软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(10):4671−4694 [doi: 10.13328/j.cnki.jos.007294] [CSTR: 32375.14.jos.007294]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

结合大语言模型和领域知识库的证券规则规约方法*

李靓果[1], 薛志一[1], 陈小红[1], 张 民[1], 陈良育[1], 李萍萍[2], 姜婷婷[2]

[1](上海高可信计算重点实验室 (华东师范大学), 上海 200062)

[2](国泰君安证券股份有限公司 数据中心, 上海 201201)

通信作者: 陈小红, E-mail: xhchen@sei.ecnu.edu.cn; 姜婷婷, E-mail: jiangtingting@gtjas.com

摘　要: 业务规则在证券领域至关重要, 它们是证券交易系统的需求的来源. 鉴于业务规则的易变性, 如何提升从业务规则交易文档中规约出软件需求的效率, 成为一个核心的问题. 证券业务规则文档具有与软件不相关描述多、专业术语多、上下文相关表述多和抽象表示多等特性, 其自动化规约需要领域相关知识的支持. 如何将领域相关知识融入自动化过程中, 成为规约的关键问题. 提出了一种结合大语言模型和领域知识库的证券领域业务规则自动规约方法, 对大语言模型通过微调、上下文学习等嵌入领域知识执行规则分类和需求信息提取等自然语言处理任务. 此外, 还通过领域知识库提供专业领域知识, 进行需求的可操作化和关系识别, 最终形成数据流形式的需求规约. 评估结果显示, 该方法能够处理各种证券交易领域的业务规则文档, 在评估数据集上的平均功能点识别率为91.97%, 达到甚至超越了领域专家的水平, 与人类参与者相比, 效率平均提高了 10 倍.

关键词: 证券领域; 业务规则; 需求规约; 大语言模型; 领域知识; 软件需求

中图法分类号: TP311

中文引用格式: 李靓果, 薛志一, 陈小红, 张民, 陈良育, 李萍萍, 姜婷婷. 结合大语言模型和领域知识库的证券规则规约方法. 软件学报, 2025, 36(10): 4671–4694. http://www.jos.org.cn/1000-9825/7294.htm

英文引用格式: Li LG, Xue ZY, Chen XH, Zhang M, Chen LY, Li PP, Jiang TT. Specification Method of Securities Rules Integrating Large Language Models and Domain Knowledge Base. Ruan Jian Xue Bao/Journal of Software, 2025, 36(10): 4671–4694 (in Chinese). http://www.jos.org.cn/1000-9825/7294.htm

Specification Method of Securities Rules Integrating Large Language Models and Domain Knowledge Base

LI Liang-Guo[1], XUE Zhi-Yi[1], CHEN Xiao-Hong[1], ZHANG Min[1], CHEN Liang-Yu[1], LI Ping-Ping[2], JIANG Ting-Ting[2]

[1](Shanghai Key Laboratory of Trustworthy Computing (East China Normal University), Shanghai 200062, China)

[2](Data Center, Guotai Junan Securities Co. Ltd., Shanghai 201201, China)

Abstract: Business rules are crucial for the securities domain and serve as the source of requirements for securities trading systems. Due to the variability of these business rules, how to improve the efficiency of specifying software requirements from business rule trading documents has become a core problem. The securities business rule documents feature numerous software-unrelated descriptions, abundant professional terms, and many context-related expressions and abstract representations, which necessitate the support of domain-specific knowledge for automatic specification. As a result, how to integrate the domain-related knowledge into the automatic process becomes a key problem for specification. This study proposes an automatic specification method for securities domain businesses integrating large language models and the domain knowledge base. It leverages the large language models, employing techniques such as fine-tuning and in-context learning to embed domain knowledge for natural language processing tasks such as rule classification and requirement information extraction. Additionally, this study also employs the domain knowledge base to provide professional knowledge and assist in the operationalization and relationship extraction of requirements. Finally, requirement specification in the form of data flow is formed. The

* 基金项目: 国家自然科学基金 (62161146001, 62372176, 62272166); 上海市可信工业互联网软件协同创新中心项目; 上海市“数字丝路”可信智能软件国际联合实验室项目 (22510750100)
收稿时间: 2024-01-31; 修改时间: 2024-04-18, 2024-08-03; 采用时间: 2024-09-19; jos 在线出版时间: 2025-06-27
CNKI 网络首发时间: 2025-06-30

evaluation results show that the proposed approach can process business rule documents in various securities trading fields, achieving an average function point identification rate of 91.97% on the evaluation dataset, which matches or even surpasses the level of experts in the domain, with the efficiency improved by an average of 10 times compared to human participants.

Key words: securities domain; business rule; requirement specification; large language model (LLM); domain knowledge; software requirement

业务规则在证券领域扮演着关键角色, 它们是一系列指导和控制证券市场运作的规定和指南. 这些规则不仅详细规定了交易行为, 如交易的时间、数量和价格等, 也对市场参与者的行为进行了详细的监管和规范. 证券交易所的业务规则文档在约束和指导券商以及其他交易参与者的行为方面发挥着至关重要的作用. 各大证券交易所都有一套详细的业务规则文档来定义和约束交易, 如《深圳证券交易所债券交易规则》[1]、《上海证券交易所交易规则》[2]等. 在开发交易系统时, 券商必须严格遵循这些文档中规定的规则. 这些业务规则是证券交易系统需求和约束的来源, 其中与软件需求相关的部分将被用作交易系统软件需求规约的一部分, 以便进行设计、实现和测试. 如何从自然语言描述的业务规则文档中发现需求, 进一步挖掘这些需求之间的关系以形成软件需求规约, 是交易系统开发中的一个关键问题. 随着市场的快速发展和监管要求的不断变化, 业务规则展现出易变性. 在这种背景下, 如何提高业务规则规约的效率成为一个关键问题.

目前, 业内主要采用人工的方法进行业务规则的规约, 手动抽取业务规则中与软件需求相关的表示, 并识别其中的关系, 形成可测试的软件需求. 与法律[3,4]、交通[5]等领域的规则文档不同, 证券领域中交易规则文档自动规约存在如下挑战: (1) 以不受限的自然语言表述的业务规则中包含了大量证券领域专业术语, 难以抽取和理解; (2) 业务规则的表达中存在许多省略和抽象的表达, 需要对其进行可操作化; (3) 业务规则之间存在着复杂的隐式关系. 这些挑战不仅要求有强大的自然语言处理能力, 还要求具备相应的领域知识, 以便实现自动的专业术语识别、需求可操作化以及需求隐式关系识别. 领域知识对于实现业务规则的自动规约至关重要, 因此, 如何将领域知识融入规约过程成为自动化的关键.

学术界已有众多研究致力于本体制导的软件需求规约过程[6−9], 它们利用本体这种知识库提供领域知识, 从而进行软件需求的规约. 随着大语言模型的兴起, 其所具备的广泛“世界知识”[10]引起了学术界的关注, 一些学者开始研究这些模型是否掌握了专业知识. 研究表明, 大语言模型在知识密集领域缺乏专业领域知识[11]. 我们的实验也证实了现有的大语言模型在证券领域交易规则文档中的领域知识不足. 受其他领域知识图谱和大语言模型结合工作[11−13]的启发, 本文提出了一种结合大语言模型和领域知识库的证券领域业务规则自动规约方法. 在这个方法中, 我们首先利用 ChatGPT[14]等大语言模型强大的自然语言处理能力, 通过微调、上下文学习等手段嵌入隐式的领域知识, 进行软件需求信息抽取. 接下来, 我们结合领域知识库中显式可表达的知识进行了需求的可操作化和关系识别, 生成数据流形式的软件需求规约. 这为专业知识密集型文档的处理和规约提供了一种有价值的借鉴. 本文的主要贡献如下.

(1) 提出了一种证券领域业务规则自动规约方法, 该方法结合了大语言模型和领域知识库, 可以从不受控的自然语言业务规则文档自动生成需求规约, 减轻需求工程师的工作负担, 提高了业务规则规约的效率.

(2) 设计了一种将领域知识有效整合到自动化需求规约中的方法, 通过大语言模型的微调、上下文学习等执行领域知识相关的自然语言处理任务, 并建立领域知识库进行关系识别, 从而生成面向功能的数据流, 保证了需求规约生成的质量.

(3) 基于提出的方法, 设计实现了证券领域业务规则自动规约工具 LLSec, 并在 5 个证券交易领域中进行了方法评估. 评估结果表明, 本文生成的需求规约在评估数据集上的平均功能点识别率为 91.97%, 达到甚至超越了领域专家的水平, 且效率比人类参与者平均提高了 10 倍.

本文第 1 节详细讨论证券领域业务规则规约中遇到的具体问题, 并据此定义方法框架. 第 2 节给出本文模型的训练及领域知识库的构建. 第 3 节提供方法的实现细节. 第 4 节进行案例研究和实验评估. 第 5 节对有效性威胁进行讨论. 第 6 节比较了相关工作. 最后第 7 节总结全文, 并对未来工作进行展望.

1 方法框架

本节将通过案例详细说明证券领域业务规则规约中遇到的具体问题, 探索了直接使用大语言模型的限制, 并

据此定义解决方法框架.

1.1 具体问题分析

本节将详细讨论在证券领域业务规则规约中面临的问题.

第一, 必须要确定哪些业务规则与软件需求相关. 在证券领域的业务规则中, 存在一部分规则与软件的直接功能无关. 这些规则通常涉及市场操作的法规、合规性标准以及交易伦理等, 它们描述了市场行为的法律约束和道德框架, 但并不直接对软件的操作功能提出要求. 因此, 我们的目标是识别并筛选出那些与软件功能性需求直接相关的有意义规则. 《深圳证券交易所债券交易规则》[1]中的规则片段如案例 1 所示. 其中, 规则 3.1.3 与软件需求没有直接关联, 因为该规则主要描述了交易所在市场的管理职责和行为, 并未直接指导或要求软件实现特定的功能或操作. 而其他 4 条规则指定了交易软件在处理债券交易时必须遵循的要求, 是软件需求分析和实现的重要依据. 因此, 我们需要过滤掉规则 3.1.3, 保留软件需求相关的其他规则.

案例 1: 深圳证券交易所债券交易规则

规则 3.1.3. 本所可以对债券交易方式实施动态调整并及时向市场公布.

规则 3.1.5. 采用匹配成交方式的, 每个交易日的 9:15–9:25 为开盘集合匹配时间, 9:30–11:30、13:00–15:30 为连续匹配时间; 采用点击成交、询价成交和协商成交方式的, 交易时间为每个交易日的 9:00–11:30、13:00–20:00.

规则 3.3.4. 采用匹配成交方式的, 债券现券的申报数量应当为 10 万元面额或者其整数倍, 卖出时不足 10 万元面额部分, 应当一次性申报; 债券通用质押式回购的申报数量应当为 1 000 元面额或者其整数倍.

规则 4.4.4. 竞买日前, 卖方可以修改竞买预约要素或者取消预约.

规则 4.1.10. 每个交易日 9:20–9:25 的开盘集合匹配阶段, 本所交易系统不接受匹配成交的撤销申报.

第二, 证券领域的业务规则使用不受限的自然语言表达, 并且包含大量的领域术语, 导致需求信息的提取变得困难. 上述的规则 3.3.4 展现了典型的证券领域业务自然语言规则文本的特点: 结构上复杂多变, 内容上包含大量领域术语以及存在一定程度的省略现象. 具体而言, 规则中包含了多个子句, 每个子句针对不同的交易类型有着不同的申报数量要求, 在表达上展现出复杂的层次关系和细节分化, 使得整体理解具有一定的挑战性. 此外, 规则中的“匹配成交方式”“债券现券”“一次性申报”“债券通用质押式回购”等词都是典型的证券领域术语而不是通用词汇. 这些文本特征使得传统的基于规则的提取方法效果不佳.

第三, 证券行业的业务规则经常包含抽象表达或依赖于特定上下文的表示方式. 为了生成可操作的需求规约, 必须对抽象概念进行具体化, 并通过上下文推断补全相关的约束, 以确保每条规则都包含完整的约束条件并且可执行. 例如, 案例 1 中的规则 4.4.4 不是一个直接可执行的规则, 因为它缺少交易市场、交易方式和交易品种等必要信息. 此外, “竞买预约要素”作为一个抽象概念, 需要进一步细化为具体的要素内容, 包括“竞买方式”“证券代码”等. 只有在这些额外信息得到补充后, 该规则才能变得具体、明确且可执行.

第四, 证券行业的业务规则之间存在着大量隐式的依赖关系, 这意味着需要考虑规则的执行顺序和相互影响. 理解这些依赖关系对于准确执行和应用规则至关重要. 例如, 案例 1 中的规则 4.1.10 规定了与撤销申报相关的时间, 该规则的有效执行前提是存在一笔已申报的订单. 因此, 它依赖于与申报相关的规则, 比如上述规定了申报数量的规则 3.3.4. 这种依赖关系表明, 一个看似独立的规则实际上可能与其他多个规则紧密相连, 需要在更广泛的规则体系中考虑其应用和实施.

综上, 我们发现, 证券领域业务规则规约自动化面临的挑战主要来源于领域知识. 前两个问题属于自然语言处理任务, 涉及的领域知识都比较隐式. 后两个问题则需要结合领域知识对需求进行可操作化和关系识别, 其中的领域知识可以显式表示. 因此, 如何有效地将隐式和显式的领域知识嵌入规约过程成为关键问题.

1.2 大语言模型的试用

大语言模型通过海量文本的训练获得了广泛的“世界知识”[10], 可能已经具备了相关的领域知识. 为了探索这

一可能性, 我们对比了目前最先进的大语言模型之一——GPT-4[14]和领域专家生成证券业务规则规约的表现. 我们使用案例 1 中的债券交易规则, 要求 GPT-4 和领域专家根据提示词生成对应的规约, 结果如图 1 所示.

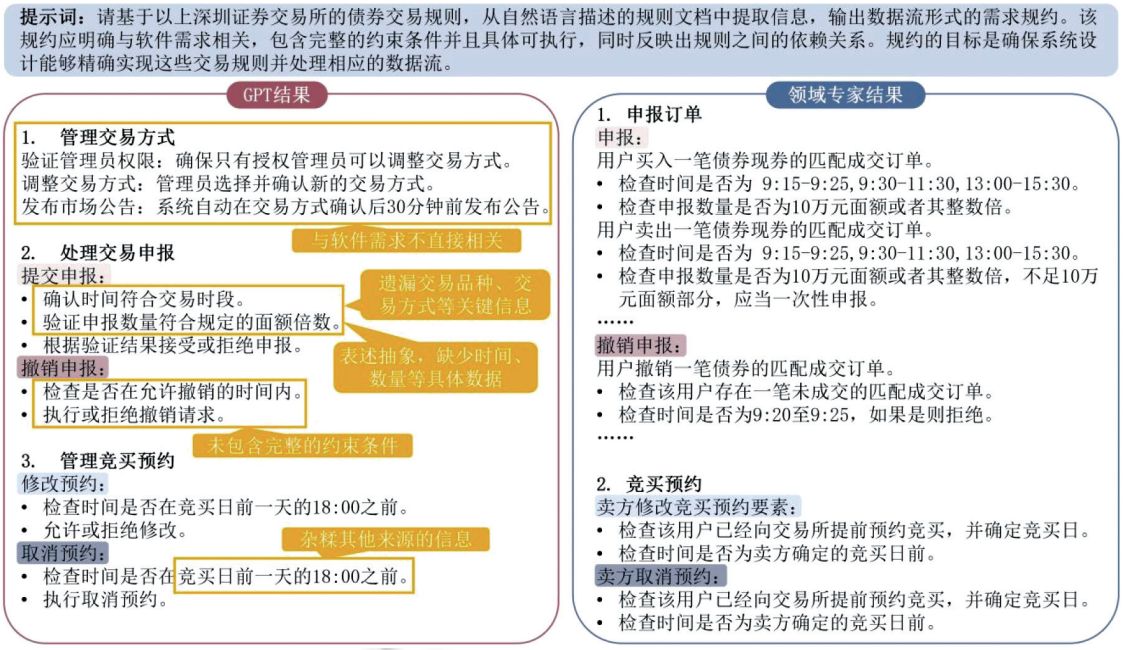


图 1 GPT-4 与领域专家生成业务规则规约的对比分析

通过分析领域专家和 GPT-4 生成的规约, 我们发现 GPT-4 在规约任务中存在以下问题.

(1) 无法精准筛选与软件需求相关的内容. 在证券领域的业务规则中, 一些规则与软件需求无关, 需要被过滤. 尽管提示中要求规约应与软件需求明确相关, 但 GPT 生成的结果中仍然包含了一些不相关的规则.

(2) 遗漏需求关键信息. 尽管 GPT 生成的规约能够准确提取一些需求信息, 但有时仍然会遗漏一些证券领域中的需求关键信息, 例如交易方式和交易品种等.

(3) 需求表述抽象, 不具有可操作性. GPT 生成的规约表述抽象, 缺乏具体的操作步骤和执行细节. 例如, “确认时间符合交易时段”这一表述未明确具体的交易时间, 导致规约在实际操作中难以实施.

(4) 需求间关系未能识别. GPT 生成的部分规约未包含需求间关系. 例如, 对于“撤销申报”, 需要在发起撤销申报前先“申报”.

(5) 幻觉与杂糅. GPT 有时会产生幻觉, 生成的规约中杂糅了一些其他来源的信息. 例如, “检查时间是否在竞买日前一天的 18:00 之前”这一内容未在深圳证券交易所的业务规则中提及, 可能引入错误, 影响规约的准确性和可靠性.

通过上述对比, 我们可以看出, 直接使用 GPT-4 生成业务规则需求规约虽显示出一定的可行性, 但也存在明显的限制. 对于一些通用的自然语言处理任务, GPT-4 表现较好, 能理解基本领域术语的含义, 并基于规则的字面表达进行简单的关系识别. 然而, 由于缺少相关领域知识, GPT-4 在执行自然语言处理任务时并不清楚哪些内容与软件需求相关以及哪些信息对于证券领域是关键的, 会产生问题 (1)、(2). 同时, GPT-4 在处理抽象表达和挖掘规则之间的依赖关系时仅能进行表面级的语义理解, 难以将抽象的表达具体化以及深入解析复杂关系, 导致问题 (3)、(4). 此外, GPT-4 因为缺少可靠的知识来源, 还可能产生幻觉, 如问题 (5), 即在生成的规约中引入与当前业务无关的错误信息. 这些都需要结合可靠的知识源——领域知识进行处理.

1.3 方法框架

为了克服上述挑战, 提出了一种新的为业务规则自动生成规约的方法. 它结合了大语言模型和领域知识库, 通

过微调、上下文学习等方式将隐式领域知识融入大语言模型, 执行其中的自然语言处理任务, 同时构建外部显式知识库进行后续的需求可操作化和关系识别, 以充分利用显式领域知识避免幻觉, 提高生成规约的质量和效率. 具体而言, 我们提出一个 4 步规约框架, 包括业务规则过滤、需求信息抽取、需求可操作化和需求关系识别, 如图 2 所示. 框架输入为自然语言描述的规则文档, 输出数据流形式的需求规约. 第 1 步业务规则过滤利用分类模型将业务规则文档中的语句分为软件需求相关规则、软件需求无关规则以及领域知识, 从而过滤掉需求无关的规则. 第 2 步需求信息抽取利用抽取模型将文档中所有软件需求相关的自然语言规则转化为形式化的软件需求. 这些形式化需求采用 Knauf 等人[15]定义的形式化业务规则 (formal business rule, FBR) 表示. 第 3 步为基于交易概念知识库的需求可操作化, 其目的是将不完整的、抽象的需求转换为具体的、可操作化的需求. 最后一步为需求关系识别, 该步骤基于交易操作序列知识识别需求的隐式依赖关系, 并据此生成数据流.

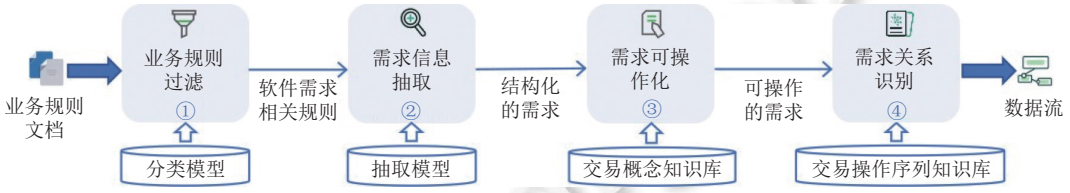


图 2 业务规则自动规约框架图

(1) 业务规则过滤. 业务规则过滤任务就是要从所有自然语言规则中筛选出软件需求相关规则, 过滤掉需求无关的规则. 筛选出的软件需求相关规则将进行后续的需求信息抽取步骤. 由于规则分类的标准通常隐含在文本语义中, 难以通过一组简单的规则明确定义, 因此我们提出构建一个分类模型, 有效识别规则文本中的关键语义信息. 该分类模型是通过在我们手动标注的语料库上对预训练模型进行微调而得到的.

(2) 需求信息抽取. 需求信息抽取任务就是要提取业务规则中的关键实体, 识别其类型并组装为形式化的软件需求. 我们利用大语言模型强大的自然语言理解和信息抽取能力来完成这个任务. 我们首先基于 GPTs 创建了针对证券领域的需求信息抽取模型, 然后利用创建的抽取模型从预处理后的规则文本中抽取出软件需求相关的关键信息, 并使用算法将这些信息组装为形式化需求. 为了使大语言模型在执行抽取任务时不遗漏关键信息, 我们构建了证券领域需求元模型, 并据此设计了信息抽取的提示词, 从而引导模型更加精准地抽取信息.

(3) 需求可操作化. 需求可操作化任务就是要从上述忠于原文的形式化需求结合交易概念知识生成具体可执行的需求规约. 这一步骤旨在解决自然语言表述抽象的问题. 主要任务包括: 1) 必要约束补全: 单条需求的约束往往不完整, 需要结合上下文信息和交易概念知识进行补全, 以生成包含全部必要约束的完整需求; 2) 抽象需求具体化: 需求中的抽象描述需要结合交易概念知识进行具体化, 转换为可操作的需求; 3) 嵌套关系处理: 需要处理需求间的引用和嵌套, 确保一条需求中包含所有相关的约束; 4) 同属性项需求的组合: 需要组合具有相同属性项的需求, 确保同一属性相关约束的完整性.

(4) 需求关系识别. 需求关系识别任务是基于交易操作序列知识识别需求间的隐式依赖关系, 生成面向功能的需求规约. 识别的依赖关系包括: 1) 基于语义的隐式依赖关系: 一些需求在语义上具有前置或后置条件, 我们使用自然语言处理 (NLP) 工具和启发式规则来识别这些关系, 生成相应的前置或后置需求; 2) 基于状态机的隐式依赖关系: 一些需求的依赖关系没有在语义上体现, 而是隐含在实际的交易流程中. 我们使用知识库中的状态机来识别这些关系, 将有关联的规则组织到一起, 并以数据流的形式输出, 形成软件需求规约.

上述前两个步骤中涉及两个特定的大语言模型: 分类模型和抽取模型, 分别实现对业务规则的过滤和需求信息的抽取. 后两个步骤中使用了两种领域知识: 交易概念知识和交易操作序列知识, 为需求的可操作化和关系识别提供了丰富且准确的信息来源. 接下来, 我们将讨论这两种大语言模型的创建和领域知识库的构建.

2 大语言模型创建与领域知识库的构建

在本节中, 我们将深入探讨分类模型和抽取模型的创建过程以及领域知识库的构建方法.

2.1 分类模型的训练

分类模型的目的是对业务规则文档中的规则进行分类. 将规则分类为软件需求相关规则、软件需求无关规则和领域知识, 从而过滤噪声规则. 对于分类模型, 我们选择对预训练模型进行微调, 理由如下: 相对于采用庞大的语言模型, 简单微调已有的预训练模型就能达到良好的分类效果, 有相关实验证明, 经过微调的 BERT 模型在金融领域文本分类任务上的 $F1$ 分数比 5 样本学习 (5-shot) 的 GPT-4 更高[16]. 由于目前缺乏公开可用的相关数据集, 需要自行构建句子分类数据集. 为此, 我们首先广泛阅读了证券领域的交易规则文档并与领域专家进行了深入交流, 制定了如下分类的标注准则.

(1) 领域知识: 这些规则描述了证券领域的专业知识、术语和概念, 用于帮助理解和应用业务规则. 它们不涉及具体的操作细节, 而是提供关于证券领域的背景知识和理论基础. 领域知识规则通常与需求没有直接关联, 而是作为参考和指导. 具体的领域知识分类见第 2.3 节.

(2) 软件需求相关规则: 这些规则约束了证券系统行为和操作, 涉及时间、数量、价格、成交方式、定价方式等具体细节, 可以通过观察、测量和操作来验证其准确性和符合性, 是需要软件严格遵循和执行的规则.

(3) 软件需求无关规则: 剩余的句子为软件需求无关规则, 这些规则不直接对软件的操作功能提出要求, 例如案例 1 中的规则 3.1.3.

基于上述准则, 我们收集并标注了证券领域的 18 篇业务规则文档, 创建了一个规则过滤数据集. 该数据集共包含 3 328 条数据, 其中有 678 条软件需求相关规则、299 条领域知识以及 2 351 条软件需求无关规则. 在训练前, 我们按照 9:1 的比例将数据集划分为训练集和验证集, 并且应用 Wei 等人[17]的方法对训练集数据进行增强, 使得增强后的训练集拥有超过 33 000 条数据, 从而有效增加数据的多样性. 我们选择中文预训练模型 Mengzi-BERT-base-fin[18]作为基础框架, 并通过微调完成规则过滤. Mengzi 是一个在中文金融语料库上预训练的基于 BERT 的模型, 拥有 103M 参数. 该模型的输入是自然语言描述的规则, 模型的输出是 3 个数字, 分别代表分类为上述 3 个类别的概率, 分类决策基于这 3 个概率值中的最大值来确定. 在训练过程中, 我们使用交叉熵作为损失函数, 并使用 AdamW 优化器[19]执行梯度下降方法. 同时, 我们设置了一个线性递减的学习率, 在 20 轮的训练中从 1E–5 递减到 0. 每轮训练中, 8 条规则被同时输入到模型进行训练. 训练完成后, 我们在验证集上评估了模型的性能. 实验结果显示, 模型的准确率达到了 99.1%, 能够较好地判别规则文本的类型.

2.2 需求信息抽取模型的创建

对于需求信息抽取模型, 我们使用了 OpenAI 推出的 GPTs 功能进行创建. 这是因为在 GPT 的试用过程中我们发现 GPT-4 可以抽取相关信息, 只是缺少对关键信息的把控. 为了提供必要的信息以及保证 GPT 输出的可控性, 我们首先对证券领域相关需求的构成要素进行了深入分析, 并基于此构建了证券领域需求元模型. 接着, 我们将元模型中的关键概念和例子融入提示词中, 通过上下文学习创建了需求信息抽取模型.

(1) 构建证券领域需求元模型. 元模型是关于特定领域模型的模型, 它定义概念并提供用于创建该领域中的模型的构建元素[20]. 通过构建元模型, 开发者可以在早期阶段更好地捕获和分析系统需求. 元模型提供了一种管理和分析复杂系统的有效手段. 我们通过 4 个步骤构建了证券领域的需求元模型.

① 需求分析: 通过对证券市场的参与者、业务流程、业务规则等的深入研究了解证券领域的核心内容.

② 定义实体和联系: 从业务规则和领域知识中提炼关键实体, 如操作人、交易品种、交易方式、交易约束等, 并确定实体之间的关系, 包括“与关系”和“或关系”.

③ 构建需求元模型: 利用已定义的实体和关系构建元模型. 构建好的元模型描述了证券领域的知识和结构, 如图 3 所示.

④ 验证和持续完善: 通过领域专家的评审与反馈, 持续调整和优化元模型的内容和结构, 以精确地定义和反映证券领域的相关特征和知识.

在我们构建的证券领域需求元模型中, 每条需求由条件子句和结果子句组成. 需求之间可能存在与、或及依赖关系. 每个条件和结果子句都由若干个约束子句构成. 约束子句是元模型中最重要的概念之一, 它包括时间、价

格、数量等 7 类不同的约束. 每类约束之间都由“与”或者“或”连接. 具体而言, 时间约束、价格约束、数量约束分别是指约束子句在时间、价格、数量方面的约束. 例如, “每个交易日的 9:00–11:30”就是对交易时间的约束. 交易方式约束是对交易规则中交易方式的限制, 在不同证券产品或市场的交易中, 交易方式可以采用多种形式. 以债券交易为例, 常见的交易方式包括匹配成交、点击成交和竞买成交等. 对于证券领域, 交易品种约束主要包含股票、债券、基金等. 交易结果约束主要表现为规则中包含的成功、失败等交易结果. 交易操作约束由操作人、操作和操作部分组成, 例如, 对于“债券投资者提交应价申报”, “债券投资者”为操作人, “提交”为操作, “应价申报”为操作部分. 在证券交易中, 操作人主要分为 5 类, 分别为交易所、券商、投资者、发行人以及监管机构.

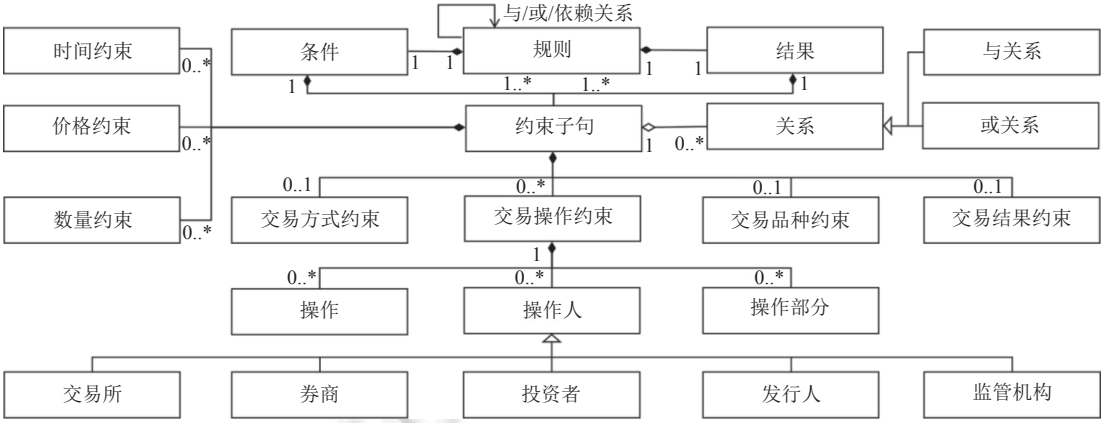


图 3 证券领域需求元模型

(2) 设计需求信息抽取实体类型. 为了提高信息抽取的精确度, 我们基于上述的元模型设计了一系列实体类型, 用以定义规则中对应的实体, 从而使得信息抽取模型更专注于与这些类型相关的信息, 减少误报和漏报. 实体类型包含交易品种、交易方式、时间、价格、数量、结果, 这与我们元模型中的概念一致. 此外, 根据元模型中的交易操作约束概念, 实体类型还包含操作人、操作和操作部分. 为了建立需求间的依赖关系, 我们定义了事件和状态, 事件表示规则中提到的前置或后置条件, 状态指示规则所处状态, 它们还可以结合状态机进行进一步的规则推理. 我们注意到, 一条规则有时会引用另一条规则中的相关约束, 为此我们增加了结合规则, 用以关联其他规则. 此外, 我们还定义实体类型操作符用于指示比较关系的词, 如大于、不足等.

特别地, 我们创新性地引入了键 (key) 和值 (value) 两种实体类型, 以提示模型识别抽取元模型中未包含的概念. 在业务规则文档中, 每条规则有着其独特的约束, 它们往往不会在文档中重复出现. 例如, 在《深圳证券交易所债券交易规则》文档中, 只有规则“采用匹配成交方式时, 债券现券的申报价格最小变动单位为 0.001 元”涉及对“申报价格最小单位”的约束. 应用上述方法, “申报价格最小单位”将被识别为键. 通过实施键-值抽取策略, 我们有效避免了关键信息的丢失, 从而确保信息抽取的全面性和准确性.

最后, 我们针对证券领域业务规则文档的信息抽取制定了共 15 种实体类型, 分别为交易品种、交易方式、时间、价格、数量、结果、操作、操作人、操作部分、事件、状态、结合规则、操作符、键以及值. 这些实体类型将被用于指导基于大语言模型的信息抽取.

(3) 创建需求信息抽取模型的提示词. 抽取模型的提示词主要包含两部分: 任务说明提示词和示例驱动提示词. 任务说明提示词向 GPT-4 介绍要完成的任务, 即解析证券交易规则并进行需求信息抽取, 并对任务中的实体类型定义、输出格式规定、特殊标注指导等细节进行了详细解释, 以确保模型能够正确理解并进行抽取. 为进一步提高信息抽取的准确率, 并确保输出的格式符合预期, 我们为需求信息抽取模型提供了来自不同文档的 100 条具体规则及其预期的信息抽取输出作为示例驱动提示词. 这些示例被纳入 GPT 的知识库中, 作为模型学习和参考的依据. 需求信息抽取模型使用的具体提示词及其设计如图 4 所示.

为了评估创建的需求信息抽取模型的效果, 我们将 3 篇业务文档中所有需求相关的规则依次输入到需求信息

抽取模型种, 获得其对应的输出结果, 并与人工标注的结果进行比较. 实验结果显示, 抽取模型信息抽取的准确率达到了 97.76%, 能够较好地识别领域术语和关键信息. 这表明上述方法为抽取领域特定自然语言文本中的关键信息提供了一种有效的解决方案.

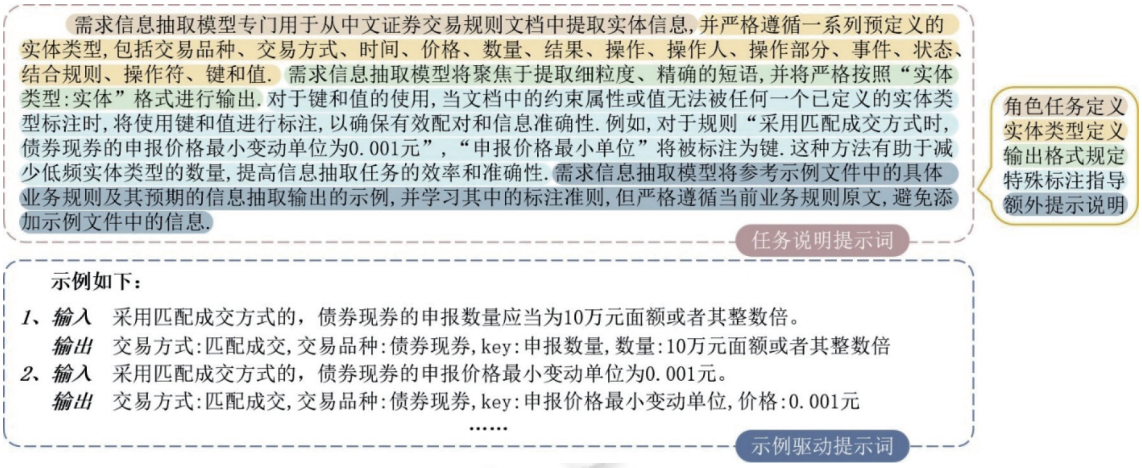


图 4 需求信息抽取模型提示词

2.3 领域知识库的构建

我们将领域知识分为两大类, 分别是交易概念领域知识和交易操作序列领域知识, 分别用于需求的操作化和关系识别. 其中交易概念领域知识关注对交易中静态概念的分类、解释和组合关系的描述, 而交易操作序列领域知识则涉及交易过程中操作执行的顺序和状态变化的相关知识.

(1) 交易概念领域知识. 交易概念领域知识主要包括解释型、分类型和组合型知识. 其中, 解释型领域知识主要是对领域特定名词的定义和解释, 比如“价格优先原则是指较高价格买入申报优先于较低价格买入申报, 较低价格卖出申报优先于较高价格卖出申报”. 分类型领域知识是对领域特有概念的分类, 比如“债券交易方式有匹配成交、点击成交、询价成交、竞买成交、协商成交以及本所认可的其他交易方式”. 组合型领域知识是对领域概念之间的组合关系的描述, 比如“意向申报的申报要素应当包括证券代码、交易方向、发送范围、证券账户号码、是否匿名等内容”.

交易概念领域知识的获取主要有两种来源: 文档自动提取和领域专家补充. 我们定义了一组启发式的规则, 结合算法自动提取文档中的领域知识. 对于规则过滤模型判断为领域知识的句子, 算法首先会依据句子中的关键字判断它对应领域知识的类别. 若包含“是指”“为”等表示解释型的词, 则为解释型知识, 算法提取要解释的名词及它对应的解释; 若包含“是一种”“有”等表示 is-a 关系的词, 则为分类型知识, 算法提取要分类的概念以及它包含的具体类别; 若包含“下列”“包括”等表示 has-a 关系的词, 则为组合型领域知识, 算法对这种知识的处理同分类型知识. 最后, 算法将领域知识整合到领域知识库中, 供需求可操作化使用. 在实际使用中, 根据测试, 我们的算法能够自动处理超过 90% 的领域知识, 其余领域知识将被手动补充到领域知识库中.

(2) 交易操作序列领域知识. 交易操作序列领域知识主要包含证券交易操作执行顺序及状态变化, 我们采用状态机描述交易过程中不同事件触发的不同交易状态, 用以对不同操作之间的先后顺序进行约束. 针对不同业务的不同交易过程, 可以用不同的状态机描述. 图 5 就是一个针对债券交易的状态机, 其交易状态有 8 个, 从未委托开始, 经过多个操作进入终结状态. 操作可能会触发状态的迁移, 如申报、撤销申报等, 也可能不触发状态迁移, 比如撤销申报失败. 这些知识将会被用来识别软件需求间的依赖关系.

最终, 我们构建了一个实用的证券领域知识库, 其中包含了 266 条交易概念领域知识和 56 条交易操作序列领域知识. 这些知识为后续步骤的执行提供了必要的基础. 通过对领域知识的整合与应用, 我们能够更有效地分析和处理模型抽取得到的形式化软件需求, 生成更全面、更高质量的需求规约.

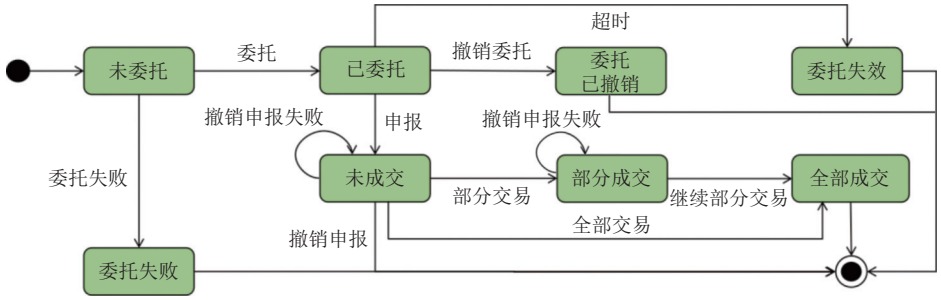


图 5 债券交易状态机

3 方法细节

3.1 需求信息抽取

对于需求信息抽取, 考虑到基于自然语言的业务规则具有高度的非结构性, 格式多样且语法复杂, 直接对原始规则进行信息抽取可能无法理解规则的准确语义, 导致抽取的准确度不高. 因此, 我们首先使用 GPT-4 对原始的业务规则进行预处理. 在此基础上进行信息抽取, 最后组装成形式化需求.

(1) 文本预处理. 这一步使用 GPT-4 将复杂的、表达不规范的业务规则进行简化和明确化. 提示词的内容主要包含两部分: 任务说明提示词和示例驱动提示词. 第 1 部分向 GPT-4 介绍要完成的任务, 即解析和简化证券交易的复杂业务规则, 并提出了一些具体的要求, 如针对指代、倒装、省略等情况的处理以及复杂情况分解成简单明确子规则的要求. 第 2 部分则提供了一个完整的示例, 用于展示期望的输出格式和处理方法, 帮助模型理解如何应用上述的任务说明, 并确保输出格式一致. 具体的预处理提示词及其设计如图 6 所示.

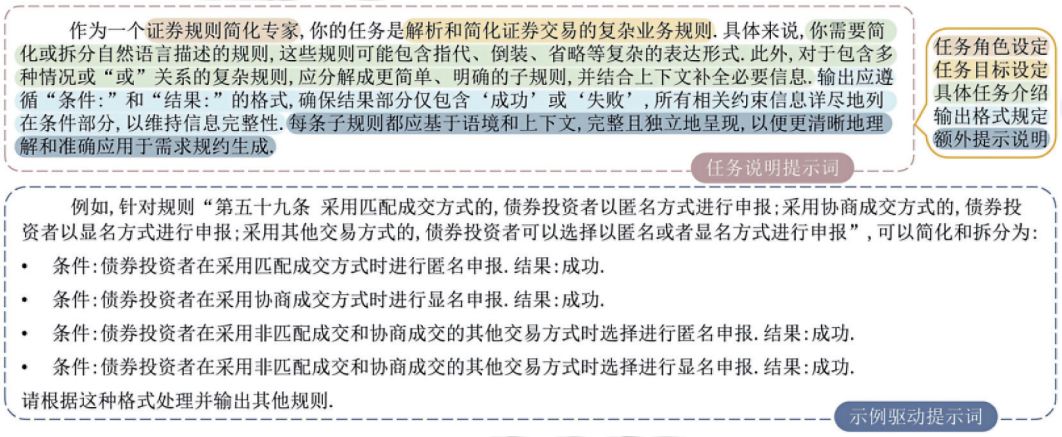


图 6 预处理提示词

预处理过程不仅需要关注文中明确表述的内容, 还需要能够理解和推断由省略所引起的隐含信息以及对复杂的情况进行拆分. 我们以案例 1 中的规则 3.3.4 为例, 说明这一步骤应该得到的结果. 针对规则 3.3.4, 预处理阶段会根据规则中涉及的交易品种和对应的申报数量要求, 将其拆分成 3 条子规则, 并分别识别每条子规则的条件和结果, 同时确保每个条件都直接关联到一个明确的结果 (成功或失败). 拆分简化后的规则如下所示.

规则ID	条件	结果
3.3.4.1	采用匹配成交方式的债券现券申报, 申报数量为10万元面额或其整数倍	成功
3.3.4.2	采用匹配成交方式, 卖出时债券现券申报数量不足10万元面额, 但一次性申报全部数量	成功
3.3.4.3	采用匹配成交方式的债券通用质押式回购申报, 申报数量为1 000元面额或其整数倍	成功

(2) 需求信息抽取. 这一步基于预处理的结果对规则进行需求信息抽取. 我们使用第 2.2 节中创建的需求信息抽取模型执行信息抽取任务. 首先, 我们将一些示例业务规则输入需求信息抽取模型, 并对它输出的结果给予反馈. 经过 5 轮迭代后, 我们将上一步的输出依次输入模型进行信息抽取. 例如, 在子规则 3.3.4.1 的条件部分, 需求信息抽取模型从输入的规则中抽取并标注了关键信息, 包括交易方式、交易品种、键和数量, 同时确保信息完全对应原文内容.

交易方式: 匹配成交	交易品种: 债券现券	键: 申报数量	数量: 1 000元面额或其整数倍

特别地, 需要注意的是, 在这里为了保证信息抽取的质量, 我们允许人与抽取模型进行必要的少量 (实验显示 5 次即可) 交互, 人的作用是核实和确认这一阶段的中间结果. 这可能比完全无交互的自动化方法效果更好. 这些任务不要求深厚的领域专业知识, 普通需求工程师即可完成.

(3) 形式化需求的组装. 在将信息抽取的结果转化为形式化需求表示的过程中, 必须深入考虑规则的结构特性, 确保组装算法充分适应多样化的语言表达习惯. 基于对业务规则表达方式和结构的分析, 我们提出了一套规则组装算法, 具体步骤如下.

1) 生成形式化需求子句. 该步骤的目标是将抽取模型输出的结果组装起来, 生成形式化子句. 一个形式化子句由实体、实体类型以及连接它们的谓词这 3 部分组成. 算法从抽取模型输出的结果中提取每一个实体类型-实体对, 并将其组装为“实体类型 is 实体”格式的形式化需求子句.

2) 组装子句. 在生成子句后, 算法将子句通过 and 连接构成形式化需求的条件部分, 即 if 语句, 并将第 3.1 节中预处理阶段识别的结果放入结果部分, 即 then 语句. 例如, 上述规则 3.3.4.1 的抽取结果组装得到的形式化需求如下所示.

rule 3.3.4.1

if 交易方式 is “匹配成交” and 交易品种 is “债券现券” and 申报数量 is “1 000 元面额或其整数倍”

then 结果 is “成功”

在这个示例中, 条件部分由 3 个子句组成, 每个子句都描述了一个具体的要求: 交易方式必须是“匹配成交”, 交易品种必须是“债券现券”, 申报数量必须是“1 000 元面额或其整数倍”. 子句之间通过逻辑“and”连接, 所有子句必须同时满足, 才能触发结果部分. 当条件满足时, 预期的结果是“成功”. 这种形式化的输出使得业务规则一目了然, 便于理解和执行.

3.2 需求可操作化

上一步生成的形式化需求是直接从业务规则文档中解析得到的, 它只包含规则的直接表述和明确约束, 但有时可能缺乏实施需求所必需的背景信息和上下文联系, 使得需求不够明确. 因此, 为了构建一个可操作的、具体的需求集, 我们需要引入交易概念领域知识和上下文信息以补全这些需求. 这一阶段的处理主要包含 4 个步骤, 分别为必要约束补全、抽象需求具体化、嵌套需求处理以及同属性项组合.

(1) 必要约束补全. 每条需求都有一系列必须要满足的约束条件, 比如该需求适用的交易市场和品种等. 如果缺少这些约束, 这条需求将无法被执行. 然而, 并非所有需求都完整包含这些约束, 因此, 我们需要对每条需求的必要约束进行补全. 单条需求需要补全的必要约束被定义在交易概念知识库中, 例如, 对于债券交易, 必要约束包括交易市场、交易品种、交易方式等. 为了补全需求中缺失的必要约束, 我们结合相应需求的具体上下文和领域知识获取对应的信息. 以案例 1 中的规则 4.4.4 为例, 原始需求并未包含交易市场、交易方式以及交易品种. 然而, 通过分析上下文, 我们得知该需求适用于深圳证券交易所的债券交易, 并且与竞买成交方式有关. 因此, 在该步骤中, 我们为这条需求补全相应约束. 补全后的需求如下, 其中, 突出显示部分是在此步骤中补充的约束.

rule 4.4.4.1

if 交易市场 is “深圳证券交易所” and 交易方式 is “竞买成交” and 交易品种 is “债券” and 时间 is “竞买日前” and 操作人 is “卖方” and 操作 is “修改” and 操作部分 is “竞买预约要素”

then 结果 is “成功”

(2) 抽象需求具体化. 证券业务文档中存在着许多抽象的表达, 需要将这些抽象需求具体化. 这一步骤主要依赖于领域知识, 将需求中的抽象表述替换成可操作的具体概念, 确保生成可执行的需求. 以上述需求 4.4.4.1 为例, “竞买预约要素”是一个抽象概念, 要使该条需求可操作, 需要将其替换成具体的竞买预约要素内容. 根据交易概念知识库中的相关知识“竞买预约要素应当包括以下内容: 竞买方式、证券代码、价格区间、数量、竞买时间等”, 具体化后需求如下. 其中, 突出显示部分是在此步骤中补充的约束.

---

rule 4.4.4.1

if 交易市场 is “深圳证券交易所” and 交易方式 is “竞买成交” and 交易品种 is “债券” and 时间 is “竞买日前” and 操作人 is “卖方” and 操作 is “修改” and 操作部分 is “竞买预约要素” and 竞买预约要素 is “竞买方式、证券代码、价格区间、数量、竞买时间等”

then 结果 is “成功”

---

(3) 嵌套需求处理. 需求中经常涉及一个需求直接引用或引述另一个需求的情况. 对于这样需求嵌套的情况, 我们将会把该需求及其引用的需求结合起来, 从而使单条需求中包含完整的约束. 例如, 对于规则 3.3.2“本所交易系统在本规则第 3.1.5 条规定的交易时间内, 接受相应的债券交易申报”, 该规则显式要求与规则 3.1.5 进行组合. 在处理这样的嵌套需求时, 会将需求 3.3.2 和 3.1.5 进行组合, 生成一条新的需求.

(4) 同属性项组合. 需求通常是独立的, 分别描述不同的交易场景, 但也可能出现多条需求都约束同一属性的情况. 对于这些需求, 我们会将其进行组合, 目的在于识别并整合约束相同属性的需求, 以确保某个字段的完整约束得以在一条需求中表达. 对于包含相同属性项的两条需求, 若它们描述不同的交易场景, 例如一条需求专门针对匹配成交, 另一条专门针对竞买成交, 这种情况下需求不应该组合; 相反, 如果两条需求描述相同的交易场景, 那么它们之间的各项约束不会发生冲突, 因此可以进行组合. 例如, 规则“债券交易的单笔最大申报数量不得超过 100 亿元面额”与规则“债券交易的申报数量应当为 10 万元面额或其整数倍”都包含同一交易场景下关于申报数量的约束, 两者之间不存在冲突, 因此这两条规则对于申报数量的约束同时有效, 应该将这些约束合并到一条需求中. 通过这种方式, 可以确保相关属性的约束在单一需求中得到完整且一致的表述.

3.3 需求关系识别

我们基于上一步需求可操作化的结果, 结合交易操作序列知识库进行需求间基于语义的和基于操作序列的隐式依赖关系识别, 并生成数据流形式的规约.

(1) 基于语义的隐式依赖关系识别. 一些业务需求在语义中隐含了与其他需求的依赖关系. 这种关系通常体现为前置与后置的逻辑关系, 一般使用“在…前”“在…后”“直到”等时间介词表达需求之间的顺序. 这些介词对于理解和执行需求至关重要, 因为它们指定了事件发生的相对时间顺序. 对于需求信息抽取模型抽取出的每个事件, 我们会从中提取这些时间介词, 并将其分为前置条件或后置条件分别进行处理.

前置条件是在执行需求之前必须满足的条件. 为了处理前置条件, 我们采用中文自然语言处理工具 HanLP[21] 进行词性标注, 提取事件中的操作、操作人、操作部分, 并将它们写为一条单独的需求, 补充为原需求的前置需求. 例如, 针对需求“报价方发出报价后, 未成交部分, 可以修改价格和数量等报价要素”, 我们在原始需求前添加一条前置需求, 其操作人为“报价方”, 操作为“发出”, 操作部分为“报价”, 结果为成功.

后置条件是在执行需求之后必须满足的条件. 为了处理后置条件, 我们采取添加反例需求的方法. 我们通过将原始需求的结果设置为失败构造了一个反例需求, 用以明确指出在不满足特定条件的情况下, 即在后置条件所规定的时间点之后执行的需求将不能成功. 例如, 对于需求“在应价方提交有效的应价申报前, 卖方申请并经本所认可后, 可以撤销竞买发起申报”, 我们会为其补充一条反例需求, 即“在应价方提交有效的应价申报之后, 卖方撤销竞买发起申报将会失败”.

(2) 基于操作序列的隐式依赖关系识别. 对于一些需求, 尽管隐式依赖关系在其具体描述中并没有直接体现, 但在实际的执行过程中, 这些需求必须遵循要特定的先后依赖关系. 例如, 在证券领域中, 可能会要求先存在一笔

已申报的订单, 然后才能对其进行撤销申报. 我们利用交易操作序列知识库帮助识别这样的隐式依赖关系. 交易操作序列知识库中交易状态的迁移通常由特定操作触发. 这些状态可能从原始需求中直接获得, 也可能结合状态机和操作推理得到. 这些状态迁移规定了操作之间存在的顺序关系. 为了识别隐式依赖关系, 我们将需求中的操作和状态与对应状态机中的事件和状态进行比较, 如果匹配, 那么它们的序列关系被视为相同. 另外, 我们也可以通过交易操作序列知识库中的状态机, 以推理出操作的前置后置状态, 从而有效确定这些需求的隐式依赖关系, 形成状态连贯的需求链.

例如, 考虑图 7 所示的场景, 其中两个需求分别与“申报”和“撤销申报”有关. 结合债券交易状态机 (图 5), 可以推断出要撤销申报, 必须先进行申报. 因此, 包含“申报”操作的需求 3.1.5.1 应该在包含“撤销申报”操作的需求 4.1.10.1 之前执行. 此外, 我们还能为相应需求补充操作执行前后的状态. 以需求 3.1.5.1 为例, 我们在 if 语句中补充前置状态“已委托”, 并在 then 语句中补充后置状态“未成交”.

图 7 基于状态机的隐式关系识别示例

(3) 规约生成. 最终, 我们基于形式化需求及其依赖关系生成数据流形式的规约. 这种规约以图形化的方式展示了需求的执行顺序和数据流动, 确保了业务流程的正确性和一致性. 以案例 1 中的规则 3.1.5、规则 3.3.4 以及规则 4.1.10 为例, 规则 3.1.5 和规则 3.3.4 都是与申报操作相关的规则, 而规则 4.1.10 则是与撤销申报操作相关的. 根据需求关系识别的结果, 需求 4.1.10 依赖于前两条需求的执行, 由此生成的数据流如图 8 所示, 该数据流从债券投资者申报一笔订单开始, 到这笔订单撤销失败终止.

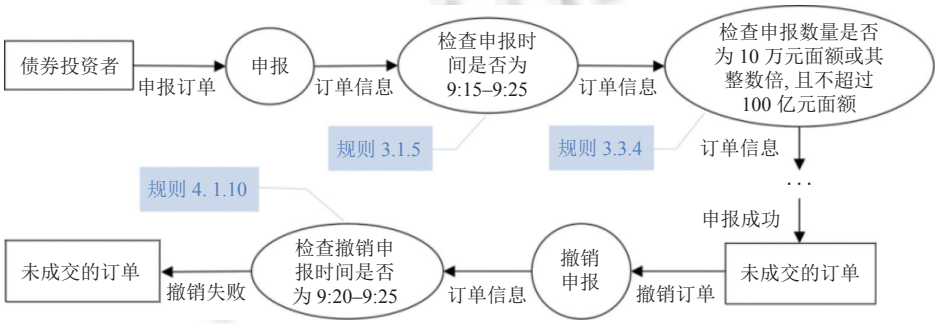


图 8 债券申报数据流

4 案例研究与评估

4.1 案例研究

本节以《深圳证券交易所债券交易规则文档》[1]为例, 应用本文所提出的方法对该文档进行了规则过滤、需求信息抽取、需求可操作化和关系识别等处理, 生成形式化的需求规约. 该文档包含 10 个章节, 详细介绍了深圳证券交易所债券交易的各项规定, 包括交易参与者的资格标准、不同类型的债券交易流程以及交易操作的基本原则等内容, 共计 164 条以自然语言描述的业务规则. 表 1 展示了该文档在每个步骤的输出统计信息, 包括规则表示形式、业务规则数及处理时间等. 下面, 我们对每一步的处理过程和结果进行详细说明.

表 1 深圳证券交易所债券交易规则文档处理统计概览

步骤	规则表示形式	业务规则数	处理时间 (s)	涉及的领域知识数	规则关系数	规约中涉及的业务路径数
业务规则过滤	自然语言	98	3.89	—	—	—
需求信息抽取	FBR形式	135	约3 000	—	—	—
需求可操作化	FBR形式	2 308	0.37	28	—	—
需求关系识别	FBR形式	2 562	5.46	12	326	2 562

(1) 业务规则过滤. 我们使用微调好的分类模型执行规则过滤任务. 首先, 文档被按句划分为 366 个独立的句子. 然后, 我们将这些句子逐个输入规则过滤模型进行分类. 分类结果显示, 文档包含 98 条软件需求相关规则、226 条软件需求无关规则以及 42 条领域知识. 抽取出的 42 条领域知识随后被算法自动整合到领域知识库中, 同时, 98 条软件需求相关的规则将会进入后续的需求信息抽取步骤. 在这个任务中, 规则过滤主要由过滤模型自动完成, 无需需求人员的参与, 处理案例文档中的 366 条数据仅耗时 3.89 s.

(2) 需求信息抽取. 我们首先使用 GPT-4 对筛选出的 98 条软件需求相关规则进行预处理. 预处理过程包括结合上下文对规则进行深入的语义理解, 对规则中的指代、省略等部分进行补全, 以及将描述多个场景的规则拆分为更小的、仅描述一个交易场景的子规则. 这些步骤能够使规则集合更加精细和具体, 有助于提高后续步骤抽取的效率和准确性. 经过这一系列的预处理步骤, 我们最终得到了一个包含 135 条子规则的集合, 每条子规则由条件和结果两部分组成.

基于预处理的结果, 我们使用需求信息抽取模型对每条子规则的条件部分按照预定义的 15 种实体类型进行需求信息抽取. 对于每条规则, 抽取模型提取其中与需求规约相关的实体, 并赋予其对应的实体类型, 形成若干结构化的“实体类型-实体”元素对. 这种结构化的数据去除了规则中冗余的部分, 保留了需求规约相关的要素, 提高了规则的可读性, 也使得规则更加容易被计算机处理和分析. 预处理和需求信息抽取步骤主要通过人工将规则输入给 GPT-4, 并获取其回答而完成. 对于案例文档的 98 条规则, 与 GPT-4 进行交互并完成整个过程所需时间大约为 3 000 s.

最后, 规则组装算法对抽取出的实体及其对应的实体类型进行组装, 最终生成了 135 条形式化的需求. 这些需求不仅展现出高度的形式化和标准化特点, 而且能够清晰地表达出每条需求的语义内容和逻辑关系. 尽管如此, 通过需求信息抽取得到的需求严格忠于原始文本, 由于需求内部缺乏必要的领域特定约束、需求之间缺乏系统性的联系等原因, 这些需求尚未形成一个可操作的整体.

(3) 需求可操作化. 我们基于交易概念领域知识对上一步得到的形式化需求进行需求可操作化. 由于每条需求都隐含着特定的上下文约束, 我们结合领域知识库对全部的 135 条形式化需求进行了必要约束补全, 以确保需求的完整性和准确性. 在此过程中, 我们将 6 条形式化需求中的抽象表述替换为具体约束. 这一步骤共引入了 28 条相关的领域知识, 涵盖了交易方式的分类、交易申报的要素等方面. 此外, 我们还处理了 3 条相互嵌套的需求以及 57 处同属性项组合. 这些被组合的属性项包括时间、数量、价格等与需求紧密相关的约束条件, 每条需求都针对交易场景的一个特定方面进行了描述, 通过组合形成了完整的约束网络. 在整个需求可操作化阶段, 我们结合领域知识对所有需求进行处理, 最终得到了 2 308 条经过需求可操作化后的需求. 这些需求包含了描述一个完整交易

场景所需的所有具体且必要的约束条件.

(4) 需求关系识别. 基于需求可操作化的结果, 我们进行了关系识别并基于需求生成了数据流. 我们首先使用算法检索需求中包括“事件”和“状态”等关于时间序列的实体类型, 识别相应需求的前置或后置条件, 从而将语义上隐式依赖的需求进行关联. 随后, 我们提取需求中的“操作”“操作部分”等实体类型, 并利用交易操作序列知识库中的债券交易状态机将操作序列上存在依赖关系的需求进行关联. 通过这些步骤, 案例文档最终抽取和识别出 2 562 条需求以及 326 对关系. 具体而言, 我们识别出了 14 对基于语义的隐式关系, 其中包括 10 个前置条件和 4 个后置条件. 另外, 我们利用债券交易状态机识别出了 312 对基于操作序列的隐式关系, 这些关系涉及状态机中的 12 个不同的状态迁移.

4.2 实验评估

为了评估所提出的方法的有效性, 我们研究了以下 4 个问题.

● RQ1: 与大语言模型、领域专家和非领域专家相比, 本文方法从不受控的自然语言业务规则文档自动规约出的软件需求的质量和效率如何?

● RQ2: 在业务规则过滤和需求信息抽取阶段, 上下文学习和微调策略对大模型性能的影响如何? 哪种策略更有效?

● RQ3: 在需求可操作化和关系识别阶段, 外部领域知识库的引入对需求规约质量的影响如何?

● RQ4: 本文方法在不同制度体系下的证券交易市场中的通用性如何?

在评估中, 我们使用功能点识别率评估生成需求规约的质量, 使用时间消耗评估生成规约的效率. 其中, 功能点是指从输入到交易系统的一个数据实体, 经过系统处理后输出执行结果的数据流中的一条约束. 功能点识别率 (*FPI*) 是指在一个数据集所有实际存在的功能点中, 人类参与者或工具生成的规约中正确的功能点所占的比例, 可表示为 $FPI = F_t/F_a$, 其中, F_t 是人类参与者或工具生成规约中正确的功能点的数目, F_a 是总功能数. 较高的 *FPI* 表示生成的规约质量较高, 包含更多正确的功能点.

基于提出的方法, 我们设计并实现了证券领域业务规则自动规约工具 LLSec. 我们将它部署在以下配置的服务器上: AMD Ryzen Threadripper PRO 5975WX CPU, 256 GB RAM, 一块 NVIDIA RTX 3090Ti GPU, 操作系统为 Ubuntu 22.04. 所有的实验都在该服务器上开展.

4.2.1 RQ1: 业务规则规约的质量和效率

● 数据集和对比对象. 由于缺乏相关公开的基准测试数据集, 需要建立专门的评估数据集. 在构建过程中, 我们考虑了上海和深圳证券交易所的常用与关键交易领域, 涉及债券、股票、基金和创业板等不同交易品种, 以及竞价交易、大宗交易和盘后定价交易等多种交易方式. 我们从上述交易领域中随机选择了具体业务, 并收集相关规则文档中关于这些业务的所有规则构成数据集. 最终我们得到了 5 个评估数据集, 分别为《深圳证券交易所创业板盘后定价交易业务规则》《上海证券交易所竞价交易申报业务规则》《深圳证券交易所股票大宗交易业务规则》《深圳证券交易所基金交易业务规则》和《上海证券交易所债券交易申报业务规则》. 这些数据集的特征如表 2 所示.

表 2 数据集特征

序号	文献	规则数	交易所	具体业务	交易品种	交易方式
1	[22]	10	深圳证券交易所	创业板盘后定价业务	股票	盘后定价
2	[2]	12	上海证券交易所	竞价交易业务	股票、基金、权证、存托凭证	竞价交易
3	[23]	12	深圳证券交易所	股票大宗交易业务	股票	大宗交易
4	[24]	12	深圳证券交易所	基金交易业务	基金	竞价交易、大宗交易
5	[25]	11	上海证券交易所	债券交易申报业务	债券	匹配成交、点击成交、询价成交、协商成交、竞买成交

为了评估我们所提出的方法的有效性, 我们为每个数据集编写了相应的需求规约, 并统计了每个数据集的规则数、需要补全的规则数、依赖关系数以及功能点数等特征信息. 在这个过程中, 我们与领域专家紧密合作. 首

先, 我们确定了要评估的业务和数据集. 随后, 在领域专家完成对比实验后, 我们共同对需求规约和依赖关系进行了修改和最终确认. 我们的目标是通过这种方法覆盖证券领域内多样化的业务场景, 以确保评估数据集和评估实验的代表性和有效性.

在对比对象的选择上, 鉴于目前证券领域业务规则的规约实际上完全依赖于人工进行, 不存在其他规约生成工具作为对比基准, 我们考虑将本文方法与人类参与者以及通用大型语言模型 GPT-4 和 GLM-4[26]的规约结果进行对比分析. 参与实验的人类参与者包括 3 位领域专家和 2 位非专家. 领域专家来自国泰君安证券股份有限公司的测试部门, 拥有平均 5–7 年的证券行业工作经验. 非专家则由平均具备 1–2 年工作经验但并非专门从事证券领域工作的研发人员组成. GPT-4 是目前最优秀的大语言模型之一, GLM-4 是清华大学研发的中文大语言模型, 曾登顶过中文大模型评估基准 c-eval 榜单. 我们使用大语言模型端到端地处理数据集中的业务规则, 从而生成规约. 参与实验的大语言模型的详细的版本说明如表 3 所示, 其中, GPT-4 和 GLM-4 将参与所有的 4 个实验, 而 Mengzi-BERT-base-fin[18]和 Llama 2[27]将在实验 2 中被用到.

表 3　参与实验的模型版本说明

模型名称	调用方式	模型架构	参数量	模型发布时间	模型使用时间
GPT-4[14]	网页版	仅解码器	约1.8万亿	2023-03-14	2024-05
GLM-4[26]	网页版	编码器-解码器	超过130B	2024-01-16	
Mengzi-BERT-base-fin[18]	本地部署	仅编码器	103M	2021-10-14	
Llama 2[27]	本地部署	仅解码器	7B	2023-07-19	

● 实验过程和结果. 我们将数据集中的规则输入到 LLSec 中, 经过一系列处理后, 得到对应的需求规约. 同时, 我们将数据集提供给领域专家和非专家, 要求他们为每个数据集生成需求规约, 并记录完成这一任务所需的时间. 对于大语言模型 GPT-4 和 GLM-4, 我们首先准备了一段引导性提示词. 这段提示包含了需求规约生成任务的具体指示, 阐明了规约所需包含的内容和格式以及需要具体化抽象表达等提示和要求. 随后, 我们使用 2 样本学习 (2-shot), 将 2 条规则样本输入模型得到对应的需求规约, 并分析并指出其中存在的问题, 以此纠正模型的输出. 最后, 我们将每个数据集的规则输入模型, 记录模型的输出. 在这个过程中, 我们不提供任何额外的引导或提示.

人类参与者和每个工具生成规约的功能点数量和识别率是由算法自动计算的. 一个规约由多个数据流组成, 每个数据流又包含从输入到输出的多个功能点. 对于功能点识别率, 算法首先针对每个数据流进行单独的功能点识别率计算. 对于每个数据集, 其整体的功能点识别率是通过计算数据集内所有数据流识别率的平均值来确定的. 据此得出的实验结果如表 4 所示, 其中, 领域专家和非专家的结果分别为对应所有参与者结果的平均值.

表 4　在 5 个数据集上与领域专家、非专家、通用大语言模型在生成规约的数据流数量、功能点识别率和时间消耗上的对比

数据集	数据集特征			领域专家			非专家			GPT-4			GLM-4			LLSec		
	#规则	#DF	#依赖关系	#DF	FPI (%)	时间 (min)	#DF	FPI (%)	时间 (min)	#DF	FPI (%)	时间 (min)	#DF	FPI (%)	时间 (min)	#DF	FPI (%)	时间 (min)
1	0	10	11	20	**89.11**	33	29	69.81	75	38	55.99	20	41	50.69	18	14	87.52	**4**
2	0	12	50	48	**93.92**	40	36	74.05	73	85	80.31	15	88	82.11	19	700	**93.92**	**5**
3	4	12	78	40	91.26	35	50	64.75	85	41	82.63	25	48	71.73	9	276	**93.27**	**5**
4	3	12	112	56	86.20	40	36	61.57	70	58	59.56	17	66	60.26	20	1 288	**90.95**	**6**
5	17	11	168	83	83.25	50	55	52.22	74	90	75.83	25	64	40.56	20	518	**94.20**	**6**

注: #DF (number of data flow) 为数据流数量

从表 4 中可以看出, 在各个数据集上, LLSec 生成的需求规约在功能点识别率方面与领域专家相媲美, 且优于非专家、GPT-4 和 GLM-4. 在数据集 2–5 中, LLSec 生成的规约的功能点识别率达到甚至超过了领域专家的水平, 尤其是在数据集 5 上, 识别率达到了 94.20%, 高出领域专家 10 个百分点. 在所有数据集中, 相较于非专家和两个

大语言模型, LLSec 生成的规约均展现了更高的识别率, 特别是在数据集 5 上, 比 GLM-4 高出 53 个百分点. 这表明我们的方法能够有效理解自然语言描述的业务规则, 并生成高质量的需求规约.

此外, 我们还观察到, 随着数据集 1–5 的功能点和依赖关系的增加, 领域专家和非专家在这些数据集上的功能点识别率整体呈现出下降趋势. 当数据集中的功能点和依赖关系较少时, 人类参与者能够较好地生成规约, 但在功能点和依赖关系较多的情况下, 他们容易忽略某些约束和依赖关系. 相比之下, 机器不受这些人类固有限制的影响. 例如, 在数据集 5 上, LLSec 和 GPT-4 均表现良好, 功能点识别率高于较为简单的数据集 4. 然而, 从表 4 中可以看出, 在各个数据集上, LLSec 生成的需求规约在功能点识别率方面与领域专家相媲美, 且优于非专家、GPT-4 和 GLM-4. 在数据集 2–5 中, LLSec 生成的规约的功能点识别率达到甚至超过了领域专家的水平, 尤其是在数据集 5 上, 识别率达到了 94.20%, 高出领域专家 10 个百分点. 在所有数据集中, 相较于非专家和两个大型语言模型, LLSec 生成的规约均展现了更高的识别率, 特别是在数据集 2 上, 比 GLM-4 高出 51 个百分点. 这表明我们的方法能够有效理解自然语言描述的业务规则, 并生成高质量的需求规约.

从表 4 中我们发现一个反例: 从特征上看, 数据集 1 最为简单, 拥有最少的功能点, 然而人类参与者和我们的方法在这个数据集上的功能点识别率都低于更复杂的数据集 2. 通过分析发现, 数据集 1 虽然功能点较少, 但它的表达难以理解, 含有较多抽象的语句和不常见的表达. 因此, 无论是人类参与者还是我们的方法, 都难以全面识别其中的功能点, 这说明除了复杂度外, 表述的清晰度也是影响规约生成质量的重要因素.

在效率上, LLSec 优于人类参与者和大语言模型. 对人类参与者而言, 非专家需要比领域专家更长的时间. 尽管大型语言模型在处理速度上胜过人类参与者, 但 LLSec 的表现更为出色, 其在单个数据集上的处理时间不超过 6 min, 相较于领域专家的效率提升了大约 10 倍. 我们的需求信息抽取步骤虽然同样需要与大型语言模型进行交互, 但所采用的需求信息抽取模型能够直接投入使用而无需进行少样本学习, 这大大减少了交互所需的时间. 此外, 我们的方法在需求可操作化和关系识别步骤不依赖于大语言模型, 因而执行速度较快, 在每个数据集上的平均运行时间不超过 10 s.

在功能点的数目上, LLSec 生成规约包含的功能点的数目远远高于其他对比对象. 经过对生成规约的分析, 我们发现这是因为我们的方法在需求可操作化步骤会依据领域知识补全规则. 由于领域知识库不仅包含当前数据集, 还有来自同文档其他规则甚至其他文档的知识, 这些知识也会被补充到规则中. 我们的方法生成的规约包含更加完整、更加丰富的功能点, 而这能够帮助需求工程师、开发工程师等使用者更好地理解业务与需求.

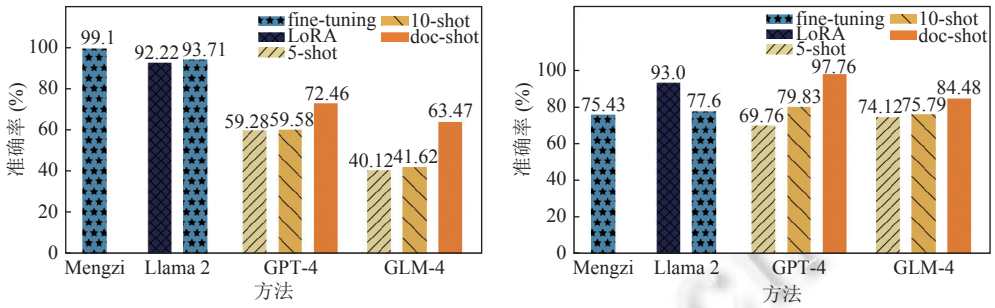
● RQ1 结论: LLSec 在生成需求规约方面表现出色, 在功能点识别率上与领域专家相媲美甚至优于专家, 并明显超过非专家和其他大语言模型, 且在效率上优于人类参与者和大语言模型.

4.2.2 RQ2: 上下文学习和微调策略的比较

● 数据集和对比对象. 通常, 将大语言模型应用于具体的下游任务有两种方法: (1) 使用提示词和示例进行上下文学习; (2) 使用数据集进行微调训练. 本实验的目的是将这两种方法在业务规则过滤和需求信息抽取两个任务上的效果进行比较. 在规则过滤任务中, 我们将构建的语料库按照 9:1 分为训练集和验证集, 使用训练集进行模型微调或上下文学习, 使用验证集评估模型的结果. 在需求信息抽取任务中, 由于规则往往嵌入在特定的上下文之中, 而不同的上下文会对规则的抽取结果产生影响, 我们从构建的语料库中随机选择一篇文档作为验证集, 剩余的数据作为训练集. 在本实验中, 我们选择的对比对象包括预训练分类模型 Mengzi (Mengzi-BERT-base-fin)[18]、预训练生成式模型 Llama 2[27]以及 GPT-4 和 GLM-4 两个对话模型, 详细的版本说明如表 3 所示.

● 实验步骤和结果. 对于规则过滤任务, 我们使用训练集的全部数据对 Mengzi 和 Llama 2 分别进行全参数微调 (full fine-tuning) 和低秩适配 (LoRA)[28], 并使用验证集对模型的结果进行评估. 全参数微调对预训练模型的全部参数进行训练调整, 而低秩适配则固定预训练模型的参数, 仅在每一层引入少量的新参数以进行训练优化. 对于 GPT-4 和 GLM-4, 我们按照第 2.1 节所述的分类标注准则指导模型执行规则过滤任务, 使用图 4 设计的提示词指导模型执行需求信息抽取任务. 我们从训练集中随机选择了 5 条、10 条和 1 篇文档 (366 条) 数据作为学习示例进行上下文学习, 然后使用相同的验证集进行评估. 受限于模型的上下文长度, 学习示例数不能无限增多. 在学

习示例为 1 篇文档时, 模型的上下文长度达到上限. 在需求信息抽取任务中, 我们使用同样的策略对模型进行微调和验证. 微调和上下文学习两种策略在业务规则过滤和需求信息抽取任务中的准确率对比如图 9 所示.



(a) 业务规则过滤任务 (b) 需求信息抽取任务

图 9 微调和上下文学习两种策略在业务规则过滤和需求信息抽取任务中的准确率对比

在业务规则过滤任务上, 从图 9(a) 中可以看出, 经过微调训练后的模型在验证集上的准确率都显著优于基于上下文学习的模型. 微调后的模型准确率均超过 90%, 其中 Mengzi 的准确率最高, 达到了 99.1%. 这表明微调策略能够有效促进模型对输入数据的理解, 从而输出精确的分类结果. 相较之下, 基于上下文学习的 GPT-4 和 GLM-4 在此任务上的表现较差. 当学习示例数较少的时候, 两个模型准确率较低, 而学习示例数的增加能够提高模型准确率. GPT-4 在学习示例为 1 篇文档时的准确率最高, 为 72.46%. 通过分析 GPT-4 和 GLM-4 模型的分类结果, 我们注意到超过 80% 的错误是由于模型错误地将与软件需求无关的规则分类为软件需求相关规则. 这一观察结果表明尽管我们在提示词中强调了分类标准并提供了丰富的示例, 基于上下文学习的模型在很大程度上仍然坚持其自身的偏见和标准, 而没有深入洞察到我们具体任务的核心要求.

在需求信息抽取任务上, 从图 9(b) 可以看出, 微调训练后的模型和基于上下文学习的模型均表现出了卓越的性能. 在微调模型方面, Llama 2 在低秩适配训练模式下取得了更高的准确率, 达到了 93%, 高于 Mengzi 模型的 75.43%. 这一结果可能是由于 Llama 2 具有更多的参数, 从而具备更强的学习能力, 能够更有效地适应复杂的需求信息抽取任务. 此外, 基于上下文学习的模型准确率也都超过了 70%, 其中, 使用 1 篇文档作为示例学习的 GPT-4 甚至达到了 97.76%, 优于微调后的 Llama 2 模型. 我们分析认为, 这种结果的原因在于 GPT-4 等大语言模型具有丰富的内部知识库, 且在信息抽取等任务上进行过预训练, 而这些任务与当前的需求信息抽取任务高度相似. 此外, 经过微调的 Llama 2 模型效果不如 GPT-4 的原因可能与模型的参数数量有关: 我们实验中微调的 Llama 2 模型拥有 70 亿参数, 而 GPT-4 拥有超过 1.8 万亿的参数, 这为 GPT-4 在处理复杂任务时提供了更高的计算能力.

● RQ2 结论: 微调和上下文学习两种适应策略均可用于规则过滤和需求信息抽取任务, 可能在不同应用场景下表现不同. 在选择适应策略时, 需要根据具体任务和数据情况进行调整.

4.2.3 RQ3: 外部知识库对需求规约质量的影响

● 数据集和对比对象. 在实验 1 中我们发现, 不同的文档具有不同的表达方式, 理解的难易程度也不同. 为了探究依赖关系的识别对生成高质量规约的影响, 评估数据集需要控制变量, 排除理解难度不同对实验结果的影响. 基于这种考虑, 我们从同一篇业务文档中抽取规则构建数据集, 这些数据集是独立同分布的. 由于《上海证券交易所债券交易规则》[25]涉及丰富的业务功能, 并且不同需求间存在一定的依赖关系, 因此我们选择它作为数据来源. 基于这篇交易规则文档, 我们构建了 4 个评估数据集 (数据集 6–9), 每个数据集包含的规则数、功能数、关系数等特征在表 5 中展示. 数据集 6 和 7 不包含依赖关系, 而数据集 8 和 9 包含依赖关系. 数据集 7–9 具有相似的规则数, 但随着依赖关系数逐渐增加, 数据流的数目逐渐增多. 我们的目的是评估在不同的功能数和关系数的情况下, 领域知识对生成规约质量的影响. 在本实验中, 我们同样选择与实验 1 相同的领域专家、非专家、GPT-4 和 GLM-4 作为我们的对比对象.

● 实验步骤和结果. 与实验 1 类似, 在本实验中我们将每个数据集提供给领域专家和非专家, 并输入到 GPT-4、

GLM-4 以及 LLSec 中, 得到相应的需求规约. 此外, 我们设置了一个消融实验, 将我们的方法中的领域知识库置空, 目的是探索知识对生成需求规约的功能点识别率的影响. 在任务设置上, 我们额外提示实验的每位参与者与大语言模型注意识别需求间的依赖关系, 并使用实体类型标签“状态”来标记具有依赖关系的数据流. 需求规约生成后, 我们使用实验 1 中的算法来计算数据流数目和功能点识别率. 实验结果如表 5 所示.

表 5　在 4 个具有不同功能数和依赖关系数的独立同分布的数据集上与领域专家、非专家和通用大语言模型在生成规约的数据流数量和功能点识别率对比

数据集	数据集特征			领域专家		非专家		GPT-4		GLM-4		LLSec (无领域知识库)		LLSec	
	#规则	#DF	#依赖关系	#DF	FPI (%)	#DF	FPI (%)	#DF	FPI (%)	#DF	FPI (%)	#DF	FPI (%)	#DF	FPI (%)
6	5	24	0	47	**95.83**	16	75.00	48	79.17	25	78.33	16	70.00	198	90.00
7	10	42	0	73	**92.24**	47	69.99	107	77.46	50	73.41	36	71.02	358	90.60
8	10	86	8	63	88.09	51	58.12	87	67.71	44	64.19	30	58.72	333	**94.31**
9	11	186	42	72	78.28	76	51.79	81	51.28	52	55.15	37	44.49	524	**94.29**

从表 5 中我们发现, 领域知识对生成规约的质量具有显著的影响. 随着数据集包含的数据流数和关系数的增加, 领域专家、非专家、GPT-4 和 GLM-4 在数据集上生成的规约的功能点识别率均呈现出下降趋势, 而 LLSec 则相对稳定, 甚至在功能和关系数增多时不降反升. 通过对实验数据的分析, 我们发现领域专家具有丰富的领域知识, 能够补充和关联大部分的需求, 但当规则复杂度提升时容易产生遗漏和错误, 导致功能点识别率逐渐下降. 对于非专家、GPT-4 和 GLM-4, 我们发现它们虽然具备一定程度的知识, 但知识水平有限, 因而在部分需求的理解和补充上表现不佳, 功能点识别率相对较低. 与此相反, 我们的 LLSec 方法通过需求可操作化和关系识别步骤有效利用了证券领域的交易概念和交易操作序列等专业知识, 生成数据流的功能点更加具体和全面. 特别地, 当我们实验中将 LLSec 的外部领域知识库置空后, 其生成的需求规约的功能点识别率产生了明显下降, 甚至低于人类测试者和一般通用大模型. 这一结果进一步强调了领域知识在生成高质量需求规约中的关键作用, 从而证实了外部知识库的应用对于提升需求规约质量的重要性.

● RQ3 结论: 外部知识库对提高需求规约的质量影响显著. LLSec 有效利用了外部知识库, 因而在处理功能数和关系数各异的规则时, 展现出了较高的和稳定性的功能点识别率.

4.2.4　RQ4: 方法的通用性

● 数据集和对比对象. 为了全面探讨和验证本文方法在不同制度体系下证券交易市场中的通用性, 我们收集了纽约证券交易所、东京证券交易所和香港证券交易所的相关规则文档, 从中挑选股票、债券等 5 个具有代表性的业务领域, 并筛选出涉及这些业务的所有规则, 构成了 5 个评估数据集 (数据集 10–14). 其中, 数据集 10 为纽约证券交易所股票交易规则[29], 数据集 11 为纽约证券交易所交易和结算规则[29], 数据集 12 为东京证券交易所股票经营规定[30], 数据集 13 为东京证券交易所债券经营规定[30], 数据集 14 为香港证券交易所交易机制[31], 每个数据集的规则数等特征展示在表 6 中. 这些来自不同交易所的业务规则分别以英文、日文和繁体中文编写, 在内容和表达方式上存在较大的差异. 鉴于领域专家和非专家对纽约、东京等交易所的交易规则不够熟悉, 存在认知偏差和较高的时间成本问题, 这里我们仅将我们的方法与 GPT-4 和 GLM-4 两种大语言模型进行比较.

表 6　在 5 种不同制度体系下的证券领域业务规则数据集上生成规约的数据流数量和功能点识别率以及与 GPT-4 和 GLM-4 模型的比较

数据集	数据集特征			GPT-4		GLM-4		LLSec	
	#规则	#DF	#依赖关系	#DF	FPI (%)	#DF	FPI (%)	#DF	FPI (%)
10	10	44	7	48	73.15	56	68.93	579	**82.95**
11	9	28	4	30	72.12	76	56.5	2497	**76.68**
12	12	30	6	27	55.04	62	51.79	368	**77.67**
13	9	22	5	20	56.49	36	52.91	537	**81.6**
14	12	38	5	25	60.24	52	69.2	3062	**79.84**

● 实验步骤和结果. 由于我们的方法仅适用于简体中文文本, 在处理英文、日文及繁体中文表示的业务规则前, 我们首先借助谷歌翻译将这些文本转换为简体中文. 转换完成后, 我们逐一对照原文, 对翻译中存在的误差和不流畅之处进行了手动修正, 确保了业务规则表述的通顺和准确性. 接着, 我们将这些简体中文的业务规则输入到我们的工具中, 生成相应的需求规约. 对于 GPT-4 和 GLM-4 模型, 考虑到它们本身具备处理英文、日文和繁体中文的能力, 为了确保比较的公平性, 我们直接将原始规则文本输入到这些大模型中, 得到了对应的需求规约. 然后, 我们将这些需求规约翻译为简体中文, 以便计算其功能点识别率. 最后, 我们使用实验 1 中提到的算法来自动计算我们的方法以及 GPT-4 和 GLM-4 生成规约的数据流数目和功能点识别率, 结果如表 6 所示.

从表 6 中我们可以发现, LLSec 在 5 个数据集上生成需求规约的功能点识别率最高, 优于 GPT-4 和 GLM-4, 并且展现出了较高的稳定性. 具体来看, GPT-4 在处理纽约交易所的业务规则 (数据集 10 和 11) 时, 功能点识别率达到了 70% 以上, 但在面对东京交易所的规则 (数据集 12 和 13) 时, 识别率下降, 甚至不足 60%. GLM-4 的表现也显示出了一定的波动性. 相比之下, 我们的方法在不同交易所的数据集上不仅达到了 80% 左右的功能点识别率, 而且数据集之间的识别率差异不超过 7%. 这显示了我们的方法在不同制度体系下的准确性和稳定性.

此外, 相较于实验 1 的结果 (表 4), 我们的方法在处理不同制度体系下的业务规则并生成需求规约的过程中功能点识别率的下降幅度相对较小. 我们的方法最初是基于深圳和上海证券交易所的业务规则进行精心设计和完善的. 在这两个交易所的相关规则文档评估中, 我们实现了高达 90% 的功能点识别率. 尽管如此, 当我们的方法应用于纽约、东京等具有不同制度体系的业务规则时, 仍然展现出了强劲的处理能力. 在这些情况下, 功能点识别率的平均降幅大约为 10%, 虽然略低于领域专家的水平, 但仍然优于非专家、GPT-4 和 GLM-4 的表现. 这一结果进一步表明了我们的方法具有较强的适应性和通用性, 即便在面对多样化的制度和文化背景时, 该方法仍能保持高性能的表现, 这对于跨文化、跨制度环境下的业务规则处理具有重要的实际意义和应用价值.

尽管如此, 我们的方法在处理不同制度体系下的业务规则时功能点识别率仍略有下降. 通过对工具生成需求规约的分析, 我们认为原因主要来源于训练数据和领域知识的不足. 不同制度体系下的业务规则具有不同的规则表达方式、专业术语和领域知识. 为了提高本文方法在不同制度体系下的功能点识别率, 我们考虑使用多语言和多文化的证券领域交易规则数据集对筛选模型和需求信息抽取模型进行上下文学习或微调训练, 以提高模型在处理不同语言表述的业务规则时的泛化能力. 此外, 我们还考虑融入专业知识, 将新的领域知识加入训练集和领域知识库, 以方便模型和算法更好地理解和处理专业术语和规则.

● RQ4 结论: 尽管证券领域不同制度体系下的业务规则存在差异, LLSec 在解析业务规则文档生成需求规约的任务中实现了 80% 左右的功能点识别率. 这一结果也表明了本文方法的通用性.

5 有效性威胁

在本节中, 我们将从内部效度、外部效度、构建效度和可靠性这 4 个方面对有效性威胁进行分析和讨论.

(1) 内部效度. 1) 大语言模型的局限性. 本文方法利用大语言模型强大的自然语言处理能力, 对自然语言规则进行了有效的过滤和抽取. 对于业务规则过滤任务, 我们通过对模型进行微调可以显著提高其分类的质量; 而对于需求信息抽取任务, 采用上下文学习方法则更为合适. 然而, 这一结论不一定适用于其他领域. 此外, 由于模型自身的局限性, 比如上下文窗口大小的限制和对隐含信息的理解不足等, 可能无法保证抽取结果完全无误. 为了解决这一问题, 我们在模型的输出后实施了人工审核流程, 以确认和验证抽取的信息, 从而增强了抽取结果的可靠性. 2) 领域知识的完备性. 从业务规则文档生成需求规约涉及各种领域知识, 我们通过不同方式整合和利用这些知识, 包括通过训练数据或提示词的方式将隐式知识自然融入大语言模型以及构建外部知识库, 从而增强模型针对特定领域的自然语言处理能力, 并结合外部领域知识库生成高质量的需求规约. 在我们的方法中, 业务规则过滤和需求信息抽取主要通过知识适配后的大语言模型完成, 而需求可操作化和关系识别则极大地依赖于外部知识库的充分性. 领域知识库的不完备可能导致需求可操作化不充分或关系识别不完整, 进而影响生成需求规约的质量. 为应对这一挑战, 我们需要不断地完善和更新领域知识库, 以确保覆盖更加全面的业务逻辑.

(2) 外部效度. 我们的研究数据集选自上海、香港、东京、纽约等不同制度体系下证券交易所的真实业务规则文档, 覆盖多种业务场景, 具有较好的代表性和普适性. 尽管如此, 鉴于证券领域业务规则的多样性和风格差异, 所选数据集不一定能够完全代表现实世界中的所有业务规则场景. 未来研究将探索本方法在更广泛数据集上的应用效果, 以进一步验证其广泛适用性. 另外, 虽然我们的方法主要针对证券领域, 但整体框架展现出较强的通用性. 结合大语言模型自身的能力和对领域知识策略性的整合和运用, 我们的方法在各种专业知识密集型领域表现出可迁移性. 关键调整包括: 1) 对大语言模型进行微调或上下文学习, 以适应对应的领域; 2) 构建相应领域特定的知识库. 由于不同领域之间存在显著差异, 上述工作是不可避免的. 尽管这些工作耗时费力, 但训练好的大语言模型和领域知识库具有可重用性, 即能够为同一领域提供持续支持, 可以跨应用和跨项目地被重复利用, 从而显著减轻所需的人力工作负担. 这种积累和共享的知识资源不仅可提高工作效率, 也将推动整个领域内技术的进步.

(3) 构建效度. 对于需求规约生成质量的评估, 目前许多研究依然采用定性分析方法, 依靠与用户和其他利益相关者的交互, 通过访谈和反馈来验证需求文档是否准确反映了业务需求和用户期望[32]. 由于评估需求规约的质量涉及对需求内容的解释及其与用户期望之间的匹配度, 在不同领域中可能存在显著差异. 在本文中, 功能点的准确识别是衡量需求规约质量的基础, 因此, 本文通过功能点识别率 (FPI) 进行评估. 较高的 FPI 表明 LLSec 能够全面捕捉业务规则中的关键信息, 生成的需求规约更为全面. 功能点识别率是生成质量的一个重要指标, 但也可能存在其他评估指标, 如规约的可读性、可维护性和可测试性.

(4) 可靠性. 由于大语言模型的输出具有随机性, 可能在不同的运行时表现出不同的行为, 特别是在解析非结构化和复杂自然语言数据时. 此外, 模型的更新和优化可能会导致结果的变动, 从而影响实验的可靠性. 然而, 经过 3 次以上的重复实验验证, 我们发现使用上下文学习和微调策略两种策略生成的需求规约总是保持较高的、相近的准确率. 这显示了即便在面临模型内在不确定性的情况下, 我们的处理流程依然稳定有效. 此外, 我们已经将我们的代码、模型和数据开源 (https://github.com/LingleLi/LLSec), 以支持研究结果的验证和复现.

6 相关工作

本文探索了证券领域的业务规则规约方法, 研究了如何将领域知识嵌入到规约过程中. 相关的工作主要包括领域知识赋能的需求规约以及业务规则规约.

6.1 领域知识赋能的需求规约

需求工程领域早期的工作主要采用基于规则的方法从政策文档或法规文本中提取软件需求. Young 等人[33]提出了一种以关键要素和模板为核心的方法, 用于从政策文档中提取软件需求. 该方法首先对政策文档中的承诺、特权和权利等要素进行分析, 以识别特定的政策陈述, 然后按照范围、行为者等维度将这些陈述分为 12 个类别, 并使用模板将分类后的政策陈述转换为软件需求. Breaux 等人[34]提出了一种从法规文本中提取数据访问需求的方法, 通过定义 6 种数据访问约束, 提取文本中的主体、动作、对象等内容, 并创建相应的规则表来记录各规则的约束. 这些提取方法虽然在一定程度上取得了成效, 但它们面临着两大主要问题. 首先, 它们完全依赖于预定义的规则或模式, 这限制了它们的通用性. 其次, 这些方法仅仅通过提取的内容明确提取的知识, 没有更多的领域知识的嵌入.

也有些研究者们以本体显式的表示领域知识, 用于软件需求的规约. 例如 Chen 等人[6,7]提出了一种结合本体和问题框架的需求建模方法, 将本体知识嵌入到问题框架的规范中, 从而引导需求分析员逐步抽取和描述软件开发问题. Bencharqui 等人[8]同样设计了一种基于本体的需求规约方法, 通过本体构建领域知识模型, 并将其嵌入到规约过程中从而提高规约的一致性和准确性. Abdalazeim 等人[9]回顾了从各种设计模型生成自然语言需求规约的方法, 包括将 UML 模型、XML 和 OntoUML 转换为自然语言规范的工作以及利用 OWL 和 RDF 本体生成自然语言文本的工作, 其中涵盖了将本体、领域知识等嵌入规约过程的应用, 能够生产更精确的需求模型和更高质量的需求规约. 然而, 这些工作需要持续地更新和维护本体和规则库.

自 ChatGPT 等大语言模型问世以来, 使用大语言模型进行需求规约成为一个新的研究热点. Jain 等人[35]提出了一种基于自然语言生成模型 T5 的方法, 用于从软件工程合同中自动识别安全性和隐私性要求. 他们将来自多个

领域的 60 余个软件工程合同数据集中的安全性和隐私性要求以微调的方式嵌入到大语言模型中, 作为额外的知识提高大语言模型的准确性. Arora 等人[36]评估了大语言模型在需求工程领域的需求获取、分析、规约和验证阶段的应用潜力. 目前, 学术界已经有工作研究将大语言模型与知识图谱等语义网络技术集成, 以增强其在需求抽取和形式化表达中的生成效能和语义精确度[11−13]. 与这些工作相比, 本文方法是首次结合大语言模型与领域知识, 应用在软件的功能需求规约上.

6.2 业务规则规约

目前, 并不存在完全自动化的业务规则规约方法, 但在业务规则的分类、需求信息抽取以及关系识别都已经有一些工作. 现在已经有很多基于机器学习的业务规则分类工作. 比如, Lafi 等人[37]提出了一种对不同类型的业务规则进行分类的方法, 将业务规则分类为定义性规则、数据规则、活动规则和参与方规则. 他们使用一种包含多个阶段的训练数据的管道方法来训练机器学习模型, 从而将业务规则自动分类, 进而增强软件需求的明确性和组织性. Herbst 等人[38]也对规则进行了分类, 他们针对数据库系统定义了一致性规则以及对有效数据库状态的完整性约束. 但是这些工作并没有对是否与软件需求相关进行分类.

目前, 大部分业务需求信息抽取工作仍然依赖人工完成, 许多研究采用手动方式将业务规则表示为形式化的逻辑表达式, 如对象约束语言 (object constraint language, OCL)[39]、自定义的业务规则建模语言[40]、语义业务词汇与规则 (semantic business vocabulary and rules, SBVR)[41,42]、礼貌逻辑程序 (courteous logic program, CLP)[43]、基于 XML 的业务规则标记语言 (business rules markup language, BRML) 等, 以清晰地描述业务规则以及这些规则之间的优先级和冲突关系. 与此同时, 也存在一些自动抽取业务规则的工作. 这些工作主要从自然语言规则文档中抽取相关信息, 将业务规则表示为某种表达形式. 例如, Bajwa 等人[44]提出了一种基于规则的算法对英语自然语言文档进行语义分析, 将自然语言规范翻译成 SBVR 业务规则. Chittimalli 等人[45]提出了一种无监督的启发式方法, 基于词性标注和句法分析进行实体抽取, 并将它们持久化到 SBVR 元模型中. Holter 等人[46]使用序列标注和少量学习识别需求语句中语义成分, 包括范围、条件和需求. Zhang 等人[47]提出了使用微调的 BERT 模型来提取文档中关键信息的方法, 他们的研究表明 BERT 只需要少量的训练数据 (少于 100 篇文档) 就能达到合理的精度. 这些规则和训练数据都是一种领域知识的嵌入.

此外, 有研究关注从抽象规则到具体规则的转换. 例如, Kardasis 等人[48]定义了抽象业务规则, 包括意图型规则和可操作化规则, 并指出要基于信息系统架构分析将高层的意图转换为具体需求和规约. 该方法支持业务规则的抽象到具体可实现的业务规则的精化关系, 信息系统架构就是其领域知识, 但只适用于一般的信息系统.

也有一些研究关注业务规则关系发现和识别. 例如, Bhattacharyya 等人[49]基于启发式规则, 使用最大熵分类器提取规则意图之间的成对关系, 将单个规则拆解为多个规则意图, 从而识别规则意图之间关系. 这种方法本质上探索的是业务规则内的关系. Schlutter 等人[50]提出一种从自然语言需求中提取知识并将其转化为语义关系图的方法. 输出的语义关系图可以用来进一步分析和理解需求, 帮助软件开发人员和项目管理人员更好地理解和组织需求信息. Mahgoub 等人[51]描述了一种从文本文档集合中自动提取关联规则的文本挖掘技术. 这种技术依赖于关键词特征来发现标记文档的关键词之间的关联规则, 从大量文本数据中提取有意义的模式和关系. 上面的关系识别工作主要集中在规则内部概念或关键词之间的显式关联上. 相比之下, 证券领域业务需求间的关系识别更加关注规则之间的隐式依赖关系, 特别是它们之间存在的一些基于操作序列的隐式关系, 不能通过文本特征去识别. 因此, 我们的方法强调结合领域知识与大语言模型结合.

7 总 结

本文提出了一种结合大语言模型和知识库的证券领域业务规则自动规约方法, 从不受控的自然语言业务规则文档中自动生成需求规约. 这种方法不仅减轻了需求工程师的工作负担, 也显著提高了业务规则规约的效率. 实验结果在 5 个不同的证券交易领域中证明了本文方法的有效性, 本文的工具在评估数据集上的平均功能点识别率为 91.97%, 达到甚至超越了领域专家的水平, 且平均效率提高了 10 倍. 本文的方法和工具为证券领域业务规则的

自动化规约提供了一个有效途径, 也为其他领域知识密集系统的知识嵌入提供了有价值的借鉴.

在未来的工作中, 我们计划不断提高抽取模型的准确率, 丰富领域知识库, 由此进一步提高自动化生成的需求规约质量, 能更快地应对业务规则的变化. 特别地, 由于证券行业的要求, 未来也考虑建立领域化的大语言模型, 以便部署到证券公司中, 真正投入使用. 同时, 我们考虑将该方法拓展到证券领域以外的其他行业中去.

References:

[1] Shenzhen Stock Exchange. Shenzhen Stock Exchange Bond trading rules. 2022 (in Chinese). https://docs.static.szse.cn/www/lawrules/rule/bond/bonds/trade/W020220127631859244722.pdf

[2] Shanghai Stock Exchange. Trading rules of Shanghai Stock Exchange (Revised in 2023). 2023 (in Chinese). http://www.sse.com.cn/lawandrules/sselawsrules/trade/universal/c/10118241/files/e773c99a2fef4fbc930bc332492156d8.docx

[3] Sleimi A, Sannier N, Sabetzadeh M, Briand L, Ceci M, Dann J. An automated framework for the extraction of semantic legal metadata from legal texts. Empirical Software Engineering, 2021, 26(3): 43. [doi: 10.1007/s10664-020-09933-5]

[4] Sannier N, Adedjouma M, Sabetzadeh M, Briand L. An automated framework for detection and resolution of cross references in legal texts. Requirements Engineering, 2017, 22(2): 215–237. [doi: 10.1007/s00766-015-0241-3]

[5] Bhuiyan H, Governatori G, Bond A, Rakotonirainy A. Traffic rules compliance checking of automated vehicle maneuvers. Artificial Intelligence and Law, 2024, 32(1): 1–56. [doi: 10.1007/s10506-022-09340-9]

[6] Chen XH, Yin B, Jin Z. Ontology-guided requirements modeling based on problem frames approach. Ruan Jian Xue Bao/Journal of Software, 2011, 22(2): 177–194 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3755.htm [doi: 10.3724/SP.J.1001.2011.03755]

[7] Chen XH, Jin Z. An ontology-guided process for developing problem frame specification: An example. In: Proc. of the 3rd Int'l Workshop on Applications and Advances of Problem Frames. Leipzig: ACM, 2008. 36–39. [doi: 10.1145/1370811.1370818]

[8] Bencharqui H, Haidrar S, Anwar A. Ontology-based requirements specification process. In: Proc. of the 10th Int'l Conf. on Innovation, Modern Applied Science & Environmental Studies. 2022. 01045. [doi: 10.1051/e3sconf/202235101045]

[9] Abdalazeim A, Meziane F. A review of the generation of requirements specification in natural language using objects UML models and domain ontology. Procedia Computer Science, 2021, 189: 328–334. [doi: 10.1016/j.procs.2021.05.102]

[10] Zhang ZH, Fang M, Chen L, Namazi-Red MR, Wang J. How do large language models capture the ever-changing world knowledge? A review of recent advances. In: Proc. of the 2023 Conf. on Empirical Methods in Natural Language Processing. Singapore: ACL, 2023. 8289–8311. [doi: 10.18653/v1/2023.emnlp-main.516]

[11] Kau A, He XZ, Nambissan A, Astudillo A, Yin H, Aryani A. Combining knowledge graphs and large language models. arXiv:2407.06564, 2024.

[12] Lu RS, Lin CC, Tsao HY. Empowering large language models to leverage domain-specific knowledge in E-learning. Applied Sciences, 2024, 14(12): 5264. [doi: 10.3390/app14125264]

[13] Pan SR, Luo LH, Wang YF, Chen C, Wang JP, Wu XD. Unifying large language models and knowledge graphs: A roadmap. IEEE Trans. on Knowledge and Data Engineering, 2024, 36(7): 3580–3599. [doi: 10.1109/TKDE.2024.3352100]

[14] OpenAI. ChatGPT. 2024. https://chatgpt.com

[15] Knauf R, Spreeuwenberg S, Gerrits R, Jendreck M. A step out of the ivory tower: Experiences with adapting a test case generation idea to business rules. In: Proc. of the 17th Int'l Florida Artificial Intelligence Research Society Conf. Miami Beach: AAAI, 2004. 343–348.

[16] Li XZ, Chan S, Zhu XD, Pei YL, Ma ZQ, Liu XM, Shah S. Are ChatGPT and GPT-4 general-purpose solvers for financial text analytics? A study on several typical tasks. In: Proc. of the 2023 Conf. on Empirical Methods in Natural Language Processing: Industry Track. Singapore: ACL, 2023. 408–422. [doi: 10.18653/v1/2023.emnlp-industry.39]

[17] Wei J, Zou K. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In: Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int'l Joint Conf. on Natural Language Processing. Hong Kong: ACL, 2019. 6382–6388. [doi: 10.18653/v1/D19-1670]

[18] Zhang ZS, Zhang HQ, Chen KM, Guo YH, Hua JY, Wang YL, Zhou M. Mengzi: Towards lightweight yet ingenious pre-trained models for Chinese. arXiv:2110.06696, 2021.

[19] Loshchilov I, Hutter F. Decoupled weight decay regularization. In: Proc. of the 7th Int'l Conf. on Learning Representations. 2019.

[20] Liu L, Özsu MT. Encyclopedia of Database Systems. 2nd ed., New York: Springer, 2018. 2245–2248. [doi: 10.1007/978-1-4614-8265-9]

[21] He H, Choi JD. The stem cell hypothesis: Dilemma behind multi-task learning with transformer encoders. In: Proc. of the 2021 Conf. on Empirical Methods in Natural Language Processing. Punta Cana: ACL, 2021. 5555–5577. [doi: 10.18653/v1/2021.emnlp-main.451]

[22] Shenzhen Stock Exchange. Special provisions for trading on the ChiNext Board of Shenzhen Stock Exchange. 2023 (in Chinese). https://www.szse.cn/lawrules/rule/repeal/rules/P020231230545310237980.pdf

[23] Shenzhen Stock Exchange. Trading rules of Shenzhen Stock Exchange (Revised in 2023). 2023 (in Chinese). https://docs.static.szse.cn/www/lawrules/rule/stock/trade/W020230217564423808793.pdf

[24] Shenzhen Stock Exchange. Detailed rules for the trading, subscription, and redemption of securities investment funds at Shenzhen Stock Exchange (Revised in 2022). 2022 (in Chinese). https://docs.static.szse.cn/www/lawrules/rule/fund/trade/W020220610536157903323.pdf

[25] Shanghai Stock Exchange. Shanghai stock exchange Bond trading rules. 2022 (in Chinese). https://bond.sse.com.cn/lawrule/sserules/trading/a/20220527/377bbbcd6fc1f3f5a7afabcb671335da.docx

[26] Du ZX, Qian YJ, Liu X, Ding M, Qiu JZ, Yang ZL, Tang J. GLM: General language model pretraining with autoregressive blank infilling. In: Proc. of the 60th Annual Meeting of the Association for Computational Linguistics, Vol. 1 (Long Papers). Dublin: ACL, 2022. 320–335. [doi: 10.18653/v1/2022.acl-long.26]

[27] Touvron H, Martin L, Stone K, *et al*. Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288, 2023.

[28] Hu EJ, Shen YL, Wallis P, Allen-Zhu Z, Li YZ, Wang SA, Wang L, Chen WZ. LoRA: Low-rank adaptation of large language models. In: Proc. of the 10th Int'l Conf. on Learning Representations. 2022.

[29] New York Stock Exchange. NYSE Rules. 2024. https://nyseguide.srorules.com/rules

[30] Japan Exchange Group. Tokyo stock exchange business regulations and related rules (as of January 4, 2024). 2024. https://www.jpx.co.jp/english/rules-participants/rules/regulations/index.html

[31] Hong Kong Exchanges and Clearing Limited. Rules of the exchange. 2024. https://www.hkex.com.hk/-/media/HKEX-Market/Services/Rules-and-Forms-and-Fees/Rules/SEHK/Whole_SEHK_c.pdf

[32] Davis A, Overmyer S, Jordan K, Caruso J, Dandashi F, Dinh A, Kincaid G, Ledeboer G, Reynolds P, Sitaram P, Ta A, Theofanos M. Identifying and measuring quality in a software requirements specification. In: Proc. of the 1st Int'l Software Metrics Symp. Baltimore: IEEE, 1993. 141–152. [doi: 10.1109/METRIC.1993.263792]

[33] Young JD, Antón AI. A method for identifying software requirements based on policy commitments. In: Proc. of the 18th IEEE Int'l Requirements Engineering Conf. Sydney: IEEE, 2010. 47–56. [doi: 10.1109/RE.2010.17]

[34] Breaux TD, Antón AI. Analyzing regulatory rules for privacy and security requirements. IEEE Trans. on Software Engineering, 2008, 34(1): 5–20. [doi: 10.1109/TSE.2007.70746]

[35] Jain C, Anish PR, Ghaisas S. Automated identification of security and privacy requirements from software engineering contracts. In: Proc. of the 31st IEEE Int'l Requirements Engineering Conf. Workshops. Hannover: IEEE, 2023. 234–238. [doi: 10.1109/REW57809.2023.00047]

[36] Arora C, Grundy J, Abdelrazek M. Advancing requirements engineering through generative AI: Assessing the role of LLMs. In: Nguyen-Duc A, Abrahamsson P, Khomh F, eds. Generative AI for Effective Software Development. Cham: Springer, 2024. 129–148. [doi: 10.1007/978-3-031-55642-5_6]

[37] Lafi M, AbdelQader A. Automated business rules classification using machine learning to enhance software requirements elicitation. In: Proc. of the 2023 Int'l Conf. on Information Technology. Amman: IEEE, 2023. 775–778. [doi: 10.1109/ICIT58056.2023.10226076]

[38] Herbst H, Knolmayer G, Myrach T, Schlesinger M. The specification of business rules: A comparison of selected methodologies. In: Proc. of the 1994 IFIP WG8.1 Working Conf. on Methods and Associated Tools for the Information Systems Life Cycle. Amsterdam: Elsevier, 1994. 29–46.

[39] Warmer J, Kleppe A. The Object Constraint Language: Getting Your Models Ready for MDA. 2nd ed., Boston: Addison-Wesley, 2003. 1–240.

[40] Jensen SH, Thummalapenta S, Sinha S, Chandra S. Test generation from business rules. In: Proc. of the 8th Int'l Conf. on Software Testing, Verification and Validation. Graz: IEEE, 2015. 1–10. [doi: 10.1109/ICST.2015.7102608]

[41] Nemuraite L, Skersys T, Sukys A, Sinkevicius E, Ablonskis L. VETIS tool for editing and transforming SBVR business vocabularies and business rules into UML&OCL models. In: Proc. of the 16th Int'l Conf. on Information and Software Technologies. Kaunas: Springer, 2010. 377–384.

[42] Object Management Group. Semantics of business vocabulary and rules (SBVR). 2024. https://www.omg.org/spec/SBVR/1.5/PDF

[43] Grosof BN, Labrou Y, Chan HY. A declarative approach to business rules in contracts: Courteous logic programs in XML. In: Proc. of the 1st ACM Conf. on Electronic Commerce. Denver: ACM, 1999. 68–77. [doi: 10.1145/336992.337010]

[44] Bajwa IS, Lee MG, Bordbar B. SBVR business rules generation from natural language specification. In: Proc. of the 2011 AAAI Spring Symp. San Francisco: AAAI, 2011. 2–8.

[45] Chittimalli PK, Prakash C, Naik R, Bhattacharyya A. An approach to mine SBVR vocabularies and rules from business documents. In: Proc. of the 13th Innovations in Software Engineering Conf. Jabalpur: ACM, 2020. 12. [doi: 10.1145/3385032.3385046]

[46] Holter OM, Ell B. Reading between the lines: Information extraction from industry requirements. In: Proc. of the 14th Int'l Conf. on Recent Advances in Natural Language Processing. Varna: INCOMA Ltd., 2023. 703–711.

[47] Zhang RX, Yang W, Lin LY, Tu ZK, Xie YQ, Fu ZH, Xie YH, Tan LC, Xiong K, Lin J. Rapid adaptation of BERT for information

extraction on domain-specific business documents. arXiv:2002.01861, 2020.
[48] Kardasis P, Loucopoulos P. Expressing and organising business rules. Information and Software technology, 2004, 46(11): 701–718. [doi: 10.1016/j.infsof.2003.12.003]
[49] Bhattacharyya A, Chittimalli PK, Naik R. Relation identification in business rules for domain-specific documents. In: Proc. of the 11th Innovations in Software Engineering Conf. Hyderabad: ACM, 2018. 14. [doi: 10.1145/3172871.3172884]
[50] Schlutter A, Vogelsang A. Knowledge extraction from natural language requirements into a semantic relation graph. In: Proc. of the 42nd IEEE/ACM Int'l Conf. on Software Engineering Workshops. Seoul: ACM, 2020. 373–379. [doi: 10.1145/3387940.3392162]
[51] Mahgoub H, Rösner D, Ismail N, Torkey F. A text mining technique using association rules extraction. Int'l Journal of Computer and Information Engineering, 2008, 2(6): 2044–2051. [doi: 10.5281/zenodo.1058536]

附中文参考文献:

[1] 深圳证券交易所. 深圳证券交易所债券交易规则. 2022. https://docs.static.szse.cn/www/lawrules/rule/bond/bonds/trade/W020220127631859244722.pdf
[2] 上海证券交易所. 上海证券交易所交易规则 (2023 年修订). 2023. http://www.sse.com.cn/lawandrules/sselawsrules/trade/universal/c/10118241/files/e773c99a2fef4fbc930bc332492156d8.docx
[6] 陈小红, 尹斌, 金芝. 基于问题框架方法的需求建模: 一个本体制导的方法. 软件学报, 2011, 22(2): 177–194. http://www.jos.org.cn/1000-9825/3755.htm [doi: 10.3724/SP.J.1001.2011.03755]
[22] 深圳证券交易所. 深圳证券交易所创业板交易特别规定. 2023. https://www.szse.cn/lawrules/rule/repeal/rules/P020231230545310237980.pdf
[23] 深圳证券交易所. 深圳证券交易所交易规则 (2023 年修订). 2023. https://docs.static.szse.cn/www/lawrules/rule/stock/trade/W020230217564423808793.pdf
[24] 深圳证券交易所. 深圳证券交易所证券投资基金交易和申购赎回实施细则 (2022 年修订). 2022. https://docs.static.szse.cn/www/lawrules/rule/fund/trade/W020220610536157903323.pdf
[25] 上海证券交易所. 上海证券交易所债券交易规则. 2022. https://bond.sse.com.cn/lawrule/sserules/trading/a/20220527/377bbbcd6fc1f3f5a7afabcb671335da.docx

李靓果(1999－), 女, 硕士生, 主要研究领域为基于知识的需求形式化与分析验证.

薛志一(2000－), 男, 博士生, 主要研究领域为神经网络的鲁棒训练和验证, 基于大语言模型的测试用例自动生成.

陈小红(1982－), 女, 博士, 副教授, 博士生导师, CCF 专业会员, 主要研究领域为需求工程, 基于知识的软件工程, 形式化方法.

张民(1982－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为形式化方法, 智能系统建模与验证.

陈良育(1980－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为智能软件工程, 智慧教育.

李萍萍(1991－), 女, 硕士, 主要研究领域为证券领域核心交易系统的功能测试及用户验收.

姜婷婷(1982－), 女, 学士, 主要研究领域为证券领域核心交易系统的上线质量保障.