性复杂逻辑结构翻译而言，语义偏移往往并非仅出现在最终输出阶段，而是在前向传播过程中受到表层词汇、局部句法和生成历史等因素影响，逐步偏离原有逻辑骨架。LSEG 的作用不在于事后判断输出是否正确，而是在中间表示层回答“当前表示应当朝哪个方向移动，才能更接近逻辑一致状态”这一问题。

另一方面，LSEG 并不在所有层和所有时刻统一施加修正，而是利用熵信号识别模型内部的不稳定状态，只在必要时刻进行干预，从而在逻辑校正与语言流畅性之间取得平衡。由于大语言模型不同层承担的功能并不相同，浅层更多涉及词汇和局部句法编码，中高层则更倾向于复杂语义与结构关系整合，因此固定式干预容易破坏模型原有的正常语义组织过程。相比之下，基于熵引导的动态注入能够提升干预的针对性，并减少不必要的副作用。

从方法定位来看，LSEG 与输出后处理方法和监督微调方法均存在明显区别。输出后处理通常在逻辑表达式生成之后再利用规则、验证器或重写策略进行修正，其优势在于可直接操作显式符号，但难以改变错误已在模型内部形成的事实。当深层表示中的错误逻辑已经逐步固化时，后处理往往只能进行局部修补。LSEG 则将干预位置前移到表示层，试图在错误尚未定型为显式符号之前抑制其形成过程，因此更贴近复杂逻辑错误的实际生成机制。与监督微调相比，LSEG

的主要作用点不在参数空间，而在推理时的中间状态空间。前者通过大量标注数据塑造模型的长期任务能力，后者则在不更新参数的前提下，对具体推理过程中的瞬时表示漂移进行定向校正，二者更适合被理解为互补而非替代关系。

从可解释性与工程实现角度看，LSEG 也具有一定优势。其关键组成部分具有较明确的功能分工：逻辑保持扰动用于刻画不变逻辑骨架，平均表示用于估计逻辑稳定中心，主方向提取用于概括共享偏移模式，熵波动用于识别不稳定状态，动态强度分配用于控制干预幅度。正是由于这些模块之间具有相对清晰的对应关系，LSEG 不仅便于分析整体有效性，也为后续消融实验和层级分析提供了更直接的解释基础。同时，该方法不依赖大规模额外监督数据，也无需重新训练模型参数，因而能够以较低成本接入已有基础模型。方向构建与推理干预之间的解耦设计，也使其在灵活性与部署效率上具有一定现实价值。

当然，LSEG 仍然存在若干前提条件与潜在局限。首先，其有效性依赖于逻辑保持扰动的质量。如果输入变体在表层改写过程中引入了新的歧义，或改变了原有逻辑关系，那么由此估计得到的逻辑稳定中心与主校正方向就可能偏离真实目标。其次，熵信号虽可作为内部稳定性的代理指标，但并不能直接等同于逻辑正确性，某些语义错误未必伴随显著熵波动，而某些熵波动也未必一定对应真实的逻辑漂移。因此，熵更适合作为状态诊断线索，而非逻辑正确性的绝对判据。此外，LSEG 所构建的分层校正方向本质上是一种统计意义上的共享方向，更适合纠正跨样本反复出现的系统性偏移，对于极端稀有或分布外的复杂结构，其作用可能相对有限。

尽管存在上述限制，LSEG 仍为显性复杂逻辑结构翻译提供了一种具有代表性的推理阶段增强思路。它表明，在大语言模型已经具备一定逻辑翻译能力的前提下，性能瓶颈未必完全来自知识或训练不足，也可能与推理过程中的内部表示稳定性密切相关。通过对这种稳定性进行定向分析与动态干预，模型在不修改参数的条件下仍有可能获得稳定增益。就这一点而言，LSEG 的意义不仅在于提出了一种具体方法，也在于为理解复杂逻辑翻译中的错误来源提供了新的分析视角。

3.3 实验结果与分析

3.3.1 实验设置

本章在 FOLIO 和 MALLS 两个数据集上开展实验。两者都可用于自然语言到一阶谓词逻辑的评测，但构建方式和难度侧重点并不相同。FOLIO 是一个人工构建并带有一阶逻辑标注的数据集，最初用于自然语言中的形式逻辑推理研究。其构建过程包括自然语言写作、逻辑公式标注、语言质量检查、逻辑质量检查以及自动验证等环节，整体标注成本较高。该数据集规模为 1430 个 example，数据具有较强的人工性和逻辑复杂度^[60]。相比之下，MALLS 是 LogicLLaMA 工作中提出的 NL2FOL 数据集，该数据集主要由 GPT 4 生成，并结合有效性检查构建而成，规模约为 34K，对句级翻译任务更友好，也更适合观察模型在复杂表

达下的泛化能力^[3]。

从任务特点上看，FOLIO 更适合检验模型在标准显性逻辑现象上的整体表现，因为其中包含较多带有明确逻辑成分的自然语言表达，例如量词、否定、合取、析取和条件关系等^[60]。MALLS 的规模更大，句式和表达来源也更丰富，因此更适合用来观察模型在复杂句级翻译场景中的稳定性。本章同时使用这两个数据集，主要是希望分别从标准逻辑结构翻译能力和复杂表达条件下的稳健性两个层面考察 LSEG 的效果。

在评价指标方面，本章采用逻辑等价性 (Logical Equivalence, LE) 作为主要指标。LE 关注的不是预测公式与参考公式在字符串层面是否完全一致，而是两者在逻辑语义上是否等价。对于 NL2FOL 任务来说，这一点尤其重要，因为两个公式即使在变量命名、谓词顺序或局部写法上存在差异，只要表达的逻辑含义相同，就应当被判为正确。在实现上，常见做法是借助定理证明器或求解器判断预测公式 ϕ 与参考公式 ψ 是否逻辑等价，例如检验 $\neg(\phi \leftrightarrow \psi)$ 是否不可满足；若不可满足，则说明二者逻辑等价。相较于精确匹配，LE 更能反映模型是否真正保留了原句的逻辑含义，因此本章采用 LE 作为核心评价标准。

3.3.2 实验结果

表 3-1 给出了 LSEG 与现有方法在 FOLIO 和 MALLS 上的 LE 结果。整体来看，Qwen 3 8B 基线已经具备一定的逻辑翻译能力，在 FOLIO 和 MALLS 上的 LE 分别达到 88.00% 和 89.70%。在接入 LSEG 后，两项结果分别提升至 92.01% 和 91.34%，说明该方法在不进行参数更新的前提下，仍然能够稳定提升模型在复杂逻辑翻译任务中的语义一致性。考虑到 Qwen 3 本身是通用大语言模型，而非专门面向 NL2FOL 训练的模型，这一结果说明推理阶段的表示干预仍然可以为逻辑翻译任务带来可观收益。

从结果上看，LSEG 在两个数据集上都取得了正向增益。在 FOLIO 上，LSEG 相比基线提高了 4.01 %；在 MALLS 上，提高了 1.64 %。前者说明该方法能够有效改善模型在标准显性逻辑场景下的翻译质量，后者则说明这种收益并没有随着句子复杂度上升而完全消失。换言之，LSEG 的作用并不只是修补少量简单样本，而是在不同难度条件下都表现出一定稳定性。

横向比较来看，LSEG 的结果已经达到甚至超过多数现有方法。在 FOLIO 上，LSEG 的 92.01% 已经明显高于 GPT 4、LogicLLaMA 和 Code4Logic 的 CodeGeeX 版本，也略高于 Symbol LLM 13B，仅略低于 Code4Logic 的 GPT 3.5 Turbo 16k 版本。在 MALLS 上，LSEG 的 91.34% 则超过了表中的所有对比方法。这说明，尽管 LSEG 并未依赖额外监督微调或更大规模模型参数，但仅通过推理阶段的表示干预，仍然可以取得具有竞争力的结果。

这一结果值得注意的地方在于，LSEG 的改进并不是通过增加训练数据规模或引入额外任务头实现的，而是通过对模型内部表示进行定向校正实现的。也就是说，这一结果在一定程度上说明，显性复杂逻辑结构翻译中的性能瓶颈并不完

表 3-1 LSEG 与现有方法在 FOLIO 和 MALLS 上的逻辑等价性结果

Table 3-1 Logical Equivalence Results on FOLIO and MALLS

方法	FOLIO LE (%)	MALLS LE (%)
Flan T5 ^[61]	70.67	68.45
Claude 1 ^[62]	74.47	77.46
GPT 4 ^[10]	85.53	84.38
LogicLLaMA ^[3]	84.90	81.98
Symbol LLM 13B ^[56]	91.59	89.41
Code4Logic (CodeGeeX) ^[23]	84.77	85.81
Code4Logic (GPT 3.5 Turbo 16k) ^[23]	92.67	90.92
Qwen3 8B ^[8]	88.00	89.70
LSEG (基于 Qwen3 8B)	92.01	91.34

全来自模型不会翻译，还可能来自模型在推理过程中没有稳定地沿着正确逻辑方向生成。当这种内部漂移被适度抑制时，即便不改变模型参数，模型仍然有机会输出更符合原句语义的逻辑表达式。

进一步看，不同方法的结果也反映出当前 NL2FOL 任务中的几条典型技术路线。以 LogicLLaMA 和 Symbol LLM 为代表的方法，更强调监督微调或逻辑约束带来的收益^[3,56]；以 Code4Logic 为代表的方法，则尝试借助代码生成作为中间形式来提升翻译能力^[23]。LSEG 与这些方法不同，它并不直接改变模型的训练过程，也不依赖额外的中间表示，而是把重点放在推理阶段的内部状态控制上。从表 3-1 的结果来看，这一路径是有效的，而且在某些数据集上具有明显优势。

当然，也需要看到 LSEG 并不是在所有设置下都取得绝对最优结果。例如，在 FOLIO 上，Code4Logic 的 GPT 3.5 Turbo 16k 版本仍然略高于 LSEG^[23]。这说明推理阶段增强并不是唯一有效的技术方向，利用更强基础模型或更合适的中间表示设计同样能够带来收益。不过，从方法属性上看，LSEG 的优势在于其部署成本更低。它不需要重新训练参数，也不依赖闭源 API 才能发挥作用，因此更适合作为现有开源模型上的轻量增强模块。

总体而言，主实验结果表明，LSEG 在两个代表性数据集上都带来了稳定提升，并且这种提升是在不更新模型参数的条件下实现的。这说明，本章提出的方法确实能够从表示层面缓解显性复杂逻辑结构翻译中的语义漂移问题，也为后续的跨模型验证和消融分析提供了基础。

3.3.3 跨模型验证结果

为了进一步考察 LSEG 是否具有跨模型适用性，本章将其应用到 LogicLLaMA 上进行验证。与作为主实验基础模型的 Qwen3 8B 不同，LogicLLaMA 是一个专

门面向自然语言到一阶谓词逻辑翻译任务构建的模型，其训练过程已经引入了针对 NL2FOL 的任务适配。因此，在这一设置下接入 LSEG，主要是为了观察：当基础模型本身已经具有较强的逻辑翻译能力时，推理阶段的表示干预是否仍然能够带来额外收益。

表 3-2 给出了 LSEG 在 LogicLLaMA 7B 和 13B 两个版本上的验证结果。可以看到，LSEG 在两个模型规模上都带来了稳定提升。对于 LogicLLaMA 7B，FOLIO 上的 LE 从 84.12% 提升到 85.24%，MALLS 上从 81.98% 提升到 83.06%；对于 LogicLLaMA 13B，FOLIO 上从 85.81% 提升到 86.90%，MALLS 上从 82.37% 提升到 83.48%。虽然提升幅度不如在 Qwen3 8B 基线上的提升那样明显，但这种跨模型、跨参数规模的一致改进，仍然说明 LSEG 并不依赖某一个特定基础模型才能发挥作用。

表 3-2 LSEG 在 LogicLLaMA 上的跨模型验证结果

Table 3-2 Cross Model Validation Results of LSEG on LogicLLaMA

方法	FOLIO LE (%)	MALLS LE (%)
LogicLLaMA 7B ^[3]	84.12	81.98
LogicLLaMA 7B + LSEG	85.24	83.06
LogicLLaMA 13B ^[3]	85.81	82.37
LogicLLaMA 13B + LSEG	86.90	83.48

这一结果至少说明了两点。第一，LSEG 的收益并不依赖于基础模型必须是未经过逻辑任务训练的通用模型。即便在 LogicLLaMA 这种已经针对 NL2FOL 任务进行过适配的模型上，LSEG 仍然能够进一步提升结果，说明其所纠正的问题并没有被原有训练过程完全消除。换言之，监督训练可以帮助模型学到更强的逻辑映射能力，但这并不意味着模型在具体推理过程中就不会发生内部表示漂移。

第二，这也说明 LSEG 与现有逻辑微调方法之间更接近互补关系，而不是替代关系。LogicLLaMA 的优势主要来自训练阶段的任务适配，而 LSEG 的作用点则位于推理阶段的中间表示控制。前者更偏向于让模型学会如何完成任务，后者则更偏向于帮助模型在生成过程中维持稳定的逻辑方向。从这个意义上说，二者分别作用于不同层面：一个作用于参数空间，另一个作用于推理时状态空间。因此，当二者结合时，能够形成一种较自然的协同效果。

当然，也需要看到，LSEG 在 LogicLLaMA 上的提升幅度相对更小。这种现象也是可以理解的。因为 LogicLLaMA 本身已经在训练阶段融入了面向 NL2FOL 的任务知识，其初始表示状态通常比通用模型更加贴近目标任务，因此可被进一步校正的空间相对有限。但即便如此，LSEG 仍然能够在两个数据集和两个模型规模上带来一致的正向增益，这恰恰进一步支持了本章的判断，即显性复杂逻辑

结构翻译中的错误并不只来自模型知识不足，也与推理过程中的状态稳定性有关。

总体来看，跨模型验证结果表明，LSEG 具有较好的模型无关性和可迁移性。它不仅能够在通用大语言模型上发挥作用，也能够为已经经过逻辑任务适配的模型提供额外收益。这说明，本章提出的方法并不是某一基础模型上的偶然技巧，而是一种在不同模型条件下都具有一定适用性的推理阶段增强机制。

3.3.4 消融实验结果

为了进一步分析 LSEG 各组成部分的实际作用，本章围绕两个问题开展消融实验：第一，熵引导的自适应调度是否必要；第二，不同层区间上的干预效果是否存在明显差异。前者主要用于检验动态状态检测在方法中的作用，后者则用于分析逻辑校正更适合发生在模型的哪些层。

表 3-3 熵引导自适应调度机制的消融结果

Table 3-3 Ablation Results of the Entropy Guided Adaptive Scheduling Mechanism

变体	FOLIO LE (%)	MALLS LE (%)
无熵引导（固定强度）	90.42	90.11
固定调度 LSEG	91.02	90.83
完整 LSEG	92.01	91.34

首先，在熵引导机制的消融实验中，本章比较了三种设置：一是移除熵引导，仅保留固定强度干预；二是保留方向注入，但采用固定调度；三是完整的熵引导 LSEG。结果如表 3-3 所示。可以看到，在去掉熵引导后，模型在 FOLIO 和 MALLS 上的 LE 分别下降到 90.42% 和 90.11%；采用固定调度时，结果略有回升，分别达到 91.02% 和 90.83%；完整 LSEG 的结果最高，分别达到 92.01% 和 91.34%。这一结果说明，单纯注入逻辑校正方向虽然已经能够带来一定收益，但如果缺少对模型内部状态变化的动态识别，干预效果仍然会受到限制。

从结果变化趋势看，完整 LSEG 相比“无熵引导”在 FOLIO 上提升了 1.59 个百分点，在 MALLS 上提升了 1.23 个百分点；相比“固定调度 LSEG”则分别提升了 0.99 和 0.51 个百分点。这说明，熵引导机制带来的收益并不是偶然的，而是稳定存在的。其原因在于，复杂逻辑翻译中的错误并不会在所有层和所有时间步上平均出现，而更可能集中在少数关键层和关键时刻。如果不考虑这种内部状态差异，而是统一施加固定强度干预，就容易出现干预不足或过度干预的问题。完整 LSEG 通过熵信号识别不稳定层，并按波动幅度分配干预强度，因此能够在逻辑校正与语义保持之间取得更好的平衡。

其次，本章进一步分析不同层区间上的干预效果。为此，本章将 Transformer 的层划分为多个不重叠的四层区间，并在每次实验中仅对某一区间施加 LSEG

表 3-4 不同层区间上的 LSEG 干预效果

Table 3-4 Intervention Effects of LSEG on Different Layer Ranges

层区间	FOLIO LE (%)	MALLS LE (%)
基线 (无 LSEG)	88.00	89.70
L0-3	83.20	87.50
L4-7	87.92	89.12
L8-11	88.63	89.84
L12-15	88.21	89.55
L16-19	87.44	89.31
L20-23	91.12	90.96
L24-27	89.52	90.20
L28-31	91.08	90.93
完整 LSEG	92.01	91.34

干预。结果如表 3-4 所示。可以看到，在浅层区间施加干预并没有带来明显收益，其中 L0 到 L3 区间甚至使 FOLIO 的 LE 从 88.00% 下降到 83.20%。相比之下，中上层区间的干预效果更为明显，尤其是 L20 到 L23 和 L28 到 L31 两个区间，在两个数据集上都表现出较强增益。不过，从整体结果看，完整 LSEG 仍然优于任一单一区间设置，这说明方法的效果并不只是来自某一固定层段，而是来自分层方向构建与动态调度机制的共同作用。

为了更直观地展示不同层区间上的收益分布，本章进一步将结果可视化，如图 3-3 所示。可以看到，浅层区间上的干预通常难以带来提升，而中上层区间，尤其是 L20 到 L23 和 L28 到 L31，对最终结果的改善更为明显。这一现象说明，逻辑相关信息在不同层上的分布并不是均匀的。浅层表示更接近词汇和局部句法特征，因此如果过早注入逻辑方向，反而可能破坏尚未完成语义整合的表示结构；中上层则更可能承担复杂语义关系组合和逻辑结构建模功能，因此也是逻辑漂移更容易影响最终输出的位置。

综合来看，消融实验说明 LSEG 的有效性并不是由单一因素决定的，而是来自两个部分的共同作用。逻辑校正方向提供了明确的修正依据，熵引导机制则决定了这些方向应在何时、何处以及以何种强度被注入。若缺少前者，方法将失去方向性；若缺少后者，方法又容易退化为粗粒度的统一干预。也正因为如此，完整 LSEG 才能在两个数据集上获得最好结果。

3.3.5 层级熵分析

除结果指标上的提升外，本章还进一步分析了模型在不同层上的平均熵变化，以更直观地理解熵引导机制为什么能够发挥作用。具体来说，本章统计了模

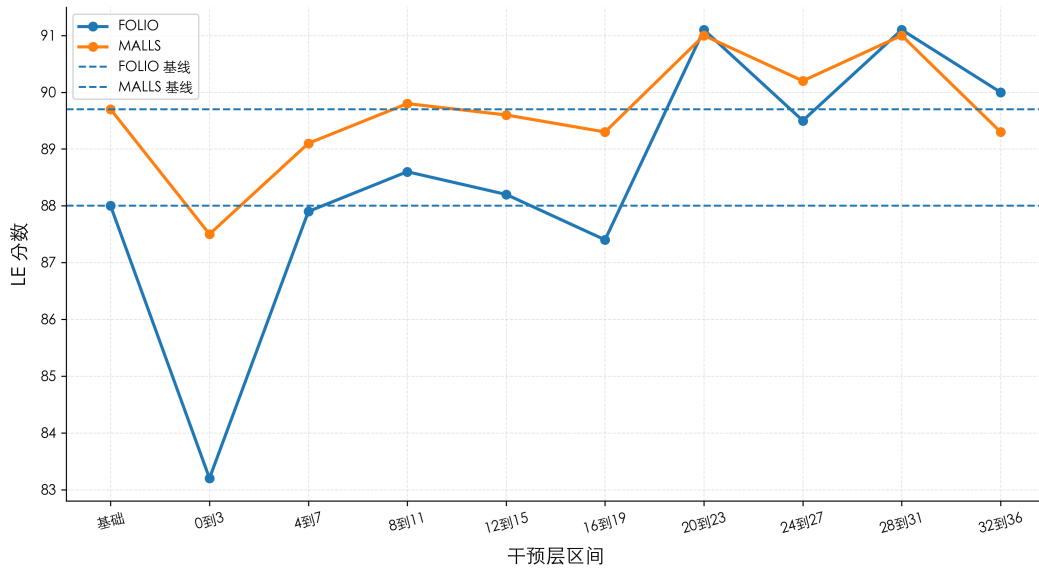


图 3-3 FOLIO 与 MALLS 上不同层区间的 LSEG 干预效果

Figure 3-3 Layerwise Intervention Effects of LSEG on FOLIO and MALLS

型在各层上的平均 token 熵，用以观察从浅层到深层的内部不确定性变化趋势。结果如图 3-4 所示。

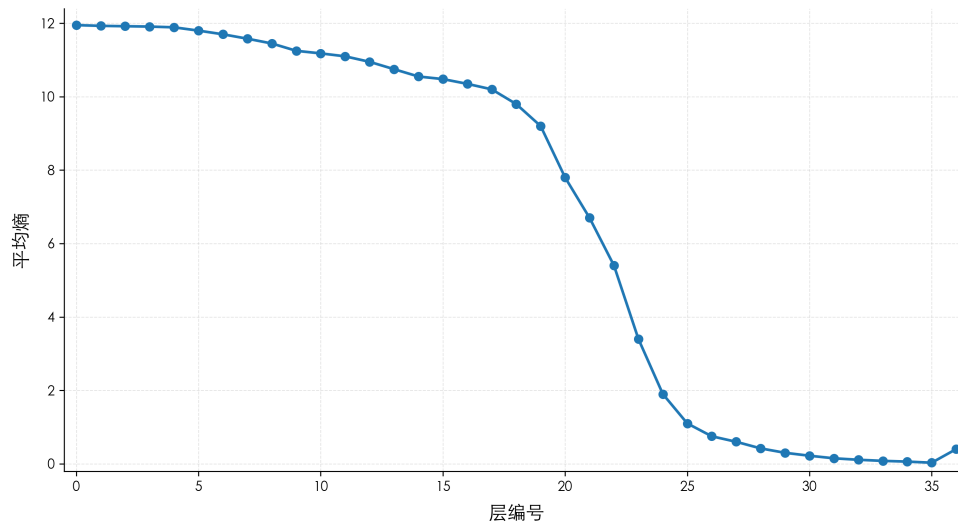


图 3-4 模型各层平均熵分布

Figure 3-4 Average Entropy Distribution Across Model Layers

从图 3-4 可以看到，模型在浅层的平均熵整体较高，且变化相对平缓；随着层数加深，熵值逐渐下降，并在中上层出现更明显的收缩趋势。这说明，模型在前几层仍然处于较为分散的候选状态，更多承担词汇和局部句法信息的编码；而到了中上层之后，内部表示开始逐步收敛，模型对下一 token 的预测也趋于集中。换言之，中上层更可能是复杂语义关系和逻辑结构逐渐成形的关键位置。

这一现象与前面的层区间消融实验结果是相互对应的。消融实验表明，中上层区间上的干预效果明显优于浅层，而层级熵分析则进一步说明，中上层恰好也

是模型内部状态发生快速收缩和重组的位置。也就是说，LSEG 并不是随机地在某些层上起作用，而是恰好作用在模型由表层编码向高层语义整合过渡的关键阶段。这为熵引导机制的合理性提供了更直接的支持。

从方法角度看，层级熵可以被视为一种内部状态变化的代理信号。当某一层相较前一层出现较大的熵波动时，往往说明模型正在经历较明显的语义重组过程。如果这一过程沿着正确逻辑方向进行，那么状态收缩意味着决策逐渐稳定；但如果模型在这一阶段已经偏离原始逻辑结构，那么后续的低熵状态反而可能意味着错误被提前固化。因此，熵变化本身并不等于逻辑正确性，但它能够反映模型在推理过程中的内部不稳定程度，从而为干预时机的选择提供依据。

进一步说，这一分析也有助于解释为什么完整 LSEG 优于固定调度方法。固定调度虽然也可以在若干层上施加校正，但它无法区分哪些层在当前时刻真正处于不稳定状态，因此更容易出现干预不足或过度干预的情况。相比之下，熵引导机制可以根据实际状态变化动态决定干预位置和强度，使逻辑方向注入更贴近模型当下的内部需求。因此，层级熵分析不仅是对实验现象的补充说明，也从内部状态角度解释了方法设计的必要性。

总体来看，层级熵分析为 LSEG 的设计提供了较清晰的经验支持。它表明，显性复杂逻辑结构翻译中的关键问题并不只是输出端的符号选择，而更可能与中上层表示整合过程中的状态波动有关。LSEG 正是利用这一点，将逻辑方向构建与熵引导调度结合起来，从而在不更新模型参数的前提下，对内部表示漂移进行更有针对性的控制。

3.4 本章小结

本章围绕显性复杂逻辑结构下的自然语言到一阶谓词逻辑翻译问题，提出了一种面向推理阶段的增强方法 LSEG。与依赖监督微调或输出后处理的方法不同，LSEG 主要从模型内部表示入手，尝试在不更新参数的前提下，对推理过程中的逻辑漂移进行定向校正。具体来说，本章首先通过逻辑保持扰动样本构建分层逻辑稳定中心，并进一步提取各层的主校正方向；随后，在在线推理阶段结合层级熵变化识别不稳定层，并在这些层上实施自适应方向注入，从而减轻复杂逻辑翻译中的语义偏移问题。

在实验部分，本章分别从主结果、跨模型验证、消融实验以及层级熵分析几个方面对 LSEG 进行了验证。结果表明，该方法在 FOLIO 和 MALLS 两个数据集上都带来了稳定提升。其中，在以 Qwen3 8B 为基础模型的设置下，LSEG 在 FOLIO 上将逻辑等价性从 88.00% 提升到 92.01%，在 MALLS 上从 89.70% 提升到 91.34%。跨模型实验进一步说明，LSEG 并不依赖某一特定基础模型，在 LogicLLaMA 上同样能够带来一致收益。消融实验则表明，方法的效果并不是由单一因素决定的，而是来自分层逻辑校正方向与熵引导调度机制的共同作用。

从结果分析来看，本章方法的有效性说明，显性复杂逻辑结构翻译中的错误并不只是输出端的符号选择问题，也与模型在推理过程中的内部表示稳定性密

切相关。当模型在中上层语义整合阶段发生偏移时，这种偏移会逐步传递到输出端，并最终表现为量词辖域、逻辑连接关系或否定附着位置上的错误。LSEG 的作用，正是在这些潜在偏移尚未完全固化为最终输出之前，对其进行有依据的方向性干预。层级熵分析也进一步支持了这一判断，即中上层既是模型内部状态快速收缩的位置，也是 LSEG 干预最有效的位置。

总体而言，本章提出的 LSEG 方法为显性复杂逻辑结构下的 NL2FOL 翻译提供了一种新的推理阶段增强思路。该方法在无需额外参数训练的条件下，通过对中间表示的分层分析与动态校正，实现了对语义一致性和逻辑稳定性的提升，也为后续研究从表示层面理解和改进逻辑翻译任务提供了进一步参考。

第4章 面向隐性逻辑承载句级的 NL2FOL 技术

4.1 引言

上一章主要讨论了显性复杂逻辑结构文本的自然语言到一阶谓词逻辑翻译问题，重点关注当句子中已经明确出现量词、否定、条件、蕴含以及嵌套结构时，大语言模型在推理过程中容易出现的表示漂移与逻辑偏移现象。对于这类文本而言，逻辑关系本身已经在表层语言中以较为明确的形式呈现出来，因此问题的关键在于，模型能否在推理过程中稳定保持这些显性逻辑结构，并将其准确映射为形式化逻辑表达式。

然而，在更广泛的自然语言场景中，逻辑关系并不总是以显式方式直接出现。大量句级文本虽然表面上缺乏明显的逻辑连接词或标准化逻辑结构，但其内部实际上蕴含着较为清晰的条件约束、因果依赖、范围限制、事件顺序或语义前提。这类逻辑关系往往并不直接体现在表层符号之中，而是分布在词语搭配、句法组织和语义组合之中，需要在更深层次上进行抽取和重构。对于自然语言到一阶谓词逻辑翻译任务而言，这意味着模型面对的不再只是如何稳定翻译已有逻辑结构的问题，而是如何从表层语言中恢复尚未显式展开的逻辑结构的问题。

与显性复杂逻辑结构相比，句级隐性逻辑关系的翻译难点具有不同特点。显性复杂逻辑结构中的困难，主要在于模型是否能够正确处理已经出现的逻辑成分及其组合关系；而句级隐性逻辑中的困难，则在于模型首先需要识别那些尚未显式形式化的关系，再进一步完成逻辑映射。换言之，这里的问题不只是逻辑翻译不稳定，更在于逻辑结构本身尚未被充分显式化。如果直接要求模型从自然语言一步生成一阶谓词逻辑表达式，模型往往容易停留在表层语义重写层面，而忽略句子内部更深层的约束关系，从而导致生成结果虽然在局部上看似合理，但在整体上缺乏完整的逻辑结构支撑。

这一问题提示我们，对于句级隐性逻辑文本，仅仅依赖端到端的自然语言到逻辑表达式映射可能并不是最合适的路径。相比一步到位地生成形式化逻辑表达式，先引入一个更接近自然语言、同时又比自然语言更具结构化特征的中间表示层，可能更有利于隐性逻辑关系的显式化与后续形式化转换。中间表示的引入，本质上是在自然语言与一阶谓词逻辑之间增加一个过渡层，使模型能够先用更易于生成和理解的结构表达句子内部的逻辑骨架，再进一步将其映射为严格的形式逻辑表达式。

在多种可能的中间表示形式中，伪代码具有较好的适配性。一方面，伪代码比自然语言更具结构性，能够较清楚地表示条件、分支、约束、依赖和执行顺序等信息；另一方面，它又不像一阶谓词逻辑那样对符号精度要求极高，因此更容易被大语言模型生成和调整。对于句级隐性逻辑关系而言，伪代码可以被视为一种介于自然语言和形式逻辑之间的桥梁表示。通过这一桥梁，模型可以先将自然语言中的隐含关系显式组织为更清晰的中间结构，再在此基础上完成到一阶谓

词逻辑的映射。这样做不仅有助于缓解直接生成 FOL 时的结构错配问题，也为后续验证和修正提供了更明确的操作对象。

基于上述考虑，本章聚焦于句级隐性逻辑关系的自然语言到一阶谓词逻辑翻译问题，提出一种基于伪代码中间表示的翻译方法。该方法的基本思路是，将句级自然语言输入先转换为伪代码形式的中间表示，对其中的逻辑关系进行显式组织与表达，再通过相应的验证与映射机制，将中间表示进一步转换为一阶谓词逻辑表达式。通过这种多阶段翻译框架，本章希望降低从自然语言直接到形式逻辑的跨度，增强模型对句级隐性逻辑关系的识别能力、结构建模能力以及最终翻译结果的可控性和可验证性。

与直接端到端翻译方法相比，本章方法的特点主要体现在两个方面。其一，它强调隐性逻辑关系的显式化过程，将逻辑结构恢复作为翻译前的重要中间步骤，而不是把全部任务压缩在一次生成之中。其二，它引入中间表示验证机制，使模型生成的伪代码不仅要在表面上可读，还要在结构上支持后续逻辑映射，从而减少逻辑遗漏和结构偏差对最终 FOL 表达式的影响。通过这种方式，本章试图为句级隐性逻辑文本的 NL2FOL 翻译提供一种更适合的建模路径。

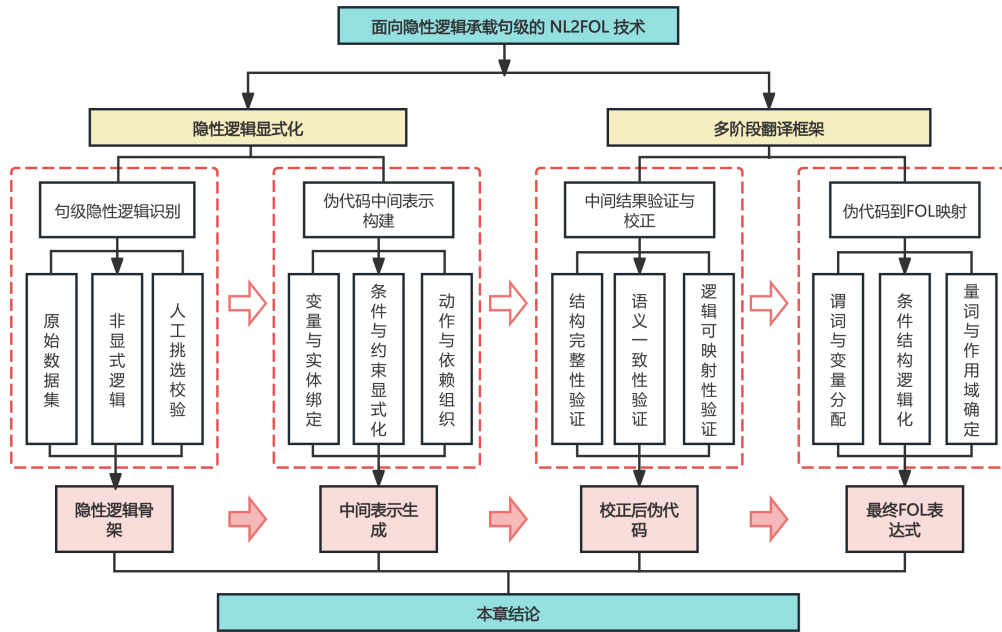


图 4-1 基于伪代码中间表示的句级隐性逻辑翻译框架

Figure 4-1 Pseudocode-Based Framework for Sentence-Level Implicit Logic Translation

本章其余内容安排如下：第 4.2 节介绍本章方法的整体设计，包括问题定义、整体框架、伪代码中间表示构建方法、验证机制以及从伪代码到一阶谓词逻辑的映射过程；第 4.3 节给出实验结果与分析，重点考察中间表示与验证机制对句级隐性逻辑翻译效果的影响；第 4.4 节对本章内容进行总结。

4.2 方法介绍

4.2.1 问题定义与研究动机

本章关注的任务，是面向句级隐性逻辑关系的 NL2FOL 技术。与显性复杂逻辑结构场景不同，这里的输入句子虽然在语义上蕴含较为明确的逻辑约束，但这些约束并不总是通过显式逻辑连接词直接呈现出来。也就是说，模型在这一任务中面对的，不只是将已有逻辑结构映射为形式化表达式的问题，更是先从自然语言中恢复潜在逻辑结构，再完成形式化翻译的问题。

设输入句子为 x ，输出的一阶谓词逻辑表达式为 y 。若沿用第三章中的基本定义，则该任务同样可以表述为条件生成问题，即在给定输入 x 的条件下，生成与原句语义严格对应的一阶谓词逻辑表达式 y 。从形式化角度看，该任务可定义为如下映射过程：

$$y = \arg \max_{y'} p_{\theta}(y' | x) \quad (4-1)$$

然而，对于句级隐性逻辑关系而言，若直接将任务视为从自然语言一步到位生成 FOL 表达式，往往会遇到明显困难。其原因在于，模型面对的并非已经显式展开的逻辑骨架，而是需要先在自然语言内部识别和组织隐含的条件关系、约束关系、依赖关系以及作用范围，再将其进一步形式化。

这种困难主要体现在三个方面。

第一，隐性逻辑关系在表层语言中往往缺乏明确的形式标记。很多句子虽然不包含明显的 if、or 等逻辑信号词，但其语义内部已经包含条件限制、存在约束或范围依赖。例如，一些带有习惯性表达、事件先后关系、能力限制、角色约束或默认前提的句子，在自然语言层面可以是简洁而流畅的，但若要将其转换为 FOL，则必须先识别其中真正起约束作用的隐含成分。

第二，直接生成 FOL 的表示跨度过大。自然语言是高灵活性、上下文依赖较强的表达系统，而一阶谓词逻辑则要求明确的变量、谓词、量词、论元关系以及严格的组合结构。对于句级隐性逻辑文本而言，模型往往尚未完成对逻辑关系的显式组织，就被要求一步映射到形式逻辑，这容易导致两类问题：一类是逻辑关系识别不完整，即模型只抓住了句子的表层语义，而遗漏了真正起约束作用的隐含关系；另一类是逻辑结构映射不稳定，即模型虽然捕捉到部分语义线索，但由于缺乏中间层组织过程，最终输出的 FOL 结构不完整或不一致。

第三，句级隐性逻辑文本中的错误往往难以通过最终输出直接定位。对于显性逻辑结构文本，若模型在量词辖域、连接关系或否定作用范围上出错，通常还能较容易地从最终 FOL 公式中观察到问题所在。但对于隐性逻辑文本而言，如果中间逻辑骨架本身就没有被正确恢复，那么最终 FOL 中呈现出的错误往往只是结果层面的症状，研究者很难判断问题究竟出在逻辑关系识别阶段，还是出在形式映射阶段。因此，仅依赖最终公式进行分析，往往不足以支持有效纠错。

基于上述困难，我们认为，句级隐性逻辑翻译的关键问题并不只是如何生成一个正确的 FOL 公式，而首先是如何让隐含逻辑关系在生成前被显式组织出

来。从这一角度看，在自然语言与一阶谓词逻辑之间引入一个中间表示层，是一种更合理的建模思路。中间表示的作用，并不是替代最终逻辑表达式，而是在两种表示系统之间提供一个过渡结构，使模型先完成隐性逻辑的显式化，再进行形式化映射。

在多种可能的中间表示形式中，本章选择伪代码作为桥梁表示，而非直接采用真实代码。这一选择主要基于以下考虑。首先，本章的目标并不是生成可执行程序，而是借助一种结构化表示显式刻画句子内部的逻辑骨架。真实代码强调语法合法性、执行语义和语言特定约束，而这些内容与本任务关注的逻辑结构恢复并不完全一致。其次，伪代码比自然语言更具结构性，能够较清楚地表示条件、分支、顺序、约束和依赖等信息；同时，它又不像一阶谓词逻辑那样对形式精度要求极高，因此更容易被大语言模型生成和调整。再次，伪代码避免了具体编程语言带来的语法负担，使模型可以把注意力更多集中在逻辑关系的显式化上，而不是程序实现细节上。最后，伪代码还更适合后续的结构验证与逻辑映射，因为其中保留的是与逻辑翻译直接相关的骨架信息，而不是与执行相关的冗余细节。

因此，本章将句级隐性逻辑翻译任务重新组织为一个多阶段过程：首先从自然语言中生成伪代码中间表示，用于显式化原句中的潜在逻辑关系；然后对中间表示进行验证与必要校正，以保证其结构完整性和逻辑可映射性；最后再将经过验证的伪代码映射为一阶谓词逻辑表达式。通过这种方式，原本跨度较大的自然语言到 FOL 翻译，被拆解为隐性逻辑显式化和形式化逻辑映射两个相对更可控的子过程，从而为提升句级隐性逻辑翻译的准确性和稳定性提供了方法基础。

4.2.2 整体框架设计

基于上述分析，本章提出一种基于伪代码中间表示的句级隐性逻辑翻译框架。该框架的核心思想是，在自然语言与一阶谓词逻辑之间增加一个结构化桥梁层，使模型不再被要求直接从自然语言一步生成最终逻辑公式，而是先通过伪代码恢复句子内部的逻辑组织方式，再在此基础上完成形式逻辑映射。相比端到端直接翻译，这一框架更强调逻辑关系的逐步显式化和结构化约束，从而降低隐性逻辑翻译中的信息遗漏与结构错配风险。

从整体流程上看，该方法主要由三个阶段构成：第一阶段是自然语言到伪代码的转换；第二阶段是伪代码中间表示的验证与校正；第三阶段是从伪代码到一阶谓词逻辑的映射。从形式化流程上看，本章方法可以表示为如下多阶段转换过程：

$$x \xrightarrow{f_{pc}} z \xrightarrow{f_{ver}} \hat{z} \xrightarrow{f_{fol}} y \quad (4-2)$$

其中， x 表示自然语言输入， z 表示模型初步生成的伪代码中间表示， $\hat{z}$ 表示经过验证与校正后的伪代码中间表示， y 表示最终输出的一阶谓词逻辑表达式。

在第一阶段，模型的主要任务是从自然语言表层表达中恢复潜在的逻辑关系，并将其组织为结构更清晰的伪代码表示。这里的伪代码并不是为了执行，而是为了显式刻画句子中的条件关系、顺序依赖、作用范围和约束结构。例如，自

然语言中原本隐含的条件触发关系，可以在伪代码中被转写为显式的条件分支；原本分散在描述性语言中的角色约束和对象限制，也可以在伪代码中被统一组织为变量和条件表达。通过这一过程，句级隐性逻辑关系从自然语言中的分散语义，转化为更具结构性的中间形式。

在第二阶段，系统并不直接把伪代码输出视为最终中间结果，而是进一步对其进行验证与必要校正。这样设计的原因在于，虽然伪代码比 FOL 更容易生成，但模型仍然可能在这一阶段出现变量不一致、条件缺失、依赖关系不完整或结构层次不清等问题。如果不对中间表示进行检查，错误就会继续传递到后续 FOL 映射阶段。验证模块的作用，正是在伪代码和最终逻辑表达式之间增加一道结构约束，确保中间必须在逻辑结构上足够完整，并且能够支持后续形式化映射。

在第三阶段，系统将经过验证的伪代码中间表示进一步映射为一阶谓词逻辑表达式。由于伪代码已经把原句中的隐含关系显式组织出来，因此这一阶段的任务重点不再是恢复逻辑关系，而是将已显式化的逻辑结构映射为严格的符号表达。相比直接从自然语言到 FOL 的一步式方法，这种分阶段方式能够显著降低形式化映射时的不确定性，也使最终逻辑公式的构造过程更具可解释性。

4.2.3 伪代码中间表示构建方法

在本章框架中，伪代码中间表示的构建是整个方法的关键起点。该阶段的目标，是将自然语言句子中原本隐含的逻辑关系转写为一种更具结构性的表达形式，使其在保持原始语义核心内容的同时，具备更明确的条件组织、变量指代和约束表达能力。换言之，这一阶段不是简单的自然语言改写，而是一种面向逻辑显式化的结构重构过程。

为了实现这一目标，本章对伪代码中间表示提出三个基本要求。

第一，伪代码应能够显式表达句子中的逻辑控制关系。对于句级隐性逻辑文本而言，很多真正关键的关系并不体现在表层逻辑词上，而隐藏在语义依赖之中。例如，一个看似陈述性的句子，实际可能包含只有在某条件满足时某事件才成立的限制关系。伪代码需要把这类关系转写为显式的条件结构、约束结构或顺序结构，而不是继续保留在模糊的自然语言叙述中。

第二，伪代码应能够保留与形式逻辑映射相关的核心语义单元。这包括实体、属性、动作、关系、条件约束以及它们之间的依赖路径。伪代码不要求像 FOL 那样立即给出精确的量词和符号组合，但至少应当把后续映射所需的基本逻辑组件组织清楚。例如，句子中哪些对象是独立变量，哪些条件控制某个动作成立，哪些事件之间存在先后或依赖关系，这些内容都应在伪代码中得到较明确体现。

第三，伪代码应避免引入与任务无关的实现细节。由于本章选择伪代码而非真实代码，其目的就在于减少具体编程语言的语法负担。因此，构造中间表示时不强调代码可执行性，也不要求采用某一特定编程语言的严格语法，而是优先保留逻辑骨架和结构层次。这种弱语法、强结构的设计，有助于模型把注意力集中

在逻辑组织而非程序实现上。

基于上述要求，本章将伪代码中间表示设计为一种半结构化的逻辑描述形式。形式上，伪代码中间表示可记为：

$$z = \langle B, C, R, D \rangle \quad (4-3)$$

其中， B 表示变量与实体绑定单元， C 表示条件与约束单元， R 表示动作与关系单元， D 表示顺序与依赖单元。它通常由以下几类基本单元组成：

(1) 变量与实体绑定单元。该部分用于标记句子中的核心对象、参与者或事件主体，使后续逻辑映射时能够形成明确的论元结构。

(2) 条件与约束单元。该部分用于显式表达句子中的限制条件、触发条件、适用范围以及默认前提，是隐性逻辑显式化的核心。

(3) 动作与关系单元。该部分用于表示实体之间的行为关系、属性关系或状态关系，使句子中的语义内容能够被组织为后续可映射的谓词结构。

(4) 顺序与依赖单元。当句子中存在事件先后关系、前提依赖或结果依赖时，该部分用于将其显式化，从而避免模型在后续映射中只保留局部语义而遗漏整体结构。

在具体生成过程中，模型接收到自然语言输入后，需要先完成语义分解，再将句子中的核心成分组织为上述结构化单元。与普通文本改写不同，这里强调的是从表层叙述中抽取逻辑骨架。例如，一个自然语言句子在原始形式下可能只是顺畅地描述某个事实，但在伪代码中则需要重新组织为若满足条件 A ，则存在关系 B ；若条件 A 不满足，则关系 B 不成立之类的显式结构。通过这种转写，原本隐含的逻辑依赖就被转化为更易于分析和验证的中间层表达。

从方法角度看，伪代码中间表示的构建可以理解作为一种逻辑重述过程。它既不是普通自然语言摘要，也不是直接符号化，而是在二者之间建立一种结构化重写。其本质作用在于，把句级隐性逻辑关系从难以直接形式化的自然语言状态转化为可以被进一步验证和映射的显式结构状态。因此，这一阶段的输出质量将直接影响后续验证和 FOL 映射的有效性。

此外，伪代码中间表示还具有一个重要优势，即它更适合支持错误定位。对于端到端直接生成 FOL 的方法而言，一旦最终结果出错，研究者往往难以判断错误究竟是出在逻辑关系识别阶段，还是出在形式化阶段。而在伪代码框架下，如果最终 FOL 结果有误，可以先检查伪代码是否已经正确恢复了条件、约束和依赖关系；如果伪代码本身就存在缺失或错配，则问题主要出在隐性逻辑显式化阶段；如果伪代码是正确的，但 FOL 仍然有误，则问题更可能出在映射阶段。由此，中间表示不仅承担桥梁功能，也为后续分析提供了诊断价值。

总体来看，伪代码中间表示构建阶段的核心，不在于让模型生成像程序一样的文本，而在于让模型以更结构化的方式显式表达句子中的隐性逻辑骨架。通过这一阶段，自然语言中的隐性约束关系被转化为更清晰、更可检查、更便于映射的中间形式，从而为后续验证与 FOL 转换奠定基础。

4.2.4 基于验证的中间结果校正机制

在完成伪代码中间表示构建之后，本章并不直接将其送入后续的一阶谓词逻辑映射模块，而是引入一个中间结果验证与校正阶段。设置这一阶段的原因在于，尽管伪代码相较于 FOL 更容易生成，但模型在这一层仍然可能出现多种错误，例如条件表达不完整、变量指代不一致、依赖关系缺失、结构层次混乱，或者伪代码虽然表面上可读，但并不足以支持后续形式化映射。若不在这一阶段进行检查，这些问题将进一步传递到 FOL 生成过程中，并最终表现为逻辑遗漏、结构错配或公式不完整。

从任务目标上看，本章所需要的验证并不是传统意义上的程序语法验证，因为伪代码本身并不要求可执行；本章更关注的是一种面向逻辑结构的中间表示验证。换言之，验证模块需要回答的核心问题不是这段伪代码能否运行，而是这段伪代码是否已经充分、准确地显式表达了原句中的隐性逻辑关系，并且能够作为后续 FOL 映射的可靠输入。因此，本章将验证目标概括为三个层面：结构完整性验证、语义一致性验证和逻辑可映射性验证。

第一，结构完整性验证主要检查伪代码内部是否具备基本的逻辑组织结构。例如，当自然语言句子中存在条件触发关系时，伪代码是否显式给出了条件块与结果块；当句子涉及多个实体和关系时，伪代码是否对这些实体进行了清晰区分；当句子存在多个约束成分时，伪代码是否保留了这些约束，而不是只保留一个局部事实。如果伪代码中出现条件缺失、变量遗漏、约束断裂或结构层级不清等问题，则说明中间表示尚不具备充分的表达能力，需要进一步修正。

第二，语义一致性验证主要考察伪代码是否忠实反映了原始自然语言的核心语义关系。由于伪代码属于中间表示，其形式相对自由，因此模型有时可能会在生成过程中引入与原句不一致的附加解释，或将某些隐性关系错误地显式化。例如，原句中只是表达某行为在特定条件下成立，模型却可能在伪代码中额外加入否则一定不成立的强约束。这类问题虽然不一定表现为结构错误，但会导致后续 FOL 结果在语义上偏离原句。因此，验证模块还需要从原句与伪代码之间的对应关系出发，检查中间表示是否出现语义扩张、语义收缩或关系替换等问题。

第三，逻辑可映射性验证则关注伪代码是否已经具备足够清晰的形式化条件，以支持后续映射到一阶谓词逻辑。对于本章方法而言，并不是所有看起来合理的伪代码都适合直接映射。例如，某些伪代码虽然在描述层面通顺，但仍然保留了较多自然语言式模糊表达，没有把变量关系、条件边界或依赖路径显式展开。这样的中间表示即使在可读性上没有明显问题，也会给后续 FOL 转换带来较大不确定性。因此，本章强调，中间表示不仅要可读，更要可映射，也就是说，它必须在结构上具备足够明确的逻辑支撑。

基于上述目标，本章将验证阶段设计为检查—反馈—校正的过程。具体而言，模型首先生成伪代码中间表示，然后由验证模块从结构、语义和可映射性三个维度进行检查。当发现问题时，系统不会直接丢弃当前结果，而是将相应的错误类型或不足信息反馈给生成模块，由模型根据反馈对伪代码进行修正。形式

上, 该过程可表示为:

$$\hat{z} = \begin{cases} z, & \text{if } z \text{ passes verification} \\ f_{\text{corr}}(z, x), & \text{otherwise} \end{cases} \quad (4-4)$$

其中, $f_{\text{corr}}(\cdot)$ 表示在原始自然语言输入 x 的约束下, 对伪代码中间表示 z 进行反馈式校正的过程。

这样做的原因在于, 句级隐性逻辑关系的显式化本身就带有一定的渐进性, 很多时候模型的第一次生成并非完全错误, 而是部分正确但结构不完整。在这种情况下, 直接重生成往往会造成额外不稳定, 而基于反馈的局部修正更有利于保留已有正确内容并补足缺失部分。

从方法作用上看, 引入验证机制至少带来两方面收益。一方面, 它可以阻断错误向后续 FOL 阶段传播。因为一旦伪代码本身存在问题, 后续映射通常只能把这些问题以更形式化的方式固化下来。另一方面, 它也为本章方法提供了更强的中间过程可控性。相较于端到端直接生成 FOL, 研究者现在可以在伪代码层观察模型究竟已经恢复了哪些隐性逻辑关系, 哪些关系仍然缺失, 以及校正后哪些结构发生了改善。这使得整个翻译过程不仅更稳, 也更易分析。

4.2.5 从伪代码到一阶谓词逻辑的映射方法

在经过验证与必要校正之后, 伪代码中间表示已经完成了对原始自然语言中隐性逻辑关系的初步显式化。此时, 后续任务的重点不再是从表层语言恢复潜在逻辑骨架, 而是将已经显式组织好的逻辑结构映射为严格的一阶谓词逻辑表达式。与直接从自然语言到 FOL 的端到端方法相比, 这一阶段的任务边界更加清晰, 因为模型面对的输入不再是模糊且多义的自然语言描述, 而是一个已经显式化了条件、约束、依赖与关系结构的中间表示。

我们认为, 从伪代码到一阶谓词逻辑的映射, 本质上是一种结构约束下的形式化转换过程。因此, 该阶段可记为如下映射过程:

$$y = g(\hat{z}) \quad (4-5)$$

其中, $g(\cdot)$ 表示从经过验证的伪代码中间表示到一阶谓词逻辑表达式的形式化映射函数。

由于伪代码已经把句级隐性逻辑关系组织为更清晰的结构单元, 因此在这一阶段, 系统主要需要完成以下几项工作: 其一, 识别伪代码中的实体、变量和关系, 并为其分配适当的逻辑符号; 其二, 将伪代码中的条件结构、分支结构、顺序依赖和约束表达映射为相应的逻辑连接关系; 其三, 确定变量作用范围及必要的量化方式, 从而得到语义完整的一阶谓词逻辑表达式。

从映射粒度上看, 本章将伪代码中的基本结构分为几类典型单元, 并分别对应到 FOL 中的不同表达方式。对于实体和对象单元, 主要映射为常量或变量; 对于属性和状态单元, 主要映射为一元谓词; 对于动作和关系单元, 主要映射为

二元或多元谓词；对于条件结构，则通常映射为蕴含或条件约束；对于存在性或适用范围信息，则需要进一步转化为全称量词或存在量词。通过这一分层映射，原本较接近自然语言的伪代码表示，被逐步转写为形式逻辑中具有严格语义定义的表达式。

在这一过程中，伪代码的一个关键价值在于，它提前为量化和结构绑定提供了中间支撑。对于句级隐性逻辑文本，若直接从自然语言生成 FOL，模型往往容易在变量指代、条件辖域和约束边界上出现不稳定。而伪代码已经把原始句子中的结构关系显式组织出来，因此这些问题在映射阶段会变得更易处理。例如，当伪代码中已经明确写出某条件控制某关系成立时，系统在映射到 FOL 时就更容易确定应使用蕴含结构而不是简单合取；当伪代码中已经把参与者和对象关系区分清楚时，系统也更容易分配谓词参数位置并保持论元一致性。

为了更直观地说明这一过程，本章给出一个简化示例。设自然语言句子为：

Only students who submit the assignment can pass the course.

对于这一句子，若直接端到端生成 FOL，模型往往容易只保留 *student*、*submit* 和 *pass* 等局部关系，而忽略 *only* 所引入的必要条件约束。本章方法首先将其转化为伪代码中间表示，例如：

```
for each person:
  if person passes the course:
    person is a student
    person submits the assignment
```

在这一伪代码中，原句中的隐性约束关系已经被显式组织为一个条件结构：通过 *if person passes the course* 引出后续必要条件，从而更清楚地表达“能通过课程的人必须满足两个前提”。在此基础上，系统再将其映射为一阶谓词逻辑表达式：

$$\forall x (Pass(x) \rightarrow (Student(x) \wedge SubmitAssignment(x))) \quad (4-6)$$

如公式 (4-6) 所示，伪代码在这里起到的作用，并不是直接替代 FOL，而是把原句中 *only* 所隐含的必要条件关系转化为更明确的中间结构，使后续逻辑映射更稳定、更可控。

从方法层面看，这种映射过程具有两个优势。第一，它降低了最终公式生成中的不确定性。因为模型不再需要在一步之内同时解决理解隐含关系和生成形式逻辑两个问题，而是只需在已经显式化的伪代码基础上进行符号化。第二，它提高了错误分析的可定位性。如果最终 FOL 公式存在问题，可以回溯检查伪代码是否已经正确表达了条件、依赖和变量关系，从而判断问题究竟出在隐性逻辑恢复阶段，还是出在形式化映射阶段。

需要指出的是，本章并不要求伪代码与 FOL 之间是一一固定、机械僵化的模板映射关系。相反，不同句子在具体表达形式上可能存在差异，因此映射过程本身仍然需要一定的语义判断与结构适配能力。但相较于直接从自然语言生成 FOL，这种判断发生在一个更清晰、更受约束的输入空间中，因此整体难度和不稳定性都会明显降低。

总体而言，从伪代码到一阶谓词逻辑的映射阶段，是本章框架中的形式化收束步骤。经过前面的显式化和验证，中间表示已经把自然语言中的隐性逻辑关系转化为较为清晰的结构骨架；而在这一阶段，这一骨架被进一步组织为严格的形式逻辑表达式，从而完成从自然语言到 FOL 的整体转换。

4.2.6 方法分析与讨论

从整体上看，本章方法的有效性主要建立在一个基本判断之上：对于句级隐性逻辑关系，困难并不只是出在形式逻辑本身太复杂，而更在于自然语言中的逻辑骨架在表层上没有被充分显式化。若在逻辑关系尚未被明确组织之前，直接要求模型一步生成一阶谓词逻辑表达式，模型很容易停留在表层语义重写层面，或者只生成部分关系，而遗漏真正决定逻辑结构的隐含约束。本章方法通过在自然语言与 FOL 之间加入伪代码中间表示层，把隐性逻辑恢复和形式逻辑映射拆分开来，从而降低了单步生成带来的结构压力。

从表示层面看，伪代码之所以适合作为桥梁，在于它同时具备两种特征。一方面，它比自然语言更结构化，能够更清楚地表达条件、分支、依赖和作用范围，因此有利于隐性逻辑的显式化；另一方面，它又不像 FOL 那样要求严格的符号精度和量化形式，因此更容易被大语言模型生成和调整。正是这种介于自然语言与形式逻辑之间的中间特性，使得伪代码能够承担桥梁表示的角色。

从方法流程上看，本章框架也具有较强的分阶段可控性。自然语言到伪代码阶段负责恢复隐性逻辑骨架，验证阶段负责检查和校正中间结构，伪代码到 FOL 阶段负责完成形式化收束。相较于一步式端到端方法，这种分阶段框架的优势在于，每一阶段的任务边界更加清晰，也更容易定位错误来源。如果最终输出存在问题，研究者可以回溯判断是隐性逻辑没有被正确显式化，还是中间表示虽然正确但在形式化映射时发生了偏差。这使得整个系统不仅更稳，也更便于分析和改进。

此外，本章方法还体现出一定的可解释性优势。端到端直接生成 FOL 时，读者通常只能看到自然语言输入与最终逻辑公式之间的跳跃式转换，中间推理过程不透明。而引入伪代码之后，系统在中间层就显式呈现出条件关系、变量关系和约束组织方式，从而为模型输出提供了一种更可观察的结构解释。对于句级隐性逻辑任务而言，这一点尤为重要，因为该任务最大的难点之一就是逻辑关系本身在原句中并不显眼，而中间表示恰好为这一部分提供了外显抓手。

当然，本章方法也有其前提和边界。首先，伪代码中间表示的质量会直接影响后续验证和映射结果。如果中间表示阶段已经遗漏了关键约束，那么后续步骤

很难完全补救。其次，验证模块虽然能够提高中间结果的结构完整性，但它依然主要依赖所定义的检查维度和反馈形式，因此其效果会受到验证规则设计的影响。再次，从伪代码到 FOL 的映射虽然比直接从自然语言到 FOL 更稳定，但并不意味着这一阶段不存在歧义，尤其当伪代码本身仍保留部分自然语言式模糊表达时，映射过程依然需要额外判断。

总体而言，本章提出的方法并不是单纯增加一个中间步骤，而是围绕句级隐性逻辑关系的特点，对自然语言到一阶谓词逻辑翻译流程进行重新组织。它的核心价值在于，通过伪代码中间表示显式恢复隐性逻辑骨架，并结合验证与分阶段映射机制，提高模型在句级隐性逻辑翻译任务中的结构建模能力和结果可控性。

4.3 实验结果与分析

4.3.1 实验数据

为验证本章所提出方法在句级隐性逻辑关系翻译任务中的有效性，本章从 NL2FOL 基准数据集的基础上进行进一步筛选与扩展，从而得到实验数据。具体而言，本章以 FOLIO 和 MALLS 作为原始数据来源，并从中筛选出符合句级隐性逻辑特征的样本，即句子不包含显式逻辑提示，但是具有逻辑关系。在此基础上进一步补充伪代码中间表示标注，最终构成本章实验所使用的句级隐性逻辑数据集约 2000 条。

采用这一数据构造方式，主要出于两方面考虑。首先，FOLIO 和 MALLS 本身已经提供了较为成熟的自然语言与一阶谓词逻辑配对数据，因此可以为本章研究提供可靠的基础语义对齐框架。其次，本章关注的重点并不是一般意义上的 NL2FOL 翻译，而是其中一类更具体的场景，即句级隐性逻辑关系的翻译。若直接使用原始数据集中的全部样本，则难以突出本章任务与显性复杂逻辑结构翻译之间的区别。因此，有必要在现有基准之上进一步筛选样本，并围绕本章方法补充中间表示层的数据。

在样本筛选过程中，本章主要依据以下原则识别句级隐性逻辑样本。

第一，样本在表层表达中不应大量依赖显性逻辑触发词。也就是说，句子不以典型逻辑连接词或显性量化结构作为主要逻辑载体，例如不以明显的全称量词、条件连接词或标准化逻辑模板直接呈现其逻辑关系。这里的目的，并不是绝对排除所有逻辑信号，而是避免样本过度接近第三章所讨论的显性复杂逻辑结构场景。

第二，样本内部应蕴含较明确的句级逻辑约束关系。这类约束关系可以表现为资格限制、条件依赖、角色要求、事件前提、范围限制、顺序约束或隐含规则等。换言之，虽然逻辑关系在表层不完全显式，但句子内部应当存在可以被恢复和形式化的逻辑骨架，而不是仅仅包含松散的语义联想。

第三，样本应具有较明确的一阶谓词逻辑映射可能性。由于本章的目标仍然是句级自然语言到 FOL 的翻译，因此被筛选的句子应能够在经过适当显式化后，

映射为结构较为清晰的一阶谓词逻辑表达式。对于那些高度依赖篇章上下文、强常识补全或复杂跨句推理才能成立的样本，则不纳入本章数据范围，而更适合留待后续篇章级研究讨论。

在完成样本筛选之后，本章进一步为每个句级隐性逻辑样本补充伪代码中间表示标注，形成自然语言句子—伪代码中间表示—一阶谓词逻辑表达式的三元组结构数据。与原始 NL-FOL 配对数据相比，这一步扩充使得本章实验数据不仅可以用于评估最终 FOL 翻译结果，还可以用于分析伪代码中间表示是否有效恢复了句子内部的隐性逻辑关系。换言之，本章的数据构造不仅服务于最终结果评估，也服务于中间过程分析。

在伪代码标注过程中，本章遵循与方法设计一致的基本原则。其一，伪代码应优先显式表达句子中的条件、约束和依赖关系，而不是简单重写自然语言表述。其二，伪代码应保留后续 FOL 映射所需的核心对象、关系和结构信息。其三，伪代码不追求真实编程语言的可执行性，而强调逻辑结构的清晰性与中间表示的可验证性。通过这一补充标注过程，本章最终形成了一个专门面向句级隐性逻辑翻译的实验子集。

从研究设计角度看，这种数据构造方式具有较明显的合理性。一方面，它建立在现有公认基准之上，因此保证了基础语义对齐和评价体系的可靠性；另一方面，它又通过任务特定筛选和中间表示扩充，形成了能够体现本章研究重点的数据形态。由此，本章实验既能够与已有 NL2FOL 研究保持可比性，又能够突出句级隐性逻辑翻译这一更具体的问题设置。

4.3.2 实验设置

在实验设置上，本节主要从三个方面对所提方法进行评估，分别是：不同模型条件下的整体翻译效果、各组成模块对最终性能的贡献，以及典型样本上的中间过程表现。与第三章主要关注最终逻辑等价性不同，本章除了考察最终 FOL 结果之外，还结合消融实验与案例分析，进一步讨论伪代码中间表示和验证机制在句级隐性逻辑翻译中的作用。

首先，在中间表示层评估方面，本章主要考察模型生成的伪代码是否能够较完整地表达句子中的核心逻辑骨架。由于伪代码本身并不是最终形式逻辑表达式，因此这一阶段更关注结构完整性、语义一致性和逻辑可映射性。例如，模型是否显式写出了原句中隐含的条件关系，是否保留了关键对象与约束信息，是否将原句中的依赖关系组织为更清晰的中间结构。这一部分评估的目的，并不是单纯判断伪代码写得像不像代码，而是判断其是否在逻辑上起到了桥梁表示的作用。

其次，在最终结果评估方面，本章仍然使用与第三章一致的逻辑等价性作为核心评价维度，即考察模型生成的一阶谓词逻辑表达式是否与参考逻辑表达式在语义上等价。这一设置保证了第四章实验结果与前文研究之间具有可比性，也使本章可以更清楚地回答一个问题：引入伪代码中间表示和验证机制之后，句

级隐性逻辑关系的最终 FOL 翻译是否得到改善。

再次，在方法有效性评估方面，本章还将通过对比不同设置来分析各个模块的作用。具体而言，实验将至少包含以下几类对比：端到端直接从自然语言生成 FOL 的基线方法；引入伪代码中间表示但不进行验证的两阶段方法；引入伪代码中间表示并结合验证机制的完整方法。通过这种逐步递进的对比，可以更清楚地区分中间表示本身带来的收益和验证机制进一步带来的收益。

从模型流程上看，本章实验中的完整方法由三个步骤构成。第一步，模型根据自然语言输入生成伪代码中间表示；第二步，验证模块对中间表示进行结构检查和必要校正；第三步，将经过验证的伪代码映射为最终一阶谓词逻辑表达式。对应地，基线方法则省略部分中间过程，例如直接端到端生成 FOL，或只保留伪代码生成但不引入验证机制。通过这些对比，本章能够更系统地考察各模块对最终翻译质量的影响。

需要说明的是，由于本章研究重点在于方法框架和中间表示机制，因此实验设置不仅关注最终准确率，还关注中间过程是否提升了系统的可分析性与可控性。换言之，即便某些样本上的最终 FOL 结果与基线相近，只要伪代码中间表示使隐性逻辑结构变得更清晰、更容易定位错误，本章方法仍然具有重要价值。这一点也是第四章与传统端到端翻译研究的重要区别。

4.3.3 整体实验结果

为考察本章所提出方法在不同基座模型上的适用性，本节在多个主流大语言模型上进行实验，对比直接翻译基线与完整方法在句级隐性逻辑翻译任务上的表现。这里的完整方法仍采用本章提出的多阶段翻译框架，即先将自然语言输入转换为伪代码中间表示，再经过中间结果验证与校正，最后映射为一阶谓词逻辑表达式。与前文主要基于单一模型分析方法有效性不同，本节进一步关注该框架是否具有跨模型的稳定收益。

表 4-1 给出了不同模型上的实验结果。从结果可以看出，在各个模型设置下，完整方法均优于直接翻译基线，表明该框架在不同模型上均表现出一定的稳定性。其中，Qwen3 8B 在两种设置下均取得了相对最优结果，表明较强的基础语义理解与生成能力有助于句级隐性逻辑关系的恢复与形式化映射；而在参数规模相对较小或结构能力稍弱的模型上，完整方法同样带来了较明显提升，这说明本章方法不仅能够增强高性能模型的逻辑翻译稳定性，也能够一定程度上缓解中等规模模型在端到端直接翻译中的结构遗漏问题。

进一步比较不同模型上的提升幅度可以发现，完整方法在不同模型上均带来了较为稳定的性能增益，说明该方法并不依赖于某一特定模型架构，而具有一定的跨模型适用性。从表 4-1 可以看出，无论是在 Qwen、Llama，还是在 Mistral、Gemma 和 InternLM 等不同系列模型上，完整方法均优于直接翻译基线，且提升幅度整体较为接近。这表明，句级隐性逻辑翻译中的主要困难并不只来自模型参数规模或通用语言建模能力的差异，更关键的问题在于模型能否准确识别自然

表 4-1 不同模型上的句级隐性逻辑翻译结果

Table 4-1 Results on Sentence Level Implicit Logic Translation across Different Models

模型	直接翻译基线 (LE%)	完整方法 (LE%)	提升值
Qwen2.5 7B	80.96	86.84	+5.88
Llama3.1 8B	78.73	84.91	+6.18
Qwen3 8B	83.41	89.62	+6.21
Mistral 7B	77.85	83.46	+5.61
Gemma 2 9B	79.62	85.57	+5.95
InternLM 2.5 7B	78.94	84.72	+5.78

语言中隐含的条件约束、资格限制与结构关系。

其中，一些基础模型在直接翻译设置下更容易出现条件遗漏、约束关系弱化或逻辑结构组织不完整等问题。这在一定程度上说明，当模型需要直接从自然语言一步生成一阶谓词逻辑表达式时，句级隐性逻辑关系所带来的表示跨度仍然较大。相比之下，完整方法通过引入伪代码中间表示，并结合验证机制，能够先对原句中的潜在逻辑骨架进行显式化组织，再完成向一阶谓词逻辑的映射，从而有效缓解直接翻译过程中的结构偏移问题。

另一方面，像 Qwen3 8B 这类本身具有较强指令跟随能力和结构化生成能力的模型，在直接翻译设置下已经能够取得相对较好的结果，但完整方法仍然能够在此基础上进一步提升逻辑等价性。这说明本文方法不仅能够弥补相对较弱模型在逻辑组织上的不足，也能够对性能较强的模型形成有效补充。整体来看，该结果表明本文方法具有较好的模型无关性，能够作为一种相对通用的句级隐性逻辑翻译增强框架迁移到不同开源大语言模型上。

4.3.4 消融实验

为了进一步分析本章方法中各组成部分的实际作用，本章设计了消融实验，重点考察两个核心问题：其一，伪代码中间表示是否确实对句级隐性逻辑关系的显式化起到了关键作用；其二，验证与校正机制是否能够进一步减少中间表示中的结构错误，并提升最终一阶谓词逻辑翻译结果的稳定性。

具体而言，本章使用 Qwen3 8B 模型进行实验，并设置以下几种变体进行比较：

完整方法：即本章提出的完整框架，包括自然语言到伪代码中间表示的生成、中间结果验证与校正，以及从伪代码到一阶谓词逻辑的映射。

去掉伪代码中间表示：在该设置中，模型不再经过伪代码中间层，而是直接从自然语言生成一阶谓词逻辑表达式。该变体用于考察中间表示层本身的贡献。

去掉验证模块：在该设置中，模型仍然先生成伪代码中间表示，再将其映射

为一阶谓词逻辑表达式，但不再对伪代码进行中间结果验证与校正。该变体用于分析验证模块在减少错误传播方面的作用。

从机制上看，去掉伪代码中间表示后，模型重新退回到端到端直接翻译路径，需要在一步生成中同时完成隐性逻辑恢复和形式逻辑表达，因而更容易遗漏原句中的潜在条件约束、资格关系或事件依赖结构。相比之下，完整方法通过引入伪代码中间层，将原本隐含的逻辑骨架先显式组织出来，再进行形式化映射，因此更有利于保持逻辑结构的完整性。

另一方面，去掉验证模块后，虽然伪代码中间表示仍然能够提供结构化过渡层，但中间表示本身可能存在条件表达不完整、变量关系不一致或逻辑可映射性不足等问题。这些问题若不在中间阶段被及时发现和修正，就会进一步传递到最终 FOL 生成阶段。因此，验证模块虽然不是本章框架中最基础的组成部分，却在提升整体稳定性方面发挥了关键作用。

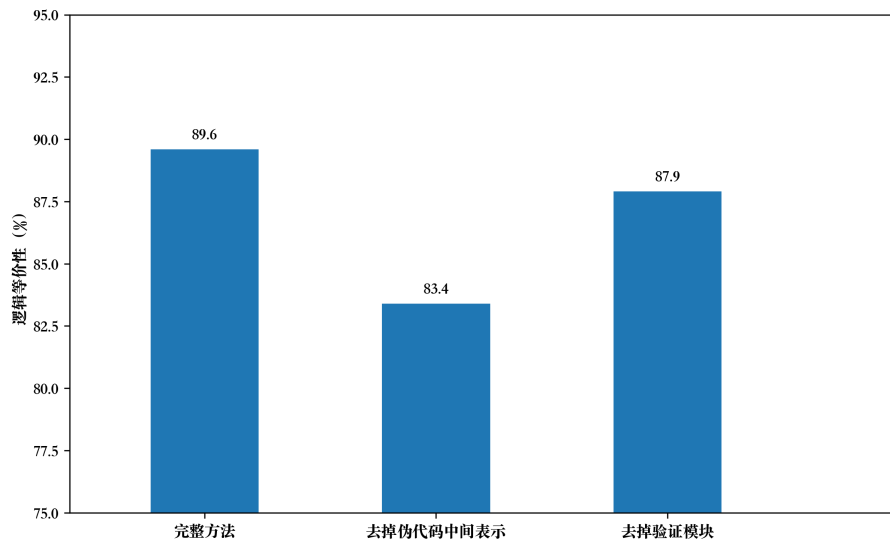


图 4-2 消融实验结果

Figure 4-2 Ablation Results of the Proposed Method

图 4-2 进一步展示了消融实验结果的可视化对比。可以看出，完整方法始终取得最优结果；去掉伪代码中间表示后，模型性能下降最为明显；去掉验证模块后，结果虽仍优于直接翻译基线，但相较完整方法已有明显回落。这一现象表明，伪代码中间表示是本章方法能够有效处理句级隐性逻辑关系的核心组成，它直接决定了模型能否先恢复并组织原句中的隐性逻辑骨架；而验证模块则在此基础上进一步提高了中间表示的结构完整性与后续形式化映射的稳定性。两者并非替代关系，而是前后衔接、相互补充的关系。

4.3.5 案例分析

为了更直观地展示本章方法在句级隐性逻辑关系翻译中的作用，下面给出一个典型示例进行分析。设输入句子为：

Students with a valid ID may enter the lab.

该句在表层上并不包含典型的显性逻辑连接词，但其内部隐含着较清晰的资格约束关系，即能够进入实验室的对象需要同时满足是学生和持有有效证件两个条件。对于这类句子，端到端直接翻译方法容易只保留其中的局部关系，例如仅捕捉 student valid ID 或 enter the lab 等表层语义成分，而未能将它们组织为清晰的条件结构。

表 4-2 展示了该句在不同方法下的输出结果。

表 4-2 句级隐性逻辑翻译案例分析

Table 4-2 Case Study of Sentence Level Implicit Logic Translation

环节	内容
自然语言输入	Students with a valid ID may enter the lab.
直接翻译结果	$\forall x (Student(x) \wedge HasValidID(x) \wedge MayEnterLab(x))$
伪代码中间表示	for each person: if person is a student and person has a valid ID: person may enter the lab
最终 FOL 结果	$\forall x ((Student(x) \wedge HasValidID(x)) \rightarrow MayEnterLab(x))$

从案例结果来看，直接翻译基线虽然捕捉到了句子中的几个核心语义成分，但在逻辑组织上存在明显问题。这样得到的逻辑表达式虽然形式上像一个公式，但并未真正表达原句所蕴含的条件结构。

相比之下，本章方法首先生成伪代码中间表示，将原句中的隐性逻辑关系显式组织为条件块。在这一中间表示中，原本分散在自然语言中的两个前提条件被明确绑定到进入实验室这一结果关系之上，从而把句级隐性逻辑骨架清晰地表达出来。在此基础上，系统进一步将其映射为最终一阶谓词逻辑表达式：

$$\forall x ((Student(x) \wedge HasValidID(x)) \rightarrow MayEnterLab(x)) \quad (4-7)$$

如公式 (4-7) 所示，完整方法生成的 FOL 不再只是对原句表层语义成分的堆叠，而是准确表达了原句中隐含的资格约束关系。这个案例表明，本章方法的关键并不只是生成一个最终正确的 FOL 公式，而是通过伪代码中间表示把原句中原本不够显式的逻辑骨架恢复出来，再借助验证与映射机制将其组织为形式逻辑表达。

进一步来看，这一案例也体现了中间表示和验证机制的互补作用。若只依赖直接翻译，模型很可能在最终输出中遗漏某个资格条件，或者把多个条件简单并

列为事实。若只生成伪代码但不进行验证，则模型仍有可能在条件表达上不够完整，例如只保留 `student` 而遗漏 `valid ID`，或者在结构上没有明确条件—结果关系。而完整方法通过中间表示显式化和验证机制约束，能够更稳定地恢复句子中的隐性逻辑约束，并把这一结构保留到最终 FOL 映射结果中。

4.3.6 方法讨论

综合以上实验结果可以看出，本章方法对于句级隐性逻辑关系翻译的改进，主要体现在两个层面。其一，它提高了模型恢复隐性逻辑结构的能力，使句子内部原本不够显式的条件与约束关系得以清晰表达。其二，它增强了翻译过程的中间可分析性和结果可控性，使研究者能够更明确地区分错误究竟发生在逻辑显式化阶段，还是发生在形式逻辑映射阶段。

从整体实验结果来看，伪代码中间表示的引入带来了最主要的性能提升，这说明本章提出的桥梁表示层确实是处理句级隐性逻辑关系的关键。相比自然语言，伪代码更有利于组织隐性条件、资格约束和依赖关系；相比一阶谓词逻辑，伪代码又保留了较强的生成友好性和结构可读性。因此，它在自然语言与形式逻辑之间形成了一个更平衡的过渡层。

从消融实验分析来看，验证模块虽然带来的提升幅度小于中间表示层本身，但依然发挥了稳定而重要的作用。它能够进一步减少中间表示中的结构遗漏和约束缺失问题，从而使最终 FOL 结果更加完整。这说明，本章方法的优势并不只是多加了一步中间表示，而是通过中间表示与结构验证的组合，在翻译流程中建立起了一种更受约束的逻辑恢复机制。

当然，本章方法也存在一定局限。首先，它依赖于伪代码中间表示生成质量，若模型在第一阶段未能较好恢复隐性逻辑骨架，则后续验证和映射的空间也会受到限制。其次，验证模块的效果在一定程度上取决于检查维度和反馈设计，若验证规则定义不充分，某些类型的结构错误仍可能被保留。再次，本章主要聚焦句级隐性逻辑关系，对于那些需要跨句上下文才能恢复完整逻辑关系的样本，本章方法的适用范围仍然有限。这些问题也为下一章进一步讨论篇章级隐性逻辑翻译提供了自然过渡。

4.4 本章小结

本章围绕句级隐性逻辑关系的自然语言到一阶谓词逻辑翻译问题，提出了一种基于伪代码中间表示的多阶段翻译方法。与第三章面向显性复杂逻辑结构的推理阶段增强方法不同，本章关注的核心问题不再是如何稳定保持已经显式出现的逻辑结构，而是如何从表层不带明显逻辑形式的自然语言句子中恢复潜在的逻辑骨架，并进一步完成形式化映射。

首先，本章分析了句级隐性逻辑关系翻译的主要困难，指出该任务的关键不只是最终 FOL 生成是否正确，更在于模型能否在翻译之前充分显式化原句中的条件关系、资格约束、依赖结构和范围限制。为了解决这一问题，本章在自然语

言与一阶谓词逻辑之间引入伪代码中间表示层，将原本一步完成的翻译任务拆分为隐性逻辑显式化和形式逻辑映射两个相对更可控的子过程。

其次，在方法设计上，本章构建了一个由伪代码中间表示生成、中间结果验证与校正、以及伪代码到一阶谓词逻辑映射组成的整体框架。其中，伪代码中间表示用于显式组织句子中的逻辑骨架，验证模块用于检查中间表示在结构完整性、语义一致性和逻辑可映射性方面的问题，而形式化映射模块则在此基础上进一步生成最终的一阶谓词逻辑表达式。通过这种设计，本章方法在保持翻译过程可解释性的同时，也增强了中间过程的可控性和结果的稳定性。

再次，在实验部分，本章基于 FOLIO 和 MALLS 数据集筛选并构造了句级隐性逻辑实验子集，并进一步补充伪代码中间表示标注，形成适用于本章任务的数据形式。实验结果表明，相较于端到端直接翻译方法，引入伪代码中间表示后，模型在句级隐性逻辑翻译任务上的逻辑等价性结果得到明显提升；在进一步加入验证与校正机制后，完整方法取得了最优表现。消融实验分析进一步说明，伪代码中间表示是提升句级隐性逻辑翻译效果的核心因素，而验证模块则在减少错误传播、提高结构完整性方面发挥了重要补充作用。

总体而言，本章研究表明，对于句级隐性逻辑关系，直接从自然语言一步生成一阶谓词逻辑表达式并不是最稳妥的路径。相比之下，通过引入伪代码中间表示对隐性逻辑骨架进行显式恢复，并结合验证机制进行中间结构约束，能够更有效地缓解句级隐性逻辑翻译中的结构遗漏和逻辑偏差问题。这一结果说明，在自然语言到形式逻辑翻译任务中，中间表示不仅是一种技术手段，也是一种重要的建模思路。上述研究为后续进一步处理更复杂的篇章级隐性逻辑关系奠定了方法基础。

第5章 面向隐性逻辑承载篇章级的 NL2FOL 技术

5.1 引言

前一章主要讨论了句级隐性逻辑关系的自然语言到一阶谓词逻辑翻译问题，重点关注在单句范围内如何借助中间表示恢复原本未显式展开的逻辑骨架。该章研究表明，当逻辑关系主要隐藏在单句内部时，通过在自然语言与一阶谓词逻辑之间增加一个结构化桥梁层，可以在一定程度上缓解直接翻译过程中存在的逻辑遗漏、结构错配和结果不稳定等问题。

然而，在更真实的自然语言场景中，逻辑关系并不总是局限于单个句子内部。大量篇章级文本中的前提、限制、结论和适用范围往往分散在多个句子之中，句与句之间通过指代、省略、承接、补充说明和事件推进等方式共同构成完整语义。对于这类文本而言，某个句子单独看时往往只呈现局部事实或表层动作，而真正决定其逻辑含义的条件来源、对象身份或规则依据，可能需要结合前后文才能完整恢复。因此，篇章级隐性逻辑翻译的关键难点，不再只是句内逻辑关系不够显式，而是逻辑信息本身具有明显的跨句分布特征。

与前两章所讨论的问题相比，篇章级隐性逻辑翻译具有更高的复杂性。对于显性复杂逻辑结构文本，主要困难在于模型能否稳定保持表层已经出现的逻辑成分及其组合关系；对于句级隐性逻辑文本，主要困难在于模型能否在单句内部恢复尚未显式化的逻辑骨架。而在篇章级场景下，模型还需要进一步解决跨句语义整合问题，即如何在多个句子之间追踪实体、连接事件、恢复约束来源，并将分散的信息组织为统一的形式逻辑表达。若仍然仅依赖单句输入，即使模型能够较处理好句内逻辑，也往往难以获得完整的逻辑翻译结果。

这一问题说明，对于篇章级隐性逻辑关系，仅依赖推理阶段局部增强或单句中间表示通常是不够的。因为这类任务所要求的核心能力，已经从句内结构恢复进一步扩展为篇章级上下文建模能力。换言之，模型需要在训练阶段就学习如何利用跨句上下文恢复潜在逻辑关系，而不仅仅是在推理阶段对已有输出进行修正。基于这一考虑，本章将研究重点转向监督微调方法，希望通过构造篇章级隐性逻辑数据，使模型直接学习从篇章上下文到一阶谓词逻辑表达式的映射能力。

为此，本章提出一种基于监督微调的篇章级隐性逻辑翻译方法。该方法以 Qwen3 0.6B 作为基础模型，围绕篇章级隐性逻辑关系构造任务特定数据，并通过监督微调使模型学习跨句信息整合、上下文约束恢复和篇章级形式化表达能力。与前两章不同，本章不再重点依赖推理阶段表示干预或句级中间表示桥接，而是通过篇章级数据驱动方式，让模型在训练中直接学习更复杂的逻辑映射模式。

考虑到篇章级隐性逻辑样本的构造与标注成本较高，本章并未采用大规模自动生成策略，而是选择小规模高质量数据构造方案。具体而言，本章以高考英语阅读材料为基础来源，从中筛选并构造需要结合上下文才能恢复完整逻辑关

系的篇章级样本，形成约 1000 条监督微调数据。相比追求数据规模，这一方案更强调数据与任务定义之间的匹配性，以及篇章依赖特征和逻辑标注的准确性。通过这种设计，本章希望验证一个核心问题：即使在参数规模较小的基础模型上，只要提供足够聚焦的篇章级监督数据，模型仍然能够获得一定的跨句逻辑翻译能力提升。

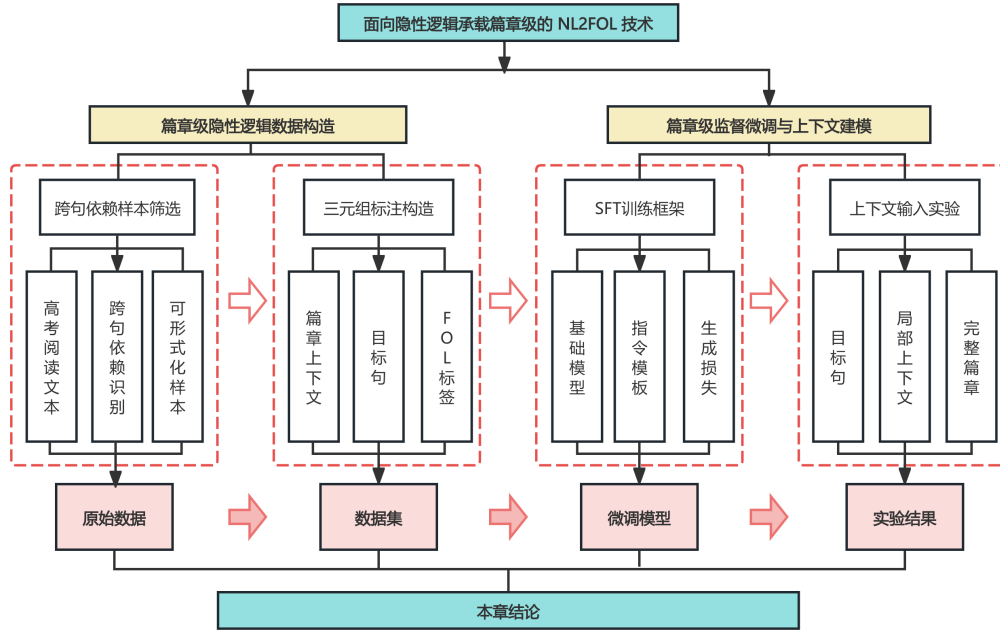


图 5-1 基于监督微调的篇章级隐性逻辑翻译方法框架

Figure 5-1 Supervised Fine-Tuning Framework for Document-Level Implicit Logic Translation

本章其余内容安排如下：第 5.2 节介绍本章方法设计，包括篇章级隐性逻辑数据构造方法、监督微调框架以及训练目标；第 5.3 节给出实验结果与分析，重点考察监督微调策略和篇章上下文输入对篇章级隐性逻辑翻译效果的影响；第 5.4 节对本章内容进行总结。

5.2 方法介绍

5.2.1 问题定义与研究动机

本章关注的任务，是面向篇章级隐性逻辑关系的自然语言到一阶谓词逻辑翻译。与前一章的句级隐性逻辑翻译不同，这里的输入不再是单个句子，而是由多个句子共同构成的篇章片段。设篇章输入为 $D = \{s_1, s_2, \dots, s_n\}$ ，其中 s_i 表示篇章中的第 i 个句子；设输出为与该篇章核心逻辑语义相对应的一阶谓词逻辑表达式 y 。则篇章级隐性逻辑翻译可以形式化为一个条件生成问题，即在给定篇章输入 D 的条件下，生成在模型概率意义上最优、且尽可能保持语义一致的目标表达式 y 。若将基础模型记为参数为 θ 的条件生成模型，则该任务可表示为：

$$y = \arg \max_{y'} p_{\theta}(y' | D) \quad (5-1)$$

如公式 (5-1) 所示, 篇章级隐性逻辑翻译在形式上仍属于条件生成任务, 但其本质复杂度已经明显高于句级场景。原因在于, 模型所依赖的逻辑信息不再局限于单句, 而是可能跨越多个句子分布。一个句子中的实体身份可能由前文引入, 一个条件限制可能由后文补充, 一个结论的成立也可能依赖于多句共同提供的前提信息。在这种情况下, 若模型只关注局部句子内容, 而未能充分整合篇章上下文, 就容易生成语义不完整甚至逻辑错误的形式化表达。

篇章级隐性逻辑翻译的困难主要体现在以下几个方面。

第一, 跨句实体指代会直接影响逻辑表达式中的变量绑定。在篇章文本中, 实体往往先在某一位置被完整引入, 随后通过代词、省略形式或角色替代在后文反复出现。若模型不能正确追踪这些跨句指代关系, 就很容易在最终逻辑表达中出现变量错配、论元混乱或对象身份不一致的问题。

第二, 前提与结论的逻辑对应关系常常跨句分布。很多篇章级样本并不是单句内给出完整逻辑结构, 而是前一句提供事实背景, 后一句给出行为或结论, 再由另一句补足规则依据或限制条件。对于这类样本, 若模型仅从目标句本身出发进行形式化, 很可能只能得到局部上合理的表达, 而无法恢复完整逻辑链条。

第三, 篇章级隐性逻辑关系往往包含更复杂的上下文依赖。例如, 某一规则是否适用, 可能取决于前文提供的身份条件; 某一事件是否能够成立, 可能依赖于前后文中的时间顺序、状态变化或因果推进。与句级场景相比, 这类信息更难通过单次局部推断获得, 因此对模型的篇章级表示与整合能力提出了更高要求。

基于上述分析, 我们认为, 篇章级隐性逻辑翻译的关键问题, 并不只是目标句内部的逻辑关系显式化, 而是如何让模型在训练阶段真正学习到跨句语义整合与篇章级逻辑映射能力。对于这一类问题, 仅依赖提示设计、推理阶段修正或局部中间表示, 通常难以稳定获得足够效果。相比之下, 通过构造篇章级隐性逻辑监督数据, 并对基础模型进行任务特定微调, 更有可能使模型直接学习上下文如何决定最终逻辑表达的映射规律。因此, 本章选择以监督微调为主要解决路径。

5.2.2 篇章级隐性逻辑数据构造方法

由于现有公开的 NL2FOL 研究大多集中在句级场景, 直接面向篇章级隐性逻辑翻译的数据资源相对缺乏, 因此本章首先需要解决数据构造问题。考虑到篇章级隐性逻辑样本的构造与标注成本较高, 本章并未从零开始完全新建一个大规模数据集, 而是采用小规模高质量的数据构造策略, 以保证样本与任务定义之间具有较高匹配度。

本章选择高考英语阅读材料作为数据基础来源, 主要有三方面原因。首先, 高考英语阅读文本长度适中, 通常由若干句构成完整语段, 既能够提供明确的篇

章上下文，又不会长到难以控制逻辑范围，适合作为篇章级实验材料。其次，这类文本在内容上普遍包含较清晰的信息推进、人物指代、因果关系、条件约束和行为结果，因此更容易从中筛选出具有跨句逻辑依赖特征的样本。再次，高考英语材料语言规范、语义边界相对明确，相较于开放域网络文本更适合进行逻辑形式化标注。

在具体构造过程中，本章将篇章级隐性逻辑样本定义为满足以下条件的文本片段。

第一，样本中的关键逻辑关系必须跨句分布。也就是说，最终逻辑表达式所依赖的信息不能全部集中在单个句子内部，而是至少需要结合两个句子以上的上下文内容才能恢复完整逻辑关系。

第二，目标句单独看不足以稳定恢复完整逻辑表达。若某个句子在脱离上下文时已经能够直接形式化，则该样本更适合归入句级任务，而不应纳入本章的篇章级实验范围。

第三，样本应具有较明确的形式化可能性。虽然本章关注的是隐性逻辑关系，但这些关系仍应能够在结合上下文之后被组织为相对清晰的一阶谓词逻辑表达式，而不是完全依赖模糊常识或开放式推断。

基于以上标准，本章从高考英语阅读材料中筛选候选篇章片段，并进一步整理为统一的数据形式。具体而言，每个样本被组织为一个三元组：

$$\mathcal{X} = (D, s_t, y) \quad (5-2)$$

其中， D 表示篇章上下文， s_t 表示目标句或目标文本片段， y 表示结合上下文之后得到的目标一阶谓词逻辑表达式。之所以采用这种三元组形式，而不是直接将整段篇章与 FOL 一一对应，是因为本章的重点并不是对整篇文本进行完全逻辑化，而是研究某一目标句的逻辑表达如何依赖其篇章上下文。采用这一形式，更有利于突出篇章依赖的作用。

在标注流程上，本章采用筛选—构造—校对的方式进行数据整理。首先，从原始高考英语材料中筛选出具有跨句依赖特征的文本片段；其次，根据篇章上下文与目标句之间的语义关系，手工构造对应的 FOL 表达式；最后，对标注结果进行二次核查，重点检查三方面内容：其一，目标句是否确实依赖上下文才能恢复完整逻辑；其二，逻辑表达式是否与篇章语义保持一致；其三，实体、变量和关系是否在跨句范围内保持一致。通过这一流程，本章最终形成约 1000 条篇章级隐性逻辑样本，并进一步划分为训练集、验证集和测试集。

从研究设计角度看，这种数据构造方式具有两方面优势。一方面，它基于真实语言材料而非纯人工杜撰例子，因此样本在篇章衔接和语义组织上更贴近自然语言使用场景。另一方面，它围绕上下文决定逻辑表达这一核心定义进行严格筛选，因此能够更直接服务于本章的研究目标。相较于追求更大规模但任务边界模糊的数据，本章更强调样本质量、篇章依赖明确性以及逻辑标注的可靠性。

5.2.3 监督微调框架设计

在完成篇章级隐性逻辑数据构造之后，本章进一步采用监督微调方法，使基础模型直接学习从篇章上下文到一阶谓词逻辑表达式的映射能力。考虑到本章的研究重点在于验证篇章级监督数据对于跨句逻辑翻译的作用，而不是追求尽可能高的绝对性能，因此本章选择参数规模较小的 Qwen3 0.6B 作为基础模型。

之所以选择小模型而非更大规模模型，主要基于两点考虑。首先，小模型在基础语言能力和复杂推理能力上相对受限，因此更能体现任务特定监督数据所带来的增益。若在参数规模较小的模型上仍能够观察到较为稳定的篇章级逻辑翻译提升，则说明数据构造与微调策略本身具有实质性作用。其次，从研究实现角度看，小模型的微调成本更低，也更适合作为篇章级隐性逻辑翻译的探索性研究起点。

本章采用的微调框架是标准的监督微调 (Supervised Fine Tuning, SFT) 方式。具体而言，模型输入由篇章上下文 D 与目标句 s_t 共同构成，输出为目标 FOL 表达式 y 。在训练阶段，模型通过大量三元组样本学习如何结合上下文恢复目标句的完整逻辑结构，并逐步形成对跨句条件来源、指代关系和约束范围的建模能力。

为了使输入格式更清晰，本章将训练样本组织为统一的指令式模板，即将篇章上下文和目标句共同放入输入端，通过显式提示要求模型输出与目标句相关的一阶谓词逻辑表达式。这样做的目的，是将监督微调任务与模型原有的指令跟随能力结合起来，使模型在训练过程中更明确地学习篇章输入—逻辑输出的任务模式。

与前两章不同，本章方法不再主要依赖推理阶段的局部修正，而是将重点放在训练阶段能力注入上。换言之，本章希望通过监督微调，使模型在参数空间中直接形成篇章级上下文整合能力，从而在推理阶段能够更自然地处理跨句逻辑关系。这种方法路径与篇章级任务本身更加匹配，因为该任务所要求的能力是全局性的上下文建模，而不仅仅是局部逻辑结构保持。

5.2.4 训练目标与实现细节

在训练目标上，本章采用标准的自回归条件生成损失。设篇章输入为 D ，目标逻辑表达式为 $y = \{y_1, y_2, \dots, y_T\}$ ，则模型训练目标可写为：

$$\mathcal{L} = - \sum_{t=1}^T \log p_{\theta}(y_t | y_{<t}, D) \quad (5-3)$$

如公式 (5-3) 所示，模型在训练过程中需要在给定篇章输入 D 以及先前已生成 token 的条件下，逐步预测目标逻辑表达式中的下一个 token。该目标与一般文本生成任务形式一致，但这里的关键在于：输出目标并不是普通自然语言，而是与篇章语义一致的一阶谓词逻辑表达式。因此，模型在训练中所学习的不只是语言续写能力，而是篇章上下文到形式逻辑的条件映射能力。

在具体实现上，本章对约 1000 条数据进行划分，其中训练集约 800 条，验证集约 100 条，测试集约 100 条。考虑到数据规模相对有限，本章更强调训练样本质量与任务边界清晰性，而不是依赖大规模数据覆盖各种语言现象。换言之，本章的方法验证重点不在于构造一个规模庞大的篇章级逻辑数据集，而在于考察：当数据定义足够聚焦时，模型是否能够通过监督微调学习到篇章级隐性逻辑翻译能力。

在推理阶段，模型直接接收篇章上下文与目标句作为输入，并生成对应的 FOL 表达式。为了进一步分析上下文的作用，本章在实验部分还将设置不同输入方式的比较，例如只输入目标句、输入局部上下文以及输入完整篇章上下文，从而考察监督微调带来的提升究竟来自一般任务适配，还是来自对篇章依赖关系的有效利用。

5.2.5 方法分析与讨论

从整体上看，本章方法的核心价值在于：将篇章级隐性逻辑翻译问题从推理时补救转向训练时学习。对于显性复杂逻辑结构或句级隐性逻辑任务，推理阶段增强和中间表示桥接能够在一定程度上发挥作用，因为这些任务中的关键信息大多仍集中在单句范围内。然而，当逻辑关系跨句分布时，模型需要具备更强的篇章级表示与整合能力，这类能力更适合通过监督微调在训练阶段注入，而不宜仅依赖推理时的局部修正。

从模型能力角度看，监督微调的意义不只是提升输出格式一致性，更重要的是让模型学习哪些上下文信息决定目标句的逻辑表达。篇章级任务的关键在于，目标句往往不能脱离前后文独立理解。因此，本章方法的真正作用不是单纯让模型看更多文本，而是通过任务特定样本让模型学习如何利用跨句上下文恢复潜在逻辑关系。这也是第五章方法与第四章方法的根本区别。第四章强调通过中间表示显式化单句内部的隐性逻辑，而第五章更强调通过监督微调学习跨句语义整合能力。

此外，本章采用小模型和小规模高质量数据的组合，也具有一定研究意义。它说明，篇章级隐性逻辑翻译的改进并不一定只能依赖超大模型和超大规模数据，有针对性的监督样本同样能够有效改善模型行为。特别是对于毕业论文研究而言，这种设计更能够突出任务建模与数据构造的作用，而不是单纯依赖模型规模优势。

当然，本章方法也存在一定局限。首先，篇章级数据构造成本较高，样本规模有限，因此模型所学习到的篇章逻辑模式仍然可能覆盖不足。其次，高考英语材料虽然在语义组织上较为规范，但文本风格相对集中，因此模型在更开放领域上的泛化能力仍需进一步验证。再次，监督微调虽然能够增强模型的篇章级逻辑建模能力，但其效果仍然依赖于训练数据的标注质量和任务定义清晰度。若样本中上下文依赖不够明确，或逻辑表达式构造不稳定，则训练效果也会受到影响。

总体而言，本章提出的监督微调方法并不是简单地把更多上下文喂给模型，

而是围绕篇章级隐性逻辑关系的特点,通过数据构造与训练目标设计,使模型学习跨句逻辑依赖与形式化映射能力。它为处理更复杂的篇章级自然语言逻辑翻译提供了一条与前两章不同但相互衔接的方法路径。

5.3 实验结果与分析

5.3.1 实验设置

为验证本章所提出方法在篇章级隐性逻辑翻译任务中的有效性,本章围绕自构建的篇章级隐性逻辑数据集开展实验。该数据集以高考英语阅读材料为基础来源,经过筛选、构造与标注,最终形成约 1000 条样本,并划分为训练集、验证集和测试集。其中,训练集约 800 条,验证集约 100 条,测试集约 100 条。与前一章的句级实验不同,本章数据中的每个样本都显式要求模型结合跨句上下文信息,才能恢复目标句的完整逻辑表达,因此更能体现篇章级隐性逻辑翻译的核心难点。

在模型设置上,本章选择 Qwen3 0.6B 作为基础模型,并采用监督微调方式进行训练。之所以选择较小规模模型,一方面是因为本章关注的重点并不是追求尽可能高的绝对性能,而是验证篇章级监督数据对跨句逻辑翻译能力的提升作用;另一方面,小模型在基础篇章建模能力上相对受限,因此更容易体现任务特定微调带来的实际收益。若在这一模型规模上仍然能够观察到较为明显的性能改进,则说明本章构造的数据和微调策略具有一定有效性。

在对比设置上,本章主要比较以下几类方法。

未微调基线:即直接使用 Qwen3 0.6B 在不给定额外监督训练的情况下完成篇章级隐性逻辑翻译。

仅目标句微调:在该设置中,模型接受监督微调,但输入仅包含目标句本身,而不提供完整篇章上下文。该设置用于考察性能提升究竟来自一般任务微调,还是来自篇章上下文信息。

局部上下文微调:在该设置中,输入包含目标句及其附近的局部上下文,而不引入完整篇章。该设置用于分析局部上下文对篇章逻辑恢复的帮助程度。

完整篇章上下文微调:即本章提出的完整方法,输入包含完整篇章上下文与目标句,由模型在监督微调后直接生成对应的一阶谓词逻辑表达式。

在评价指标方面,本章延续前文的设定,采用逻辑等价性作为核心评价标准,用以衡量模型生成的一阶谓词逻辑表达式与参考表达式在语义上是否等价。由于篇章级任务的重点在于逻辑关系恢复而非表面字符串一致,因此这一指标能够更真实地反映不同方法在篇章隐性逻辑翻译中的表现。

5.3.2 整体实验结果

从表 5-1 可以看出,未微调基线的逻辑等价性为 71.81%,说明对于参数规模较小的基础模型而言,在缺乏任务特定训练的情况下,直接完成篇章级隐性

逻辑翻译仍然较为困难。经过监督微调后，仅目标句输入的结果提升至 76.43%，表明即使不引入完整篇章上下文，任务特定训练本身也能够一定程度上改善模型的逻辑翻译表现。

进一步地，在引入局部上下文后，模型性能上升至 80.35%，说明上下文信息确实对篇章级逻辑恢复具有明显帮助。完整篇章上下文微调方法取得了最高结果，为 84.90%，相比未微调基线提升了 13.09 个百分点，也比仅目标句微调提升了 8.47 个百分点，相比局部上下文微调也进一步提升了 4.55 个百分点。这一结果表明，本章提出的篇章级监督微调方法能够较有效地提升小模型在跨句逻辑翻译任务中的表现，而更完整的篇章上下文建模对于最终性能提升具有重要作用。

表 5-1 本章方法在不同训练设置下的整体实验结果

Table 5-1 Overall Results of Different Training Settings in This Chapter

方法设置	LE (%)
未微调基线	71.81
仅目标句微调	76.43
局部上下文微调	80.35
完整篇章上下文微调	84.90

从表 5-2 的对比结果来看，通用主流模型在篇章级隐性逻辑翻译任务上整体表现较强，但不同模型之间仍然存在明显差异，说明即使是具备较强指令跟随和推理能力的通用模型，在面对跨句指代、省略补全和篇章级语义整合时，仍然难以稳定恢复完整逻辑结构。相比之下，本文方法取得了 84.90% 的结果，明显优于 Llama3.1 8B 和 Qwen3 8B，表明针对篇章级逻辑恢复任务进行专门训练能够在该实验设置下带来明显提升。

同时也可以看到，GPT 4o 的结果达到 89.95%，仍然显著高于本文方法。这表明更强的通用模型在知识覆盖、长上下文理解和复杂推理稳定性方面依然具有优势。不过，考虑到本文方法所基于的模型规模更小，且主要通过任务特定监督微调实现性能提升，其最终结果已经接近部分主流大模型，说明该方法在效率与效果之间具有较好的平衡。

从整体趋势来看，实验结果反映出两点结论。第一，监督微调本身是有效的。即便只使用目标句而不引入篇章上下文，模型在经过任务特定训练后仍然取得了较明显的提升，这说明本章构造的篇章级逻辑数据能够帮助模型更好地适应自然语言到 FOL 的输出模式。第二，篇章上下文信息对最终性能提升具有关键作用。仅目标句微调、局部上下文微调和完整篇章上下文微调之间的逐步提升，说明篇章级隐性逻辑翻译的难点并不只在于单句理解本身，而在于模型能否有效利用跨句上下文恢复完整逻辑结构。

表 5-2 与主流模型的对比结果

Table 5-2 Comparison with Mainstream Models

模型	LE (%)
Llama3.1 8B	80.12
Qwen3 8B	83.63
GPT 4o	89.95
本章方法	84.90

从方法意义上看,这一结果也说明,本章选择监督微调作为主要路径是合理的。对于篇章级隐性逻辑任务,逻辑关系往往分散在多个句子中,仅依赖提示或局部推理难以稳定恢复完整结构。相比之下,通过任务特定微调,模型可以在训练阶段直接学习上下文如何决定逻辑表达,因此更适合处理这类跨句依赖问题。

5.3.3 案例分析

为了更直观地展示本章方法在篇章级隐性逻辑翻译中的作用,下面给出一个简化案例进行分析。设篇章输入如下: Tom forgot his student card this morning. Students without valid identification are not allowed to enter the library. He waited outside until his teacher arrived.

在该片段中,第三句他一直在外面等到老师到来为目标句。若单独看这一句,模型只能得到一个关于等待行为的局部描述,而无法确定其背后的逻辑原因。真正决定其逻辑含义的信息分散在前两句中:第一句给出了 Tom 忘带学生证这一状态,第二句提供了图书馆准入规则。因此,只有结合完整篇章上下文,模型才能恢复 Tom 因不满足进入图书馆的身份验证条件而被阻止进入的逻辑结构。

表 5-3 展示了该案例在不同方法下的输出结果。

从表 5-3 可以看出,仅目标句输入时,模型往往只能抓住等待在外面这一表层事件,因而生成的逻辑表达只是局部行为描述,无法恢复其背后的规则依赖。相比之下,完整方法在结合上下文后,能够进一步抽取出实体身份、证件条件和图书馆准入规则之间的关系,从而生成更接近完整语义的逻辑表达。这一案例表明,篇章级隐性逻辑翻译的关键不在于目标句本身有多复杂,而在于目标句所依赖的逻辑前提往往分散在上下文之中。只有当模型能够有效整合这些跨句信息时,最终形式化结果才可能完整。

5.3.4 方法讨论

综合以上实验结果可以看出,本章方法对篇章级隐性逻辑翻译的改进主要体现在两个层面。其一,它通过任务特定监督微调提升了小模型的篇章级逻辑映

表 5-3 篇章级隐性逻辑翻译案例分析
Table 5-3 Case Study of Document Level Implicit Logic Translation

环节	内容
篇章输入	Tom forgot his student card this morning. Students without valid identification are not allowed to enter the library. He waited outside until his teacher arrived.
仅翻译目标句的结果	WaitOutside(Tom)
完整方法结果	$(Student(x) \wedge \neg HasValidID(x)) \rightarrow \neg Enter(x, Library)$

射能力，使模型不再仅依赖原有通用语言能力，而是能够更有针对性地学习自然语言篇章到形式逻辑表达之间的对应关系。其二，它通过完整篇章上下文输入强化了跨句信息利用，使模型在推理阶段能够更充分地整合上下文中的前提、规则和指代关系，从而减少逻辑遗漏和结构断裂。

从结果趋势来看，监督微调本身提供了最基础的提升，而完整篇章上下文则进一步释放了这种提升。也就是说，篇章级隐性逻辑翻译的性能改善并不是单靠更强生成能力就能获得，而是依赖于模型是否真正学会了如何利用上下文。未微调基线之所以表现较弱，并不是因为模型完全不能理解单句内容，而是因为它缺乏将多句信息整合为统一逻辑结构的能力。监督微调的作用，正是让模型在训练阶段学习这种篇章级映射规律。

此外，本章采用小模型与小规模高质量数据的组合，也体现出一定的方法学意义。实验结果表明，即使基础模型规模有限，只要训练数据足够聚焦且任务定义清晰，模型仍然能够获得较明显的篇章级逻辑翻译能力提升。这说明，在毕业论文研究场景中，任务定义和数据构造的重要性并不低于模型规模本身。相较于盲目追求更大模型和更多数据，本章方法更强调针对性监督对于复杂逻辑任务的促进作用。

当然，本章方法仍存在一定局限。首先，数据规模相对有限，因此模型学习到的篇章逻辑模式仍然可能覆盖不足，特别是在更复杂的开放域文本中，其泛化能力仍需进一步验证。其次，高考英语材料在风格和语篇结构上具有较强一致性，这虽然有利于构造高质量数据，但也意味着模型可能更容易适应该类文本，而在更自由的真实语篇中表现仍有不确定性。再次，本章目前主要验证的是篇章级监督微调的可行性，对于更复杂的篇章推理结构、长距离依赖和多事件链逻辑，还需要进一步扩展数据和方法。

总体而言，本章研究表明，篇章级隐性逻辑翻译的关键难点在于跨句语义整合，而监督微调为解决这一问题提供了一条直接且有效的路径。通过围绕篇章依

赖特征构造任务特定数据，并在较小规模模型上进行训练，本章方法在篇章级自然语言到一阶谓词逻辑翻译任务中取得了较为稳定的提升，也为整篇论文从显性逻辑、句级隐性逻辑进一步拓展到篇章级隐性逻辑提供了完整收束。

5.4 本章小结

本章围绕篇章级隐性逻辑关系的自然语言到一阶谓词逻辑翻译问题，提出了一种基于监督微调的篇章级隐性逻辑翻译方法。与前两章分别聚焦显性复杂逻辑结构和句级隐性逻辑关系不同，本章进一步将研究对象扩展到篇章层面，重点讨论当逻辑信息跨句分布时，模型如何结合上下文恢复完整逻辑结构，并在此基础上生成形式化表达。

首先，本章分析了篇章级隐性逻辑翻译的主要困难，指出该任务的核心问题不再只是单句内部的逻辑关系显式化，而是如何在多个句子之间追踪实体、整合前提、恢复规则来源并建立完整的跨句逻辑依赖。针对这一特点，我们认为，仅依赖推理阶段局部增强或句级中间表示方法通常难以稳定解决此类问题，因此将研究重点转向监督微调，通过训练阶段能力注入来提升模型的篇章级逻辑建模能力。

其次，在方法设计上，本章提出以 Qwen3 0.6B 为基础模型，围绕篇章级隐性逻辑关系构造任务特定监督数据，并通过监督微调使模型直接学习从篇章上下文到一阶谓词逻辑表达式的映射能力。具体而言，本章以高考英语阅读材料为基础来源，从中筛选并构造需要结合上下文才能恢复完整逻辑关系的样本，形成约 1000 条篇章级隐性逻辑数据，并组织为篇章上下文、目标句和逻辑表达式三元组结构。通过这种数据构造方式，本章将篇章级跨句依赖显式纳入训练过程，使模型能够在训练中学习更复杂的上下文整合模式。

再次，在实验部分，本章比较了未微调基线、仅目标句微调、局部上下文微调和完整篇章上下文微调等多种设置。实验结果表明，监督微调能够显著提升小模型在篇章级隐性逻辑翻译任务中的表现，而完整篇章上下文输入则进一步带来了更明显的性能改进。

总体来看，本章研究表明，对于篇章级隐性逻辑关系，自然语言到一阶谓词逻辑翻译的难点主要在于跨句信息整合，而基于监督微调的训练方式为解决这一问题提供了一条较为直接且有效的路径。通过构造聚焦篇章依赖特征的小规模高质量数据，并在基础模型上进行任务特定训练，本章方法在较小模型规模条件下仍然取得了较为稳定的改进。这一结果不仅为篇章级自然语言逻辑翻译提供了新的实现思路，也与前两章共同构成了从显性复杂逻辑结构、句级隐性逻辑到篇章级隐性逻辑的完整研究链条，为全文研究提供了较为完整的收束。

第6章 总结与展望

6.1 全文总结

本文围绕基于大语言模型的自然语言到一阶谓词逻辑翻译(Natural Language to First Order Logic, NL2FOL)问题展开研究,重点关注现有方法在复杂逻辑场景下存在的语义偏移、隐含逻辑关系恢复不足以及篇章上下文建模能力有限等问题。针对这些问题,本文从显性复杂逻辑结构、句级隐性逻辑关系和篇章级隐性逻辑关系三个层面开展了较为系统的研究,并形成了相互衔接的技术路线。

首先,针对显性复杂逻辑结构文本中容易出现的语义偏移问题,本文提出了一种面向推理阶段的增强方法 LSEG (Logic Structure and Entropy Guided Inference Controlling)。该方法不依赖额外参数更新,而是从模型内部表示层面出发,通过逻辑保持扰动样本构建逻辑稳定中心与分层逻辑校正方向,并结合层级熵变化识别推理过程中的不稳定状态,对相应隐状态实施自适应干预。实验结果表明,该方法能够在不改变模型参数的条件下,有效提升模型在显性复杂逻辑翻译任务中的语义一致性与结果稳定性。相关分析进一步说明,复杂逻辑翻译中的错误并不只是输出端的符号问题,也与推理过程中内部表示的稳定性密切相关。

其次,针对句级文本中隐性逻辑关系难以直接显式建模的问题,本文提出了一种基于伪代码中间表示的多阶段翻译方法。该方法在自然语言与一阶谓词逻辑之间引入结构化桥梁层,使模型先将句子中的条件、约束与依赖关系显式化为伪代码中间表示,再通过验证与校正机制保证中间表示的结构完整性、语义一致性和逻辑可映射性,最后完成到一阶谓词逻辑表达式的映射。实验结果表明,伪代码中间表示能够有效降低自然语言与形式逻辑之间的表示跨度,验证机制则进一步增强了翻译过程的可控性与稳定性。该研究表明,对于句级隐性逻辑关系,仅依赖端到端直接翻译往往难以充分恢复原句中的潜在逻辑骨架,而中间表示提供了一种更加合适的建模路径。

再次,针对篇章级文本中逻辑关系跨句分布、上下文依赖较强的问题,本文进一步研究了面向篇章上下文的自然语言到一阶谓词逻辑翻译方法。通过构造篇章级隐性逻辑翻译数据,并设计相应的监督微调框架,本文系统考察了不同上下文输入方式对跨句逻辑恢复的影响。实验结果表明,随着上下文信息由目标句扩展到局部上下文,再扩展到完整篇章,模型在篇章级隐性逻辑翻译任务上的表现持续提升。这说明,篇章级 NL2FOL 的关键难点不在于单句表达本身,而在于模型是否能够结合更完整的上下文信息,对跨句逻辑关系进行整体建模。

综合来看,本文的研究表明, NL2FOL 并不是一个可以由单一方法统一解决的简单任务,而是一个需要根据不同逻辑承载方式进行分层处理的复杂问题。对于显性复杂逻辑结构,关键在于抑制推理过程中的语义漂移;对于句级隐性逻辑关系,关键在于通过中间表示实现逻辑显式化;对于篇章级隐性逻辑关系,关键在于结合上下文信息恢复跨句依赖。围绕这些关键问题,本文分别提出了相应方

法，并在实验中验证了它们的有效性。

总体而言，本文围绕自然语言到一阶谓词逻辑翻译任务中的若干核心困难，分别从推理阶段控制、中间表示构建和篇章上下文建模三个方向开展研究，在一定程度上提升了大语言模型在不同复杂逻辑场景下的翻译能力，也为后续构建更可靠、更可解释的神经符号推理系统提供了方法参考。

6.2 未来展望

本文围绕自然语言到一阶谓词逻辑翻译任务中的几个关键问题进行了研究，并分别从显性复杂逻辑结构、句级隐性逻辑关系和篇章级隐性逻辑关系三个方面进行了方法设计与实验分析。虽然本文已经取得了一定结果，但从任务本身的复杂性来看，仍有一些问题没有被充分解决，也还有继续展开的空间。

一方面，本文针对不同类型的逻辑承载方式分别提出了对应方法，这有助于更有针对性地处理各自的难点。但在真实文本中，显性逻辑结构、句内隐含关系和跨句依赖关系往往同时存在，并不会严格按照任务划分被分开处理。因此，如何在统一框架中更自然地兼顾不同层面的逻辑信息，仍然是一个值得继续讨论的问题。若后续能够把推理阶段控制、中间表示构建和篇章上下文建模更好地结合起来，模型在复杂文本中的适应性应当还能继续提升。

另一方面，本文在句级隐性逻辑翻译中引入了伪代码中间表示，并验证了这种思路的有效性。现有结果说明，在自然语言与形式逻辑之间增加一个结构化过渡层，确实有助于减小直接映射带来的困难。不过，当前的伪代码表示仍然主要适用于逻辑关系较清晰的句子，对于包含默认前提、例外情况、语义歧义或更复杂组合关系的文本，其表达能力仍然有限。后续研究中，如何设计更合适的中间表示，或者如何针对不同类型的隐性逻辑选择不同的过渡形式，仍值得进一步探索。

对于篇章级隐性逻辑翻译，本文已经说明上下文信息对跨句逻辑恢复具有重要作用，但目前的研究仍主要建立在上下文扩展和监督微调的基础上，对篇章内部更深层的组织结构涉及得还不够多。比如，长文本中的指代关系、事件推进关系、主题变化以及段落之间的语义衔接，都可能影响最终逻辑结构的构建。后续若能在长上下文建模、篇章表示和关键信息选择等方面继续深入，篇章级 NL2FOL 的结果还有进一步提升的可能。

此外，本文虽然在部分方法中加入了验证与校正机制，但整体上对错误类型的分析还不够细。当前很多结果只能说明输出是否正确，却还难以清楚地区分错误究竟来自逻辑关系识别、中间结构构建，还是最终形式化映射阶段。若后续能够围绕量词范围、变量绑定、逻辑连接关系和语义等价性等方面建立更细的分析框架，不仅有助于更准确地定位问题，也能为模型改进提供更直接的依据。

最后，本文的实验主要还是围绕标准数据集和构造数据展开，距离真实应用场景还有一定距离。自然语言到一阶谓词逻辑翻译的意义，并不只是提升基准数

据上的指标,更重要的是看它能否真正支撑后续推理、验证和分析任务。今后若能进一步结合复杂问答、规则理解、知识验证、法律文本分析等具体场景,研究其与知识库、验证工具和外部推理模块的协同方式,NL2FOL 的应用价值会更加清晰。

总体而言,自然语言到一阶谓词逻辑翻译仍然是一个尚未充分解决的问题。随着大语言模型能力的提升以及神经符号方法的发展,这一方向仍具有较大的研究空间。围绕统一建模、中间表示、篇章理解、结果验证和实际应用等方面继续推进,仍然具有现实意义。

参考文献

- [1] Zettlemoyer L, Collins M. Online learning of relaxed ccg grammars for parsing to logical form [C]//Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). 2007: 678-687.
- [2] Dong L, Lapata M. Language to logical form with neural attention [J]. arXiv preprint arXiv:1601.01280, 2016.
- [3] Yang Y, Xiong S, Payani A, et al. Harnessing the power of large language models for natural language to first-order logic translation [C]//Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024: 6942-6959.
- [4] Schick T, Dwivedi-Yu J, Dessì R, et al. Toolformer: Language models can teach themselves to use tools [J]. Advances in Neural Information Processing Systems, 2023, 36: 68539-68551.
- [5] Zhao W X, Zhou K, Li J, et al. A survey of large language models [J]. arXiv preprint arXiv:2303.18223, 2023, 1(2).
- [6] Liu A, Feng B, Xue B, et al. Deepseek-v3 technical report [J]. arXiv preprint arXiv:2412.19437, 2024.
- [7] Guo D, Yang D, Zhang H, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning [J]. arXiv preprint arXiv:2501.12948, 2025.
- [8] Yang A, Li A, Yang B, et al. Qwen3 technical report [J]. arXiv preprint arXiv:2505.09388, 2025.
- [9] Wang P, Bai S, Tan S, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution [J]. arXiv preprint arXiv:2409.12191, 2024.
- [10] Achiam J, Adler S, Agarwal S, et al. Gpt-4 technical report [J]. arXiv preprint arXiv:2303.08774, 2023.
- [11] Huang J, Chang K C C. Towards reasoning in large language models: A survey [C]//Findings of the Association for Computational Linguistics: ACL 2023. 2023: 1049-1065.
- [12] Markovits H, Vachon R. Reasoning with contrary-to-fact propositions [J]. Journal of Experimental Child Psychology, 1989, 47(3): 398-412.
- [13] Xu J, Fei H, Pan L, et al. Faithful logical reasoning via symbolic chain-of-thought [J]. arXiv preprint arXiv:2405.18357, 2024.
- [14] Cheng F, Li H, Liu F, et al. Empowering llms with logical reasoning: A comprehensive survey [J]. arXiv preprint arXiv:2502.15652, 2025.
- [15] Olausson T, Gu A, Lipkin B, et al. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers [C]//Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023: 5153-5176.
- [16] Lalwani A, Kim T, Chopra L, et al. Autoformalizing natural language to first-order logic: A case study in logical fallacy detection [J]. arXiv preprint arXiv:2405.02318, 2024.
- [17] Ryu H, Kim G, Lee H S, et al. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning [J]. arXiv preprint arXiv:2410.08047, 2024.
- [18] Rudin C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead [J]. Nature machine intelligence, 2019, 1(5): 206-215.

- [19] Ji Z, Lee N, Frieske R, et al. Survey of hallucination in natural language generation [J]. *ACM computing surveys*, 2023, 55(12): 1-38.
- [20] Sarker M K, Zhou L, Eberhart A, et al. Neuro-symbolic artificial intelligence: Current trends [J]. *Ai Communications*, 2022, 34(3): 197-209.
- [21] Nye M, Tessler M, Tenenbaum J, et al. Improving coherence and consistency in neural sequence models with dual-system, neuro-symbolic reasoning [J]. *Advances in Neural Information Processing Systems*, 2021, 34: 25192-25204.
- [22] 邓鹏. 基于语义增强的事件图谱补全和推理研究 [D]. 中国科学院计算技术研究所, 2023.
- [23] Liu J. Few-shot natural language to first-order logic translation via code generation [C/OL]//Chiruzzo L, Ritter A, Wang L. *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Albuquerque, New Mexico: Association for Computational Linguistics, 2025: 10939-10960. <https://aclanthology.org/2025.naacl-long.547/>. DOI: 10.18653/v1/2025.naacl-long.547.
- [24] Winograd T. Understanding natural language [J]. *Cognitive psychology*, 1972, 3(1): 1-191.
- [25] Montague R. The proper treatment of quantification in ordinary english [C]//*Approaches to natural language: Proceedings of the 1970 Stanford workshop on grammar and semantics*. Springer, 1973: 221-242.
- [26] Woods W A. Progress in natural language understanding: an application to lunar geology [C]//*Proceedings of the June 4-8, 1973, national computer conference and exposition*. 1973: 441-450.
- [27] Warren D H, Pereira F C. An efficient easily adaptable system for interpreting natural language queries [J]. *American journal of computational linguistics*, 1982, 8(3-4): 110-122.
- [28] Androutsopoulos I, Ritchie G D, Thanisch P. Natural language interfaces to databases—an introduction [J]. *Natural language engineering*, 1995, 1(1): 29-81.
- [29] Zettlemoyer L S, Collins M. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars [J]. *arXiv preprint arXiv:1207.1420*, 2012.
- [30] Wong Y W. Learning for semantic parsing using statistical machine translation techniques [M]. Computer Science Department, University of Texas at Austin, 2005.
- [31] Wong Y W, Mooney R. Learning synchronous grammars for semantic parsing with lambda calculus [C]//*Proceedings of the 45th annual meeting of the association of computational linguistics*. 2007: 960-967.
- [32] Liang P, Jordan M I, Klein D. Learning dependency-based compositional semantics [J]. *Computational Linguistics*, 2013, 39(2): 389-446.
- [33] Clarke J, Goldwasser D, Chang M W, et al. Driving semantic parsing from the world's response [C]//*Proceedings of the fourteenth conference on computational natural language learning*. 2010: 18-27.
- [34] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks [J]. *Advances in neural information processing systems*, 2014, 27.
- [35] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [J]. *arXiv preprint arXiv:1409.0473*, 2014.
- [36] Yin P, Neubig G. A syntactic neural model for general-purpose code generation [J]. *arXiv preprint arXiv:1704.01696*, 2017.
- [37] Jia R, Liang P. Data recombination for neural semantic parsing [J]. *arXiv preprint arXiv:1606.03622*, 2016.

- [38] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners [J]. *Advances in neural information processing systems*, 2020, 33: 1877-1901.
- [39] Dong Q, Li L, Dai D, et al. A survey on in-context learning [C/OL]//Al-Onaizan Y, Bansal M, Chen Y N. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Miami, Florida, USA: Association for Computational Linguistics, 2024: 1107-1128. <https://aclanthology.org/2024.emnlp-main.64/>. DOI: 10.18653/v1/2024.emnlp-main.64.
- [40] Kojima T, Gu S S, Reid M, et al. Large language models are zero-shot reasoners [J]. *Advances in neural information processing systems*, 2022, 35: 22199-22213.
- [41] Liu P, Yuan W, Fu J, et al. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing [J]. *ACM computing surveys*, 2023, 55(9): 1-35.
- [42] Lu Y, Bartolo M, Moore A, et al. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity [C]//*Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022: 8086-8098.
- [43] Rubin O, Herzig J, Berant J. Learning to retrieve prompts for in-context learning [C]//*Proceedings of the 2022 conference of the North American chapter of the association for computational linguistics: human language technologies*. 2022: 2655-2671.
- [44] Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models [J]. *Advances in neural information processing systems*, 2022, 35: 24824-24837.
- [45] Wang X, Wei J, Schuurmans D, et al. Self-consistency improves chain of thought reasoning in language models [J]. *arXiv preprint arXiv:2203.11171*, 2022.
- [46] Zhou D, Schärli N, Hou L, et al. Least-to-most prompting enables complex reasoning in large language models [J]. *arXiv preprint arXiv:2205.10625*, 2022.
- [47] Yao S, Yu D, Zhao J, et al. Tree of thoughts: Deliberate problem solving with large language models [J]. *Advances in neural information processing systems*, 2023, 36: 11809-11822.
- [48] Turpin M, Michael J, Perez E, et al. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting [J]. *Advances in Neural Information Processing Systems*, 2023, 36: 74952-74965.
- [49] Hu E J, Shen Y, Wallis P, et al. Lora: Low-rank adaptation of large language models. [J]. *ICLR*, 2022, 1(2): 3.
- [50] Touvron H, Martin L, Stone K, et al. Llama 2: Open foundation and fine-tuned chat models [J]. *arXiv preprint arXiv:2307.09288*, 2023.
- [51] Lake B, Baroni M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks [C]//*International conference on machine learning*. PMLR, 2018: 2873-2882.
- [52] Matthei E H. The modularity of mind: An essay on faculty psychology by jerry a. fodor [J]. *Language*, 1984, 60(4): 976-979.
- [53] Kahneman D. *Thinking, fast and slow* [M]. macmillan, 2011.
- [54] Evans J S B. Dual-processing accounts of reasoning, judgment, and social cognition [J]. *Annu. Rev. Psychol.*, 2008, 59(1): 255-278.
- [55] Parisi A, Zhao Y, Fiedel N. Talm: Tool augmented language models [J]. *arXiv preprint arXiv:2205.12255*, 2022.
- [56] Xu F, Wu Z, Sun Q, et al. Symbol-LLM: Towards foundational symbol-centric interface for large language models [C/OL]//Ku L W, Martins A, Srikumar V. *Proceedings of the 62nd*

- Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Bangkok, Thailand: Association for Computational Linguistics, 2024: 13091-13116. <https://aclanthology.org/2024.acl-long.707/>. DOI: 10.18653/v1/2024.acl-long.707.
- [57] Yao S, Zhao J, Yu D, et al. React: Synergizing reasoning and acting in language models [C]// The eleventh international conference on learning representations. 2022.
- [58] Cheng Z, Xie T, Shi P, et al. Binding language models in symbolic languages [J]. arXiv preprint arXiv:2210.02875, 2022.
- [59] Liu S, Ye H, Xing L, et al. Reducing hallucinations in vision-language models via latent space steering [J]. arXiv preprint arXiv:2410.15778, 2024.
- [60] Han S, Schoelkopf H, Zhao Y, et al. Folio: Natural language reasoning with first-order logic [C]//Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. 2024: 22017-22031.
- [61] Chung H W, Hou L, Longpre S, et al. Scaling instruction-finetuned language models [J]. Journal of Machine Learning Research, 2024, 25(70): 1-53.
- [62] Perez E, Ringer S, Lukosiute K, et al. Discovering language model behaviors with model-written evaluations [C]//Findings of the association for computational linguistics: ACL 2023. 2023: 13387-13434.

致 谢

行文至此，论文已接近尾声。这三年的硕士生涯，我有过低谷，也有过高光。一路走来，虽不是那么轻松，但那些认真面对过的困惑、压力与选择，也沉淀为此刻落笔的平静与笃定。能够完成这篇论文，并顺利走到今天，离不开许多人的帮助。在此，我谨向所有关心、帮助过我的人致以最诚挚的谢意。

衷心感谢我的导师王石老师。本文从选题确定、框架设计到后续实验改进，都离不开王老师的悉心指导。在科研工作中，王老师总能给出恰到好处的建议，既不会让我的思路流于空泛，也不会将问题局限在狭窄的视角之中。刚开始接触研究工作时，我常常带着较强的工程思维，更倾向于借助现有工具去解决眼前的问题，却较少从科研的角度去思考问题的本质，进而提出更具创新性的解决方案。是王老师一次次耐心地指导和启发我，帮助我逐渐建立起更加规范的研究思维。此外，王老师在日常生活中也对我关怀备至。除了在学术上给予我悉心指导之外，王老师也让我感受到一种积极、乐观、从容面对困难的生活态度。这种影响不仅让我在科研过程中受益良多，也让我在面对学习与生活中的压力时，能够以更加平和坚定的心态继续前行。

感谢课题组的各位老师和同学，包括裴晓雪老师、资康莉老师、杨竞同学、孙严顺同学、罗天宇同学、姜之晓同学、杨俊祺同学等。能够在求学阶段与大家共同探讨科学问题，是一段十分难得而珍贵的经历。同时，大家在面对科研、学习和生活中各种问题时所展现出的态度与应对方式，也让我受益良多。

感谢我的家人和朋友们。人生路上，能够拥有一些始终愿意陪伴、倾听和关心自己的人，已是难得的幸运。

最后，感谢一路走来的自己。过去很长一段时间里，我总是在追逐一个又一个被认为“正确”的目标，也曾因此迷茫过、怀疑过，不知道自己真正热爱的是什么，又想成为什么样的人。所幸在不断学习、尝试与思考的过程中，我逐渐明白了自己所珍视和追求的东西。或许前路仍然漫长，能力也仍然有限，但我愿意继续认真走下去，努力在自己所热爱的方向上，留下哪怕微小却真实的一笔。

至此，过往可念，来路可敬，前路可期。愿自己始终清醒，也始终坚定。

2026年6月

作者简历及攻读学位期间发表的学术论文与其他相关学术成果

作者简历：

2019 年 9 月——2023 年 6 月，在西安电子科技大学人工智能院获得学士学位。

2023 年 9 月——2026 年 6 月，在中国科学院计算技术研究所攻读硕士学位。

已发表（或正式接受）的学术论文：

- (1) Yuwei Huang, Shi Wang, Kangli Zi, Tianyu Luo, Jiang Zhixiao, Ruiheng Wang and Yufei Wang. LSEG: A Fine-tuning Free Method for NL2FOL via Logic-Structure and Entropy Guided Inference Controlling. Findings of the Association for Computational Linguistics: ACL 2026, accepted.

参加的研究项目及获奖情况：

获奖情况：

- (1) 2023 年中国科学院大学学业奖学金
- (2) 2024 年中国科学院大学学业奖学金
- (3) 2025 年中国科学院大学学业奖学金

