心於至善

面向空间推理领域的神经符号框架的设计与实现 贾梁 东南大学

学校代码: 10286
分 类 号: 000
密 级: 公开
U D C: 000
学 号: 222171

工程硕士学位论文

面向空间推理领域的神经符号框架的设计与实现

(学位论文形式: 应用研究)

研究生姓名: **贾梁**

导 师 姓 名: **张志政 副教授**

申请学位类别 **工程硕士** 学位授予单位 **东 南 大 学**

工程领域名称 **电子信息** 论文答辩日期 **2025 年 5 月 30 日**

研 究 方 向 **计算机技术** 学位授予日期 **2025 年 6 月 20 日**

答辩委员会主席 **翟玉庆** 评 阅 人 **倪庆剑**

张祥

2025 年 4 月 5 日

学校代码: 10286
分 类 号: 000
密 级: 公开
U D C: 000
学 号: 222171

工程硕士学位论文

面向空间推理领域的神经符号框架的设计与实现

(学位论文形式: 应用研究)

研究生姓名: **贾梁**

导 师 姓 名: **张志政 副教授**

申请学位类别 **工程硕士** 学位授予单位 **东南大学**

工程领域名称 **电子信息** 论文答辩日期 **2025 年 5 月 30 日**

研 究 方 向 **计算机技术** 学位授予日期 **2025 年 6 月 20 日**

答辩委员会主席 **翟玉庆** 评 阅 人 **倪庆剑 张祥**

2025 年 4 月 5 日

东南大学

工程硕士学位论文

面向空间推理领域的
神经符号框架的设计与实现

专 业 名 称: **电 子 信 息**

研究生姓名: **贾 梁**

导 师 姓 名: **张志政 副教授**

RESEARCH AND IMPLEMENTATION OF VISUAL QUESTION ANSWERING TECHNOLOGY BASED ON ANSWER SET PROGRAMMING

A Thesis submitted to

Southeast University

For the Professional Degree of Master of Engineering

BY

Jia Liang

Supervised by:

Associate Prof. Zhang Zhizheng

School of Computer Science and Engineering

Southeast University

2025/4/5

东南大学学位论文独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得东南大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

研究生签名：_____ 日 期：_____

东南大学学位论文使用授权声明

东南大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权东南大学研究生院办理。

研究生签名：_____ 导师签名：_____ 日 期：_____

摘　要

空间推理能力是智能机器人等具身人工智能的重要基础。CLEVR 数据集作为空间推理领域的经典数据集，能够较好地模拟夹取、移动物体等情境，从而对智能体的空间推理能力进行有效评估。然而，CLEVR 数据集生成的图像均基于完整场景，回答问题所需的信息直接可见，因而无法充分考察模型在处理部分可见场景问题上的能力。近年来，视觉语言模型（Visual Language Model, VLM）在空间推理任务上取得了一定成果，但已有研究表明，当面对部分可见场景时，其回答准确率显著下降，这一不足限制了其在机器人领域的进一步应用。与此同时，回答集编程（Answer Set Programming, ASP）作为一种符号型知识表示与推理方法，已在公理化及缺省的空间关系表示与推理方面表现出较强能力。通过利用深度学习技术对感知信息进行处理，生成输入图像与问题的逻辑符号表示，并结合 ASP 进行推理与解答的方法，相较于单纯依赖深度学习的视觉语言模型，在空间推理任务中取得了更为优异的表现。然而，现有神经符号框架在 ASP 规则的自动拓展方面存在不足，需要依赖人工干预进行规则扩充，从而增加了开发人员的工作量。

为了解决上述问题，本文构建了一个视觉问答数据集，并提出了一种面向空间推理领域的神经符号框架。本文主要工作如下：

1. 数据集构建：在原有 CLEVR 数据集的基础上，本文新增部分可见场景的问题，从而构建了一个能够更全面考察模型空间推理能力的视觉问答数据集。
2. 神经符号框架设计：基于 DSPy 开发平台，本文提出的框架融合了 DSPy 在自动化提示生成和优化过程中的优势，其整体架构包括视觉场景理解、语义解析、规则蒸馏与迭代反馈等模块。相较于现有框架，本文新增的规则蒸馏模块在 ASP 求解器进行推理前调用大语言模型，通过利用 ASP 知识库中的先验知识对语义解析与视觉场景理解生成的 ASP 规则进行补充，并在经过一致性检验后将新获得的知识保存至知识库，从而实现了 ASP 规则的自动拓展能力。

通过对比实验，本文将所提出的神经符号方法与直接提示视觉语言模型的方法进行了比较。实验结果显示，在三种大语言模型上，神经符号方法在回答部分可见场景问题时的准确率平均提升了 12.3%，充分验证了该方法在提升大语言模型空间推理能力及跨模型迁移性方面的有效性。此外，消融实验结果表明，规则蒸馏模块对整体框架的问答准确率具有显著促进作用。最后，基于所设计的神经符号框架，本文实现了一个自动规划的课程教学演示问答原型系统，为教师开展机器人空间推理演示教学提供了有效支持。

关键词： 多模态，回答集编程，视觉问答，空间推理

Abstract

Currently, visual language models have achieved promising results in tasks such as visual question answering and image captioning; however, they often perform poorly in spatial reasoning—particularly in scenarios where scenes are only partially visible—resulting in inadequate understanding and inference of spatial relationships. Answer Set Programming (ASP) is a formal method for knowledge representation and reasoning, and several studies have demonstrated its strong capability in comprehending spatial relationships and conducting spatial reasoning. In the neural-symbolic system for visual question answering, deep learning techniques are used for perception to generate logical symbolic representations of input images and questions; ASP is then employed for reasoning to produce the corresponding answers. Compared with standalone visual language models, this neural-symbolic system exhibits superior performance in spatial reasoning. Nevertheless, existing visual question answering datasets that assess spatial reasoning ability suffer from issues such as overly simplistic question designs, limited diversity in object types and attributes, and a lack of varied linguistic expression in questions. Furthermore, current neural-symbolic systems generally exhibit poor generalization, with the process of writing and maintaining prompt templates being overly complex, resulting in relatively high development difficulty. They also require manually designed additional rules for new scenarios, which limits their scalability. To address these issues, this thesis constructs a visual question answering dataset and proposes a neural-symbolic framework specifically oriented toward spatial reasoning. The main contributions of this work are as follows:

1. A visual question answering dataset is constructed based on the CLEVR dataset. This dataset enhances the difficulty by considering factors such as the number of reasoning steps required, the diversity of object types and attributes, and the variability and ambiguity of question language. This allows for a more rigorous evaluation of a model’s capability in tackling complex spatial reasoning tasks.
2. A neural-symbolic framework tailored for spatial reasoning is designed, achieving system integration between an ASP solver and a large language model. Developed on the DSPy platform, the framework leverages DSPy’s notable advantages in automated prompt generation and optimization. Architecturally, it comprises modules for visual scene understanding, semantic parsing, rule distillation, and iterative feedback, thereby enhancing the visual question answering system’s ability to address spatial reasoning problems.
3. ased on the proposed neural-symbolic framework, a neural-symbolic system for visual question answering is designed and implemented.

Through these contributions, the proposed neural-symbolic framework demonstrates strong performance and adaptability in enhancing the spatial reasoning capabilities of visual language models, particularly in handling complex and partially observable scenarios.

Keywords: Multi-modal, Answer Set Programming, Visual Question Answering, Spatial Reasoning

目录

插图目录

表格目录

算法目录

第一章　绪论

1.1　研究背景

随着计算机视觉（Computer Vision）和自然语言处理（Natural Language Processing）技术的迅猛发展，跨模态智能理解成为人工智能研究的重要方向之一。其中，视觉问答（Visual Question Answering, VQA）[1] 任务因其广泛的应用前景和挑战性，受到了学术界和工业界的广泛关注。

视觉问答是一种多模态任务，它要求计算机能够基于给定的图像内容，理解并回答关于该图像的自然语言问题。例如，给定一幅包含动物的图片，系统需要能够回答“这只动物是什么颜色?”或“图片中有几只猫?”等问题。这一任务的核心在于多模态信息的深度融合，即如何在视觉特征和语言信息之间建立有效的联系。

视觉语言模型（Visual Language Model, VLM）是一种多模态模型，能够同时处理图像和文本信息，并生成与图像内容相关的自然语言响应。如图1.1所示为 VLM 的一个简单流程示例。VLM 通过将视觉编码器与大语言模型结合，赋予模型“看”与“理解”的能力。与传统的计算机视觉模型不同，VLM 不受固定类别集或特定任务 (如分类或检测) 约束。在大量文本和图像/视频字幕对的语料上进行重新训练，VLM 可以用自然语言进行指导，并用于处理许多典型的视觉任务以及新的生成式 AI 任务，例如摘要和视觉问答。如图1.2所示为 VLM 的通用架构，包括视觉编码器、投影器和大语言模型（Large Language Model, LLM）三个部分。视觉编码器（如 CLIP 模型）具有图像与文本的关联能力，负责提取图像特征。投影器由一组网络层构成，负责将视觉特征转换为 LLM 可理解的标记（Token）。LLM 负责生成文本输出，支持对话、推理等任务，目前任何现有的 LLM（如 ChatGPT、Llama、DeepSeek 等）都可以用来构建 VLM。

尽管 VLM 在图像分类[2]、字幕生成[3]、目标检测[4]、视频理解[5] 和文档解析[6] 等各种任务中取得了良好效果，但这些模型在空间推理方面仍然面临重大挑战。具体表现为难以区分”左”和”右”、”上”和”下”等简单空间概念，以及”前”与”后”、”内”与”外”、”近”与”远”等

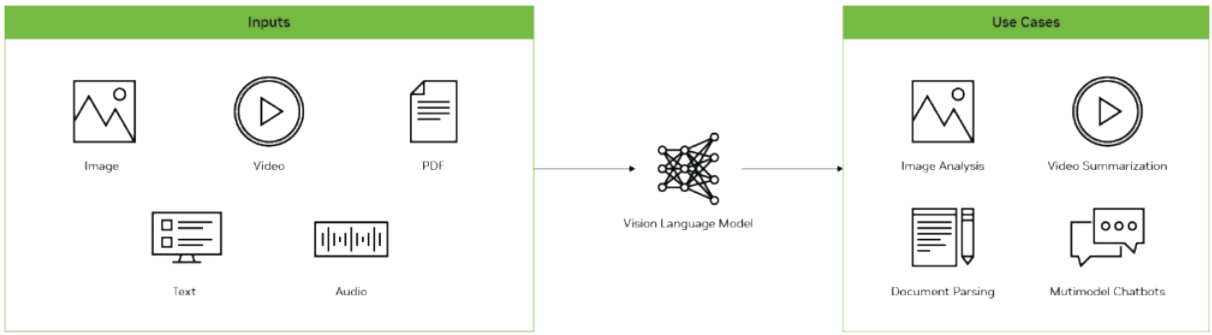


图 1.1: 视觉语言模型用例

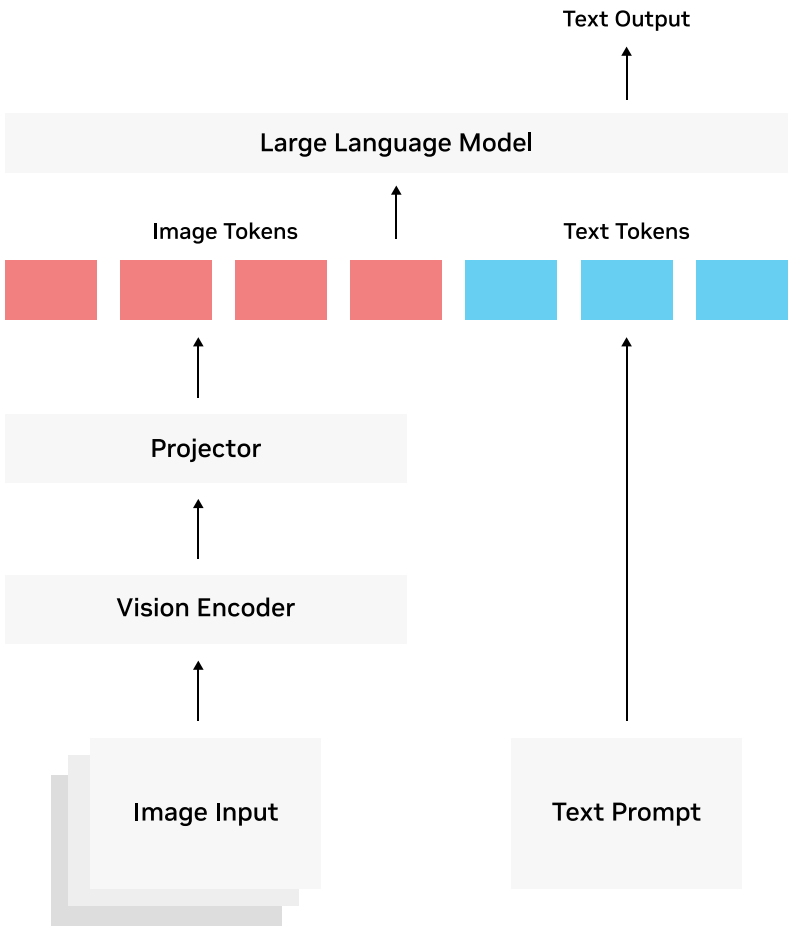


图 1.2: 视觉语言模型的通用三部分架构

复杂空间关系[7][8][9]。此外，VLM 在处理细节模糊或者部分遮挡的场景时，其表现类似于视力不佳的人，难以捕捉到图像中的细节信息，从而影响其问答或者描述的准确性，这表明目前 VLM 对处理部分可见场景的问题仍有很大的局限性[10]。正确理解并推理空间关系的能力，不仅对视觉理解至关重要，而且对智能机器人[11][12] 和增强现实[13] 等领域的现实应用也至关重要。在这些领域中，精确的空间感知能力对于导航[14]、物体操控[11] 和与现实世界的交互[15] 等任务至关重要。在自动规划课程教学中，需要使用智能机器人为学生进行操纵物块、绕避障碍物等动作的演示。而实际场景中，机器人可能会因为物体遮挡而无法直接看到目标物体的位置，进而需要结合已知信息进行推理，这就要求机器人的空间推理系统在部分可见的场景下具备推理能力，像人类一样根据周围环境信息和已有先验知识，做出正确的决策。因此，对多模态模型的空间推理能力展开研究，增强其对部分可见场景问题的推理能力，具有十分重要的现实意义和应用价值。

为了解决多模态模型在空间推理方面的问题，研究者们提出了一些新的方法。近年来，神经符号方法（Neuro-Symbolic Methods）得到了广泛关注。在多模态空间推理场景中，神经符号方法使用深度学习技术进行感知，为输入的图像和问题分别生成符号表示，再使用符号系统进行推理求解，可以有效提升多模态模型在空间推理方面的能力。

在神经符号方法中，神经网络的选择十分多样，通常包括卷积神经网络（Convolution Neural Network, CNN）、循环神经网络（Recurrent Neural Network, RNN）及其变体、图神经网络（Graph Neural Network, GNN）和 Transformer 架构及 LLM。相比其它神经网络，LLM 用于神

经符号系统，具有以下突出优势：

1. 丰富的预训练知识库。LLM 在大规模语料上进行预训练，具备广泛的世界知识和语言理解能力，这使得它们在处理涉及常识推理和多领域知识整合时，比专门针对单一任务训练的网络更具优势。
2. 自然语言理解与生成能力。LLM 能够将符号系统产生的抽象逻辑或规则，用自然语言进行解释和反馈，从而降低了人机交互的门槛，并使得系统的推理过程更易于理解和验证。
3. 零样本/少样本学习能力。由于在大规模数据上学习到了通用的语言模式，LLM 在面对新领域或数据较少的任务时，往往能够直接发挥作用，从而提高神经符号系统的泛化能力。
4. 灵活的上下文建模。LLM 能够捕捉长距离依赖和上下文信息，这对于将符号系统中离散的知识点进行有机整合，以及在复杂场景中实现动态推理具有重要意义。

结合以上各项理由，将 LLM 作为神经符号系统中的神经模块，能够与符号系统形成互补，既利用符号逻辑的严谨性，又发挥神经网络在语义理解和自然语言生成方面的优势，从而构建更强大、更灵活的智能系统。

在神经符号方法中，符号系统的选择同样多种多样，例如知识图谱嵌入（Knowledge Graph Embeddings, KGE）、逻辑神经网络（Logical Neural Networks, LNN）都为信息表示和推理提供了新思路。然而，在当前的研究和应用中，回答集编程（Answer Set Programming，ASP）往往被更多地采用作为符号系统。ASP 是一种声明式编程范式，可用于解决复杂的人工智能问题，其起源于对逻辑编程、非单调推理和知识表示的研究。相比其它符号系统，ASP 被更多用于神经符号方法中，主要是由于以下几点优势：（1）明确的逻辑语义与可解释性。ASP 基于严格定义的逻辑规则，其推理过程透明、易于解释。这一特性在需要精确推理和结果可验证的任务中尤为重要[16]；（2）强大的非单调推理能力。ASP 自然支持非单调逻辑推理，能够处理默认情况和异常情形，这使其在面对现实世界中不完全或动态变化的信息时表现出色[16]；（3）成熟的求解器与工具支持。像 Clingo 这样的 ASP 求解器已经经过长期验证，具有高效、稳定的特点，能够处理大规模问题，这为实际应用提供了有力保障[17]；（4）便于与神经网络等模块整合。ASP 的声明式表达方式使其易于与神经网络输出的信息对接，形成互补优势，从而提升整体系统的推理能力[18]。ASP 的这些特性使得它作为符号系统的杰出代表，被广泛应用于神经符号方法中。

尽管神经网络采用了预训练的 LLM，而符号系统部分选用了 ASP，但将二者有效结合在一起仍然是一项复杂的任务。为此，近年来出现了一些神经符号框架，试图通过模块化设计和统一接口来简化 LLM 与 ASP 的集成开发流程[19]。尽管现有的框架为 LLM 结合 ASP 的开发过程提供了诸多便利，降低了使用门槛，然而其在 ASP 规则的自动拓展方面存在不足，需要依赖人工干预进行规则扩充，在这一方面仍存在改进空间。

为了测试模型解决空间推理问题的能力，涌现了诸如 CLEVR、GQA、COCO 等数据集。这些数据集由问题答案对（Question and Answer Pair，以下简称 QA 对）组成，并且包含问题对应的场景图像。其中，CLEVR 数据集因高度结构化的场景构造、精确标注的物体属性及复杂的空间关系，而被广泛用于检测智能机器人的环境感知模块在空间推理方面的能力。然而，

CLEVR 数据集的图像均是基于完整场景生成，所有回答问题的信息在对应的图像中都直接可见，缺乏现实环境中常见的部分可见性的情形，不能有效考察模型解决部分可见场景问题的能力[20]。

基于上述背景，本文重点关注提升视觉问答系统对部分可见场景问题的推理能力，并对 CLEVR 数据集进行改进，以对使用神经符号方法的视觉问答系统回答该类问题的能力进行定量评估。

1.2 相关研究现状

基于以上背景，本节主要从视觉问答的发展与现状、空间推理的发展与现状、视觉问答数据集的发展与现状、生成 ASP 程序的发展与现状、神经符号方法在空间推理中的发展与现状来进行综述。

1.2.1 视觉问答的发展与现状

VQA 任务最早由 Antol[21] 等人在 2015 年提出，标志着 VQA 研究早期阶段的开始。这一阶段的 VQA 系统采用比较简单的架构，主要包括视觉编码器、语言编码器和融合模块。Ren[22] 等人将采用预训练的卷积神经网络（Convolutional Neural Network, CNN）作为视觉编码器提取图像特征，采用循环神经网络（Recurrent Neural Network, RNN）作为语言编码器或提取问题特征。Malinowski[23] 等人在视觉编码器上同样采用 CNN，在语言编码器上则是采用长短期记忆（Long Short-Term Memory , LSTM）来生成问题的答案。融合模块通常采用简单的拼接或注意力机制融合视觉和语言特征。这些系统在一些简单的 VQA 数据集上取得了不错的性能，但在处理复杂的空间推理问题时表现较差。

随着计算机视觉和自然语言处理技术的发展，VQA 系统逐步引入了更高级的特征提取与融合方法。例如，Yao[24] 等人将区域提议网络（RPN）引入到 VQA 模型中，以增强对图像中物体的感知能力。注意力机制也逐渐引入 VQA 模型中，例如 Xu[25] 等人提出堆叠注意力网络（SANs），通过多层次注意力机制，逐步聚焦于图像和问题的关键部分，从而提升了 VQA 模型的推理能力。

在 2019 年前后，得益于基于 Transformer 的预训练语言模型的兴起，VQA 研究取得了显著进展。如 BERT、ViLBERT 和 LXMERT 等模型通过联合训练大规模的视觉和语言数据，学得如何在同一嵌入空间中表示视觉和文本信息，极大程度上提升了 VQA 的准确性和鲁棒性。例如，Tan[26] 等人提出了 LXMERT 模型，将预训练的 BERT 模型与视觉编码器相结合，实现了对图像和问题的联合编码。

如今，VQA 的研究已经进入了全新阶段。随着大规模视觉语言模型（如 GLIP、GPT-4 等）的应用，VQA 模型已经能够处理更加复杂的任务，不仅处理特定问题时效果出色，而且可以进行跨模态推理，通过多种信息源之间的互动生成更加丰富和精准的答案。OpenAI 的研究团队[27] 在 CLIP 模型中，应用了自监督学习的方法，利用海量的图文对数据进行训练，实现了视觉和语言的统一表示，且训练过程无需人工标注，充分体现了自监督学习的优势。Salesforce 的研究团队[28] 在 BLIP 模型中，通过生成式预训练任务，提升了视觉语言模型在理解和生成方面的性能。

1.2.2　空间推理的研究现状

近年来，研究人员在利用 LLM 进行空间推理方面取得了显著进展。Hong[29] 等人将重点放在从多视角图像重建场景，例如通过点云或神经场构建三维表示，并结合密集语义特征进行增强，并将这些三维表示和密集特征整合到 LLM 中。然而，该方法存在诸如 3D 表征与自然语言之间的模态鸿沟导致性能下降、多视角图像并不能保证一定可用的问题。Gu[30] 等人避免直接将 3D 表征融入 LLM，尝试构造场景图并将其与 LLM 进行融合。然而，最新研究[8] 表明，当坐标信息以文本形式提供时，LLM 很难有效利用这些信息，进而削弱了 LLM 理解和削弱空间关系的能力。

1.2.3　多模态语言模型在空间推理中的局限性

当前，多模态语言模型在空间推理方面的能力仍存在显著局限。Cohn[31] 等人在 RCC-8 框架下对 GPT-4 在定性空间推理任务中的表现进行了评估，发现尽管 GPT-4 能够理解一些简单的空间关系，但在处理复杂的空间关系时，往往无法准确应用 RCC-8 的规则，导致推理精度较低。此外，GPT-4 在不同推理步骤之间表现出明显的不一致性，且其推理过程缺乏明确的策略，往往依赖于经验或直觉做出决策，但这些决策往往缺乏可解释性，难以追踪其推理路径。Bang[32] 等人也指出，GPT 在空间推理任务中面临的挑战尤为突出，特别是在涉及多个空间区域、动态变化的场景或高维空间结构时，模型倾向于依赖已知的简单规则进行推理，但在面对复杂或不常见的空间配置时，难以有效处理。此外，GPT 在基于空间关系进行推理时，往往会出现错误，无法从准确的空间关系中得出合乎逻辑的结论。特别是在涉及常识性空间理解的任务中，GPT 的表现较差，难以理解如物体重叠、邻接、相交等基本的空间常识性假设。

为了解决这一问题，研究者们提出了一些新的方法，如基于神经符号方法的空间推理框架。这些方法通过将大语言模型与符号推理方法相结合，实现了对复杂空间推理问题的有效建模。例如，Wang[33] 等人提出了一种基于大语言模型和 Answer Set Programming（ASP）的神经符号框架，用于解决复杂空间推理问题。该方法通过将大语言模型与 ASP 求解器相结合，实现了对复杂空间推理问题的高效建模。此外，研究者们还提出了一些其他的神经符号方法，如基于知识图谱的推理方法、基于逻辑规划的推理方法等，以增强大语言模型在空间推理方面的能力。

1.2.4　ASP 在空间推理中的应用

ASP 作为一种形式化的知识表示和推理方法，已经在空间推理中得到了广泛的应用。ASP 具有表达能力强、推理效率高、易于理解和调试等优点，适用于解决复杂的空间推理问题。有一些学者在这一方面做了一些研究。例如，Wałęga[34] 等人提出 ASPMT(QS)，将非单调空间推理与基于理论的答案集编程相结合，整合了定性和定量的空间信息，解决了传统空间推理方法在处理复杂空间变化和组合约束时的局限性。Baryannis[35] 等人 3 将轨迹建模为由不重叠区域构成的序列，并提出了多种 ASP 编码方案（包括专门优化的编码和通用编码）以利用合成表来保证约束的一致性，展示了 ASP 在空间推理中的应用潜力。这些研究表明，ASP 在空间推理中具有很大的潜力，可以有效解决复杂的空间推理问题。

1.2.5 神经符号方法在空间推理中的应用

尽管研究者将 ASP 成功引入空间推理问题，取得了一些研究成果，但 ASP 在感知能力方面仍然存在一些不足，如对图像和自然语言的理解能力较弱，难以处理复杂的视觉问答问题。神经符号方法的出现，为解决这一问题提供了新的思路。神经符号方法引入深度学习技术，为 ASP 求解器提供了更加丰富的输入信息，从而提升了空间推理的准确性和效率。Tejas[36]等人“逻辑透镜”（Lens of Logic, LOL）模型，采用了神经符号方法，利用了问题注意力和逻辑注意力机制，以识别和理解问题汇总的逻辑连接词，并且引入了 Fréchet 兼容性损失（Fréchet-Compatibility Loss），确保组件问题的答案与组合问题的答案在推理过程中保持一致性。Thomas[37] 等人提出了一种结合神经网络和符号推理的方法，该方法能够有效地进行空间推理和逻辑推理，从而在复杂地视觉问答任务中提高准确性，展示了符号推理与深度学习结合的潜力，推动了神经符号方法在视觉问答中的应用。Pan[38] 等人对神经符号方法进行了改进，引入了一个自我优化模块，利用符号求解器的错误提示信息来修正符号表示，从而提高推理的准确性和可靠性。

1.2.6 视觉问答中复杂空间推理的难点

空间推理作为视觉问答系统所需的重点核心能力，涉及对图像中物体间拓扑关系、方位、尺寸等多种空间信息的理解、建模和推理。然而，现有方法在空间推理中仍面临诸多挑战。主要表现在以下几个方面：

复杂空间关系的建模与表示

空间推理需要处理多层次关系，如拓扑关系（包含、相邻）、方位（左/右、前/后）、动态轨迹（移动路径）等。传统方法依赖预定义逻辑规则（如 RCC-8 拓扑模型），但难以适应开放域场景的多样性[39]。例如，自然语言中的“靠近”这个词汇，缺乏量化阈值，难以定量界定两个物体之间的距离在什么场景下为“靠近”这一关系，导致符号逻辑无法精确映射[40]。另外，基于注意力机制的模型虽然能定位目标区域，但难以捕捉长距离或者隐含的空间关联，进而导致错误定位。

多模态对齐与语义鸿沟

视觉与文本模态的语义对齐，是 VQA 模型能够顺利进行空间推理的基础。但是存在跨模态特征映射的困难以及稀疏空间关系的问题。目前，有一些研究者试图解决这些问题。在跨模态特征映射这一问题上，Costanzino[41] 等人提出了一种使用轻量级多层感知器（MLPs）的方法，训练两个映射函数，基于正常样本预测一个模态的特征，通过比较实际特征和预测特征的不一致性来检测工业场景中的异常。在解决稀疏空间关系的问题方面，Zhang[42] 等人提出了 VoT 模型，通过 VoT 提示来提升 LLMs 的空间推理能力，通过生成视觉表示增强了模型对。这些方案在一定程度上解决了多模态对齐和语义鸿沟问题，但仍存在一定局限性。具体而言，多数模型通过全局或局部注意力加权融合多模态特征，但未显式建模空间关系的层次性。另外，一些模型通过子任务分解实现推理，但对空间逻辑的组合泛化能力有限。

动态场景与实时推理的挑战

动态场景（如移动物体的避障路径规划）要求模型实时更新空间状态，但现有方法存在增量式推理不足和数据驱动的固有局限。具体而言，现有方法往往依赖于离线训练的模型，无法实时更新空间状态，导致推理结果复杂空间推理任务滞后，无法满足实时推理的需求。例如，符号推理（如 ASP）需重新求解完整的逻辑程序，导致延迟较高<empty citation>。另外像一些基于监督学习的模型，依赖于静态数据集（如 CLEVR），无法适应动态环境中的连续变化。最新的一些研究成果，如微软提出的 MVoT 框架通过生成可视化中间推理步骤，实现了结合文本与图像信息背景下，对空间关系表示的动态调整，且在复杂场景中比传统思维链（Chain of Thought, CoT）的稳健性提升 20%。

数据集偏见与评估瓶颈

现有 VQA 数据集（如 CLEVR、VQAv2）存在显著偏差，如问题答案分布不均、问题类型单一等，导致模型在特定问题上表现优异，具体而言主要是对物体识别、属性描述等静态 VQA 任务表现良好，但在真实场景下泛化能力不足，因为在真实场景中，物体间的位置关系处于时刻变化之中。数据集需要能够对动态空间变化的推理能力进行测试。以 CLEVR 为代表的部分 VQA 数据集，其图像内容均为简单的几何图形，通过特定的脚本和渲染引擎进行生成，虽然其生成的图像可以是 3D 几何体图像，但相比现实生活场景而言，仍过于简单，无法模拟真实物理世界中的物体运动、视角变化等动态场景。有一些研究者也指出了这些问题。Shrestha[40] 等人指出，像 CLEVR 这一类合成的数据集，虽然能够测试多步逻辑，但其几何简单性无法反映真实世界的复杂性，并且在问题设计上也隐含对特定答案的倾向性，如“左侧”常与特定物体绑定，进而导致模型依赖表面统计规律，而并非真实推理。

目前，已有一些研究者使用零样本学习的方法，通过引入从未见过的问题-答案组合，测试模型的泛化能力。例如，Yang[43] 等人提出了一种零样本学习的方法，通过引入从未见过的问题-答案组合，测试模型的泛化能力。此外，新的一些 VQA 数据集如 VSR，通过控制答案分布来减少语言先验，以测试模型的纯视觉推理能力。

可解释性与鲁棒性不足

空间推理需要透明化的推理过程，以支持安全验证和决策解释。然而，现有的方法存在黑箱和对抗脆弱性等问题，无法提供可解释性保障。目前，有一些研究者试图在这一方面进行改进，如通过可视化路径的方式，提高模型的可解释性。例如，Li[44] 等人提出了多模态思维可视化框架（MVoT），旨在通过生成推理轨迹的图像可视化，增强多模态大语言模型（MLLMs）在复杂空间推理任务中的表现。该方法通过在自回归 MLLMs 中引入标记差异损失，显著提高了视觉连贯性和保真度。实验结果表明，MVoT 在多个任务中表现出的性能出色，尤其是在思维链（Chain of Thought，CoT）表现很差的场景中，展现出了显著的改进。Shah[45] 等人提出了一种基于循环一致性的训练框架，旨在增强 VQA 模型对语言变化的鲁棒性。该方法通过双向训练和循环一致性，提高了模型对语言变化的适应性，从而提高了模型的鲁棒性和泛化能力。

1.3 研究目标与内容

本课题的主要研究目标是研究并设计一种新的神经符号框架，通过实验，证明该框架能够有效提升大语言模型在复杂空间推理问题上的性能。最终将该神经符号框架接入视觉问答系统。

本文的主要研究内容包括以下三个方面：

(1) 构造了一个多模态语言模型。
(2) 研究融合大语言模型及 ASP 的神经符号流水线，并使用 DSPy 来进行实现。通过 DSPy，将 ASP 求解器与大语言模型实现系统集成。在本文构建的视觉问答数据集上，进行对比实验验证。此外，为了评估，还进行了消融实验。
(3) 以本文中所提出的神经符号框架为核心，设计并实现一个视觉问答系统。

1.4 研究方法与技术路线

本文针对以上研究目标和研究内容，综合多种方法进行研究。本文的研究主要涉及三个重点内容：

(1) 对于视觉问答数据集的构建，首先，用案例分析法研究 CLEVR 数据集，分析其特点和不足，为新数据集的构建提供理论依据和设计方向。然后采用文献研究法，调研现有的视觉问答数据集，学习它们在数据集构造过程中的方法和策略。
(2) 对于面向空间推理领域的神经符号框架的研究，首先，分析现有神经符号方法在空间推理方面的优势和不足。其次，进行模型构建。最后，通过实验验证，进行对比实验和消融实验，评估设计的神经符号框架的性能。
(3) 对于视觉问答系统的设计与实现，首先，进行需求分析，明确系统的功能和性能要求。其次，设计视觉问答系统的架构和模块划分，确定各模块实现所需的技术方案。再次，基于前文的研究成果，实现视觉问答系统的各个模块，并进行系统集成和测试。最后，通过实验验证，评估视觉问答系统的性能和可用性。

具体的技术路线如图1.3所示。

1.5 论文结构

本文共分为六个章节，各章节的主要内容具体如下：

第一章为绪论，总体介绍本文的研究背景及意义、相关研究现状与不足、研究目标与研究内容、研究方法与技术路线及本文的结构安排。

第二章为背景知识，对本文涉及到的主要技术进行介绍，具体包括 ASP 程序语法及 ASP 求解器、GLIP 以及 DSPy。

第三章为数据集构建。对本文所用的视觉问答数据集进行详细介绍，包括数据集的构造目的、数据集的研究方向、数据集的设计流程以及对数据集质量的验证。

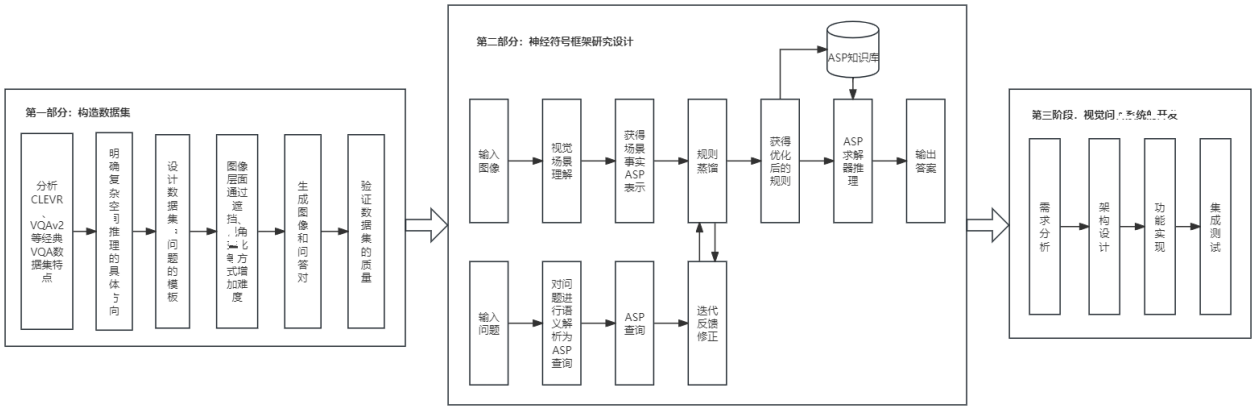


图 1.3: 技术路线图

第四章为神经符号框架的研究设计。对本文设计的神经符号框架进行详细介绍，包括流水线总体架构、视觉场景理解、语义解析、知识蒸馏、迭代反馈和规则修正、ASP 推理等模块的设计。

第五章为实验及结果分析。通过一系列实验，验证本文所提出的神经符号框架对复杂空间推理任务的提升效果，以及其在不同大语言模型架构上的泛化能力。

第六章中对本文工作加以总结，分析本文的创新点和不足之处，并对未来的研究方向进行展望。

第二章　背景知识

2.1　回答集编程

2.1.1　语法

项（terms）是 ASP 程序中最基本的元素，项可以是常量、变量或者函数。常量以符号常量或者数字表示（如 4、a）。变量以字符串来表示，要求首字母必须大写（如 Dog、Person）。函数由函数符号与若干参数构成，例如 $f(t_1,...,t_n),(n \geq 0)$。各参数 $t_i,(i=1,2,...,n)$ 也是项。若项中不存在函数符号与变量，则称该项为实例化项，否则称该项为非实例化项。

原子（atom）由谓词和项共同组成，例如 $q(t_1,...,t_n),n \geq 0$。其中，q 为 n 元谓词的标识符，$t_1,...,t_n$ 为项，n 为 0 时，谓词之后的括号可以省略。原子用于表示不同项之间的关系，例如：$dog(animal)$、$family(jensen,lucy)$、$tomorrow\_sunny$，分别表示狗是动物、jensen 和 lucy 是家人以及明天晴天。如果原子中的每一项都是实例化的，则该原子是实例化原子，否则，该原子是非实例化的原子。例如，$dog(animal)$ 是实例化原子，$family(jensen,lucy)$ 是非实例化原子。原子 $q(t_1,...,t_n)$ 的强否定形式为 $\neg q(t_1,...,t_)n$，其中符号 $\neg$ 是经典逻辑中的否定，$\neg q(t_1,...,t_n)$ 表示各项 $t_i(1 \leq i \leq n)$。不符合 q 所描述的关系，$\neg\neg a = a$，与经典逻辑中的排中律相同。文字（literal）可以是原子 $q(t_1,...,t_n)$，也可以是原子的强否定形式 $\neg q(t_1,...,t_n)$。若文字对应的原子是实例化的，则称该文字是实例化的。

定义 2.1.1 (规则). 规则 r 是具有如下形式的式子：

$$l_0\, or ...\, or\, l_m \leftarrow l_{m+1},...,l_n\, not\, l_{n+1},...,\ not\, l_p. \tag{2.1}$$

其中 $p \geq n \geq m \geq 0, l_i$ 代表文字，not 为新的逻辑连接符，通常称作缺省否定符（default negation）或者失败即否定（Negation As Failure, NAF），又称为若否定，$not\, l_i$ 被读作“没有理由相信 l_i 为真”，但是这并不表明 l_i 为假。对 ASP 程序而言，$notnota$ 不与 a 等价。规则 r 读作“如果相信 $l_{m+1},...,l_n$ 为真，并且没有理由相信 $l_{m+1},...,l_p$ 为真，则相信 $l_0\, or\, ...\, l_m$ 为真”。将 $notl_i$ 称为缺省文字（default literal），文字与缺省文字共同组成拓展文字（extended literal）。

规则 r 左侧的文字称为头部，可以表示为 $head(r)=\{l_0,...,l_m\}$。规则 r 右侧的文字称为体部，可以表示为 $body(r)=\{l_{m+1},...l_n,not\, l_{n+1},...,not\, l_p\}$，规则的体部可以分为正体部与负体部，正体部表示为 $body^+(r)=\{l_{m+1},...,l_n\}$，负体部表示为 $body^-(r)=\{l_{n+1},...,l_p\}$ 。因此，规则 r 可以表示为：

$$head(r)\ \leftarrow\ body^+(r),\ notbody^-(r) \tag{2.2}$$

一个 ASP 程序是有限条(2.2)所示的规则组成的集合。根据规则各部分所满足的相关条件，可以分别定义事实、约束、正规规则、缺省规则和严格规则，如定义(2.1.2)-定义(2.1.6)所示。

定义 2.1.2 (事实). 当规则 r 满足 $head(r) \neq \emptyset, body(r)=\emptyset$ 时，被称为事实（*fact*）。

定义 2.1.3 (约束). 当规则 r 满足 $head(r)=\emptyset, body(r) \neq \emptyset$ 时，被称为约束（*constraint*）。

定义 2.1.4 (正则规则). 当规则 r 满足 $|head(r)| = 1$ 时，被称为正则规则（*normal rule*）。

定义 2.1.5 (缺省规则). 当规则 r 满足 $body(r)^- \neq \emptyset$ 时，被称为缺省规则（*default rule*）。

定义 2.1.6 (严格规则). 当规则 r 满足 $body(r)^- = \emptyset$ 时，被称为严格规则（*definite rule*）。

根据上述各类规则的定义，本文进一步定义简单逻辑程序、缺省逻辑程序和正规逻辑程序如下。

定义 2.1.7 (简单逻辑程序). 若程序 P 由有限条严格规则组成，则 P 被称为简单逻辑程序。

定义 2.1.8 (拓展逻辑程序). 若程序 P 由有限条缺省规则组成，则 P 被称为拓展逻辑程序。

定义 2.1.9 (正则逻辑程序). 若程序 P 除约束外，由有限条正则规则组成，则 P 被称为正则逻辑程序（*Normal Logic Program, NLP*）。

若不做特别说明，本文考虑的程序均为不含约束的正规逻辑程序，即程序中的每条规则头部有且仅有一个文字。

2.1.2 语义

文字的可满足性语义解释和程序的模型

作为基础理论，首先给出 Herbrand 域、Herbrand 基的定义，分别如定义 2.10 和 2.11 所示，进一步定义规则与程序的实例化，最终在实例化程序下讨论文字的可满足性语义解释和可满足性。

定义 2.1.10 (Herbrand 基). P 是一个 *ASP* 程序，将程序 P 中出现的常量和函数形成的所有不含变量的项的集合称为 *Herbrand* 域，使用 $\mathcal{U}_P$ 表示。

定义 2.1.11 (Herbrand 域). P 是一个 *ASP* 程序，出现在程序 P 的规则中的谓词符号与 $\mathcal{U}_P$ 中的项组成的所有可能的不含变量的原子集合称为程序 P 的 *Herbrand* 基，使用 $\mathcal{A}_P$ 表示，为了保持简洁，通常简写为 $\mathcal{A}$。

定义 2.1.12 (规则的实例化). 给定 P 中的一个规则 r，使用 $ground(r)$ 表示通过使用 $\mathcal{A}_P$ 中的常量对应地替换 r 中变量而获得的规则集合，这一映射被称为规则的实例化。

定义 2.1.13 (程序的实例化). 程序 P 由一系列规则的集合组成，程序 P 的实例化 $ground(P)$ 表示为式(2.3):

$$ground(P) = \bigcup_{r \in P} ground(r) \tag{2.3}$$

定义 2.1.14 (文字的可满足性语义解释). 定义文字的可满足性语义解释 $I = \langle I^+, I^- \rangle$, 其中 $I^+ \cup I^- \subseteq \mathcal{A}_P$ （P 是实例化程序），$I^+ \cap I^- = \emptyset$。I^+ 表示为已知为真的文字集合，I^- 表示已知为假的文字集合。当 $I^+ \cap I^- = \mathcal{A}_P$ 时，I 被称作完全可满足性语义解释（*complete interpretation*）。若 I 不是完全可满足性语义解释，则 I 中存在真值未确定（*undefined*）的文字。

定义 2.1.15 (程序的模型). 假设 P 是一个实例化的回答集程序，$I = \langle I^+, I^- \rangle$ 是一个可满足性语义解释。若文字 $l \in I^+$，则称 I 满足 l，记作 $I \models l$；对缺省文字 $notl$，若 $l \in I^-$，则称 I 满足 $notl$，记作 $I \models notl$。对于文字集合 S，若 $\forall l \in S, I \models l$, 则称 I 满足集合 S，记作 $I \models S$，当一个规则 r 满足 $I \not\models body(r)$ 或 $I \models head(r)$ 时，I 满足规则 r。当 I 满足程序 P 的所有规则时，称 I 是 P 的一个模型。

回答集语义

定义 2.1.16 (一致性文字集合). 给定一个文字集合 S，若 S 中不同时包含 l 和 $\neg l$，则集合 S 是一致性文字集合，其中，$l \in S$ 是集合 S 中的任意文字。

定义 2.1.17 (可满足性，Satisfiability). 给定一致性文字集合 S，文字 lit 与规则 r，若 $lit \in S$，则文字 lit 满足集合 S。若 $head(r) \cap S \neq \emptyset$，则 S 满足头部。若 $body^+(r) \subseteq S, S \cap body^-(r) = \emptyset$，则集合 S 满足体部。若 S 满足规则 r 的头部或者 S 不满足规则 r 的体部，则 S 满足规则 r，记作 $S \models r$。给定一个 *ASP* 程序 P，若 $\forall r \in P, S \models r$，则 S 满足程序 P。

定义 2.1.18 (规则的适用与阻塞). 给定一致性文字集合 S 与规则 r，S 满足 $body(r)$，则称规则 r 在集合 S 下是适用的（*applicable*），否则称规则 r 在集合 S 下被阻塞（*block*）。

例 2.1.1. 给定 *ASP* 程序 P_3，该程序包含两条规则：

$$r_1 : p \leftarrow q, s.$$
$$r_2 : s \leftarrow t.$$

下面通过分析说明集合 $S = \{p, q\}$ 对程序 P 的可满足性。

(1) 因为 $p \in S$，所以文字 p 与文字 q 均满足 S；
(2) 因为 $S \cap head(r_1) = \{p\} \neq \emptyset$，所以 S 满足 $head(r_1), S \models r_1$；
(3) 因为 $body^+(r_2) \subsetneq S$，所以 S 不满足 $body(r_2)$，所以 $S \models r_2$；
(4) 因为 S 满足程序 P 中的每一条规则，所以 S 满足程序 P。

定义 2.1.19 (Gelfond-Lifshitz 规则（GL 规约）). 假设程序 P 是给定的实例化程序，S 是一致性文字集合，l 是文字，$S \subseteq Lit(P), l \in Lit(P)$, P 关于 S 的 GL 规约结果 P^S 定义为式(2.4)：

$$P^S = \{head(r) \leftarrow body^+(r) \mid r \in P, body^-(r) \cap S = \emptyset\} \tag{2.4}$$

定义 2.1.20 (简单逻辑程序回答集[<empty citation>]). 假设 P 是简单逻辑程序，程序 P 的回答集是 $S \subseteq Lit(P)$ 满足：

(1) 对于程序 P 中的每一条规则 $l_0 \leftarrow l_1, ..., l_m$，如果 $l_1, ..., l_m \in S$，那么 $l_0 \in S$。
(2) $S \subseteq Lit(P)$ 且不存在 S 的任何子集也满足程序 P 的每一个规则，即 S 是满足程序 P 的最小集合；
(3) 如果 S 包含互补的文字，那么 $S = Lit(P)$；

定义 2.1.21 (拓展逻辑程序回答集). 假设 P 是包含缺省否定的扩展逻辑程序，S 是实例化文字集合，如果 S 是 P^S 的回答集，则 S 是 P 的回答集。如果程序 P 没有回答集或者只有一个包含互补文字的回答集 $Lit(P)$，那么程序 P 是不一致程序，否则，程序 P 是一致性程序。

可以看出，使用 *GL* 规约，可以将扩展逻辑程序程序转换为简单逻辑程序，从而得到扩展逻辑程序的回答集。下面通过一个例子说明。

例 2.1.2 (GL 规约与问答集). 给定一个实例化的 *ASP* 程序 P_4：

$$r_1 : p(a) \leftarrow notp(b)$$
$$r_2 : p(b) \leftarrow notp(a)$$
$$r_3 :\leftarrow p(b).$$

程序 P_4 可能的回答集有四个：$S_1 = \emptyset, S_2 = \{p(a)\}, S_3 = \{p(b)\}, S_4 = \{p(a), p(b)\}$, 根据回答集的定义依次验证：

(1) $P^{S_1} = \{p(a) \leftarrow .\} \cup \{p(b) \leftarrow .\} \cup \{\leftarrow p(b).\}$，事实 $(p(b) \leftarrow .)$ 与约束 $(\leftarrow p(b).)$ 同时在 P^{S_1} 中出现，因此 S_1 不是 P^{S_1} 的回答集，所以 S_1 不是 P 的回答集。

(2) $P^{S_2} = \{p(a) \leftarrow .\} \cup \{\leftarrow p(b).\}$，而 $S_2 \models ((p(a) \leftarrow .))$ 且 $S_2 \models (\leftarrow p(b).)$，因此 $S_2 \models P^{S_2}$，且不存在 $S_2' \subset S_2, S_2' \models P^{S_2}$，所以 S_2 是 P^{S_2} 的回答集；

(3) $P^{S_3} = \{p(b) \leftarrow .\} \cup \{\leftarrow p(b).\}$，事实 $(p(b) \leftarrow .)$ 与约束 $(\leftarrow p(b).)$ 同时在 P^{S_3} 中出现，因此 S_3 不是 P^{S_3} 的回答集，所以 S_3 不是 P 的回答集；

(4) $P^{S_4} = \{\leftarrow p(b).\}$，可知 P^{S_4} 的回答集为 $\emptyset$，因此 S_4 不是 P^{S_4} 的回答集，所以 S_4 不是 P 的回答集。

well-founded 语义

上世纪 90 年代，良基语义（well-founded semantics）的概念由 Van Gelder A 提出，对于任意的回答集程序，都存在对应的良基模型（well-founded models），并且在多项式时间内可以计算出这个模型[46]。该语义模型是本文对 NAF 文字进行解释和对不一致程序原因进行分类的重要基础，下面介绍这一模型的定义及计算方法。

定义 2.1.22 (立即结论). P 是一个 *ASP* 程序，$S \subseteq \mathcal{A}_P \square V \subseteq \mathcal{A}_P$，则集合 S 关于 P 和 V 的立即结论（*immediate consequence*）记作 $\mathcal{T}_{P,V}(S)$，定义如式(2.5)所示：

$$T_{P,V}(S) = \{a | \exists r \in P, head(r) = a, body^+(r) \subseteq S, body^-(r) \cap V = \emptyset\} \tag{2.5}$$

考察上式可以发现，当集合 V 确定时，关于集合 S 的函数 $T_{P,V}(S)$ 是单调的。因此，当集合 V 固定时，该函数存在最小不动点（Least Fixed Point, LFP），使用 $lfp(.)$ 表示。

定义 2.1.23 (良基模型). P 是一个 *ASP* 程序，P^+ 是程序 P 中的全部严格规则的集合，序列 $(K_i, U_i)_{i \geq 0}$ 定义如下：

$$K_0 = lfp(T_{P^+}) \qquad K_i = lfp(T_P, U_{i-1})$$
$$U_0 = lfp(T_{P,K_0}) \qquad U_i = lfp(T_{P,K_i})$$

若 $\langle K_j, U_j\rangle = \langle K_{j+1}, U_{j+1}\rangle$，且不存在 $0 \leq k < j$，使得 $\langle K_k, U_k = \langle K_{k+1}, U_{k+1}\rangle$（j,k 是正整数），则 P 的良基模型 $WF_P = \langle W^+, W^-\rangle$，满足 $W^+ = K_j$，$W^- = \mathcal{A}\backslash U_j$

例 2.1.3 (良基模型). 程序 P_5 包含以下四条规则：

$$r_1 : q \leftarrow, notp. \qquad r_2 : p \leftarrow, notq.$$
$$r_3 : a \leftarrow b. \qquad r_4 : b \leftarrow .$$

首先得到 $P_5^+ = \{a \leftarrow b\} \cup \{b \leftarrow .\}$，而 $lfp(T_{P_5^+}) = a, b$，因此 $K_0 = a, b$；进一步计算得到 $U_0 = lfp(T_{P_5^+, K_0}) = \{a, b, p, q\}$，$K_1 = lfp(T_{P_5^+, U_0}) = \{a, b\} = K_0$，$U_1 = lfp(T_{P_5^+, K_1}) = \{a, b, p, q\} = U_0$。因此 $\langle K_0, U_0\rangle = \langle K_1, U_1\rangle$ 且不存在 $0 \leq k < 1$，$\langle K_k, U_k\rangle = \langle K_{k+1}, U_{k+1}\rangle$。因此程序 P_5^+ 的良基模型为 $WF_{P_5^+} = \langle \{a, b\}, \{\}\rangle$，$p$ 和 q 在良基模型中属于未定义（$undefined$）。

2.1.3　求解器

求解器是实现 ASP 语义推理的核心工具，其工作原理是：通过搜索逻辑程序的最小模型（即回答集）来解决复杂的组合优化问题。

求解器的工作流程可以划分为两个阶段：基础化（Grounding）和求解（Solving）。基础化是将一阶逻辑程序实例化为命题逻辑形式，而求解则是基于 CDCL 算法搜索满足约束的模型。

截至目前，已经有众多 ASP 求解器。学术界和工业界所用的主流的求解器，可分为两类：单阶段求解器（如 Clasp、DLV）、多阶段求解器（如 Clingo）。两类求解器的主要区别在于，单阶段求解器独立地进行基础化与求解，而多阶段求解器则是集成了基础化与求解的交互式系统。

表2.1对目前的主流求解器的适用场景及核心优势进行了对比。因为 Clingo 求解器具有免费开源、使用简单、运行高效的特点，本文在求解 ASP 程序时，使用 Clingo 进行相关实验。

2.2　GLIP

GLIP（Grounded Language-Image Pretraining）是一种融合视觉与语言的预训练模型，旨在提高模型在目标检测、图像理解以及跨模态任务上的能力。GLIP 通过对大规模的图像-文本对数据进行联合训练，使得模型能够理解自然语言查询，并在图像中定位相应的目标，从而实现零样本（zero-shot）或少样本（few-shot）的目标检测任务。其架构如图2.1所示。

GLIP 采用了大规模的图像-文本数据进行预训练，使用基于 Transformer 的架构，并结合视觉和语言的特征进行跨模态学习。其训练过程主要包括：（1）文本引导的目标检测：输入不仅包含图像，还包含文本描述，使得模型能够基于自然语言查询来识别目标。（2）跨模态对齐：通过对比学习和注意力机制，使得视觉特征与语言特征在共享的表示空间中进行对齐。（3）自监督学习：利用大规模无标注数据进行自监督学习，提高模型的鲁棒性和泛化能力。

相比以往的目标检测技术，GLIP 的突出优势在于：（1）零样本检测能力：无需重新训练，即可识别新的类别。（2）泛化性强：由于使用了开放词汇的文本描述，GLIP 在实际应用中更加灵活。（3）跨领域适应性：可以应用于医学影像分析、自动驾驶、机器人视觉等多个领域。

求解器	核心技术	适用场景	核心优势
Clasp	多线程 CDCL 情性子句学习	大规模组合优化 工业级应用	支持并行搜索 性能竞赛冠军
Clingo	增量式基础化 多阶段编程	动态规则生成 交互式推理	集成 Python API 支持在线更新程序
DLV	数据库优化 存在线量化推理	知识表示型问题 语义 Web	高效处理复杂规则 支持非确定域
WASP	分层弱约束优化 冲突驱动剪枝	偏好推理 多目标优化	加权约束高效处理
Smodels	稳定模型算法 部分求值	教学与研究 基础模型验证	算法透明易扩展
LPARSE	基础化预处理	规则实例优化	与 Smodels 配合使用
IDP	基于模型的推理 扩展一阶逻辑	复杂知识库管理	支持类型系统 高阶推理
Alpha	并行求解引擎	超大规模问题	GPU 加速搜索

表 2.1: 扩展后的系统对比

目前，GLIP 在多个计算机视觉任务中表现优异，典型的应用包括：开放词汇目标检测（Open Vocabulary Object Detection, OVOD）、跨模态信息检索、复杂视觉问答（Visual Question Answering, VQA）、自动标注和数据增强。

2.3 大语言模型

2.3.1 概述

大语言模型（Large Language Model，LLM）是自然语言处理（Natural Language Processing, NLP）领域的重要研究方向。 它们通过在大规模文本数据上进行训练，学习语言的复杂模式和结构，能够生成连贯且有意义的文本。 这使得 LLM 在机器翻译、文本摘要、问答系统等任务中表现出色。

目前，研究和应用领域主流的 LLM 有 GPT、Llama、DeepSeek。下面本文将对以上几种主流模型的特点和适用任务展开介绍。

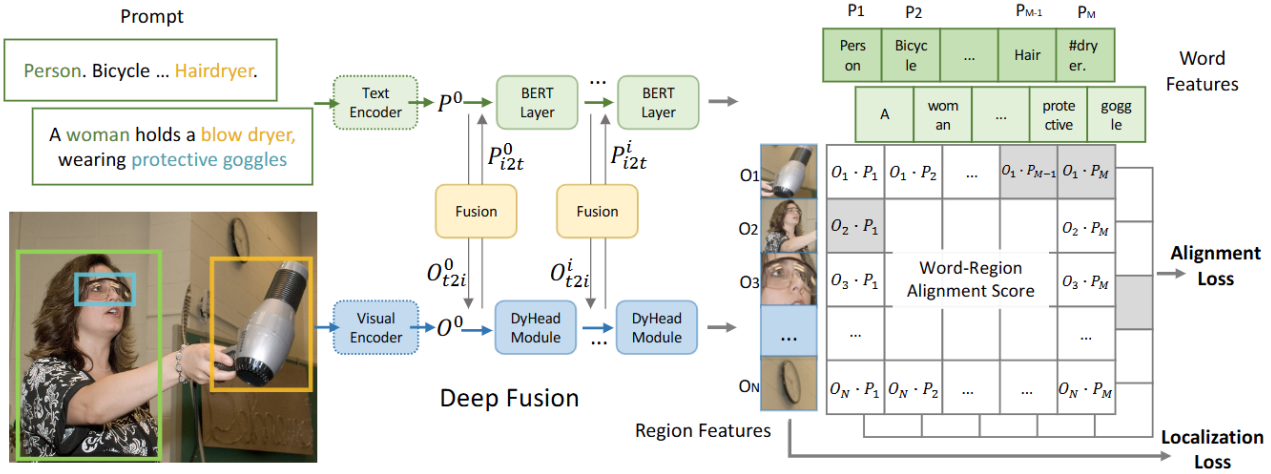


图 2.1: GLIP 结构示意图

2.3.2　GPT

目前，GPT 系列最新的模型为 GPT-4o，其在架构方面采用了高达 1.8T 参数的多模态 Transformer，引入联合注意力机制，支持文本-图像-语音跨模态联合推理。此外，GPT 采用了动态窗口扩展技术，长文本生成连贯性指标提升 37%。

训练策略上，采用了强化学习对齐（RLHF），通过 3 阶段训练（SFT → RM → PPO），有害内容生成率降至 0.3%。并且采用了 数据蒸馏的方案，其使用合成数据占比达 45%，有效减少了网络噪声数据污染。

GPT 目前在 多模态理解方面表现优异，图像描述任务 BLEU-4 达 0.74，视频内容摘要准确率 91%。然而 GPT 训练成本高昂，价格较高，每百万 token 费用 7.5 美元，是 DeepSeek 的 53 倍。

2.3.3　Llama

Llama 是由 Meta 开源的 LLM，其在架构上采用 分组查询注意力（GQA），键值头数量压缩至查询头的 1/4，70B 模型推理显存需求降至 48GB。此外，还采用了 RoPE 位置编码，动态旋转嵌入技术提升长文本连贯性，32k 上下文下困惑度降低 12%。

训练策略上，Llama 采用了课程学习策略，分阶段增加数据复杂度，代码生成任务准确率提升 19%。此外，Llama 还支持 量化部署，4bit 量化版可在消费级显卡（RTX 4090）运行，大大降低了模型的部署应用门槛。

Llama-3 整合 CLIP 视觉编码器，使其在多模态领域具有较强的应用能力，图像问答 VQA 准确率 82%。

2.3.4　DeepSeek

在架构层面上，DeepSeek 采用了混合专家架构（MoE）、多头潜在注意力（MLA）以及动态路由策略，以更好地对训练过程进行优化，降低对硬件的需求。其采用细粒度专家分割（256 个路由专家 +1 个共享专家），激活参数仅占全量参数的 1/10，推理速度提升 3 倍。此外，

DeepSeek 还通过低秩键值压缩技术，将 KV 缓存需求降低 60%，支持 64k token 长上下文处理（约 3-4 万字）。基于负载均衡优化算法，专家利用率提升 30%，在代码生成任务中 HumanEval 得分达 53.8%。

在训练优化上，DeepSeek 使用了 FP8 混合精度训练的方案，结合 BF16 与 FP8 量化，单卡 H800 训练吞吐量提升 2.1 倍，175B 模型训练成本降至 OpenAI 的 1/10。此外，也采用了强化学习优先（RL-First）的策略，DeepSeek-R1 通过纯强化学习训练实现复杂推理，数学证明任务准确率比 GPT-4o 高 15.4%。

相比目前其它的 LLM，DeepSeek 对硬件的需求更小，应用场景拓展到了移动端本地部署。此外，DeepSeek 的高推理速度也使得其被应用于实时风险分析领域，某银行系统响应延迟压缩至 200ms。

2.3.5 LLM 总结

目前，LLM 的产业化应用已渗透至许多行业，在金融、教育、法律等领域大显身手。

头部投行部署 GPT-4o 模型实现高频交易策略优化，实时解析财经新闻、财报电话会议录音，构建情感指数（Sentiment Index），预测标普 500 指数波动率的误差率仅 2.3%；通过强化学习动态调整投资组合，在 2024 年美股波动期间最大回撤控制在 5% 以内，优于人工策略的 12%；LLM 自动解析《巴塞尔协议 III》等法规文件，识别交易记录中的异常模式，某欧洲银行借此将反洗钱审查效率提升 40 倍。

在线教育平台“学而思”集成 LLaMA-3 模型开发智能辅导系统，根据学生答题数据（如错误知识点关联图）动态生成习题，某初中数学实验班平均成绩提升 23%；采用多维度评估框架（逻辑性、语法、创新性），与教师评分的一致性系数（Cohen’s Kappa）达 0.81；7×24 小时解答学科问题，在江苏省高三模拟考中，系统对物理压轴题的解题正确率超过 95%。

某顶级律所联合 DeepSeek-R1 模型构建法律智能平台，扫描百万字合同文本，自动标记非常规条款（如对赌协议中的隐藏风险），准确率 98.7%；基于历史案例数据库，对知识产权纠纷案的胜诉概率预测与法院判决吻合度达 89%；将法律意见书起草时间从 40 小时压缩至 2 小时，且符合《民法典》最新司法解释要求。

尽管 LLM 取得突破性进展，仍需解决以下问题。

1. 高质量语料短缺与数据多样性不足的问题。训练 GPT-4 和 Gemini Ultra 需要 4-8 万亿单词，而人类高质量文本数据可能在 2026 年前耗尽。中文语料尤其匮乏，全球大模型训练集中文占比仅 1.3%，且国内缺乏垂直领域专用数据集。此外，共享语料库存在大量噪声，例如社交媒体内容碎片化、标注不一致等问题，直接影响模型性能。例如，化工行业安全检查单需通过 LLM 重新筛选和优化低相关性条款。
2. 模型扩展定律（Scaling Laws）的失效。OpenAI 新一代模型 Orion 在训练进度 20% 时达到 GPT-4 水平，但后续投入更多资源后性能提升不足 5%，显示传统“堆参数”模式已遇瓶颈。此外，GPT-4 单次推理成本达数万次浮点运算，日均调用百万次的企业算力成本超数万美元，而模型能力提升与成本增长不成正比。
3. 幻觉（Hallucination）与可靠性缺陷问题。LLM 在缺乏数据支撑时可能编造虚假信息，例如谷歌 Bard 错误描述天文照片来源。金融、医疗领域因此面临业务风险，某法律分析

系统需人工审核降低误判率。

4. 推理效率与速率限制。GPT-4 推理需 40GB 显存，平均响应延迟超 3 秒，远超用户容忍阈值。异步函数调用技术（如 AsyncLM）通过并行化处理，可将任务加速 5.4 倍，缓解了这一问题，但仍需硬件优化。

2.4 DSPy

DSPy（Declarative Self-improving Python）是由斯坦福大学自然语言处理（NLP）实验室开发的开源框架。它实现了从传统提示工程到编程驱动范式的范式转变，专门用于基于基础模型的应用程序开发。通过模块化系统构建，DSPy 支持快速迭代并优化提示与模型权重，尤其擅长开发从简单分类器到复杂检索增强生成（RAG）流水线及智能体循环的多样化应用。

2.4.1 核心特性

声明式编程接口

DSPy 提供声明式 API，开发者可通过定义输入输出行为（如 question -> answer）构建 LM 流水线，无需深入设计复杂提示模板。这种抽象化设计大幅简化开发流程，提升代码可维护性。

自动优化机制

框架内置优化器（如 BootstrapFewShot、MIPRO），可自动调整提示模板或微调模型权重。例如，仅需少量标注数据即可通过编译（compile()）实现提示优化与参数更新，显著降低人工调试成本。

模块化系统设计

DSPy 将复杂任务拆解为可复用模块（如 ChainOfThought、ReAct、RAG），支持通过组合模块构建多阶段推理流程。例如，法律文档分析系统可分解为条款提取、风险识别、建议生成三个独立优化的子模块。

工具链集成

框架兼容 LangChain 等工具链，支持将现有 LECL（LangChain 表达式语言）封装为 DSPy 模块，实现提示优化与评估流程的无缝对接。此外，内置评估框架（dspy.Evaluate）支持自定义指标（如准确性、完整性）进行系统性验证。

2.4.2 应用场景与典型案例

智能问答系统

通过定义签名（如 context, question -> answer）和优化器，构建基于检索增强的问答系统。例如，使用 dspy.RAG 模块结合知识库实现高精度回答生成。

检索增强生成

DSPy 有助于创建将信息检索与文本生成相结合的 RAG 系统，从而提高生成内容的质量和相关性。

智能体与复杂推理

通过组合 ChainOfThought（思维链）与 ReAct（推理-行动循环）模块，构建支持多步骤决策的客服机器人。

2.4.3 DSPy 总结

将 DSPy 纳入论文研究，可为语言模型编程提供前沿视角。其三大创新——声明式编程（替代手工提示）、自动优化 （降低人工干预）、模块化架构 （提升系统可靠性）——有效解决了传统方法的脆弱性与低效问题。例如，法律文档分析系统的案例表明，DSPy 可将开发周期缩短 40%，同时输出稳定性提升 35%。这一框架为构建可扩展、高鲁棒性的 AI 系统提供了方法论基础，是探索下一代语言模型技术的理想工具。

2.5 本章小结

本章就开展本文研究所需的背景知识展开了详细介绍。首先就 ASP 展开介绍，并围绕其语法、语义、求解器展开详细叙述。其次，对本文所用的目标检测模型 GLIP 展开介绍。随后，介绍了目前主流的 LLM：GPT、Llama 和 DeepSeek，特别是各自的架构特点和适合的应用场景，并对当前 LLM 发展需要解决的问题和 LLM 目前已经应用的领域展开介绍。最后，本章就 DSPy 这一 LLM 开发的高效框架展开介绍，突出了其模块化设计和自动优化提示词的显著优势。

第三章　数据集

对模型的空间推理能力进行充分评估，高质量的数据集必不可少。本章将详细介绍如何构造用于空间推理 SRASP 数据集，并对数据集的质量进行测试。

3.1　设计目标

尽管 CLEVR 数据集至今仍被广泛应用于多模态模型的空间推理能力研究，但其存在着一些较为明显的问题，其中比较凸出的是，CLEVR 数据集中的场景是完全可观察的，模型可以直接从图像中获取所有必要信息。现实世界中，有很多信息并不能直接从图像中获取。人类基于图像对问题进行回答时，除了直接从图像中观察到的信息，往往也需要使用已经积累的先验知识。本文在此对空间推理的问题进行如下定义：如果回答该问题所需要的全部知识均已包含在图像中，即回答问题不需要使用先验知识，那么称该问题为完全可观察问题。同理，如果图像中并不包括回答该问题所需的全部知识，也即回答该问题需要使用已经积累的先验知识，那么称该问题为不完全可观察问题。

为了解决这一问题，本章基于 CLEVR 数据集，构建一个名为 SRASP 的数据集。该数据集的主要设计目标聚焦于以下几个方面：

(1) 考察模型对不完全可观察问题的解答能力。通过引入部分可观察性，模拟现实世界中物体被遮挡或隐藏的场景，要求模型在不完整的视觉信息下进行推理，考察模型真正解决问题的能力。
(2) 考察模型对推理密集型问题的解答能力。现实世界的同一场景中，往往存在多个物体，物体之间的关系多样，解决空间推理问题需要多个步骤，逐步解决。通过增大解决问题所需的推理跳数，可以确保模型无法通过简单的模式匹配或者直接观察得出答案，进而真正考察模型的思考和推理能力。
(3) 考察模型的知识整合能力。正如在数学证明中定理之间相互印证产生新的定理一样，回答问题所需要的知识，往往也是需要将现有知识进行结合产生新知识，才能最终解决问题。在 SRASP 数据集中，为每个场景提供一组逻辑约束作为背景知识，模型需将这些先验知识与观察到的视觉信息相结合，以生成正确的答案。

3.2　物体基本属性与约束

与 CLEVR 数据集中的几何体一样，SRASP 数据集的图像中的每个物体，均有形状、尺寸、材质、颜色四种属性。每种属性的可能取值如下：

(1) 形状：圆锥体、球体、圆柱体和立方体。
(2) 尺寸：小、中、大。
(3) 材质：橡胶、金属。

(4) 颜色：红色、蓝色、绿色、黄色、灰色、棕色、紫色、青色。

除上述四种基本属性之外，图像中的物体也有“所在区域”这一属性，可取值为 0、1、2、3。由于所有图像都被划分成 4 个区域，故图像中物体也会处在某一个区域之中。为了研究问题方便，本文规定每个物体只能在图像的一个区域中，不能同时跨多个区域。物体包含这一属性能够为指定约束提供便利。在本数据集中，约束是一组规则的集合，对场景生成与推理而言密不可分，其主要作用包括：（1）限制物体的属性组合，如对颜色、形状等进行限制；（2）定义物体之间的关联关系；（3）支持对遮挡物体的推理，当物体被遮挡时，通过约束可以缩小潜在答案的范围。

基于对物体的应用范围，约束可以划分为以下三类：

1. 区域约束：仅作用于特定区域的局部规则。例如，“区域 1 中所有物体形状必须为立方体”。
2. 跨区域约束：涉及多个区域的全局规则。例如，“区域 1 和区域 2 中同颜色物体的总数不超过 2 个”。
3. 全局约束：适用于整个场景的通用规则。例如，“所有物体必须属于至少一个区域”或“不允许存在完全相同的两个物体属性组合”。

约束通过使用 ASP 来进行表示。例如：

```
:- object(X), at(X, 0), not hasProperty(X, shape, cube), not hasProperty
   ↪(X, shape, cylinder).
```

表示如果 X 在区域 0，那么它的形状必须是立方体或圆柱体。

3.3 构造流程

在确定图像中物体的基本属性之后，本文进一步确定了构建数据集的如下步骤流程：

1. 生成一组由约束定义的环境，记作 $Environment_i$。
2. 生成一个完整的场景图 $Complete_i$。该场景图完全符合上一步生成的环境 $Environment_i$ 的要求。
3. 通过从完整场景图 $Complete_i$ 中删除一个物体 Obj_i，来生成一个部分场景图 $Partial_i$。
4. 生成一个对于部分场景图 $Partial_i$，对物体 Obj_i 的有关情况进行提问的问题 Q_i。

3.4 环境表示定义

SRASP 中的环境是由一组约束来定义的。与场景相比，环境是一个更抽象一层的概念。针对某个特定的环境，可以生成以其为模板的场景。可以理解为，环境是场景的抽象，场景是环境的实例。每个约束决定了特定环境下物体的属性限制。本文首先设计了 10 个约束模板，所有模板使用 ASP 来进行表示。每个环境最多通过 20 个不同的实例化后的模板来创建，也即每个环境中最多包含 20 个不同的约束。部分约束模板的 ASP 编码表示以及对应表示含义见表3.1。最终，一共生成了 50 个环境，数据集中的每个场景都是由其中一个环境进行实例化后生成的。环境的具体示例见附录.1.1。

模板	描述
模板 1（取值约束）	`:- object(X), at(X, R), not hasProperty(X, P1, V1).` 解释: 对区域 R 中的所有物体，它们 P1 属性的取值均为 V1。 具体实现: :- object(X), at(X, 0), not hasProperty(X, color, red).
模板 2（否定约束）	`:- object(X), at(X, R), hasProperty(X, P1, V1).` 解释：对区域 R 中的所有物体，它们的 P1 属性的取值，均不能为 V1。 具体实现：:- object(X), at(X, 0), hasProperty(X, material, metal).
模板 3（恰有 N 个约束）	`:- #count{X: hasProperty(X, P1, V1), object(X), at(X, R)} != N.` **解释**：在区域 R 中，恰好有 N 个物体的 P1 属性的取值为 V1。 具体实现：:- #count{X: hasProperty(X, size, small), object(X), at(X, R')} != 2.
模板 4（至少有 N 个约束）	`:- #count{X1, X2: sameProperty(X1, X2, P1), object(X1), object(X2), at(X1, R1), at(X2, R2)} < N.` 解释：在区域 R1 和区域 R2 中，至少有 N 对物体，它们的 P1 属性的取值都是 V1。 具体实现：:- #count{X1, X2: sameProperty(X1, X2, shape), object(X1), object(X2), at(X1, 1), at(X2, 2)} < 1.
模板 5（或约束）	`:- object(X), at(X, R), not hasProperty(X, P1, V1), not hasProperty(X, P1, V2).` 解释：区域 R 中的所有对象都具有属性 P1 的 V1 值或属性 P2 的 V2 值。 具体实现：:- object(X), at(X, 1), not hasProperty(X, color, yellow), not hasProperty(X, color, blue).

表 3.1: 部分约束模板示例

3.5　场景表示定义

场景是环境的实例。SRASP 数据集以场景图的形式表示场景，其节点表示使用其属性进行注释的对象，边表示对象之间的空间关系（前、后、左、右）。在 SRASP 中，除了场景图表示之外，也用 ASP 对场景进行表示。以下展示图**??**中部分场景的 ASP 表示：

```
%场景中的物体
object(0). object(1). object(2). object(3).

%物体的属性
```

```
at(0, 2).
hasProperty(0, color, green).
hasProperty(0, size, large).
hasProperty(0, material, rubber).
hasProperty(0, shape, cylinder).
....

%物体间的空间关系
front(1, 0). right(1, 0). ...
```

其中涉及到的谓词的功能如下：谓词 `object` 用于定义不同的物体（所有物体的名称用 0，1 等数字来表示）。谓词 `hasProperty(Object, Attribute, Value)` 用于将对象的名为 Attribute 的属性的值设置为 Value。对象之间的空间关系用谓词 `left`、`right`、`front`、`behind` 来表示，例如 `left(A, B)` 表示 B 位于 A 的左侧。

3.6 图像生成

图像生成基于前文中定义的场景图。场景图是一种对场景中物体、属性及其空间关系的结构化描述，而环境则由一系列约束条件所决定，这些约束可能包括空间分布、物体间的相对关系以及特定属性的限制。因此，场景图的生成问题可以归结为一个复杂的推理问题：在给定的环境约束（基于 ASP 的规则）以及场景中预期的物体数量 n 这两个前提下，完成以下任务：

1. **区域划分**：将每个物体合理分配到预定义的四个区域之一，确保满足诸如区域容纳量、相邻关系及物体间可能的干扰等约束；
2. **属性赋值**：为每个物体赋予颜色、尺寸、形状和材质等属性，其取值必须与环境中规定的约束条件一致。例如，某些区域可能只允许出现特定颜色或尺寸范围的物体；
3. **关系一致性**：在属性分配过程中，还需要确保各物体之间的关系（如邻近、对称或排斥关系）符合逻辑规则，从而保证场景图整体的合理性和一致性。

为了解决上述推理问题，本文采用了 ASP 的方法。ASP 作为一种声明性逻辑编程范式，特别适合解决复杂约束和组合优化问题。在本系统中，ASP 求解器的工作流程主要包括以下几个步骤：

1. **约束建模**：将场景的环境约束以及物体属性赋值规则形式化为 ASP 规则。此步骤需要充分利用逻辑公式来描述物体的空间位置、属性取值范围以及各类关系约束；
2. **回答集计算**：在输入了物体数量 n 和所有相关约束之后，ASP 求解器会计算出满足所有约束条件的解集。每一个答案集代表一种物体属性及区域分配的合理配置，即一种可能的场景图；
3. **解集筛选与随机采样**：由于满足所有约束的配置方案可能数量巨大，为了使后续图像生成过程具有一定的随机性与代表性，系统从所有可能的解集中随机采样一百万个场景

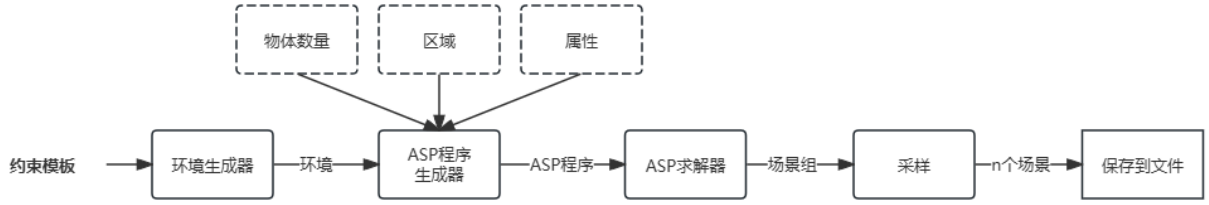


图 3.1: 生成环境以及该环境中的完整场景的流水线

图。这一随机采样策略既保证了生成场景的多样性，也为后续的图像生成提供了充分的候选数据。

在获得充分的场景图后，系统利用 Blender3 进行图像渲染。Blender3 是一款高效且功能丰富的三维渲染软件，它能够基于场景图的结构信息生成逼真的图像。渲染过程包括以下几个关键步骤：（1）场景搭建，根据场景图中各物体的位置信息及属性参数，在 Blender3 中自动搭建三维场景；（2）光照与材质设置，对各物体的材质、光照、纹理等进行设置，确保渲染出的图像在视觉上具有真实感；（3）图像生成，利用 Blender3 的渲染引擎，将搭建好的场景生成最终图像。

图3.1直观展示了从环境约束到场景图构建，再到图像生成的整个流水线过程，为后续工作的深入探讨提供了坚实的理论与实践基础。

3.7　问题表示定义

在 SRASP 数据集中，每个问题均围绕部分场景中缺失物体的四个关键属性之一：颜色、大小、形状和材质。问题的设计目标在于通过推理补全场景信息，从而考察系统在不完整场景下对物体属性关系的理解能力。为此，本文将自然语言描述的问题转化为基于 ASP 的形式化表示，使得问题的求解过程可以通过逻辑推理得到明确答案。

数据集中的每个问题最初以自然语言的形式提出，例如：“与中等大小的红色物体的材质相同的，另一个圆柱体的颜色是什么？”为了使问题具有可操作性，本文设计了对应的 ASP 编码，如下所示：

```
query(Q) :- hasProperty(X, color, Q),
hasProperty(X, shape, cylinder),

hasProperty(Y, size, medium),
hasProperty(Y, color, red),
same_material(Y, X),
X != Y.
```

在该编码中，`query(Q)` 表示需要推导出满足条件的物体 X 的颜色 Q。此外，通过多个 `hasProperty` 和 `same_material` 条件，明确限定了参与推理的物体之间的属性关系，并利用 X != Y 排除自反情况。这种表示方式不仅使问题的语义精确、结构清晰，而且便于通过 ASP 求解器进行自动求解，从而为数据集中的问题构建提供统一、标准的表达形式。

在问题生成过程中，必须对每个问题的答案范围进行严格控制。对于涉及属性 A（其中 $A \in \{color, size, material, shape\}$）的问题，其可能的解集 S 的大小满足 $1 \leq |S| \leq |A|$。其中，$|A|$ 表示属性 A 所有可能取值的数目。例如，对于尺寸属性，若其可能的取值集合 $\{large, medium, small\}$，则 $|size| = 3$。

如果某个问题生成的解集数量恰好为 $|A|$，例如问题答案为“尺寸可以为 large、medium 或 small”，则该问题在特定场景下并未起到区分或推断作用，因此被判定为无效。这一设计思路确保了数据集中每个问题在解答上都具有针对性和挑战性，从而避免生成普遍适用的、无区分价值的答案。

3.8 问题生成

在 SRASP 数据集中，每个问题均围绕部分场景中缺失物体的某一关键属性展开，如颜色、大小、形状或材质。为此，本文设计了一套模板，用以指导自然语言问题的生成。以下为问题模板样例：

```
What shape is the < Z2 > (size) < C2 > (color) < M2 > (material) [that is]
< R > (relation) the < Z > (size) < C > (color) < M > (material) < S > (
   ↪shape) ?
```

其中，<Z2>、<C2>、<M2> 表示待查询对象的已知属性（例如尺寸、颜色、材质），由随机策略从完整场景图中选取；<R> 为空间关系（如 left、right、front、behind），其取值既满足随机性，又依赖于完整场景中物体间的真实空间分布；<Z>、<C>、<M>、<S> 则代表参考对象的属性，通过对完整场景图中与查询对象具有特定空间关系的候选对象进行筛选而确定。这种模板化设计不仅使自然语言问题的结构化描述成为可能，而且便于后续转换为 ASP 的形式化表示，从而实现问题求解的自动化。

问题模板的实例化过程基于与图像对应的完整场景图。具体步骤包括：

1. 场景图构建与部分场景生成。利用完整场景图构建方法，将真实场景中的物体、属性以及空间关系进行抽象建模；从完整场景中随机移除一个物体，以构造部分场景图，此移除的对象即为“查询对象”，其缺失的属性将作为问题求解目标。
2. 已知属性的随机选取。对于查询对象，模板中的已知属性（例如 <Z2>、<C2>、<M2>）由随机采样策略确定，确保不同问题之间在属性分布上具有较好的随机性和代表性；同时，选取的属性应满足数据集整体的“问题类型平衡”要求，避免某一属性出现频率过高或过低。
3. 空间关系的确定。对于模板中表示空间关系的部分（<R>），取值虽然随机，但参考对象的选择依赖于完整场景图中的物体空间布局。具体而言，从与查询对象存在 <R> 关系的物体中进行候选对象的筛选，从而确保问题中提及的空间关系具有实际语义意义。

为了使问题具有可操作性和求解性，所有生成的问题均转化为 ASP 形式。转换过程中包括以下步骤：

1. 规则构建。将自然语言问题中的各项约束（属性约束、空间关系约束、对象排他性约束等）以 ASP 规则的形式表达；

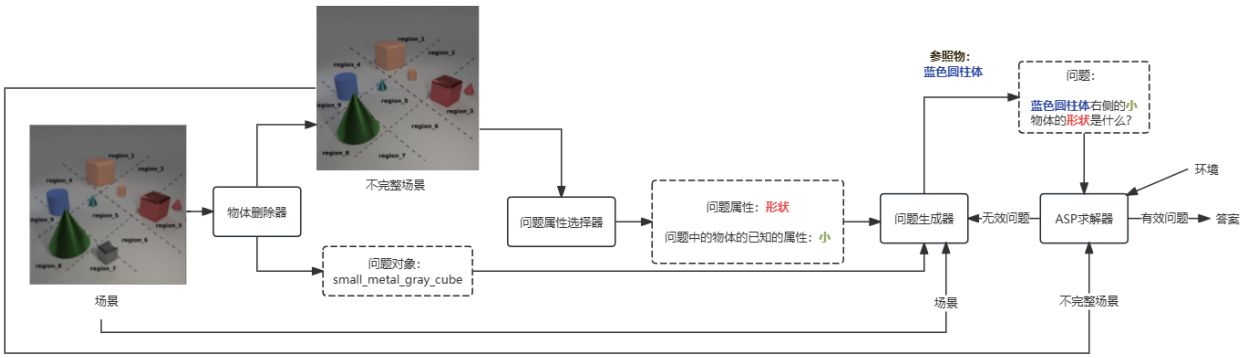


图 3.2: 生成部分场景和问题，并进行标记的流程

2. 约束整合。同时将部分场景图与环境约束作为求解器的输入，确保问题求解过程在完整逻辑下进行；

3. 解集判定。由 ASP 求解器计算出问题的所有可能解。针对每个属性 $A \in \{color, size, material, shape\}$，若生成的解集 S 满足 $1 \leq |S| \leq |A|$，则问题被认为具有合理的答案区分性；若解集规模等于 $|A|$ 说明答案涵盖了属性的全部取值，缺乏针对性，此时问题将被判定为无效并从数据集中剔除。

图3.2展示了从完整场景图构建、部分场景生成、问题模板实例化、ASP 表示转换，到最终问题求解的流水线过程。该流程通过随机采样，使生成的问题在属性组合和空间关系上呈现多样性；通过对解集规模的判定机制，有效过滤掉普遍适用或无区分意义的问题，确保每个问题都能够准确反映场景中物体属性的关系；使用模板化设计和 ASP 表示为后续数据集扩充与新问题类型的加入提供了灵活性和统一标准。

3.9　数据集分析

3.9.1　统计分析

问题模板的统计分布及问题数量在 5 到 9 之间的查询属性分布统计见图3.3。本数据集是基于 CLEVR 数据集进行生成的，在问题模板方面，采用了 CLEVR 数据集中的六种问题模板。此外，也展示了特定类型的问题在不同场景物体数量下的分布情况。

问题分布的统计图见图3.4。从统计图中可得知，有关颜色和形状的问题在 SRASP 数据集中占比最高，分别是 39% 和 37.6%，关于大小和材质的问题则相对较少，分别只占到了 13.1% 和 10.3%。此外，统计图中也展示了不同类型问题的答案集分布。在生成数据集的过程中尽量实现均衡分布，避免多数问题指向相同答案集的情况。例如，当问题涉及物体尺寸时，其潜在解可能为 {大、中}、{大、小}、{小、中}、{大}、{中} 或 {小}。

答案的分布统计。

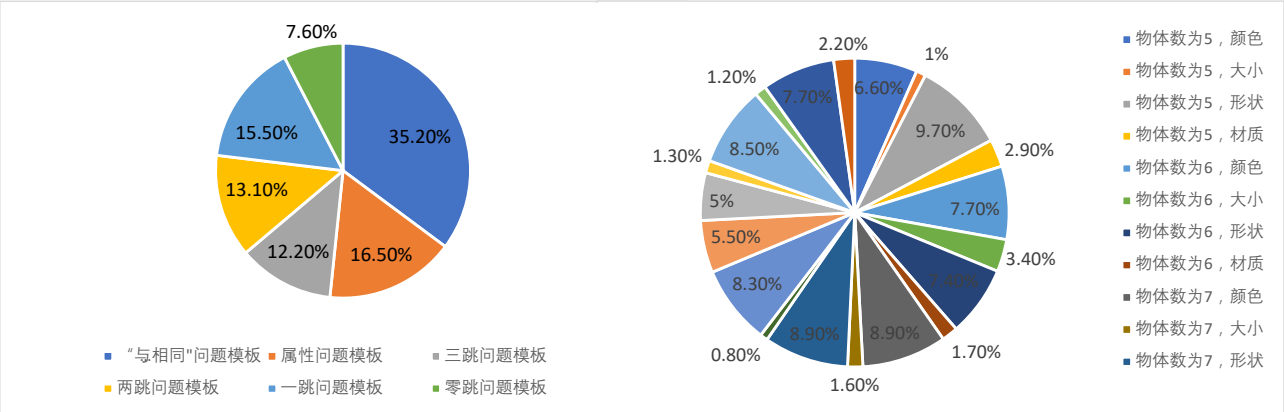


图 3.3: 问题模板统计及问题数量在 5 到 9 之间的查询属性分布统计

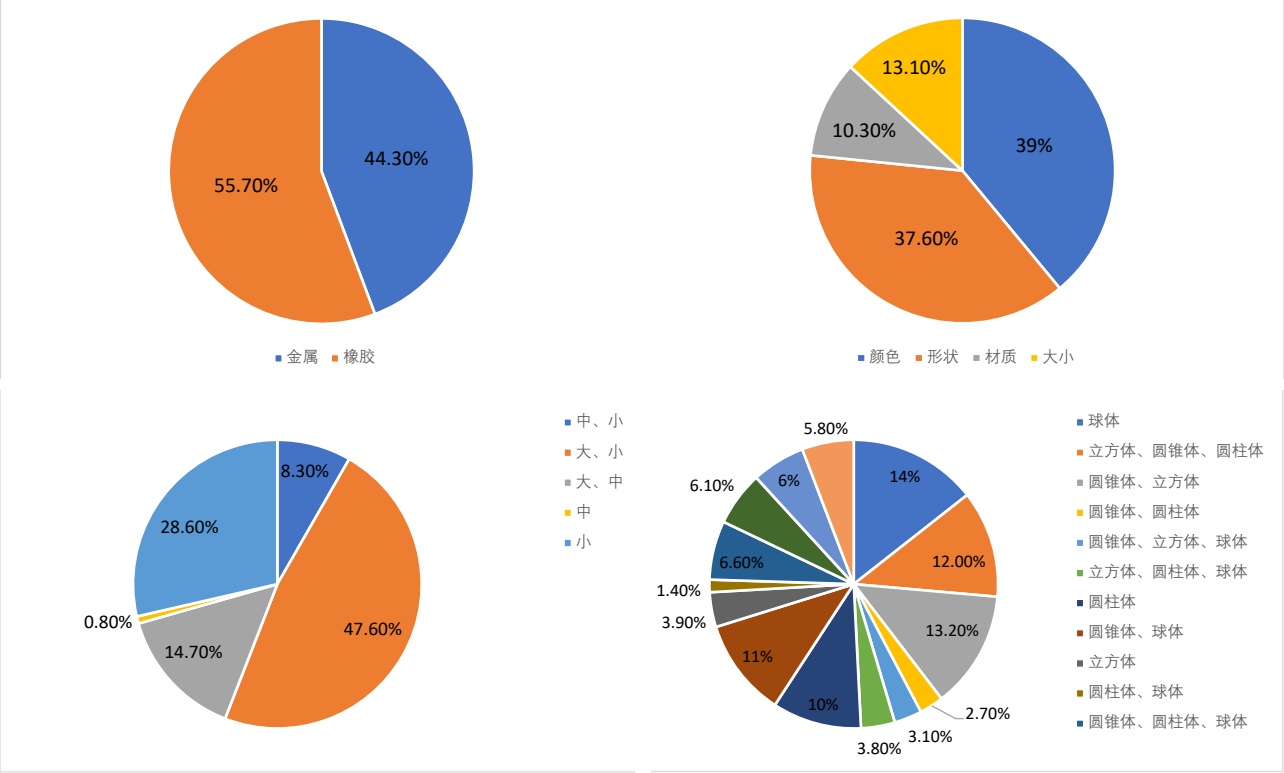


图 3.4: 问题分布统计图

3.9.2　质量分析

质量保证

为了保证数据集的质量，采用双盲审核机制，邀请两名评审员各自独立对数据集进行评审，验证每个问题的答案、推理解答所需步数及问题是否可观察。在评审过程中，如果两名评审员同时判定该问题出现错误，那么将该问题剔除。双盲评审的结果见表**??**，其中展示了评审员 1 和评审员 2 分别与初始标注的一致性，以及两名评审员之间的一致性。通过 Fleiss's Kappa 衡量的评审员间的一致性超过 0.8，证明了数据集的可靠性。

	答案是否正确	推理所需步数	问题是否可观察
初始标注与评审员 1	81.2	84.4	89.6
初始标注与评审员 2	84.2	85.6	85.4
评审员 1 与评审员 2	80.1	83.8	86.9
平均值	81.8	84.6	87.3

表 3.2: 评审员 1、2 和初始标注之间的标注者间一致性

难度保证

本文通过记录 2 位评审员在 SRASP 数据集上回答问题时的正确率、以及所需检索信息的次数、回答问题所需要的跳数，来对数据集的难度进行判断，并于其它现有的 VQA 数据集进行比较。实验结果见表**??**，其中明显可以看出，SRASP 回答问题所需的跳数，明显大于现有数据集 VQAv2，另外 SRASP 所需的检索信息的次数也更多一些，这些都证明了回答 SRASP 数据集所需的外部知识更多，考虑次数更多，数据集难度更大。另外，人类在本文构造的 SRASP 数据集上的准确率最低，进一步侧面印证了本文构造数据集的挑战性。

	回答问题正确率	推理所需跳数	检索信息次数
SRASP	81.2	84.4	89.6
CLEVR	84.2	85.6	85.4
GQAv2	80.1	83.8	86.9
平均值	81.8	84.6	87.3

表 3.3: 人类评审员在不同 VQA 数据集上回答问题的表现

3.10　本章小结

本章介绍了 SRASP 数据集的构造过程，重点描述了如何基于完整场景图生成部分场景图，并通过 ASP 实例化问题模板以确保问题的多样性和合理性。

首先，本章阐述了对象移除的原则，即如何选择查询对象以及如何保证其属性在问题类型上的均衡性。随后，详细说明了查询模板的填充策略，包括查询属性的选取、参考对象的

确定以及空间关系的合理性，以确保生成的问题符合实际场景。最后，本章介绍了 ASP 求解器在问题生成过程中的作用，即利用 ASP 规则对部分场景进行推理，以确定查询属性的可能取值范围，从而生成符合逻辑约束的高质量视觉问答数据。

通过上述方法，SRASP 数据集不仅保证了问题的可解释性，还增强了对复杂空间关系的推理能力，为视觉问答任务提供了更具挑战性的数据支持。

第四章　神经符号框架的架构设计与实现

4.1　引言

本章对神经符号框架的架构进行详细的介绍，包括流水线总体架构、视觉场景理解、语义解析、迭代反馈和规则修正、规则蒸馏、ASP 推理等模块的设计和实现。框架的目标如下：

(1) 增强 VQA 系统在复杂空间关系推理方面的能力。
(2) 使用 Dspy 来完成对 LLM 的提示、优化，以降低神经符号方法的开发难度，并增强其可扩展性。
(3)

4.2　框架总体架构

框架的总体架构如图4.1所示。其中，LLM 在整个框架中的作用有以下两点：

(1) 在语义解析模块中，LLM 对自然语言问题进行语义解析，将问题以 ASP 的形式表示出来，以便后续输入到迭代反馈模块中进行修正。
(2) 在迭代反馈模块中，LLM 对 ASP 表示进行多次迭代优化，其中包括解析错误、基础化错误等。每一轮迭代都将根据上一轮的错误信息进行针对性的修正。
(3) 在规则蒸馏中，LLM 结合 ASP 知识库中的先验知识，对迭代反馈修正后 ASP 规则进行进一步补全，为 Clingo 求解器推理提供更全面的知识。
(4) 在求解结果翻译中，LLM 对 Clingo 求解器输出的以 ASP 表示的结果进行翻译，使其以自然语言形式，针对提出的问题进行作答。添加规则，对规则进行一致性检查以及整合错误信息的反馈。

本文在语义解析模块和迭代反馈模块中，分别采用不同的 LLM，各自进行微调，以更好地满足不同任务的需求。

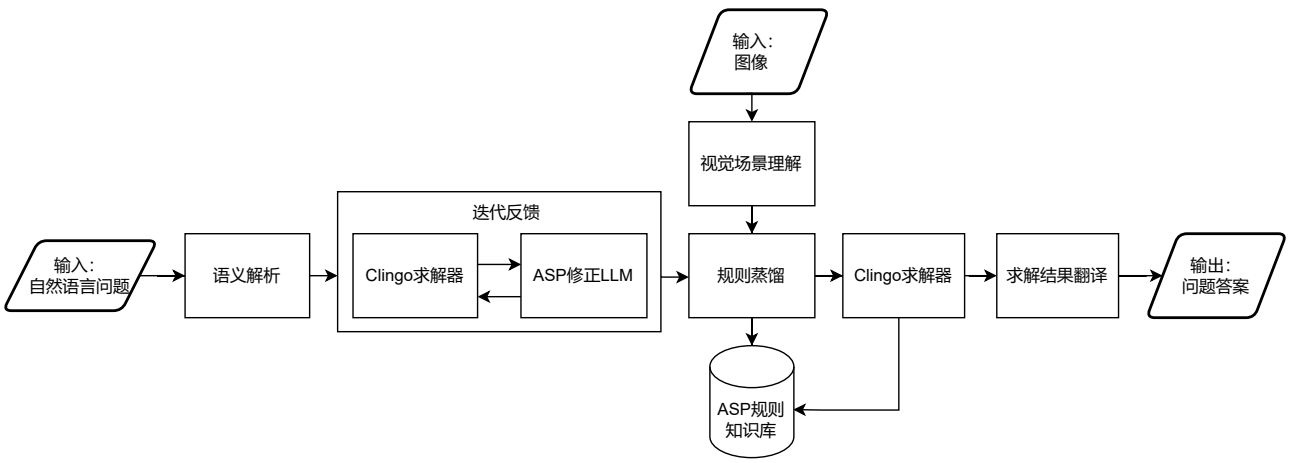


图 4.1: 面向空间推理领域的神经符号框架示意图

4.3 视觉场景理解

视觉场景理解是本文所设计流水线中的核心模块，其目标是从输入图像中提取结构化信息，包括物体的属性（如形状、颜色、大小等）、位置信息以及物体间的空间关系。这些信息将被进一步转换为 ASP 的形式，为后续基于逻辑推理的语义解析与问答模块提供形式化的知识基础。

本章主要围绕以下四个方面展开详细阐述：目标检测、空间位置提取、空间关系提取以及场景图生成。最后，本文给出一个完整的视觉场景理解演示案例，并讨论在实现过程中遇到的关键技术挑战及相应解决方案。

4.3.1 目标检测

目标检测是视觉场景理解的最基础的任务，其目标是从图像中识别出所有相关物体，并标注其边界框、类别和属性。本文选择 GLIP 模型作为目标检测的实现工具，原因在于其独特的语言-视觉预训练特性。基于大规模的图文对数据，GLIP 进行大量预训练，能够根据自然语言描述（如“红色立方体”）直接定位图像中的对应物体。这种能力特别适合 VQA 任务，使问题中的语言信息与图像内容高效对齐成为可能。

目标检测是视觉场景理解的基础任务，其任务是从输入图像中识别出所有相关物体，并为每个物体确定其边界框、类别和相关属性。本文采用了 GLIP 模型作为目标检测的核心工具，其主要优势体现在：

(1) **语言-视觉预训练能力**：GLIP 在大规模图文对数据上进行预训练，能够根据自然语言描述（例如“红色立方体”）直接定位图像中的对应物体，从而实现图像内容与问题描述的高效对齐；
(2) **高效性与鲁棒性**：模型在处理多样化场景时具有较高的检测准确率，适合构造后续依赖视觉信息的场景图。

具体实现流程如下：

(1) **图像预处理**：将原始图像调整至 GLIP 模型要求的输入分辨率（例如 800×1333 像素），以保证检测精度；
(2) **文本提示构造**：根据数据集构造要求，设计一组覆盖目标类别及属性的自然语言提示。本文选取的提示短语包括描述物体大小（如“大物体”、“小物体”）以及颜色与形状组合（如“红色立方体”、“蓝色球体”、“绿色圆柱体”等）；
(3) **目标检测与属性解析**：将图像和文本提示输入 GLIP，获得每个检测物体的边界框和类别标签。随后对类别标签进行解析，将复合描述（如“红色立方体”）拆解为单独属性（color=red, shape=cube）。同时，通过计算边界框面积（$(x_2 - x_1) \times (y_2 - y_1)$），结合预设阈值进一步推断物体的大小属性。

例如，对于一张包含“红色立方体”和“蓝色球体”的图像，GLIP 可能检测出如下信息：

1. 物体 1：类别为“红色立方体”，边界框坐标为 (x_1, y_1, x_2, y_2)；
2. 物体 2：类别为“蓝色球体”，边界框坐标为 (x_3, y_3, x_4, y_4)。

4.3.2　空间位置提取

在完成目标检测后，接下来的任务是为每个物体确定其精确的空间位置，这对于后续空间关系的推理至关重要。本文采用以下两种方式提取物体位置信息：

1. **二维中心点计算**：利用目标检测得到的边界框信息，计算物体在图像平面内的中心点坐标。公式如下：
$$x_c = \frac{x_1 + x_2}{2}, y_c = \frac{y_1 + y_2}{2}$$
该中心点坐标用于描述物体在二维图像中的位置；

2. **三维位置信息获取**：若图像同时包含深度信息（例如通过 Blender 渲染生成的场景），则可直接从深度图中提取物体的 z 值，从而获得物体在三维空间中的位置，表示为 (x, y, z)。

提取出的位置信息将以 ASP 事实的形式进行存储，例如：position(obj1, x1c, y1c, z1) 表示物体 1 的中心点位置。

4.3.3　空间关系提取

空间关系的提取是实现复杂场景理解与多步逻辑推理的关键。本文从以下几个角度对空间关系进行提取：

(1) **二维空间关系**：基于物体的中心点坐标计算物体之间的相对位置。例如：

- 若物体 A 的 x_c 小于物体 B 的 x_c，则判定 A 位于 B 的左侧，表示为 `left(objA, objB)`；
- 类似地，通过比较 y_c 坐标可判定上下关系（例如，若物体 A 的 y_c 小于 B，则 A 在 B 之上，记作 `above(objA, objB)`）。

(2) **三维空间关系**：利用深度信息，对物体间的前后关系进行判断。例如，若物体 A 的 z 值小于物体 B 的 z 值，则判定 A 位于 B 的前面，记作 `in_front_of(objA, objB)`。

(3) **遮挡关系**：结合边界框重叠情况与深度信息进行判断。如果两个物体边界框存在重叠且 A 的深度值明显小于 B，则可以认为 A 遮挡 B，记作 `occludes(objA, objB)`。

所有提取到的空间关系均转换为 ASP 事实，如：left(obj1, obj2) 表示物体 1 在物体 2 的左边，in_front_of(obj1, obj2) 表示物体 1 在物体 2 的前面；occludes(obj1, obj2) 表示物体 1 遮挡了物体 2。

4.3.4　场景图生成

场景图是将目标检测、空间位置与空间关系综合融合成的统一结构化表示，其主要构成如下：

(1) **节点表示**：每个节点对应图像中的一个物体，并附有相应的属性（如 color, shape, size, material）以及位置信息；

(2) **边的构建**：节点之间的边用于表示物体间的空间关系，如 left_of、in_front_of 等。

在构建过程中，首先为每个检测到的物体创建一个节点，并记录其属性及位置；随后，依据前述空间关系，将相应的有向边添加到图中。最终，整个场景图将被转化为 ASP 事实，以支持后续符号推理任务。

4.3.5 实现细节与实例

为便于说明，下面给出一个具体示例。假设输入图像包含如下场景：

(1) 一个红色大立方体位于图像左侧；

(2) 一个蓝色小球体位于图像右侧，且位于红色立方体的前方。

经过 GLIP 目标检测，得到如下检测结果：

- 物体 1：类别为“红色立方体”，边界框为 (50, 100, 150, 200)；
- 物体 2：类别为“蓝色球体”，边界框为 (250, 50, 350, 150)。

进一步计算得到：

- 物体 1 的中心点坐标为 $(100, 150)$；
- 物体 2 的中心点坐标为 $(300, 100)$。

假设深度信息显示：物体 1 的 z 值为 50，物体 2 的 z 值为 75，则可提取以下空间关系：

- `left_of(obj1, obj2)`（因为 $100 < 300$）；
- `above(obj2, obj1)`（因为 $100 < 150$）；
- `in_front_of(obj2, obj1)`（因为 $75 > 50$，在三维空间中深度值较大的物体更靠后，需根据具体定义调整）。

最终生成的 ASP 事实示例如下：

```
color(obj1, red).
shape(obj1, cube).
size(obj1, large).
position(obj1, 100, 150, 50).

color(obj2, blue).
shape(obj2, sphere).
size(obj2, small).
position(obj2, 300, 100, 75).
```

```
left_of(obj1, obj2).
above(obj2, obj1).
in_front_of(obj2, obj1).
```

4.3.6　技术挑战与解决方案

在实际实现过程中，本模块面临如下关键技术挑战，并提出相应解决方案：

(1) **检测准确性**：GLIP 在处理物体密集或遮挡严重的场景时可能存在误检问题。为此，本文引入非极大值抑制（NMS）以及深度信息辅助过滤，提高检测精度；

(2) **空间关系鲁棒性**：二维关系受视角影响较大，为增强鲁棒性，本文在有深度信息的条件下优先采用三维关系，并结合几何约束（例如边界框重叠）进行补充；

(3) **计算效率**：面对大规模图像数据，为保证系统实时性，本文通过批量处理图像以及优化提示设计，有效降低了 GLIP 的推理延时。

4.3.7　小结

本节详细介绍了基于 GLIP 的视觉场景理解模块的整体架构与实现细节。通过对目标检测、空间位置与空间关系的综合提取，最终构建了能够转化为 ASP 事实的场景图，为后续语义解析和逻辑推理提供了坚实的数据基础。下一节将介绍如何利用 LLM 将自然语言问题转化为 ASP 查询，实现语义解析与神经符号推理的无缝衔接。

4.4　语义解析

语义解析的主要任务是，通过 LLM，使用上下文学习的方法，将自然语言问题转为用 ASP 进行表示，以便与视觉场景理解提取的场景事实结合进行逻辑推理。

直观上，LLM 可能很难直接解决复杂的推理问题。然而，LLM 已经在理解文本输入并将其转化为形式化程序方面取得了巨大成功，例如程序代码[47] 和数学方程[48]。接下来，本文将介绍通过微调 LLM，根据自然语言问题，生成正确的 ASP 程序。

4.4.1　ASP 模板设计

根据第三章构造的数据集，本文设计了一组 ASP 模板，以覆盖数据集中可能出现的问题类型。以下将针对各问题类型介绍相应的 ASP 模板设计。

基础存在性问题

基础存在性问题是最简单的问题类型，其形式为“是否存在一个满足条件的物体”。本文设计了以下 ASP 模板：

```
模板：是否存在一个[颜色][材质][形状]的物体？
提示：是否存在红色金属立方体？
编码：exists :- object(ID, red, metal, cube, _, _, _).
```

三维空间关系问题

```
模板：物体A（[属性]）是否在物体B（[属性]）的[方位]方，且两者在Z轴上[关系]？
提示：红色球是否在蓝色立方体的左上方？
编码：left_above(O1, O2) :- object(O1, red, _, ball, X1, Y1, Z1), object(O2,
    ↪ blue, _, cube, X2, Y2, Z2), X1 < X2, Z1 > Z2 + 10.
exists :- left_above(O1, O2).
```

多跳推理问题

多跳推理的跳数的取值范围为 2-5，主要考察模型的推理能力。本文对此设计了以下 ASP 模板：

```
模板：若[物体A属性]在[物体B属性]的[方位1]，且[物体B属性]在[物体C属性]的[方位2]
    ↪，那么[物体A属性]相对于[物体C属性]的位置是什么？
提示：若红色球在蓝色立方体左边，且蓝色立方体在绿色圆柱体前面，那么红色球相对于
    ↪绿色圆柱体的位置？
编码：transitive_left_front(O1, O3) :- left(O1, O2), front(O2, O3).
final_relation(O1, O3) :- transitive_left_front(O1, O3), object(O1, red, _,
    ↪ball, _, _, _), object(O3, green, _, cylinder, _, _, _).
```

多参考系问题

```
模板：以[物体属性]为参照物，[目标物体属性]位于其哪个方向？
提示：以蓝色立方体为参照物，红色球是否在其右后方？
编码：local_right_behind(Target, Ref) :- object(Ref, blue, _, cube, Xr, Yr,
    ↪Zr), object(Target, red, _, ball, Xt, Yt, Zt), Xt > Xr, Zt < Zr.
exists :- local_right_behind(Target, Ref).
```

动态反事实问题

```
模板：如果移除[物体属性]，那么[某条件]是否成立？
提示：如果移除所有红色物体，是否还存在比蓝色立方体大的球？
```

```
编码：hypothetical_world(ID) :- object(ID, _, _, _, _, _, _), not (object(ID
    ↪, red, _, _, _, _, _), removed(ID)).
hypothetical_condition :- hypothetical_world(ID1), object(ID1, _, _, ball, _
    ↪, _, S1), object(ID2, blue, _, cube, _, _, S2), S1 > S2.
```

对抗性样本问题

```
模板：图中是否有[数量]个[属性]物体？注意[干扰条件描述]。
提示：是否有3个红色球？注意反光物体可能是玻璃材质而非球体。
编码：valid_ball(ID) :- object(ID, red, glass, ball, _, _, _), not material(
    ↪ID, metallic).
count(N) :- N = #count{ ID : valid_ball(ID) }.
```

4.4.2　ASP 查询生成

为了对语义解析模块的专用 LLM 进行微调，以更好地生成 ASP 查询，本文首先根据上述设计的 ASP 模板，生成训练数据，再对该 LLM 进行训练。生成的数据集共包括 10 万条训练数据，各类型占比见表。

目前，工业界和学术界已有的 LLM 有很多，如 GPT、Llama3、DeepSeek 等。为了选择最适合的 LLM，本文首先对主流 LLM 在生成 ASP 查询这一方面进行评估。LLM 的参数大小、架构及训练数据源见表4.1。从表4.1中，不难看出 DeepSeek R1 1.5B 是在这一组 LLM 中最小的模型。

模型	参数量	架构	训练数据源
GPT-4 turbo	200B	多模态统一网络	网络数据
DeepSeek R1	1.5B–70B	MoE + MLA	网络数据
LLaMa3	8B–70B	纯解码器 + 分组查询注意力机制	网络数据

表 4.1: LLM 对比详细信息，其中参数量以十亿（B）为单位。

最终获得的数据集包括 10 万条训练数据，将数据集按照 8:2 的比例划分成训练集和验证集，并保持每个问题类别的数据分布比例保持一致。

为了能够进行高效微调，最终选择参数量最小的 DeepSeek R1 1.5B 作为语义解析模块的 LLM。DeepSeek R1 1.5B 模型采用了分组查询注意力（GQA）机制，将查询头划分为 2 组共享键值投影，相比传统多头注意力（MHA）减少 25% 内存占用。同时引入局部滑动窗口注意力（窗口大小 4096）与全局注意力交替层，平衡长程依赖建模与计算效率。

为了验证 LLM 生成 ASP 查询的正确性，此处通过 Python API 调用 ASP 求解器 Clingo。Clingo，从而定义一个函数 $f(P) = AS(P)$，其中 $AS(P)$ 表示程序 P 的所有回答集（可能为

空）。对于 LLM 根据提示 x 生成的 ASP 程序 y $P_L(|x)$，以及与 x 对应的能够真实表示问题的 ASP 程序 y^*，按照以下步骤来进行验证：

1. 事实集合构建：构造一组事实，用集合 F_{y^*} 表示，其代表了问题 x。

2. 程序合成。将 F_{y^*} 与 y 合并，得到新的 ASP 程序 $P = y \cup F_{y^*}$。同理，将 y^* 与 F_{y^*} 合并，得到新的 ASP 程序 $P^* = y^* \cup F_{y^*}$。

3. 语法命中：调用 Clingo 计算 $f(P)$，若未发生解析错误，则判定为语法命中。

4. 语义验证：进一步计算 $f(P)$ 与 $f(P^*)$，分别得到 $AS(P)$ 与 $AS(P^*)$。若 $AS(P)$ 与 $AS(P^*)$ 完全匹配，则判定为语义命中。

此后，基于 DSPy 框架发起对 LLM 的调用，将自然语言问题转化为 ASP 查询。此处使用 DSPy 的 Template 组件定义提示模板，指导 LLM 如何生成指定格式的 ASP 程序。其中，也使用了 Example 组件，用于构建优化过程中使用的示例数据，封装输入和输出的对应关系，以帮助 LLM 进行学习。与传统的 LLM 开发流程相比，DSPy 框架自动化管理了提示模板和示例数据的构建过程，有效降低了手动调试和反复迭代的复杂度，从而大幅简化了整个 LLM 相关的开发流程，显著提高了开发效率和模型优化的便捷性。

4.4.3 实验与结果分析

为检测语义解析模块的性能，本文使用上一小节构造的数据集进行实验。考核指标选取上一节定义的语法命中率和语义命中率。将未经过训练的 LLM 与本文经微调后的 DeepSeek R1 1.5B 模型进行对比。在实验过程中，对实验重复进行 5 次，以降低 LLM 采样过程的随机性带来的影响。

实验结果如表**??**及图**??**所示。语法正确并不能保证语义正确，例如，Gemma 7B 实现了 45% 的语法正确率，但是语义正确率明显偏低。根据实验结果分析，没有任何模型能够做到全方面正确，轻量级模型的综合表现最差。此外，GPT-4.0 turbo 实现了 100% 的语法正确率。同时也注意到，虽然 Gemini 和 LLaMa3 70B 的参数规模和前述模型相当，但是准确率却明显较低。本文的经微调后的 DeepSeek R1 1.5B 模型在所有模型中的语义准确率最高。根据实验结果，也能够看出，当生成的 ASP 程序在语法上正确时，其语义正确的可能性也会更大。所有模型在每种类型的问题上的语法正确率和语义正确率的对比见表4.3。

4.5 迭代反馈与规则修正

迭代反馈模块是整个神经符号集成管道的核心部分，其主要功能在于提升由 LLM 生成的 ASP 程序的正确性和可执行性。本文所提出的迭代反馈机制结合了自然语言解析、逻辑程序生成与符号求解三大过程，其整体流程可以描述为：初步生成、逻辑程序执行、错误信息反馈、程序修正，直至达到预设的迭代次数或程序满足执行要求为止。

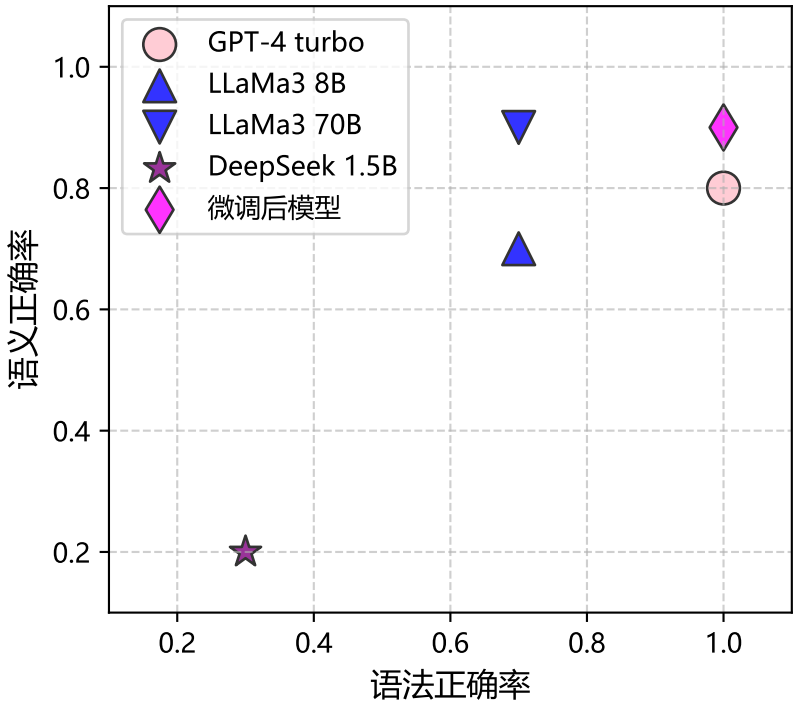


图 4.2: 各模型的语法正确率及语义正确率

模型	语法正确率	语义正确率
GPT-4 turbo	1	0.8
Llama3 8B	0.7	0.6
Llama3 70B	0.7	0.8
DeepSeek R1 1.5B	0.5	0.6
微调后模型	0.8	0.8

表 4.2: 各模型在语法和语义方面的对比分析

模型	基础存在性		三维空间关系		多跳推理		多参考系		动态反事实		对抗性样本	
	语法	语义	语法	语义	语法	语义	语法	语义	语法	语义	语法	语义
GPT-4.0 turbo	1	0.9	1	0.7	1	0.9	1	0.8	1	0.7	1	0.8
LLama3 8B	0.9	0.6	0.7	0.5	0.8	0.6	0.6	0.7	0.5	0.7	0.7	0.5
LLama3 70B	0.6	0.7	0.7	0.7	0.7	0.9	0.7	0.9	0.6	0.8	0.8	0.8
DeepSeek R1 1.5B	0.5	0.6	0.7	0.5	0.6	0.6	0.4	0.7	0.4	0.5	0.5	0.7
微调模型	1	0.9	0.8	1	0.6	0.9	0.9	0.8	0.9	0.8	0.6	0.4

表 4.3: 不同模型在各问题类型上的语法正确率和语义正确率的对比

4.5.1 反馈流程概述

首先，LLM 根据输入的自然语言描述生成初始的 ASP 程序，并将该程序交由 Clingo 求解器进行执行。若在执行过程中出现任何错误（例如语法错误、变量命名不一致、或逻辑不严谨导致的求解失败），C lingo 将输出相应的错误信息。这些错误信息作为反馈被传递给 LLM，指导其对初始生成的 ASP 程序进行针对性修正。修正后的 ASP 程序再次输入 Clingo 求解器执行，如此循环最多进行三次，最终输出经过优化的 ASP 程序，用于后续的正式推理。

4.5.2 错误类型与修正策略

不同类型的错误信息对应着 ASP 程序中不同的问题，因此需要采用不同的修正策略。具体来说，本文设计了多套提示模板，用以指导 LLM 针对性地进行程序修正，其主要错误类型包括：

1. 解析错误：如语法错误、未定义谓词、或不符合 ASP 语法要求的结构错误。针对该类错误，提示模板会明确指出错误的所在位置及其可能原因，并要求 LLM 调整相应的语法结构。

2. 基础化错误：主要涉及变量命名不一致或不匹配等问题。此时，提示模板会强调变量的统一性和正确性，要求 LLM 对变量命名进行标准化处理。

3. 求解阶段错误：包括过度约束或逻辑矛盾导致的求解失败。对于这类问题，提示模板将引导 LLM 重新审视程序中约束条件和规则之间的逻辑关系，并做出必要的修改。

每一轮反馈过程中，LLM 都将结合 Clingo 返回的错误信息和预定义的提示模板进行针对性修正，从而使生成的 ASP 程序在下一轮执行中更趋于正确和可执行。

4.5.3 DSPy 托管 LLM 调用

为了高效管理这一 LLM 与 ASP 求解器之间的复杂的反馈交互过程，本文采用了 DSPy 框架。DSPy 具备以下优势：(1) 模块化管理：DSPy 将整个反馈流程拆分为若干功能模块，如

ASP 程序生成、错误信息解析、提示模板更新及程序修正等。模块之间通过明确定义的接口进行通信，保证了信息的准确传递和记忆保留，从而实现参数的自适应调整和流程优化。(2) 自动化优化：DSPy 支持对 LLM 提示词及参数进行自动优化，极大地降低了人工调试的工作量。在多次与 LLM 交互后，系统能够根据反馈动态调整提示模板，最终得到一个较优的提示方案，以提高整体系统的修正效率和程序可执行率。(3) 日志记录与调试：为了便于调试和系统性能分析，DSPy 在整个反馈过程中对各模块的输出日志进行详细记录，包括错误信息、状态提示以及每一轮迭代的修正详情。

4.5.4　DSPy 优化器选择

本文根据公开资料，对比了 DSPy 框架中提供的三种优化器的适用任务及场景，如表4.4所示。最终，本文选择了 BootstrapFewShot 优化器，其主要理由如下：（1）标注成本低：传统的人工标注 ASP 事实需要较长时间（平均每个样本约 3.5 分钟），而 BootstrapFewShot 优化器针对少样本场景设计，仅需 15-20 个标注样本即可启动优化。（2）分层次优化：BootstrapFewShot 优化器将优化过程拆分为“教师优化”和“学生训练”两个阶段。教师优化阶段负责生成高质量的示例，而学生训练阶段则通过知识蒸馏完成对提示模板的优化，以确保整体修正效果的提升。（3）资源控制：该优化器支持限制最大错误次数和迭代轮数，避免了无限循环和资源浪费，从而提高系统的稳定性与效率。

基于本文有限的标注样本和实际应用需求，选择 BootstrapFewShot 优化器能够在保证优化效果的同时，大幅提高反馈迭代的效率和系统的总体性能。

表 4.4: Dspy 框架支持的优化器比较

优化器名称	主要功能	适用场景
BootstrapFewShot	通过在提示中自动生成并包含优化示例来扩展签名。	少样本学习
BootstrapFewShot-WithRandomSearch	在 BootstrapFewShot 的基础上，对生成的示例进行多次随机搜索，选择优化后的最佳程序。	少样本学习，需要更高精度
MIPRO	在每个步骤中生成指令和少量示例，使用贝叶斯优化来有效地搜索模块中的生成指令和示例空间。	需要复杂指令和示例优化的场景
BootstrapFinetune	将基于提示的 DSPy 程序提炼为较小语言模型的权重更新，微调底层大型语言模型以提高效率。	需要微调模型以提高效率的场景

4.6　规则蒸馏

经过迭代反馈修正后的 ASP 编码，已经能够顺利通过 Clingo 求解器执行，同时在语法和语义上保证正确。然而，要正确进行推理，可能还需要先验知识配合。规则蒸馏的目的在

于，利用 LLM 的知识和已有 ASP 知识库的知识，对迭代反馈修正后得到的 ASP 编码进行补全，减轻知识工程师手动编写规则的负担，最终得到解决问题所需的足够的 ASP 规则，供后续 Clingo 求解器使用。

本模块的技术路径如下：当系统检测到现有的 ASP 规则集无法推导出问题的正确答案时，将会

4.6.1 预提示

预提示的作用是，为 LLM 提供背景信息和任务说明。具体而言，预提示包括以下内容：（1）任务背景说明。介绍 VQA 任务场景，说明问题和图像已经被解析为 ASP 表示；（2）ASP 语法与表示。讲解 ASP 的语法以及如何使用 ASP 表示场景、问题和答案，使 LLM 了解输入格式；（3）初始理论。给 LLM 提供 ASP 规则，这些规则来源于本地的 ASP 知识库；（4）任务要求。明确向 LLM 要求，输出内容不仅包括新规则的 ASP 代码，并且要求规则尽可能通用，不应包含具体事实。

该机制通过增量式规则扩展实现 ASP 知识库的动态演进：每当系统识别出现有理论无法推导正确结果时，会触发 LLM 的规则生成模块，输出符合 ASP 语法的逻辑规则补充到 ASP 知识库中。整个过程形成” 问题识别-规则生成-理论扩展” 的闭环优化路径。

以下为节选的部分提示词。

```
你的任务是对ASP知识库进行维护，通过动态更新规则确保其能够解答各类问题。ASP知识
    ↪库中的现有规则已能够满足对部分问题的处理需要，
需根据输入的问题的ASP表示进行增量式规则拓展。输入提示将包含一个或多个采用ASP表
    ↪示形式的问题。请严格遵守以下规则：

1. 仅输出新增的ASP规则。
2. 禁止将事实作为规则添加。不允许在规则头(head)中使用具体常量实例。
3. 将规则的泛化程度实现最大化。你输出的规则应具备以下特征：使用大写字母变量
    ↪（如X,Y,Z）替代具体常量，通过谓词逻辑建立抽象关系而非具体实例关联。
4. 完全禁用自然语言。你的输出中，不得出现任何自然语言成分，包括：错误提示信
    ↪息、代码注释说和解释性文字。
```

4.6.2 规则蒸馏算法

向 LLM 进行预提示之后，对迭代反馈后的 ASP 代码和视觉场景理解模块得到的 ASP 代码，合并记作 p，执行如下步骤（最多重复 r 次）：

1. 提示。将 p 作为附加输入提示 LLM，并得到响应 R。
2. 求解。将响应 R 与 ASP 知识库中的初始知识 I 结合，得到新的 ASP 规则集 Res。基于 Res，调用 Clingo 求解器对问题进行求解。此后首先进行语法检查，若检测到语法错误，则将错误信息传给 LLM，并要求 R 进行修改，最多重试 m 次。此外进行语义检查，验

证求解器的输出是否与预期答案一致，如果不一致则将实际答案和预期答案传给 LLM，要求其对 R 进行更新，最多重试 m 次。

3. 回归测试。为了避免新增加的规则破坏知识库中规则的一致性，需要对 Res 进行测试。只有当所有的测试都通过之后，才会保留 Res，否则将会忽略 R。

以上算法有 2 个参数，r 是每个示例的重试次数，m 是修正错误规则的重试次数。r 和 m 的默认值都为 1。

为了对规则蒸馏 LLM 进行训练，补充 ASP 知识库，在此处采用了 GQA 数据集以拓展现有的 ASP 知识库。对 GQA 数据集中的问答数据进行预处理后，每条数据均以（问题，场景，答案）的三元组形式给出，数据集中的实例被用来检测当前 ASP 理论的不足，并引导 LLM 生成补充规则，以便在语法和语义检查通过后将新规则融入系统中，从而提升整体推理能力。

4.7 ASP 推理

ASP 推理阶段中，ASP 求解器接收经过优化后的 ASP 查询语句、视觉场景理解模块提取的 ASP 事实以及已有常识的 ASP 表示，进行逻辑推理，最终获得答案。本文使用 Clingo 求解器来进行求解，其工作过程分为基础化（grounding）阶段和求解（solving）阶段。

基础化阶段将 ASP 程序中的变量替换为常量，生成一个新的 ASP 程序。Clingo 通过使用内置的基础化器分析程序，生成所有可能的规则。例如，对规则 $a(X) : -b(X), notc(X).$，其将被展开为所有可能的值为 X 的实例。

求解阶段中，Clingo 采用类似 SAT 求解器的冲突驱动答案集求解（CDNL）方法。CDNL 方法通过迭代地添加约束，直到找到一个满足所有约束的解，或者证明无解。具体而言，Clingo 基于如下步骤进行求解：（1）初始化。从空分配开始（没有原子被标记为真）；（2）选择与分配。选择一个未分配的原子，猜测其真值为真或假；（3）传播。根据当前分配和规则，推导其他原子的真值。例如，若规则 $a : -b, notc.$ 满足，b 为真且 c 不在答案集中，则 a 必须为真；（4）冲突检测。若分配导致规则冲突（如与已有的约束条件相抵触），记录冲突原因；（5）回溯与学习。若发生冲突，回溯到之前的选择点，学习冲突子句以避免未来类似错误；（6）验证。当所有原子分配完成后，检查是否为答案集，即确保它是程序的模型，且最小化（无子集也能满足规则）。

相比其它的 ASP 求解器，Clingo 在以下几个方面进行了优化：（1）从冲突中学习新的规则，为将来进一步的搜索提供指导；（2）使用多线程实现并行化，同时搜索多个路径，加速求解过程；（3）使用启发式方法决定分配顺序，例如优先选择高影响力的原子；（4）预测可能发生的冲突，选择可能导致冲突的原子优先分配，减少搜索空间。

4.8 求解结果翻译

经过 Clingo 得到的输出结果，以 ASP 谓词表示，对用户而言并不友好。需要将其转为自然语言。Clingo 输出的是由逻辑谓词构成的答案集，例如谓词 **is(A, left, B)** 表示“A 位于 B 的左侧”。由于正常情况下，Clingo 输出的都是预定义的谓词，故采用构造同义词字典的方案，

将 ASP 中的逻辑谓词与自然语言表述对应起来。LLM 可基于这些模板对解析后的逻辑结果进行填充和修正，生成标准化且易于理解的描述。

4.9 实验与结果分析

4.9.1 对比试验

为全面评估神经符号框架的有效性以及泛化能力，本文选取了目前主流的三种 LLM：DeepSeek R1、Llama3 和 GPT-4 turbo，在其上应用神经符号框架，进行对比。以上既包含 DeepSeek 这类轻量级专用模型，也涵盖 GPT-4 turbo 这类通用型先进系统，而 Llama3 性能和效率之间取得了平衡，属于居中水平的模型。通过在多个基座上进行实验，证明神经符号框架对不同 LLM 的泛化能力。

基线模型一选取直接提示 VLM 的方式，即直接将自然语言问题和图像输入到 VLM 中，并不给予任何的额外提示。直接提示 VLM 的方式虽然简单，却是评估模型的关键基准，因为直接提示方法能够反映模型在没有任何外部推理辅助机制的情况下，自身对空间问题的处理能力。

基线模型二采用“事实 + 规则”的提示方法。该方法的核心思路是：指示 LLM 使用预先定义的谓词，将输入的自然语言问题转换为结构化事实，然后 LLM 应用相关逻辑规则，通过自然语言推理答案。“事实 + 规则”的提示方法满足了将原始自然语言问题转为结构化符号表示，以配合视觉场景理解所得的 ASP 事实以及原有常识，共同进行推理的核心需求。同时，该方法使用具有精确参数结构的谓词对 LLM 进行提示，使得 LLM 可以创建一致的中间态的知识表示，为问题解答提供便利。综合来看，“事实 + 规则”的提示方法作为一种简化流程，既保留了形式化推理的优势，同时能够避免在生成 ASP 程序时和 LLM 进行多次交互，进而降低对计算资源的要求，降低了成本，也减少了对外部求解器的依赖。

实验结果见表4.5，本文提出的神经符号框架在 DeepSeek R1、LLama3 和 GPT-4.0 turbo 上均超过了两种比较基准方法，证明大语言模型与逻辑推理结合对于解决复杂空间推理问题的有效性。

4.9.2 消融实验

尽管 LLM 在语义解析任务中表现出色，但其生成的 ASP 程序的正确率仍有较大的提升空间。Feng[49] 等人的研究表明，将自然语言直接转为逻辑规则的成功率一般较低。而神经符号框架通过在 LLM 和外部 ASP 求解器之间设立反馈循环机制，使 LLM 能够根据 ASP 求解器在试图求解 ASP 程序之前的语法检查结果，对生成的 ASP 程序中的错误进行修正，极大提高了将自然语言问题转为 ASP 程序的准确率。

为进一步探讨神经符号框架的迭代反馈机制的作用，本文重点关注 ASP 求解器执行过程中常发生的三类主要错误：解析错误、实例化失败以及求解阶段失败。另外，即便生成的 ASP 程序执行成功，也有很大可能由于自然语言问题与 LLM 生成的 ASP 程序之间不一致，而产生与真实答案偏离的结果。

在测试中，迭代反馈机制显著提升了所有模型的性能。如图4.3所示，经过两轮迭代反馈

模型及方法	基础存在性问题	三维空间关系问题	多跳推理问题	多参考系问题	对抗性样本问题	总体
DeepSeek R1						
直接提问	60.5	38.2	74.5	78.2	48.2	59.9
事实 + 规则提示方法	69.7	47.3	80.6	80.9	56.6	67.0
神经符号框架方法	77.8	58.9	85.4	81.8	42.8	69.3
Llama3						
直接提问	68.4	26.9	65.7	72.5	55.8	57.9
事实 + 规则提示方法	75.3	46.8	70.2	81.4	57.1	66.2
神经符号框架方法	79.1	53.1	83.3	80.5	60.8	71.4
GPT-4.0 turbo						
直接提问	77.2	45.4	60.9	57.6	58.2	59.9
事实 + 规则提示方法	85.3	54.3	68.2	61.5	50.4	64.0
神经符号框架方法	91.1	65.6	80.5	72.7	64.8	74.9

表 4.5: 不同模型及方法在各问题类型上的表现

之后，DeepSeek R1 的可执行率从 51.3% 提升到了 84.2%，LLama3 的可执行率从 60.2% 提升到了 89.7% ，GPT-4.0 turbo 的可执行率从 64.6% 提升到了 91.1%。在准确率方面，DeepSeek R1 的准确率从 57.8% 提升到了 79.3%，LLama3 的准确率从 61.1% 提升到了 85.7%，GPT-4.0 turbo 的准确率从 69.7% 提升到了 93.4%。以上结果表明，LLM 与 ASP 求解器之间的迭代反馈机制能有效解决自然语言到逻辑程序的转换过程中面临的问题。对准确率和可执行率的提升，主要集中在第一轮迭代反馈中，此后继续迭代的效果呈现边际递减趋势。由于计算资源有限，本次消融实验仅进行了三轮。实验结果有力证明，迭代反馈机制在提升 ASP 程序的可执行率和正确率方面具有明显效果，充分验证了神经符号方法的有效性。

4.10　本章小结

本章详细介绍了神经符号框架的整体架构设计与实现过程，阐述了从视觉场景理解、语义解析、迭代反馈与规则修正到规则蒸馏，再到最终的 ASP 推理的完整流水线。首先，框架总体架构明确了 LLM 在语义解析和反馈优化中的双重作用，为复杂空间关系推理提供了有效支撑。接着，在视觉场景理解部分，本文基于 GLIP 模型实现了目标检测，并通过中心点与深度信息提取物体的二维及三维位置信息，再结合空间关系提取技术构建场景图，并将提取的信息转化为 ASP 事实，为后续逻辑推理打下坚实基础。

在语义解析模块中，利用 LLM 的上下文学习能力，通过微调生成与自然语言问题对应的 ASP 程序，设计了多种 ASP 模板以覆盖基础存在性、三维空间关系、多跳推理、多参考系、动态反事实以及对抗性样本等问题类型。实验结果表明，经过微调和 DSPy 框架支持下的提示优化，生成的 ASP 查询在语法与语义上均取得了较高准确率，验证了 LLM 在语义解析任务中的有效性。

随后，通过迭代反馈与规则修正模块，系统实现了 LLM 与 ASP 求解器之间的闭环交互：

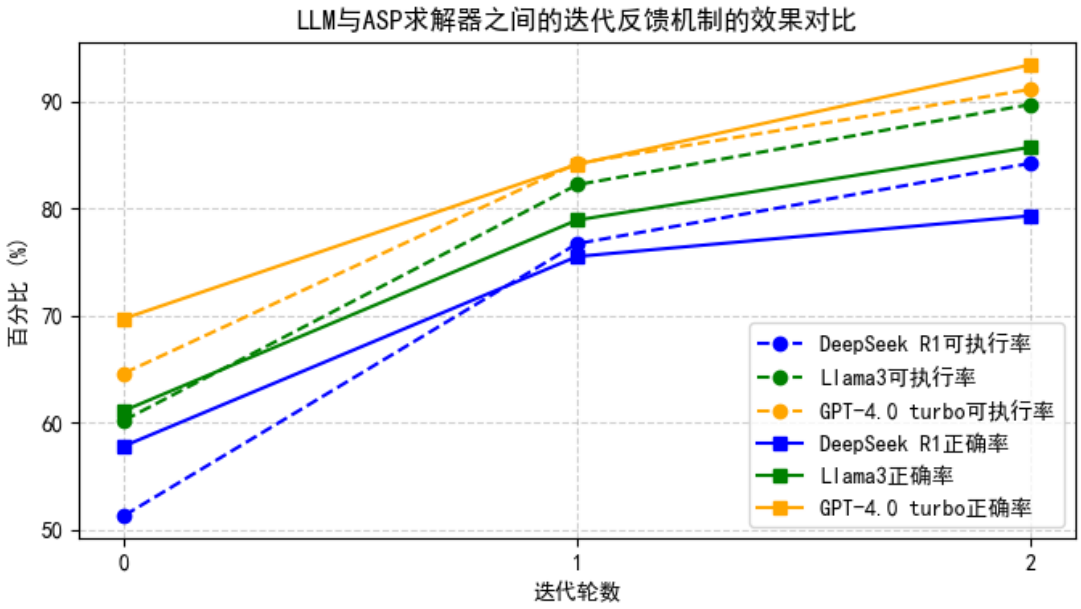


图 4.3: LLM 与 ASP 求解器之间的迭代反馈机制的效果对比

Clingo 求解器对生成的 ASP 程序进行执行检查，并反馈错误信息，LLM 据此优化 ASP 程序，经过多轮迭代后显著提升了程序的可执行率和语义正确率。接着，通过规则蒸馏模块，以迭代反馈与规则修正模块得到的 ASP 代码和视觉场景理解得到的 ASP 代码为输入，以 ASP 知识库为基础，调用 LLM 对 ASP 规则进行补充，同时对 ASP 知识库进行更新完善。最后，在对比试验和消融实验中，本文综合比较了不同 LLM 模型和提示方法的表现，验证了神经符号框架在处理复杂空间推理问题上的泛化能力与优势，同时也证明了迭代反馈机制对整体性能提升的重要作用。

总体来说，本章不仅构建了一个完整而高效的神经符号推理流水线，而且通过详细实验与消融分析，验证了各模块间协同工作的有效性，为后续基于逻辑推理的 VQA 系统提供了坚实的理论和实践基础。

第五章　问答系统的构建

5.1　需求分析

本系统旨在基于本文提出的神经符号框架，构建一个视觉问答系统，既能利用深度学习技术对图像进行特征提取，用 LLM 对自然语言进行语义解析，又能通过 ASP 求解器实现推理，同时为用户提供友好、直观的前端交互界面。

5.1.1　功能需求

核心功能上，需要实现以下几个功能：（1）用户输入与对话处理。系统需要能够支持文本输入、图像输入，未来远期可以考虑加入语音输入。此外，提供自动补全、拼写检查等输入辅助功能，以改善用户的输入体验。（2）实时对话回复。系统能够通过 Dspy 的 SDK，发起对本地 LLM 的调用，而 LLM 能够实时处理用户的问题，返回高质量的回答。另外，要能够支持多轮对话，能够保持上下文关联，具有记忆功能。（3）对话历史记录。系统需要能够自动保存用户每次的对话记录，以供用户查看。（4）帐户与认证。系统需要支持用户注册、登录、账号设置等功能。（5）数据统计与反馈。系统需要收集交互数据（如访问量、对话时长、常见问题等），供后续模型优化使用。另外，要提供用户反馈入口，支持用户评价回复质量。

辅助功能方面，主要围绕以下几点：（1）响应式设计。前端能够自适应桌面、平板和手机等不同设备。（2）辅助说明。提供 FAQ、帮助文档和使用指南，方便用户进行查阅。（3）安全与异常处理。能够对恶意输入、不符合规范的输入进行拦截和提示。

5.1.2　非功能需求

非功能需求主要聚焦于以下几个方面：（1）性能需求。系统需要能够在较短时间内完成对用户问题的回答，如果确实因为某些原因导致无法及时回答，需要能够给出合理的提示，例如“正在思考...”等。另外，各个微服务应支持高并发访问，保证系统能够承受一定规模的大流量访问。（2）可拓展性。对系统要按照主要功能来进行拆分，形成若干个微服务，并且各个微服务之间要能够相互独立，实现松耦合。当系统需要进行拓展时，只需要增加新的微服务，而不需要对原有的微服务进行修改。（3）可维护性。代码和文档需要保持清晰，易于理解和修改，特别是对于接口文档，要使用 Swagger 等工具进行规范化管理，以便于后续的维护和拓展。另外，要提供清晰的日志记录、监控报警机制，使用一些监控工具，如 Prometheus、Grafana 等，对系统的运行状态进行监控，及时发现问题并解决。（4）安全性。用户的数据传输和存储需要进行加密，对于用户的隐私数据，例如手机号、密码等，需要进行脱敏处理，以保证用户的隐私安全。此外，对登录、注册等操作，需要进行身份验证和权限控制，防止恶意攻击。（5）可用性和用户体验。在整体外观上，系统需要提供简洁友好的用户图形界面，降低使用门槛。

方法	URL	请求参数名及其类型与作用	备注
POST	/api/object_detection	image（图片，通过文件上传提供）、text_prompt（文本提示，用于指导模型检测与文本相关对象）	视觉场景理解服务接口

表 5.1: 主要接口列表

5.2 架构设计

5.2.1 总体架构

对本系统的总体架构设计如图5.1所示。

后端部分采用微服务架构设计，包括视觉场景理解服务、语义解析服务、符号推理服务、迭代反馈服务、答案生成服务、用户交互服务、日志与监控服务、数据存储服务等。各微服务之间通过 API 网关进行通信，API 网关负责请求的路由、负载均衡、安全认证、限流等功能。

具体到各个微服务的主要职能如下：（1）视觉场景理解服务负责对用户上传的图片进行目标检测，并使用预定义的谓词，使用 ASP 程序来表示场景中的物体的相关信息；（2）语义解析服务负责将用户的自然语言问题转化为 ASP 规则，以便于后续的推理；（3）符号推理服务负责对 ASP 规则进行推理，得到问题的答案；（4）迭代反馈服务负责对 ASP 规则进行修正；（5）答案生成服务负责将符号推理服务输出的以 ASP 谓词表示的答案转化成自然语言；（6）用户交互服务负责提供用户接口，接收用户的输入，调用其他服务，返回结果；（7）日志与监控服务负责记录系统的运行日志，监控系统的运行状态，及时发现问题并解决；（8）数据存储服务负责存储系统的数据，包括用户的信息、对话记录、ASP 规则等。

前端部分，对 UI 界面的开发，选择 Element UI 框架，其具有丰富多样的按钮、输入框等搭建网页所需的元素，开发快捷方便。同时，使用 Axios 向后端发送请求，实现前后端的数据交互。使用 Vue 构建组件化的前端页面，方便代码的维护和拓展。Vue Router 用于实现前端路由，实现页面的跳转。Vuex 用于实现状态管理，方便组件之间的数据传递。

在中间件上，系统引入了 Sentinel 进行流量控制，对系统的流量进行监控，实现限流、熔断等功能。RocketMQ 用于实现消息队列，实现微服务之间的异步通信。Redisson 用于实现分布式锁，保证系统的并发安全。

5.2.2 接口设计

接口设计总体按照服务划分，每个服务提供一个接口供其它服务调用。接口之间的数据传输采用 JSON 格式，采用 RESTful API 设计风格。具体的接口设计方案见表**??**。

5.2.3 安全设计

安全设计主要聚焦数据存储加密和传输安全，以确保系统的安全性和数据的机密性，防止数据泄露和未授权访问。

在存储加密上，针对数据库中存储了用户密码以及身份证号、手机号等高度敏感信息，根据字段的分类，采取不同措施，予以保护。对于用户密码，采用 SHA-256 算法对明文密码进行加密后存储。对于除密码外的其它高度敏感字段（如身份证号、银行卡号），使用 AES 加密函数，搭配密钥管理。

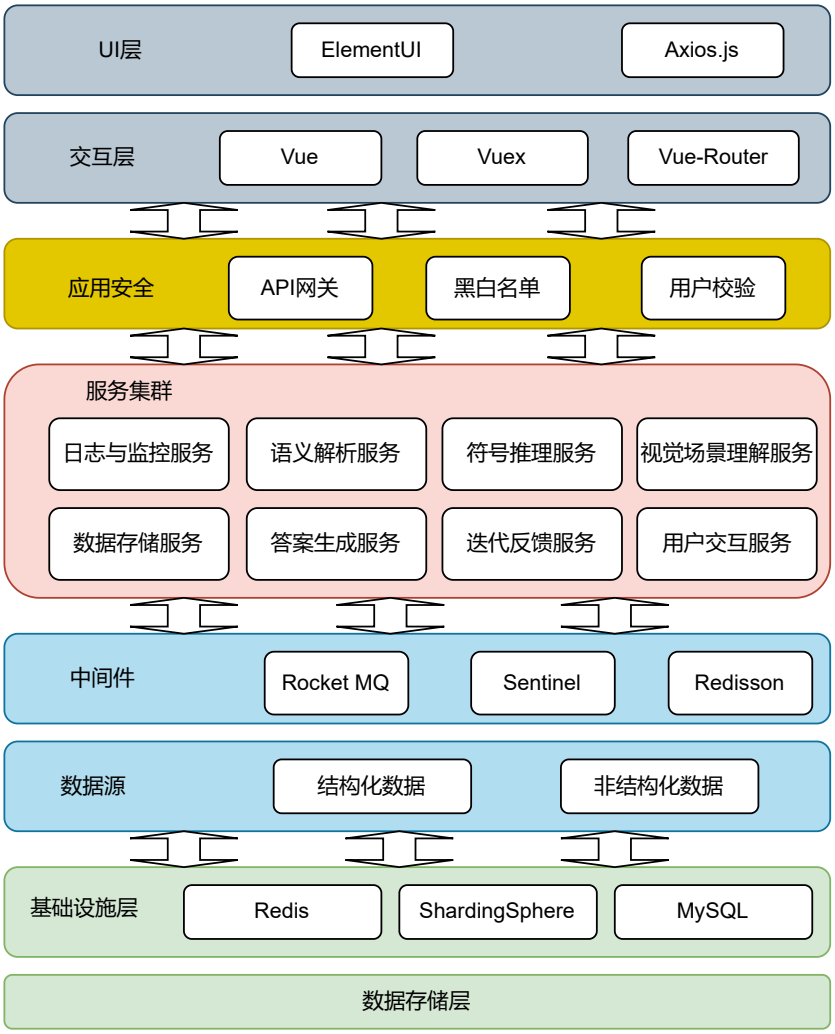


图 5.1: 系统总体架构

在传输安全上，采用 HTTPS 加密传输的方式，以防止中间人攻击。同时，使用 JWT 进行身份验证。为了防止数据在传输过程中被篡改，使用 HMAC-SHA256 进行数据完整性校验。

5.2.4　数据库设计

设置 User、Rule、Log、Dialog 以上四个数据库。下面，对这四个数据库的职责及内设表进行介绍。

User 数据库主要用于存储用户、角色、权限、会话信息等。内部包含如下表：（1）用户表 users，用于存储用户信息；（2）用户组表 roles，用于存储用户组信息；（3）用户组关系表 user_roles 用于存储用户与用户组之间关系；（4）用户状态表 user_session，用于存储会话或者登录状态记录。

Rule 数据库专门用于存储 ASP 规则及其相关信息。内部包含 rules 表和 rule_conditions 表。rules 表为主规则表，rule_conditions 表为规则条件表。具体表字段见表5.2。

Log 数据库存储系统运行日志、错误日志、操作记录等信息。其内部包含如下表：（1）系统日志表 system_logs，用于记录系统一般日志；（2）错误日志表 error_logs，用于记录错误信息；（3）审计日志表 audit_logs，用于记录关键操作审计日志。

字段名称	数据类型	说明
rule_id	INT AUTO_INCREMENT PRIMARY KEY	规则唯一标识
rule_name	VARCHAR(255)	规则名称
rule_head	VARCHAR(255)	规则若论部分
priority	INT	优先级
status	ENUM('active', 'inactive')	状态标识
description	TEXT	描述或备注
created_at	DATETIME DEFAULT CURRENT_TIMESTAMP	创建时间
updated_at	DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新时间

表 5.2: rule 表字段设置及说明

字段名称	数据类型	说明
condition_id	INT AUTO_INCREMENT PRIMARY KEY	条件唯一标识
rule_id	INT	外键，关联到主规则表
condition_text	VARCHAR(255)	单个条件表达式，例如"X > 10"
sequence	INT	条件顺序号，用于保持条件排列
created_at	DATETIME DEFAULT CURRENT_TIMESTAMP	创建时间
updated_at	DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	更新时间

表 5.3: rule_condition 表字段设置及说明

Dialog 数据库存储所有用户与系统的对话记录。其内部包含如下表：（1）对话主表 dialogs，用于记录每个对话的基本信息；（2）对话消息表 dialog_messages，用于存储对话中用户发出的消息与 LLM 给出的每条回复。

5.3 缓存设计

为系统使用缓存的目标包含以下两点：（1）提高响应速度。在整个 VQA 系统中，需要 LLM 参与的语义解析、迭代反馈以及视觉场景理解环节，所需时间较长，目前较难以大幅度优化。为尽可能减少模型调用次数，以提高响应速度，改善用户体验故为系统引入缓存；（2）降低后端负载。如果访问量超出系统的承载能力，可能会导致部分服务宕机。以 Redis 为代表的缓存，单机 QPS 最高可达 10 万，远超数据库等请求链路上的其它环节的承载能力。引入缓存，可一定程度上减少对数据库和 LLM 的直接访问次数，缓解 VQA 系统在高并发场景下的压力。

为了更好地做到成本与性能之间的平衡，本文采用了多层次缓存架构，共分两层。

第一层是本地内存缓存，用于缓存短生命周期、访问频率非常高的数据，如最近对话状态、会话上下文的部分数据。在实现方式上，使用 Java 的 ConcurrentHashMap 配合 LRU 算法对缓存进行管理。

第二层是分布式缓存，本系统选用 Redis，用于缓存跨多个服务器共享的数据，如会话历史、用户信息、通用问答的缓存结果等。分布式缓存可以保证在集群环境下数据的一致性和

高可用性，并支持数据持久化和数据过期策略。至于选用 Redis 作为分布式缓存的理由，是因为其支持丰富的数据结构、持久化存储和分布式部署，同时具有高性能。

5.4　功能展示

开发的前端页面如图5.2所示，采用了模仿 ChatGPT 的网页样式。用户可在其中自行选择模型，包括本文中微调后的模型、OpenAI 的 GPT 系列模型、Google 的 Gemini 系列模型、Meta 的 Llama 系列模型以及阿里巴巴的 QWen 系列模型，默认选用本文的微调后模型。用户可以在下方的菜单栏中，点击第二个按钮，以上传图片。同时，历史对话也会在左侧栏中予以显示。在输入框旁，也添加了语音输入按钮，点击后系统将会调用百度的语音识别 API，为用户提供语音输入服务。

上传图片并提出问题、系统给予答复后的效果如图5.3所示。用户发送的文本和图片将会在同一个消息框内显示。用户要求系统重新生成回复，或者引用系统的某条回复，以让系统能够更加贴合用户的语境。

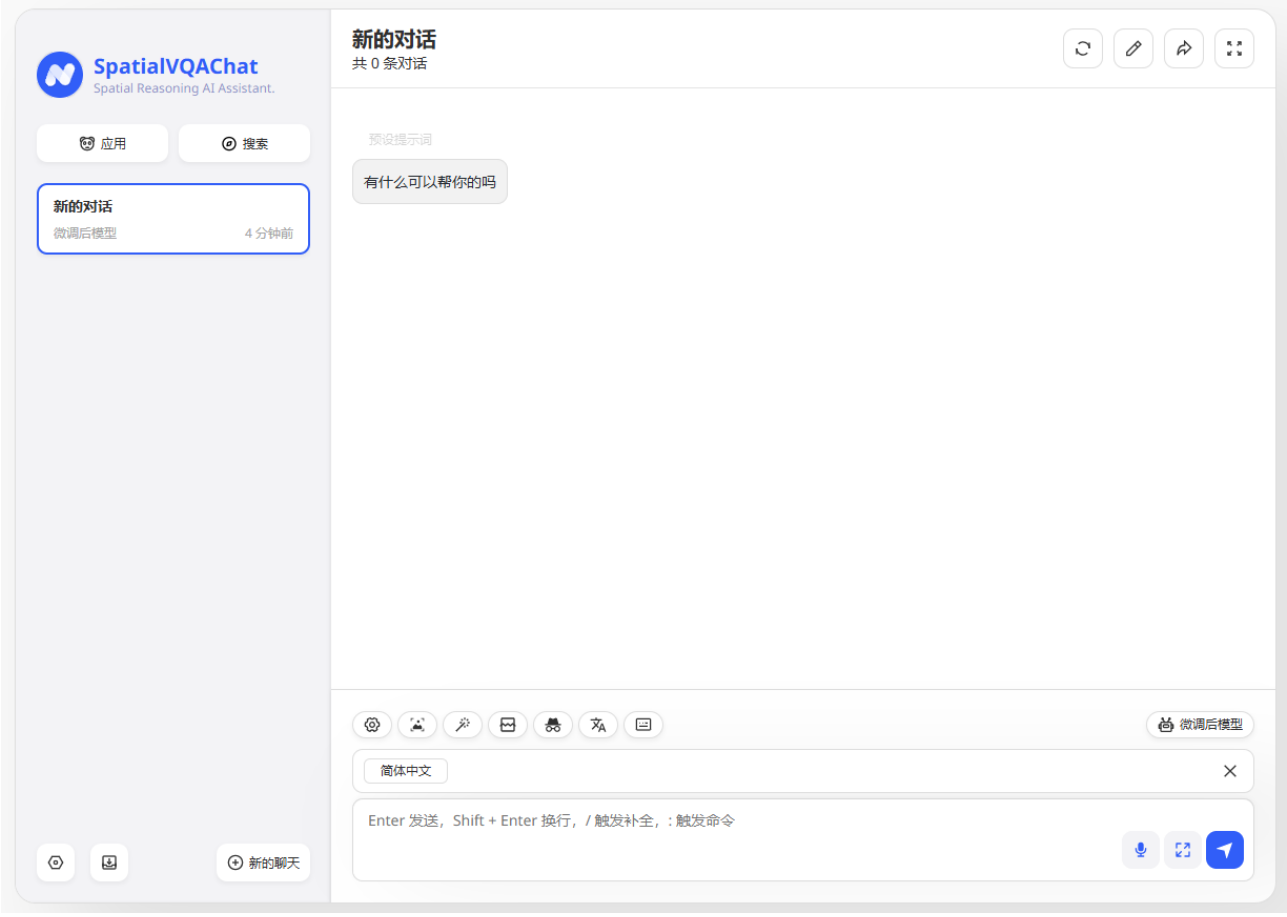


图 5.2: 新对话页面

5.5　集成测试

系统的部署运行环境的硬件配置如下：（1）CPU 为 Intel Core i9-13900K；（2）内存为 128GB；（3）显卡为 3 张 NVIDIA GeForce RTX 3090 并联，显存均为 24GB；（4）操作系统为

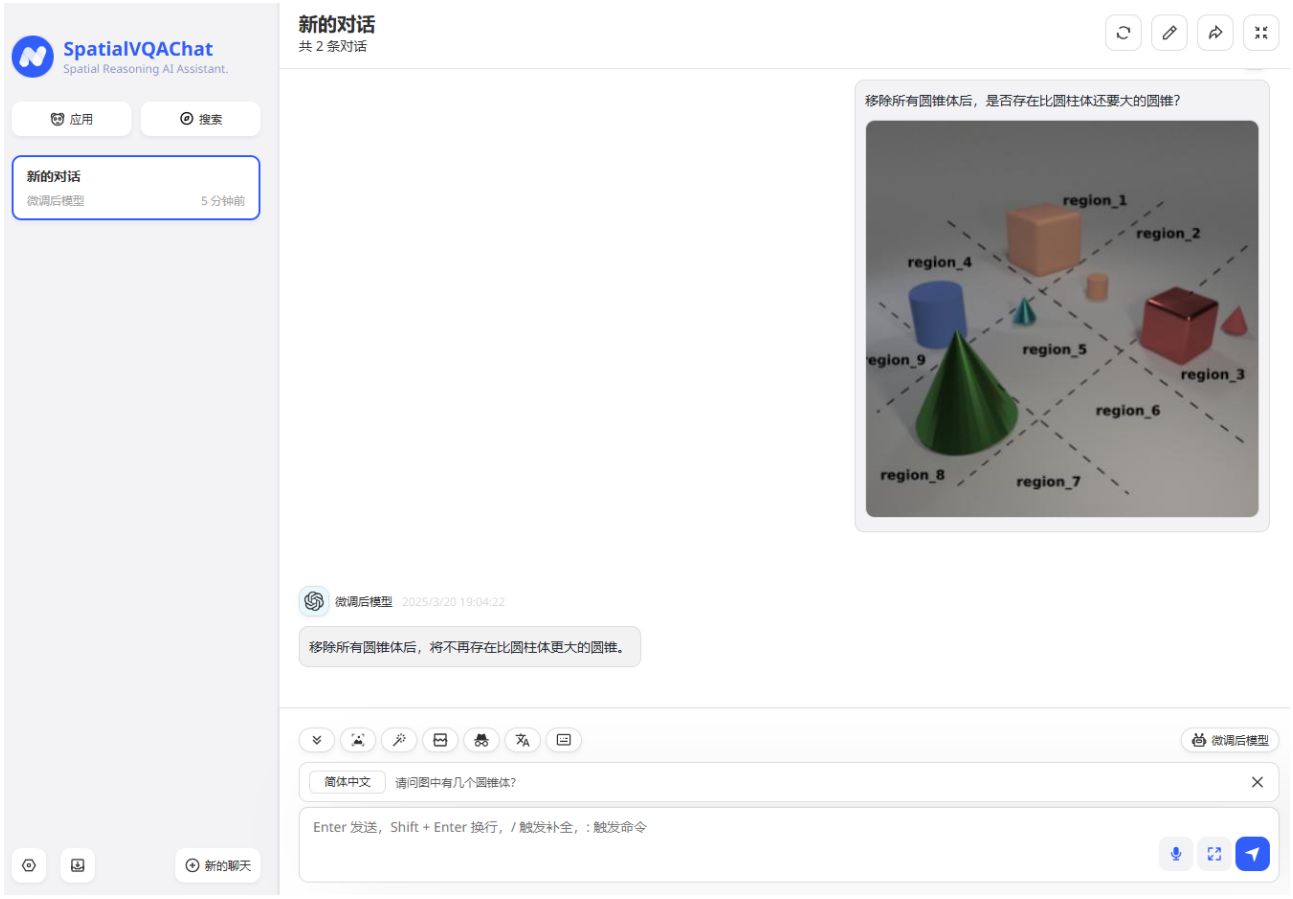


图 5.3: 上传图片并提问后的效果

Ubuntu 20.04 LTS；开发工具为 IntelliJ IDEA、IntelliJ Pycharm 和 Microsoft Visual Studio Code。

前述需求中，需要通过测试以量化评估的主要为性能需求。因此，本文通过压力测试，对系统的性能进行评估。本文使用 JMeter 工具进行压力测试。JMeter 是一种开源的压力测试工具，它可以用于对 Web 应用程序、FTP 服务器、数据库等进行性能测试。JMeter 可以模拟多个用户同时访问服务器，以此来测试服务器的性能。此外，使用 SkyWalking 对整个应用进行性能监控。图5.4为 SkyWalking 对本系统中语义解析服务的监控面板，在其中可以直观监测服务的平均响应时间、成功率、每分钟调用数等信息。

本文使用第三章中构造的数据集作为测试数据，测试的接口为用户交互接口。在开发各个接口时，已经在各个接口上设置了监控埋点。在 SkyWalking 中，能够看到各个接口的调用次数，以及接口的响应时间等信息。即便是只对某一个接口发起模拟请求，进行压力测试，也可以通过 SkyWalking 来查看其它所有相关接口的相关参数。用户交互接口处于整个调用链路的起点，对其进行压测，能够观测到整个 VQA 系统在执行一次任务时，所有相关接口的情况，故对其进行压测。

首先，确定压测目标。本系统作为一个原型系统，拟定的用户群体规模上限在 20 人左右，则可以假设并发用户数目标为 15 人。通过对 ChatGPT、DeepSeek、豆包等现有的聊天机器人的使用的观察，发现目前普遍限制同一用户在同一时刻只能进行一个对话，若在原有对话仍在生成答案的同时，切换到新对话，将会导致原有对话的生成中断。所以，本原型系统在压测目标上，设置每个用户的线程数都为 1。对以上各个接口，压测时长设置为 30 分钟。各用

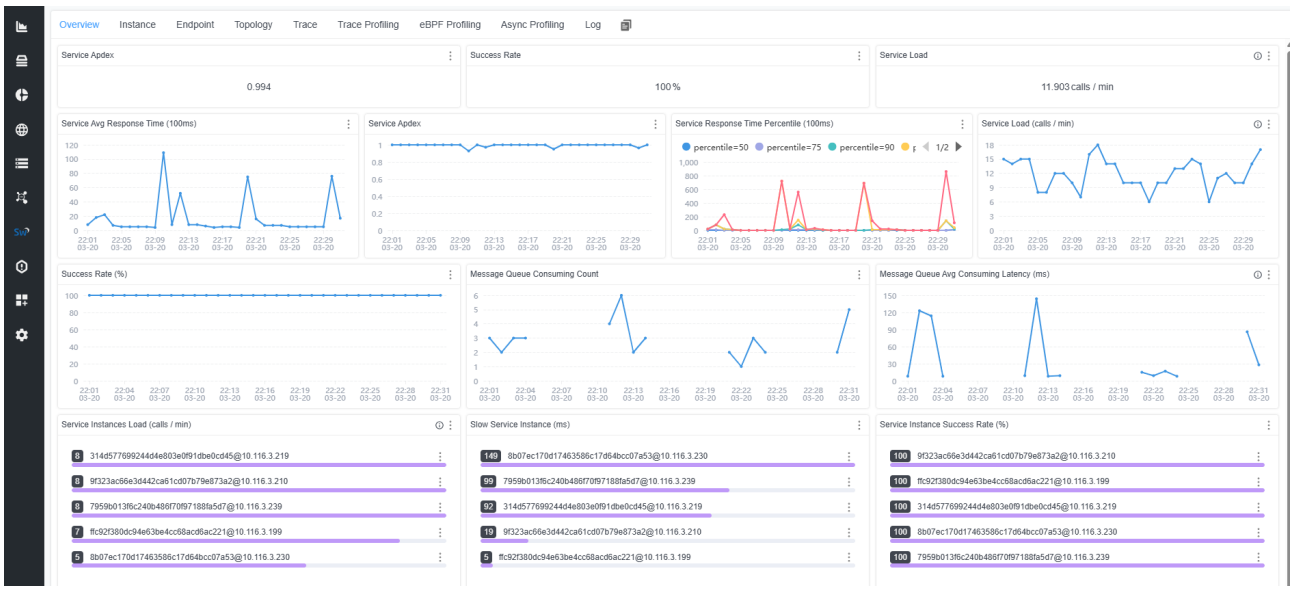


图 5.4: 链路监控面板

户线程加载，选用阶梯式加载的方式，随着时间推移，逐步增加并发用户数，以便观察系统在压力逐步增大时的性能变化。

其次，执行压测。在压测过程中，实时监控接口的性能指标，包括响应时间、吞吐量和错误率，同时监控系统的 CPU 使用率、内存使用率。

最终测试结果见表??。在 15 个并发用户下，系统响应整体较快，能满足大多数用户对视觉问答的需要。TPS 表现整体稳定，错误率也比较低。总体而言，压测结果表明，该系统能够满足其作为原型系统的要求，符合设计预期。

参数	指标	参数	指标
平均响应时间	5.1s	90% 响应时间	4.9s
TPS	12	错误率	0.5%
CPU 使用率	60.9%	内存使用率	56.8%

表 5.4: 压力测试结果

5.6　本章小结

本章设计并实现了以面向空间推理领域的神经符号系统框架为核心的 VQA 系统，首先从功能需求和非功能需求的角度展开，明确了实现 VQA 系统所需的工作重点。然后介绍了系统的整体架构，并对系统的数据库、安全方面做了具体介绍。最后，对系统的主要功能进行了展示，并对系统进行压力测试，证明了系统的实际应用效果以及对可用性。

第六章　结论与展望

6.1　本文工作总结

本文的主要工作如下：

1. 构建旨在考察模型复杂空间推理能力的数据集。基于CLEVR数据集，本文构造了SRASP数据集，并通过统计分析、双盲审核等机制，保证了数据集的质量和难度。
2. 设计了一种面向空间推理领域的神经符号框架。该框架包含视觉场景理解、语义解析、迭代反馈与规则修正、规则蒸馏、ASP推理模块，并采用对比实验进行验证，证明该框架在不同的LLM上均可提升系统对空间推理问题的解答能力，具有较强的泛化能力。
3. 设计实现了一个视觉问答系统。该原型系统以本文设计的面向空间推理领域的神经符号框架为核心，采用微服务设计理念对各模块进行拆分，且采用了Redis缓存等方案对系统进行优化，以尽可能缩短响应时间，提升用户体验。最终通过压力测试等技术手段，验证了该原型系统能够承受预想的用户体量和并发请求量，符合设计预期。

6.2　未来工作展望

本文设计的面向空间推理的神经符号框架在视觉问答系统中，对增强其空间推理能力起到了积极作用，但仍有许多方面有待完善。未来的工作可以从以下几个方面展开。

迭代反馈机制的改进。目前，迭代反馈机制采用的是方案是，最多进行三轮迭代。如果在三轮迭代中，某一轮生成的ASP程序能够顺利被Clingo执行（即不再出现语法、基础化或推理错误），则认为程序已经达到预期的正确性，此时可以提前停止迭代。这一方案在一定程度上实现了自适应性，实现了效率和效果之间的平衡，然而仍有进步的空间。可以考虑的改进方向有以下几点：

1. 利用元学习或者贝叶斯优化的方法，自动搜索最优的迭代次数和反馈策略。通过在不同任务或数据集上的交叉验证，可以自适应地调整迭代策略，使其在不同场景下达到最佳表现。
2. 采用强化学习的方法，训练一个Agent，使其会根据当前生成ASP程序的状态和错误信息动态决定是否继续迭代。类似于LLM-ARC中自动反馈调整的思路，Agent可以通过奖励信号来学习何时停止迭代，以达到最佳的平衡效果[50]。

对数据集的改进。本文构造的SRASP数据集中，为了简化对相关问题的研究，采用的是Blender引擎渲染后的静态的3D几何图像。然而目前应用领域对多模态模型的要求越来越高，静态图像为主的VQA数据集显然并不能充分考察模型对动态空间变化的推理能力。此外，本文构造的SRASP数据集，图像的内容是相对较为简单的几何图形，通过脚本进行生成而得，无法模拟真实物理世界中的物体运动、视角变化等动态场景。尤其是当前智能机器人领域发

展加快，对智能体的要求越来越高，智能体需要实时理解自身运动与物体位置的关系（如“转弯后如何避开障碍物”），而静态问答无法满足需求。可以考虑的改进方向大致有以下几点：

1. 生成动态空间问题。通过 Unity 或者 Unreal 等物理引擎的精确控制和程序化生成策略，在合成场景中模拟动态交互，自动生成涉及运动、视角变化和因果推理的动态空间问题。
2. 增加人类视角问题。本文的 SRASP 数据集中的问题，均为相机视角下的问题。可以在生成问题时，增加更多 VQA 系统在图像中的人类视角的问题，以考察 VQA 系统在视角适应上的能力。

参考文献

[1] Goyal Y, Khot T, Summers-Stay D, et al. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering[C]. in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 6904-6913.

[2] Pratt S, Covert I, Liu R, et al. What does a platypus look like? generating customized prompts for zero-shot image classification[C]. in: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023: 15691-15701.

[3] Alaluf Y, Richardson E, Tulyakov S, et al. Myvlm: Personalizing vlms for user-specific queries[C]. in: European Conference on Computer Vision. 2024: 73-91.

[4] Kuo W, Cui Y, Gu X, et al. F-vlm: Open-vocabulary object detection upon frozen vision and language models[J]. ArXiv preprint arXiv:2209.15639, 2022.

[5] Huang D A, Liao S, Radhakrishnan S, et al. Lita: Language instructed temporal-localization assistant[C]. in: European Conference on Computer Vision. 2024: 202-218.

[6] Lv T, Huang Y, Chen J, et al. Kosmos-2.5: A multimodal literate model[J]. ArXiv preprint arXiv:2309.11419, 2023.

[7] Fu X, Hu Y, Li B, et al. Blink: Multimodal large language models can see but not perceive[C]. in: European Conference on Computer Vision. 2024: 148-166.

[8] Majumdar A, Ajay A, Zhang X, et al. Openeqa: Embodied question answering in the era of foundation models[C]. in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024: 16488-16498.

[9] Zhang J, Cai M, Xie T, et al. Countercurate: Enhancing physical and semantic visio-linguistic compositional reasoning via counterfactual examples[J]. ArXiv preprint arXiv:2402.13254, 2024.

[10] Rahmanzadehgervi P, Bolton L, Taesiri M R, et al. Vision language models are blind: Failing to translate detailed visual features into words[EB/OL]. 2025. https://arxiv.org/abs/2407.06581. arXiv: 2407.06581 [cs.AI].

[11] Gao J, Sarkar B, Xia F, et al. Physically grounded vision-language models for robotic manipulation[C]. in: 2024 IEEE International Conference on Robotics and Automation (ICRA). 2024: 12462-12469.

[12] Nasiriany S, Xia F, Yu W, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms[J]. ArXiv preprint arXiv:2402.07872, 2024.

[13] Konenkov M, Lykov A, Trinitatova D, et al. Vr-gpt: Visual language model for intelligent virtual reality applications[J]. ArXiv preprint arXiv:2405.11537, 2024.

[14] Huang C, Mees O, Zeng A, et al. Visual language maps for robot navigation[C]. in: 2023 IEEE International Conference on Robotics and Automation (ICRA). 2023: 10608-10615.

[15] Driess D, Xia F, Sajjadi M S, et al. Palm-e: An embodied multimodal language model[J]., 2023.

[16] Gelfond M, Lifschitz V. The stable model semantics for logic programming[J]. Proceedings of the 5th International Conference on Logic Programming (ICLP/SLP), 1988: 1070-1080.

[17] Gebser M, Kaminski R, Kaufmann B, et al. Answer Set Solving in Practice[M]. Morgan & Claypool Publishers, 2012.

[18] Garcez A S, Broda K, Gabbay D M. Neural-Symbolic Learning Systems: Foundations and Applications[M]. Springer, 2002.

[19] Wang R, Sun K, Kuhn J. Dspy-based Neural-Symbolic Pipeline to Enhance Spatial Reasoning in LLMs[EB/OL]. 2024. https://arxiv.org/abs/2411.18564. arXiv: 2411.18564 [cs.AI].

[20] Sam Abraham S, Alirezaie M, de Raedt L. CLEVR-POC: Reasoning-Intensive Visual Question Answering in Partially Observable Environments[C]. in: Calzolari N, Kan M Y, Hoste V, et al. Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024). Torino, Italia: ELRA, 2024: 3297-3313.

[21] Antol A, Agrawal A, Lu J, et al. VQA: Visual Question Answering[J]. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015: 2425-2433.

[22] Ren M, Kiros R, Zemel R S. Exploring Models and Data for Image Question Answering[C]. in: Proceedings of the IEEE International Conference on Computer Vision. 2015: 2953-2961.

[23] Malinowski M, Fritz M. Neural-Image-Question Embedding for Visual Question Answering[C]. in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 2015-2023.

[24] Lu Y, Wang J, Gao J, et al. Look, Read and Answer: Towards Universal Visual Question Answering Models[C]. in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 10577-10586.

[25] Xu K, Ba J, Kiros R, et al. Stacked Attention Networks for Image Question Answering[C]. in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 2011-2019.

[26] Tan H, Bansal M. LXMERT: Learning Cross-Modality Encoder Representations from Transformers[C]. in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2019: 5100-5111.

[27] Radford A, Kim J W, Hallacy C, et al. Learning Transferable Visual Models From Natural Language Supervision[J]. ArXiv preprint arXiv:2103.00020, 2021.

[28] Li J, Li D, Xiong C, et al. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation[J]. ArXiv preprint arXiv:2201.12086, 2022.

[29] Hong Y, Lin C, Du Y, et al. 3d concept learning and reasoning from multi-view images[C]. in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 9202-9212.

[30] Gu Q, Kuwajerwala A, Morin S, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning[C]. in: 2024 IEEE International Conference on Robotics and Automation (ICRA). 2024: 5021-5028.

[31] Cohn A, et al. Evaluation of GPT-4 on Qualitative Spatial Reasoning Tasks[J]. Journal of Artificial Intelligence Research, 2023, 70: 1-20.

[32] Bang Y, Cahyawijaya S, Lee N, et al. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity[C]. in: Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers). 2023: 675-718.

[33] Ishay A, Yang Z, Lee J. Leveraging Large Language Models to Generate Answer Set Programs[J]. ArXiv preprint arXiv:2307.07699, 2023. arXiv: 2307.07699 [cs.AI].

[34] Wałęga P A, Bhatt M, Schultz C. ASPMT(QS): Non-Monotonic Spatial Reasoning With Answer Set Programming Modulo Theories[Z]. 2015. arXiv: 1506.04929 [cs.AI].

[35] Baryannis G, Tachmazidis I, Batsakis S, et al. A Trajectory Calculus for Qualitative Spatial Reasoning Using Answer Set Programming[J]., 2018. arXiv: 1804.07088 [cs.AI].

[36] Gokhale A, Banerjee S, Baral C. Mutual Information Maximization for Robust Multimodal Representations[C]. in: Proceedings of the Neural Information Processing Systems (NeurIPS). 2020.

[37] Eiter T, Higuera N, Oetsch J, et al. A Neuro-Symbolic ASP Pipeline for Visual Question Answering[J]. Theory and Practice of Logic Programming, 2022, 22(5): 739-754.

[38] Pan L, Albalak A, Wang X, et al. LOGIC-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning[C]. in: Findings of the Association for Computational Linguistics: EMNLP 2023. 2023: 4000-4012.

[39] 李智涛, 周之平, 叶琴. 基于空间注意力推理机制的视觉问答算法研究[J]. 计算机应用研究, 2021, 38(3): 952-955.

[40] Shrestha R, Kafle K, Kanan C. Answer Them All! Toward Universal Visual Question Answering Models[C]. in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019: 6559-6568.

[41] Costanzino A, Ramirez P Z, Lisanti G, et al. Multimodal Industrial Anomaly Detection by Crossmodal Feature Mapping[C]. in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2024: 12345-12354.

[42] Wu W, Mao S, Zhang Y, et al. Mind's Eye of LLMs: Visualization-of-Thought Elicits Spatial Reasoning in Large Language Models[C]. in: Adv/lances in Neural Information Processing Systems (NeurIPS). 2024.

[43] Yang P, Wang J, Gan R, et al. Zero-Shot Learners for Natural Language Understanding via a Unified Multiple Choice Perspective[C]. in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022: 7010-7022.

[44] Li Z, Zhou Z, Ye Q. Imagine while Reasoning in Space: Multimodal Visualization-of-Thought[J]. ArXiv preprint arXiv:2501.07542, 2025.

[45] Shah M, Chen X, Rohrbach M, et al. Cycle-Consistency for Robust Visual Question Answering[C]. in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019: 6647-6656.

[46] Van Gelder A, Ross K A, Schlipf J S. The Well-Founded Semantics for General Logic Programs[J]. Journal of the ACM, 1991, 38(3): 619-649.

[47] Gao L, Madaan A, Zhou S, et al. PAL: Program-Aided Language Models[C]. in: Proceedings of the 40th International Conference on Machine Learning (ICML): vol. 202. 2023: 10764-10799.

[48] He-Yueya J, Poesia G, Wang R E, et al. Solving Math Word Problems by Combining Language Models with Symbolic Solvers[J]. ArXiv preprint arXiv:2304.09102, 2023.

[49] Feng J, Xu R, Hao J, et al. Language Models Can Be Deductive Solvers[C]. in: Findings of the Association for Computational Linguistics: NAACL 2024. 2024: 4026-4042.

[50] Kalyanpur A, Saravanakumar K K, Barres V, et al. LLM-ARC: Enhancing LLMs with an Automated Reasoning Critic[EB/OL]. 2024. https://arxiv.org/abs/2406.17663. arXiv: 2406.17663 [cs.CL].

附录

.1 SRASP 数据集示例例

.1.1 环境

生成的每个场景都需要满足其对应环境中的所有约束。以下是 SRASP 中所有环境共享的通用规则，所有规则均以 ASP 表示：

```
1. property(color, gray). property(color, red).
2. property(color, blue). property(color, green).
3. property(color, brown). property(color, purple).
4. property(color, cyan). property(color, yellow).
5. property(shape, cube). property(shape, cylinder).
6. property(shape, sphere). property(shape, cone).
7. property(size, small). property(size, medium).
8. property(size, large).
9. property(material, rubber).
property(material, metal).
10. region(0). region(1). region(2). region(3).
11. right_R(0, 0). right_R(0, 1). right_R(0, 2).
right_R(0, 3).
12. right_R(1, 1). right_R(1, 3).
13. right_R(2, 0). right_R(2, 1). right_R(2, 2).
right_R(2, 3).
14. right_R(3, 1). right_R(3, 3).
15. left_R(R1, R2) :- right_R(R2, R1).
16. front_R(0, 0). front_R(0, 1). front_R(0, 2).
front_R(0, 3).
17. front_R(1, 0). front_R(1, 1). front_R(1, 2).
front_R(1, 3).
18. front_R(2, 2). front_R(2, 3).
19. front_R(3, 2). front_R(3, 3).
20. behind_R(R1, R2) :- front_R(R2, R1).
21. sameProperty(X1, X2, P) :- hasProperty(X1,P,V),}
22. hasProperty(X2,P,V), X1!=X2.
23. same_color(X,Y):- sameProperty(X, Y, color).
24. same_size(X,Y):- sameProperty(X, Y, size).
```

```
25. same_shape(X,Y):- sameProperty(X, Y, shape).
26. same_material(X,Y):- sameProperty(X, Y, material).
27. 1{hasProperty(X, color, V) :
28. property(color, V)}1 :- object(X).
29. 1{hasProperty(X, material, V) :
30. property(material, V)}1 :- object(X).
31. 1{hasProperty(X, shape, V) :
32. property(shape, V)}1 :- object(X).
33. 1{hasProperty(X, size, V) :
34. property(size, V)}1 :- object(X).
35.1{at(X, R): region(R)}1 :- object(X).
36.:- sameProperty(X1, X2, color),
37. sameProperty(X1, X2, material),
38. sameProperty(X1, X2, size)},
39. sameProperty(X1, X2, shape),
40. object(X1), object(X2), X1!=X2.
41.exceed_region_capacity(R) :42. #count{X: object(X), at(X, R)} >= 4,
   ↪region(R).
43:- exceed_region_capacity(_).
```

以上通用规则对应的自然语言含义如下：

```
1-9. 对象必须具有四个属性维度：颜色、形状、尺寸、材质。
1-4. 颜色属性取值范围（8种）：灰色、红色、蓝色、绿色、棕色、紫色、青色、黄色。
5-6. 形状属性取值范围（4种）：立方体、圆柱体、球体、圆锥体。
7-8. 尺寸属性取值范围（3级）：小、中、大。
9. 材质属性取值范围（2种）：橡胶、金属。
10. 场景划分为四个空间区域，编号为0、1、2、3。
11. 当对象A位于区域0时，其右侧对象B的合法区域：0/1/2/3。
12. 当对象A位于区域1时，其右侧对象B的合法区域：1/3。
13. 当对象A位于区域2时，其右侧对象B的合法区域：0/1/2/3。
14. 当对象A位于区域3时，其右侧对象B的合法区域：1/3。
15. 方位对称性规则：若对象A在对象B右侧，则对象B必在对象A左侧。
16. 当对象A位于区域0时，其前方对象B的合法区域：0/1/2/3。
17. 当对象A位于区域1时，其前方对象B的合法区域：0/1/2/3。
18. 当对象A位于区域2时，其前方对象B的合法区域：2/3。
19. 当对象A位于区域3时，其前方对象B的合法区域：2/3。
20. 方位对称性规则：若对象A在对象B前方，则对象B必在对象A后方。
27-28. 颜色属性强制单值约束：每个对象必须且只能具有一个颜色值。
29-30. 材质属性强制单值约束：每个对象必须且只能具有一个材质值。
```

31-32. 形状属性强制单值约束：每个对象必须且只能具有一个形状值。
33-34. 尺寸属性强制单值约束：每个对象必须且只能具有一个尺寸值。
35. 空间位置强制单值约束：每个对象必须且只能被分配至一个区域。
36-40. 对象差异性原则：任意两个对象不得在所有四个属性（颜色/形状/尺寸/材质）上↪完全一致。
41-43. 区域容量限制：每个空间区域最多容纳3个对象。

以下用 ASP 表示的约束，用来表示图**??**所示场景所属的特定环境。

```
44. object(0..4).
45. :- object(X), at(X, 0),
hasProperty(X, size, large).
46. :- object(X), at(X, 0),
hasProperty(X, shape, cylinder).
47. :- object(X), at(X, 0),
hasProperty(X, shape, cone).
48. :- object(X), at(X, 1),
hasProperty(X, size, small).
49. :- object(X), at(X, 1),
hasProperty(X, shape, cone).
50. :- object(X), at(X, 1),
hasProperty(X, material, rubber).
51. :- object(X), at(X, 1),
hasProperty(X, shape, cube).
52. :- object(X), at(X, 2),
not hasProperty(X, size, medium).
53. :- object(X), at(X, 2),
not hasProperty(X, material, metal).
54. :- object(X), at(X, 2),
hasProperty(X, material, rubber).
55. :- object(X), at(X, 2),
hasProperty(X, shape, sphere).
56. :- object(X), at(X, 2),
hasProperty(X, shape, cube).
57. :- object(X), at(X, 3),
hasProperty(X, size, small).
58 :- object(X), at(X, 3),
not hasProperty(X, material, metal),
59. not hasProperty(X, color, blue).
60. :- #count{X1, X2: sameProperty(X1, X2, shape),
```

```
61. object(X1), object(X2), at(X1, 3), at(X2, 2),
62. hasProperty(X1, color, yellow),
63. hasProperty(X2, color, yellow)} >= 4.
64. :- #count{X1, X2: sameProperty(X1, X2, color),
65. object(X1), object(X2),
66. at(X1, 0), at(X2, 3)} >= 2.
```

以下是对前述的规则的逐行解释：

```
44. 场景中共存在5个对象。
45. 区域0内禁止存在大尺寸对象。
46. 区域0内禁止存在圆柱形对象。
47. 区域0内禁止存在圆锥形对象。
48. 区域1内禁止存在小尺寸对象。
49. 区域1内禁止存在圆锥形对象。
50. 区域1内禁止存在橡胶材质对象。
51. 区域1内禁止存在立方体对象。
52. 区域2内所有对象必须为中尺寸。
53. 区域2内所有对象必须为金属材质。
54. 区域2内禁止存在橡胶材质对象。
55. 区域2内禁止存在球形对象。
56. 区域2内禁止存在立方体对象。
57. 区域3内禁止存在小尺寸对象。
58-59. 区域3内所有对象必须满足以下条件之一：金属材质或者蓝色外观。
60-63. 区域3与区域2内黄色对象组合规则：相同形状的黄色对象配对组数 1。
64-66. 区域0与区域3联合约束：  具有相同颜色的对象配对组数 =0（严格禁止）。
```

.1.2 问题

.1.3 回答集

.1.4 推理步骤

.2 数据集统计

.2.1 问题模板分布

致谢

到这里，我的学生时代就要告一段落了，真的要从学生变成社会人了。虽然不想承认，但还是要接受这个现实。心中感慨万千，在此写下这篇致谢，权且当作一个简短的回忆录。

第一，我想感谢我的导师。张老师是一位非常和蔼可亲的老师，他极少发脾气，平时见到我们同学们一直都是笑呵呵的，并且给我们发了不少助研金，能让我在硕士阶段不仅不用向父母要钱，还能自己存下不少存款，让很多同龄的硕士研究生都非常羡慕。其次，他在培养上认真负责，对同学们的开题和毕业论文都十分上心，能让我们顺利毕业。再者，他放我们出去实习，这一点在学院的众多导师中，其实还是比较难得的。当下，找互联网企业的对口工作，或者是央国企，都十分看重实习经历，想找一个好点的算法或者后端的工作，都得要最少最少一段，一般都要两段实习经历。张老师能够让我们离校实习，对我们学生个人的求职，无疑是十分重要的一点。

第二，我想感谢东大。依稀记得还在南师大准备考研的时候，每天我都会播放东大校歌《临江仙》，力争让自己在精神上“皈依”东大，支撑自己每天能学进去 11 个小时。没考进来之前，每天都是念着东大的好，考进来之后，开始经常吐槽学校。但思前想后，还是要感谢东大。东大作为 985，还是给了我一些普通 211 享受不到的平台和资源。能让我简历至少通过一些单位的筛选，能有资格考选调生、公务员特招，能享受到更好的资源。如果我还在南师大，我是不敢想象我有机会参加就业办的活动，公费去广州和青岛游玩的。

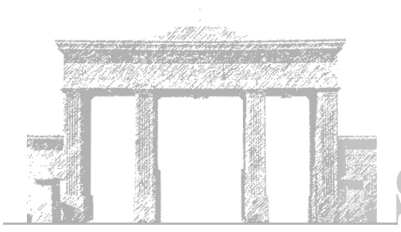
第三，我想感谢我的亲人。我的父母是传统的山东农村人，他们没什么文化，见识也不多，做不到像双公务员家庭那样，给子女在经济、升学、求职上给予很多帮助，但是他们确实也是尽己所能，让我有机会考大学。难能可贵的是，他们至少没有阻挠我去自主做一些重大决策，在我追求个人发展的路上，他们没有像一些农村父母那样去当“拦路虎”，我想这也是比较难能可贵的一点。我也想感谢我的姐姐和姐夫，他们在经济上给予了我一些帮助，每次我到北京考试或者实习，他们都热情接待我。他们两人从山东一路考到北京，双方父母都没啥本事，凭自己的努力和时代的东风，在北京打下了一片天地，着实让我羡慕。

第四，我想感谢南京这座城市。我从小一直在村镇上，拿个快递都要骑电动车去 2 公里外的镇上，娱乐生活极其匮乏，也没什么玩得来的同龄人。南京给了我大城市的感觉，玄武湖、紫金山、长江，让我体验到这座城市的魅力。我还记得 2018 年 8 月 30 日那天的六点多钟，天色已经黑了，我坐着客车跨过长江二桥，看到夜幕下奔涌的长江的感受。七年来，我走遍了南京城区，从北京西路到九乡河西路，从燕子矶到禄口，桥林、岔路口、麒麟门等等这些地名我都如数家珍，南京的一草一木，山山水水，都融入我的血脉之中。我也学会了一些南京方言，宿管和食堂的阿姨一度都问我：“小伙子啊是南京人啊?”。苏 A、3201、025，这几个南京的同义词，一直都萦绕在我的脑海之中。其实硬要说南京好，也没那么好，但是也没那么坏。我是个比较懒的人，一旦在一个城市呆久了，就懒得换地方，本能地进入了舒适圈，就不想再出来。虽然由于客观就业形势，我无法留在南京工作，没能通过名校优生和江苏省考进入公务员序列。但我今后不论身处何地，我都会一如既往的关心南京，支持南京。

当然，最重要的，还是要感谢我自己。我出身农村，从小放养，没上过补习班，父母也不懂选大学选专业，本科误入统计专业，后来下定决心从南师大统计学跨考到东大计算机专业，全身心复习 10 个月，每天学习 11 个小时，最终考了 395 分考到东大。我自己没有什么天资，最多是勤奋一点，愿意去卷。经济下行，内卷加剧，不少应届生都找不到工作，更遑论薪水较高的工作，我作为学历一般，本科非计算机专业，实习经历不够“卷”的学生，能够签一个互联网大厂的工作，实属不易。并且在实习转正后，并没有放弃求职，而是继续参加四个省市的选调生考试、国考、江苏省考、广东省考，尽管只是进面，没有最终考上，但至少还是积极主动去试过，试图去为自己争取机会。作为只能吃到时代黑利的自己，已经不错了。没赶上好时候，就只能在思想上去“比烂”，当一当阿 Q，让自己过得心里舒服点，不然自己更难受，不是吗？

祝福一路上帮助过我的亲人和朋友，祝福母校，祝福南京，祝福自己。希望我的美股账户早日能到 100 万美金，早日考上公务员。

心求至善



SOUTHEAST UNIVERSITY