软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007557] [CSTR: 32375.14.jos.007557]

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

基于大语言模型的非功能需求生成方法*

欧阳柳波, 叶巧莹, 孟心如, 杜漫茹

(湖南大学 信息科学与工程学院, 湖南 长沙 410082)

通信作者: 欧阳柳波, E-mail: oylb@hnu.edu.cn

摘　要: 在软件工程领域中, 非功能需求 (NFR) 获取一直是需求工程实践中的重要内容, 但容易被忽视. 传统的 NFR 获取方法主要依赖需求工程师的经验和人工分析, 不仅效率低下, 而且容易出现遗漏和不一致. 近年来, 大语言模型在自然语言处理领域取得突破性进展, 为自动化获取非功能需求提供了新的技术手段. 然而, 直接使用大语言模型生成非功能需求常面临知识幻觉、领域专业性不足等问题. 为此, 提出了一种基于大语言模型的非功能需求自动获取方法, 实现高质量的非功能需求生成. 构建了包含 3 856 条功能需求和 5 723 条非功能需求的结构化关联数据集, 形成 22 647 对 FR-NFR 关联关系. 通过融合检索增强生成 (RAG) 技术, 构建了包含 3 个核心模块的系统化解决方案: 基于最大边际相关性算法的语义案例检索模块、面向非功能需求生成的提示工程模块和基于参数优化的大语言模型生成模块. 通过软件工程专家的专业评分和对 BLEU、ROUGE 等自动评分指标的多维度评估, 实验结果表明方法在需求的完整性、准确性和可测试性等方面优于现有方法.

关键词: 软件需求工程; 非功能需求生成; 大语言模型; 检索增强生成; 提示工程

中图法分类号: TP311

中文引用格式: 欧阳柳波, 叶巧莹, 孟心如, 杜漫茹. 基于大语言模型的非功能需求生成方法. 软件学报. http://www.jos.org.cn/1000-9825/7557.htm

英文引用格式: Ouyang LB, Ye QY, Meng XR, Du MR. Non-functional Requirements Generation Method Based on Large Language Model. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7557.htm

Non-functional Requirements Generation Method Based on Large Language Model

OUYANG Liu-Bo, YE Qiao-Ying, MENG Xin-Ru, DU Man-Ru

(College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China)

Abstract: In software engineering, eliciting non-functional requirements (NFR) remains a critical yet often overlooked task in requirements engineering practice. Traditional NFR elicitation methods predominantly rely on the experience and manual analysis of requirements engineers, leading to inefficiency, omissions, and inconsistencies. Recent breakthroughs in large language models (LLM) in natural language processing have provided new technological means for the automated NFR elicitation. However, directly employing LLM for NFR generation often faces challenges such as hallucination and insufficient domain expertise. To address these issues, this study proposes an automated NFR elicitation method based on LLM to achieve high-quality NFR generation. A structured and correlated dataset comprising 3 856 functional requirements and 5 723 NFR is constructed, establishing 22 647 FR-NFR association pairs. The proposed method integrates retrieval-augmented generation (RAG) technology through three core modules: a semantic case retrieval module based on the maximum marginal relevance algorithm, a prompt engineering module designed for NFR generation, and an optimized LLM generation module. Through professional evaluation by software engineering experts and automatic metrics including BLEU and ROUGE, experimental results demonstrate that the proposed method outperforms existing approaches in terms of completeness, accuracy, and testability of requirements.

Key words: software requirements engineering; non-functional requirements (NFR) generation; large language model (LLM); retrieval-augmented generation (RAG); prompt engineering

---

* 基金项目: 国家自然科学基金 (62172153); 国家社会科学基金 (21BZX055)
收稿时间: 2025-04-24; 修改时间: 2025-05-31, 2025-07-08, 2025-08-15; 采用时间: 2025-09-23; jos 在线出版时间: 2025-12-24

软件工程是一门运用系统化、规范化和可度量的方法来开发、运行和维护软件的工程学科[1]. 作为其核心组成部分, 需求工程[2]专注于软件需求的系统化获取、分析、记录和验证过程. 在软件开发生命周期中, 需求分析是最为关键的阶段[3], 其质量与软件项目的成功息息相关. 需求分析涵盖了软件系统的功能需求 (functional requirement, FR) 和非功能需求 (non-functional requirement, NFR). 功能需求描述系统应完成的功能, 而非功能需求则体现系统在可观察属性上的表现, 如可靠性、性能、可用性等[4], 直接影响用户体验和系统的长期可持续性.

在当今互联网和大数据技术迅猛发展的背景下, 软件系统的规模和复杂性日益增长, 这无疑提高了对系统质量、可用性和成功率的期望. 正因如此, 非功能需求在软件工程领域扮演着越来越重要的角色. 然而, 在实际的软件开发过程中, 非功能需求的获取和管理始终面临诸多挑战. 根据 Ameller 等人[5]的研究综述, 在需求工程的早期阶段, 功能需求与非功能需求之间的界限往往较为模糊, 只有在进一步分析和细化过程中才能逐步明确. 除此之外, 非功能需求还具有抽象性和跨功能性的特点. 一方面, 非功能需求的抽象性体现在其描述通常缺乏具体、可操作的技术细节, 而更多地依赖于高层次的质量目标. 例如, “系统必须具有高可用性”这一需求在获取阶段可能难以明确量化目标或对应的实现方式, 需结合具体场景进行逐步分解. 另一方面, 非功能需求的跨功能性意味着它通常不是某一单一模块或功能的属性, 而是影响整个系统或多个模块的综合质量表现. 例如, “安全性”这一需求可能涉及身份验证模块、数据传输加密机制以及日志审计等多个功能模块的协同设计, 进一步增加了非功能需求的复杂性. 正是这些挑战, 使得非功能需求的准确识别和获取成为需求工程研究的重要方向, 并激发了众多相关研究的开展.

传统的非功能需求获取方法主要依赖于需求工程师与项目涉众的直接互动[6], 虽然信息准确但效率低下, 并且高度依赖需求工程师的个人经验和专业水平. 近年来, 随着人工智能技术的发展, 基于机器学习和神经网络的非功能需求自动提取方法逐渐兴起, 使得非功能需求获取可以不再依赖需求工程师[7]. 但这类方法仍局限于从已有文档中识别显式表达的非功能需求, 难以发现潜在的质量需求[8]. 部分研究开始关注功能需求与非功能需求之间的关联关系, 尝试通过分析功能需求来启发潜在非功能需求的获取, 但现有方法仍过度依赖人工进行分析而未实现自动化[9]. 如何充分利用功能需求与非功能需求之间的内在联系, 基于输入的功能需求自动且高效地生成相应的非功能需求, 实现从“识别已有”到“主动生成”的转变, 是本文面临的关键挑战.

近年来, 大语言模型 (large language model, LLM) 在自然语言处理领域取得了突破性进展[10]. 这些模型在海量的文本语料库上进行预训练, 从数十亿个单词中学习通用的语言模式和语义表示. LLM 在多种自然语言处理任务中展现出显著性能, 并可通过少量样本学习快速适应新任务, 为非功能需求的自动生成带来了新的机遇. 然而, 直接应用 LLM 生成非功能需求仍面临一些挑战: 首先, LLM 容易产生与事实不符的“幻觉”输出[11], 这种“幻觉”现象源于模型在预训练过程中学习的是语言的统计模式而非确切的事实知识, 且在生成回答时会过度依赖这些统计模式而非严格的逻辑推理; 其次, 尽管 LLM 在预训练阶段展现出强大的泛化能力, 但它们在面对现实世界中多样化的特定任务时, 往往缺乏针对性的有效支持, 导致它们难以生成高质量且符合实际项目需求的专业性内容. 为了克服这些局限, 研究者提出了检索增强生成 (retrieval-augmented generation, RAG) 框架[12], 通过从外部知识库中检索相关知识片段, 来指导 LLM 生成更可靠、更专业的内容. 将 RAG 与 LLM 相结合, 有望实现高质量非功能需求的自动生成.

在此背景下, 本文提出了一种非功能需求自动生成方法, 该方法以大语言模型为核心, 结合检索增强生成技术, 显著提升了非功能需求获取的效率和质量. 该方法首先构建了一个从历史项目中提取的功能需求与非功能需求相互关联的结构化案例库, 利用需求向量化技术构建为一个支持高效语义检索的知识库. 在非功能需求的生成过程中, 首先利用案例库检索与输入的功能需求高度相关的非功能需求知识片段, 这一步骤能够从大量历史数据中快速锁定最具价值的参考信息. 随后, 将检索到的知识嵌入精心设计的提示模板中, 构建信息丰富、条理清晰的上下文. 最后, 输入该上下文至大语言模型, 充分发挥其强大的自然语言生成能力, 合成高质量、专业性强的目标非功能需求. 本文通过广泛的实验, 证明了这一方法在提高非功能需求获取效率的同时, 也兼顾了需求的专业性和全面性, 旨在为软件工程领域的需求获取提供新的思路.

本文第 1 节介绍非功能需求获取的相关方法和研究现状. 第 2 节介绍本文所需的相关理论与技术基础, 包括需求工程和检索增强生成的相关理论. 第 3 节详细描述数据集的构建方法. 第 4 节提出基于大语言模型的非功能

需求生成框架. 第 5 节通过实验验证了所提方法的有效性. 第 6 节总结全文并展望未来研究方向.

1 相关工作

从需求获取的来源来看, 现有的非功能需求获取方法可以分为两大类: 基于涉众交互的直接获取方法和基于已有资料的间接获取方法. 前者主要通过与项目相关方的直接互动来获取需求, 后者则依托于对现有项目文档和相关资料的分析.

在基于涉众交互的方法中, 传统做法主要依赖需求工程师通过访谈、工作坊、文档分析等多种技术手段从项目涉众处获取需求, 如客户-开发人员会议[13,14]、用户访谈和技术专家访谈[15]以及原型法[16]等方式. 这类方法虽然能够获取第 1 手需求信息、深入理解涉众需求, 但往往耗时较长且严重依赖于需求工程师的个人经验. 为了提升效率, 研究者开始探索更加结构化的方法. 例如, Kopczyńska 等人[17]提出的结构化非功能需求获取方法, 通过预设的会议框架和质量分析模板来规范获取过程. 同时, 随着敏捷开发方法的普及, 出现了更加灵活的非功能需求获取方式, 如专门的非功能需求获取工作坊[18]、迷你质量属性工作坊[19]等研讨会形式. 这些方法通过群体互动和快速反馈来提升需求获取的效率, 但仍然需要大量的人力投入.

在基于资料分析的间接获取方法中, 早期研究主要集中在从现有文档中提取显式表达的非功能需求. 文献 [20] 指出, 非功能需求可以在应用程序的文档、图像和其他软件制品中找到, 然而, 由于功能和质量在逻辑上可能相互关联, 功能需求和非功能需求往往交织在一起难以分辨[4]. 为此, 研究者提出了多种自动化提取方法, 从早期的基于业务流程模型[21]和 OCR 技术处理项目文档[22], 发展到近期基于机器学习和深度学习的方法[23]. Rashwan 等人[24]提出了非功能需求类型的本体注释和支持向量机分类器的 ML 算法, 自动将非功能需求句子归入其本体类别. Kaur 等人[25]提出了一种改进的深度学习方法, 即 BERT-CNN, 用于非功能需求的提取和分类. 同样, 在 Rahman 等人[26]的研究中, 也采用了递归神经网络对非功能需求进行分类. 另一方面, Yahya 等人[27]倾向于采用混合深度学习模型, 结合 RNN 模型和 2LSML 来识别和分类从移动应用程序用户评论中获取的非功能需求. 这些方法在提高非功能需求获取效率方面取得了显著进展, 但其本质仍是一种“提取”工作, 难以发现未被描述的潜在质量需求.

为了摆脱现有文档的局限性, 研究者们开始探索基于功能需求推导潜在非功能需求的方法, 这一思路基于两个重要观察: (1) 在软件开发过程中, 功能需求通常先于非功能需求被获取和确定, 且通常描述更为完整; (2) 功能需求与非功能需求之间存在密切的关联关系[28], 例如, 一个“用户登录”的功能需求可能隐含了安全性、响应时间、可用性等多个质量属性的要求. 基于这一认识, Rahman 等人[29]提出了基于用例扩展的非功能需求获取技术, 通过为每个功能需求设计引导性问题来系统性获取相关非功能需求. 同时, 历史项目经验表明, 相似功能场景往往对应着类似的非功能需求模式. 基于该观察, Ramos 等人[30]开发了基于协同过滤的非功能需求推荐系统, 通过分析历史项目中功能需求与非功能需求的对应关系, 为新项目推荐相关非功能需求. 这种方法能够有效复用已有经验, 但其效果在很大程度上依赖于历史数据的质量和覆盖范围, 且存在冷启动问题, 即在缺乏足够历史数据的情况下难以提供有效推荐.

近期大语言模型在自然语言理解和知识推理方面的突破为非功能需求获取提供了新的可能性, 但相关研究仍然匮乏且亟待开发. 与传统的机器学习方法相比, 大语言模型不仅能够理解功能需求的上下文语境, 还可以基于其在软件工程领域的预训练知识进行需求推理. 这种能力特别适合从功能需求中识别潜在的非功能需求, 因为它可以理解功能描述中隐含的质量属性暗示. 然而, 大语言模型的预训练知识具有一定的局限性: 一方面可能产生与实际工程实践不符的推理结果, 另一方面难以准确把握特定项目的具体需求情境.

综上所述, 现有的非功能需求获取方法面临以下挑战: 传统的直接获取方法依赖人工经验且效率低下; 基于已有文档的提取方法难以发现未明确描述的需求; 而基于功能需求推导的方法虽然前景广阔, 但目前仍缺乏有效的自动化支持. 针对这些问题, 本文认为可利用检索增强生成技术来增强大语言模型的需求获取能力, 通过检索相似功能场景的历史非功能需求案例作为参考, 不仅可以提升模型推理结果的可靠性, 还能将历史项目的实践经验与新项目的具体情境有机结合, 从而实现更加准确和高效的非功能需求获取.

2 基础知识

2.1 非功能需求基础理论

在软件需求工程中, 需求被划分为两大类[31]: 功能需求和非功能需求. 功能需求明确定义了软件系统必须完成的具体任务, 是从用户角度对软件系统行为的描述, 包括系统应该提供的具体服务、操作和功能. 与功能需求相辅相成, 非功能需求构成了软件工程中另一个至关重要的概念. 虽然非功能需求在软件工程发展的早期阶段就已出现, 但至今尚未形成一个统一的定义. Chung 等人[4]将非功能需求定义为“软件系统的质量属性和约束条件”, 突出了非功能需求在确保软件质量方面的重要性. 而随着软件工程理论的发展, Glinzd[32]进一步将非功能需求定义为“描述系统服务或功能如何交付的需求”, 强调了其对系统实现方式的指导和制约作用. 从工程实践角度, IEEE 830-1998 标准[33]将非功能需求界定为“除功能需求外的所有需求, 包括但不限于性能需求、外部接口需求、设计约束和软件质量属性”. 这一标准为非功能需求的识别和实现提供了一个全面的框架, 确保了软件系统在满足功能需求的同时, 也能符合各种非功能的质量标准.

Pohl[31]认为非功能需求分为两类: 不明确的非功能需求和质量需求, 其中不明确的非功能需求需要进行精化和细化, 否则会造成不同涉众以完全不同的方式来理解该需求, 从而导致潜在的风险. 如无法满足涉众的期望、无法客观证明或检查该需求是否在最终系统中正确地实现. 质量需求, 作为非功能需求的一个重要分支, 是指那些描述软件系统的质量属性和约束条件的需求, 它们是确保软件满足用户明确或潜在需求的关键因素. 因此, 为了全面理解和有效管理这些质量属性, 质量模型的概念应运而生. ISO/IEC 25010 标准[34]定义了一个产品质量模型, 包含 9 个主要的质量特征: 功能适用性、性能效率、兼容性、交互能力、可靠性、安全性、可维护性、灵活性和无害性.

2.2 检索增强生成

大语言模型在实际应用中面临着知识时效性[35]、知识局限性[36]以及幻觉[37]等关键问题. 知识时效性体现在模型训练数据的截止时间限制, 导致无法获取最新信息; 知识局限性则表现为模型对特定领域知识的掌握不足; 而幻觉问题则可能导致模型生成虚假或不准确的内容. 这些问题不仅影响模型的实际应用效果, 也增加了维护和更新的成本. 为了有效解决上述问题, 研究者提出了检索增强生成技术. 该技术通过将信息检索与文本生成相结合, 在保持模型基础能力的同时, 实现了知识的动态扩充, 其主体工作流程如图 1 所示.

在 RAG 实现的过程中, 首先需要从各种格式的数据源 (如 Word 文档、PDF 文件或 HTML 网页) 中提取和清洗数据. 经过清理后, 数据被转换为标准化的纯文本格式, 以便于进一步处理. 为了克服 LLM 在处理长文本时的上下文限制, 文本会被切分为较小的块 (Chunks), 这一过程通常称为“分块”. 每个文本块随后会被转化为数字向量, 这一转换通常依赖于预训练的嵌入模型, 每个嵌入向量携带了文本块的语义信息, 便于在后续查询中高效检索和匹配. 在此基础上, 索引的构建将这些文本块及其对应的嵌入向量存储为键值对, 由此问答系统能够快速定位到最相关的文本块, 从而生成与查询相关的上下文答案. 索引不仅提升了检索的效率和准确性, 还为处理大规模数据集提供了可扩展的解决方案. 通过利用向量化的表示, 系统能够根据语义相似度而非传统的关键词匹配来找到最匹配的内容, 从而显著提升了答案的质量与上下文相关性.

接下来是检索环节, 主要任务是从知识库中精确检索出与当前查询相关的信息, 为后续增强和生成环节提供支撑. 在完成知识库构建和查询向量生成后, 系统进入相似度计算与匹配阶段, 该阶段基于预定义的相似度度量 (如余弦相似度、欧氏距离等) 计算查询向量与知识库向量间的相似程度. 考虑实际应用中的效率要求, 系统通常采用近似最近邻搜索算法, 如局部敏感哈希或基于量化的方法, 在保证检索质量的同时降低计算开销.

在最后的生成阶段, 首先将检索到的上下文信息来增强原始查询作为输入, 接着调用 LLM 生成回复. 此过程中生成策略的控制直接影响输出质量, 系统通常采用多样化的解码方法组合. 在温度参数配置方面, 较低的温度值有助于生成更确定性的答案; 而在创造性任务中, 适当提高温度值可增加输出的多样性. 同时, 系统结合 Top-k 和核采样等技术, 通过动态调整候选词元池的大小和分布, 在保证生成质量的同时引入适度的随机性. 对于强调真实性的场景, 系统还会引入基于检索信息的约束生成机制, 确保生成内容不偏离事实基础. 在生成过程中, 系统通过

设置输出长度限制、重复惩罚等参数控制生成文本的整体质量, 同时针对不同类型的检索信息, 系统采用相应的引用策略, 在保持生成内容流畅性的同时确保信息溯源的准确性. 当涉及多份检索文档时, 系统会通过引入文档权重和可信度评分, 动态调整不同来源信息的影响力, 从而提升生成内容的可靠性和连贯性.

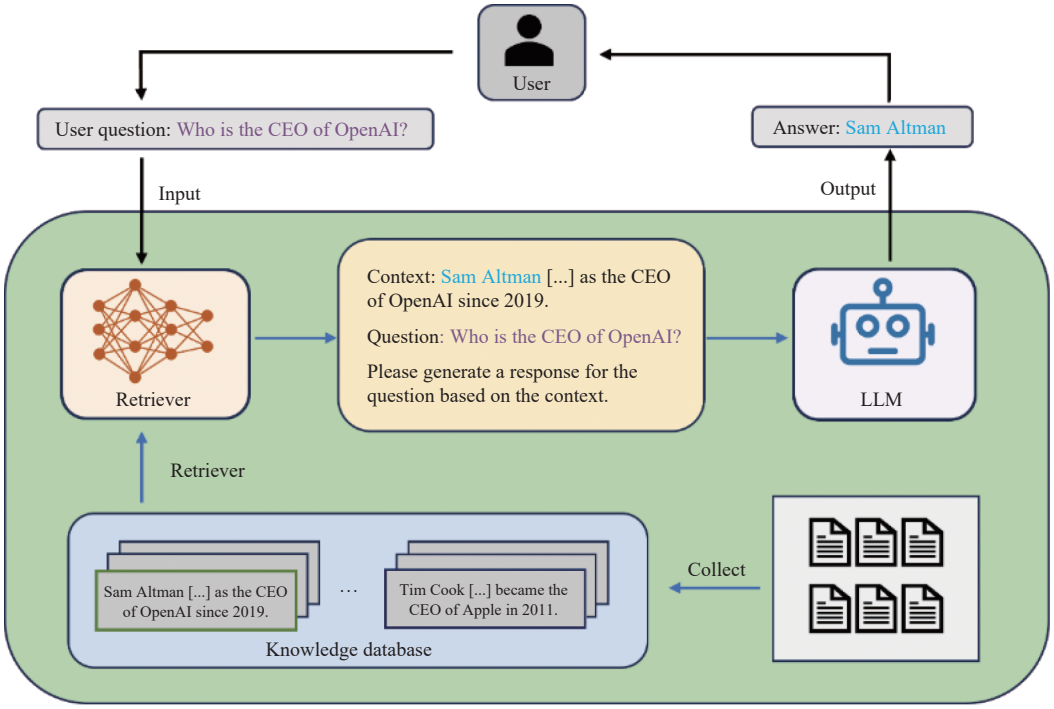

图 1 RAG 流程示意图

3 数据集构建

鉴于目前尚缺乏公开可用的功能需求与非功能需求 (FR-NFR) 关联数据集, 本文构建了一个高质量的 FR-NFR 关联数据集, 以便在后续基于大语言模型的研究中, 作为检索增强生成的知识库, 支持非功能需求的自动生成.

3.1 数据收集与预处理

为了保证知识库的质量, 并提高检索增强生成方法的有效性, 本文将基于以下 3 个核心准则选择数据源.

(1) 知识可靠性: 优先选择经过同行评议或业界认可的数据源.

(2) 领域覆盖均衡性: 为保证检索增强生成技术的有效性, 需确保数据集在企业级应用、移动应用、Web 服务等关键领域的均衡分布, 避免检索结果的领域偏差.

(3) 案例完整性: 每个功能需求与非功能需求的对应关系必须清晰完整, 且包含足够的上下文信息, 以支持后续的语义相似度计算和案例检索.

此外, 为确保数据的代表性和多样性, 本文采用多源数据融合策略, 基于这 3 个核心准则从 3 个不同渠道收集和整理需求数据, 数据来源和特征总结在表 1 中. 首选数据来源是软件工程领域广受认可的 PROMISE 数据集中的 tera-PROMISE (详见 http://promisedata.org/repository). 该数据集规模较小, 由 625 个需求句子组成. 这些句子中, 255 个是功能需求, 370 个是非功能需求. 非功能需求被标记为 11 个类别, 分别是可用性、合法性、外观和感觉、可维护性、可操作性、性能、可伸缩性、安全性、可用性、容错性和可移植性. 在原始数据中, 功能需求与非功能需求之间的关联关系是通过数字标识符进行标记的. 为了便于后续处理和分析, 本文开发了专门的 Python 脚本, 将这些关联关系转化为结构化的 JSON 格式, 同时保留了原始的语义关联信息, 这种转化不仅提高了数据的可用

性, 也为后续的需求分析和模型训练奠定了基础. 然而, PROMISE 数据集的规模较小, 可能无法充分反映现实世界中功能需求与非功能需求关联的复杂性和多样性. 为此, 本文引入了来自 Mendeley 数据平台的第 2 个数据源 (详见 https://data.mendeley.com/datasets/4ysx9fyzv4/1). 该数据集主要聚焦于 Web 应用和移动应用领域, 包含了约 6000 条经过专业人员标注的需求描述, 这些需求数据来源于实际的软件需求规格说明文档, 具有较强的实践指导意义. 在原始数据中, 功能需求与非功能需求的数量比例约为 1:2, 这一比例仅反映了需求描述的原始分布, 而非最终的功能需求-非功能需求关联对数量, 因为在实际系统中, 一个非功能需求往往会影响多个功能需求的实现.

表 1　数据来源及特征

数据来源	需求描述的数据规模	主要特征
PROMISE数据集	625条	11类非功能需求特征; 关联关系明确; 标准数据集
Mendeley数据平台	6000余条	侧重Web应用和移动应用; 来源于实际需求规格说明
开源平台 (Zenodo/GitHub)	3000余条	跨领域覆盖; 包含企业级应用和中小型应用

为进一步扩充数据集的覆盖面, 本文还从多个开源平台收集了大量软件需求规格说明文档, 如从学术资源平台 Zenodo 获取了数十个大型项目的需求文档, 经筛选和整理, 得到 35 个需求描述较为完整的文档, 从中提取 3000 余条需求描述. 这些项目包括 EAsyAnon 审计系统、AFFIRMO 医疗系统等企业级应用, 其需求文档具有规范性强、完整度高的特点. 同时, 为平衡数据集中不同规模项目的分布, 本文还从 GitHub 平台以“requirement document”“srs-document”等关键词为主题标签收集了一批中小型项目的需求文档, 包含图书馆管理系统、学生信息管理系统、导航软件、手机应用软件等多种类型, 以确保数据集具有广泛的领域覆盖性. 这种跨规模、跨领域的采样策略有效提升了数据集的代表性和普适性.

在完成原始数据收集后, 本文设计了一套系统化的预处理流程, 以提升数据质量.

第 1 步是文本清理, 主要包括特殊字符处理、多余空格删除、大小写统一和标点规范化等基础工作. 在此基础上, 本文还利用拼写检查工具, 自动识别和修正文本中的明显拼写错误.

第 2 步是需求描述的标准化处理, 这包括统一需求的表达格式, 如采用“系统应该 (the system shall)”的规范句式; 拆分复合需求, 确保每条需求描述仅包含单一的功能或质量属性; 建立统一的术语映射表, 消除不同来源文档中同义词表达的差异. 考虑到不同项目领域中术语语义的复杂性, 本文采用 3 层术语映射架构: (1) 全局通用术语表: 涵盖软件工程领域的标准术语, 共计 186 个标准术语. (2) 领域特定术语表: 针对 Web 应用、移动应用、企业级系统等不同应用领域建立专门术语表, 处理领域内的特有表达方式, 如 Web 领域的“响应时间”与移动应用领域的“启动时间”的语义对应关系, 各领域术语表平均包含 120–150 个术语. (3) 项目冲突术语表: 记录在多个项目中发现语义冲突的术语及其上下文特定含义, 通过人工专家判断确定优先映射规则, 共记录冲突术语 62 个. 术语映射的构建过程采用半自动化方法: 首先使用基于词向量相似度的自动聚类算法 (相似度阈值设为 0.75) 识别潜在的同义术语组, 然后由两名软件工程专家独立审核聚类结果, 统一同义词.

第 3 步是数据格式转换, 将所有来源的需求数据统一转化为规范的 JSON 格式, 每条记录包含唯一标识符、需求描述文本、需求类型标签等字段. 数据格式的处理过程如图 2 所示.

为了保证预处理结果的可靠性, 检验预处理的有效性, 本文建立了需求质量量化评估体系.

(1) 上下文完整性指数 (*CI*): 评估一条功能需求描述的完整程度, 包括输入条件、功能行为、输出结果等关键元素的覆盖情况;

(2) 语义表达规范性指数 (*SI*): 评估一个项目内需求描述的标准化程度, 包括术语使用的一致性、句式的规范性等.

CI 根据需求描述中包含“输入条件”“功能行为”和“输出结果”这 3 个要素的比例评分 (0–3 分), 计算公式为:

$$CI = (E_i + E_b + E_o)/3,$$

其中, E_i、E_b、E_o 分别表示需求描述中是否包含输入条件、功能行为、输出结果这 3 个要素评分.

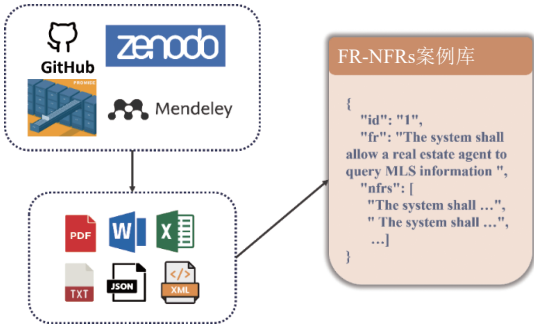

图 2　数据格式处理

SI 通过术语一致性和句式规范性计算得分, 计算公式为:

$$SI = 0.6 \times T + 0.4 \times S,$$

其中, T 表示术语一致性得分, S 表示句式规范性得分. T 基于前述 3 层术语映射表处理后的结果进行计算, 评估整个数据集中术语使用的全局一致性, 计算公式为:

$$T = \frac{N_{\text{standard}}}{N_{\text{total}}},$$

其中, N_{standard} 为一个项目内使用标准术语的需求描述数量, N_{total} 为一个项目内的需求描述总数.

句式规范性基于预定义的需求句式模板库进行评估, 该模板库包含 8 类标准句式模板: 功能性模板 (如“系统应该 [动词][对象]”)、性能模板 (如“系统应在 [时间] 内完成 [操作]”)、安全性模板 (如“系统应确保 [数据/操作] 的 [安全属性]”) 等, 每类模板包含 3–5 个具体变体, 总计 32 个标准模板. 句式规范性计算公式为:

$$S = \frac{N_{\text{conforming}}}{N_{\text{total}}},$$

其中, $N_{\text{conforming}}$ 为一个项目内符合标准模板的需求描述数量.

通过上述评估体系对预处理后的数据集进行质量评估, 结果显示: *CI* 平均得分为 2.47 分 (满分 3 分), 表明 82.3% 的需求描述包含完整的功能要素; *SI* 平均得分为 0.89 分 (满分 1 分), 其中术语一致性达到 91.8%, 句式规范性达到 84.7%. 对于 *CI* 低于 2.0 分或所在项目的 *SI* 低于 0.7 分的需求描述 (共计 312 条, 占总数的 8.1%), 均进行了专家复核, 其中 203 条通过人工完善后重新纳入数据集, 109 条因质量过低被剔除.

最后, 本文还建立了需求溯源机制, 记录每条需求的来源信息和处理历史, 便于后期的质量追踪和验证.

通过上述数据收集和预处理流程, 本文构建了一个包含 3 856 条功能需求和 6 723 条非功能需求的规范数据集 (完整数据集可以在这里获取: https://www.kaggle.com/datasets/yimingzuozhe/fr-nfr-datasets). 与现有公开数据集相比, 本文的数据集在规模、多样性和质量控制等方面都有显著优势, 为后续的功能需求-非功能需求关联分析和自动生成研究提供了可靠的数据基础.

3.2　标注方法与流程

完成数据预处理后, 需求之间的关联关系仍然需要通过专业的人工判断来确定. 为建立高质量的功能需求-非功能需求关联知识库, 本文设计了一套系统化的专家标注方案, 旨在识别和确认每个功能需求与哪些非功能需求存在实质性关联. 该方案包括标注团队的组建、标注规则的制定、标注流程的实施以及质量控制等多个环节.

在标注团队的组建方面, 本文招募了 15 名具有不同背景的专业人员参与标注工作. 这些人员被平均分为 5 个标注组, 每组由 3 名成员组成, 包括 1 名软件需求工程师、2 名软件设计开发人员.

在标注规则的设计上, 采用二元判定方法, 即由专家判定功能需求和非功能需求之间是否存在关联关系. 这一选择基于以下考虑: 首先, 二元判定可以显著降低标注者之间的主观判断差异, 提高标注结果的一致性; 其次, 简化的判定规则有助于提高标注效率, 使专家能够处理更大规模的数据; 最后, 二元关联关系更适合后续的自动化处理.

标注过程采用了严格的流程控制和质量保证机制, 每个标注组独立处理分配的需求数据, 组内 3 名成员首先

各自独立完成标注工作, 然后通过多数投票的方式解决分歧. 为了确保标注质量的一致性, 采用 Fleiss' Kappa 系数[38]评估组间的标注一致性, 该系数特别适用于多人标注的情况, 能够有效衡量不同标注者之间的一致程度. 为确保标注质量, 本文将阈值设定为 0.6, 该系数的计算公式如公式 (1) 所示:

$$\kappa = \frac{\overline{P} - \overline{P_e}}{1 - \overline{P_e}} \tag{1}$$

其中, $\overline{P}$ 和 $\overline{P_e}$ 的计算方式如公式 (2) 和 (3) 所示:

$$\overline{P} = \frac{1}{Nn} \sum_{i=1}^{N} \sum_{j=1}^{k} n_{ij}(n_{ij} - 1) \tag{2}$$

$$\overline{P_e} = \sum_{j=1}^{k} p_j^2 \tag{3}$$

其中, N 是需求数量, n 是每个需求的标注者数量, k 代表分类数 (本文中 $k = 2$, 表示"关联"和"不关联"), n_{ij} 是第 i 个需求被分到第 j 类的标注者数量, p_j 是所有标注中第 j 类的比例.

完成专家标注后, 为便于后续检索和生成任务的开展, 本文将标注结果转换为结构化的 JSON 格式, 每条记录包含 3 个主要字段: (1) 唯一标识符 (id): 用于追踪和引用具体的需求对; (2) 功能需求描述 (FR): 包含完整的功能需求文本; (3) 关联的非功能需求列表 (NFR): 以数组形式存储所有与该功能需求存在关联的非功能需求描述.

4 方法设计

基于第 1 节的分析可知, 利用大语言模型实现非功能需求的自动生成面临着两个主要挑战: 如何有效利用已有的工程经验以及如何提升生成内容的质量.

针对经验复用, 本文受案例推理 (case-based reasoning, CBR) 理论[39]启发. 该理论认为, 相似问题往往具有相似的解决方案, 这一特点与非功能需求的获取过程高度契合. 在实际项目中, 相似的功能通常会关联相似的质量属性要求. 基于这一认识, 本文采用 CBR 作为检索增强生成的理论依据, 通过建立功能需求与非功能需求的对应关系案例库, 利用 RAG 检索该案例库可以增强 LLM, 为新的功能需求找到相似的历史案例, 从而为非功能需求的生成提供可靠的参考基础.

提示学习理论研究如何通过精心设计的提示来引导模型生成符合预期的输出, 为提高大语言模型的生成质量提供了理论指导. 在本文中, 提示学习理论主要指导了 5 个方面的设计: 首先是角色定位, 通过明确的专家角色设定增强模型的专业输出能力, 如将模型定位为具有丰富软件工程经验的需求分析师; 其次是任务背景, 通过提供领域相关的上下文信息和历史案例, 帮助模型更好地理解任务需求; 第三是分析步骤, 采用任务分解和思维链策略, 引导模型进行系统化的需求分析; 第四是输出格式, 通过明确的格式规范和质量要求, 确保生成内容的规范性和可测试性; 最后是情感激励, 通过建立积极的交互氛围和责任感, 提升模型的主动性和创造性.

基于上述理论, 本文提出了一种结合 RAG 的基于大语言模型的非功能需求自动生成方法, 其中 RAG 旨在通过在生成过程中引入外部知识来增强大语言模型的输出质量, 其核心思想是: 在模型生成回答之前, 首先从知识库中检索与当前输入相关的知识片段, 将这些知识作为上下文提供给模型, 从而帮助模型生成更准确、更可靠的响应. 这种方法可以有效缓解大语言模型的知识幻觉问题, 同时提供领域相关的专业知识支持. 在本文中, 通过引入经过专家验证的历史需求案例作为参考, 显著降低模型生成"幻觉"内容的风险, 同时作为对非功能需求的补充, 提升非功能需求生成的质量和可靠性, 后文图 3 展示了本方法的基本流程.

RAG 通常有 3 个阶段: 检索、增强和生成. 对应到本文方法的 3 个紧密协作的核心模块: 基于语义相似度的案例检索模块、面向非功能需求生成的提示工程模块以及基于大语言模型的需求生成模块.

4.1 基于语义相似度的案例检索模块

在 RAG 框架中, 检索阶段的质量直接决定了知识增强的效果. 与传统 RAG 方法主要关注事实性知识的检索不同, 在非功能需求生成场景下, 检索的目标是寻找语义相似的功能需求及其对应的非功能需求对. 这种检索不仅

需要准确理解输入功能需求的语义内涵, 还需要从第 3 节所构建的 FR-NFR 关联数据集中获取最具参考价值的历史案例, 具体效果如图 4 所示. 基于语义相似度的案例检索模块是本文的核心组成部分之一, 其主要任务是通过检索与输入的功能需求在语义上相似的历史案例, 为后续的非功能需求生成提供可靠的上下文知识. 该模块的设计旨在充分利用大规模语言模型的文本嵌入能力, 构建一个高效的向量化案例库, 并通过语义相似度检索技术, 快速定位与输入功能需求最相关的历史案例. 本节将从向量化案例库的构建、语义相似度检索、检索结果的排序与重排等方面详细介绍该模块的设计与实现.

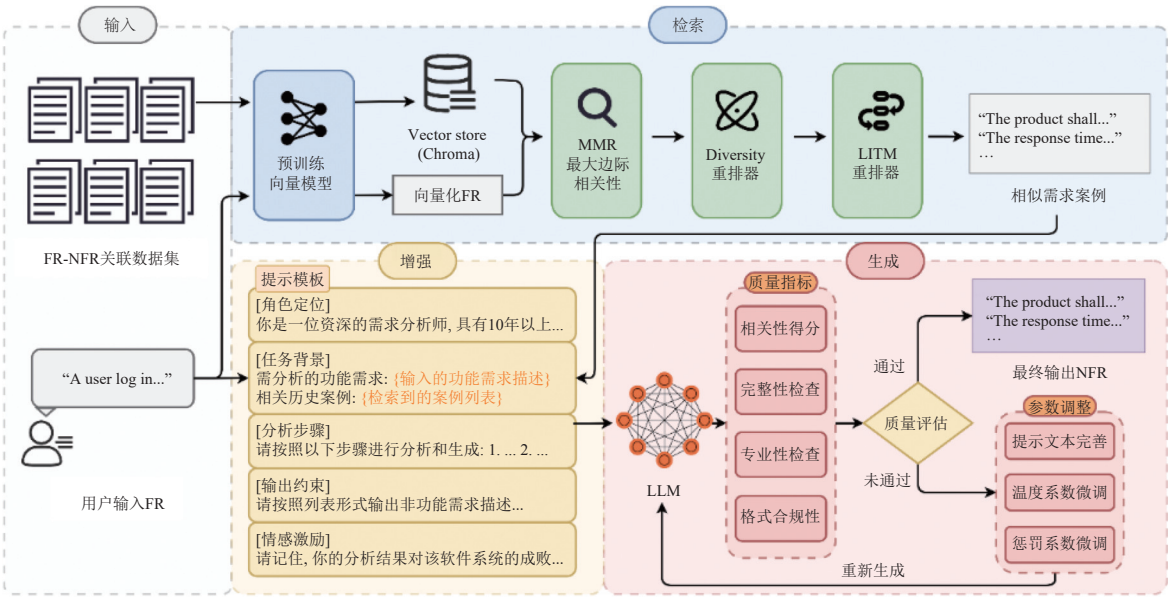

图 3　基于大语言模型的非功能需求自动生成方法总览图

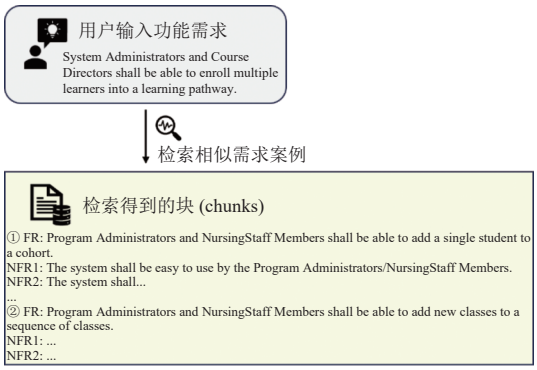

图 4　检索模块示例

本文采用基于 Transformer 架构的文本向量化方法, 选择在大规模文本语料上预训练的 text-embedding-ada-002 模型作为向量化工具来生成密集向量, 使用 Chroma 作为向量数据库来存储功能需求的向量表示. 接下来, 本文将 MMR 算法应用于语义相似度检索过程, 能够在保证检索结果与输入功能需求高度相关的同时, 避免返回语义上过于相似的案例, 从而提升检索结果的多样性和实用性

在将检索结果输入给 LLM 之前, 还有一些问题需要关注. 首先, LLM 的上下文窗口利用效率直接影响其生成质量, 任何浪费的空间或重复的内容都会限制本文可以提取和生成的答案的深度和广度, 适当地布局上下文窗口的内容是一项微妙的平衡行为. 然而, 通过向量相似性检索到的文档虽然高度相关, 但是检索结果很多重复且数量

众多, 可能导致 LLM 所获得的实际有效信息受限, 难以生成更加全面的结果. 除此之外, Liu 等人[40]发现, LLM 很难将注意力集中在长上下文中间的相关段落上, 而是着重把注意力放在文本开头和结尾的位置, 从而忽略了位于中间位置的文本. 为了解决这些挑战, 本文引入了多样性重排器和迷失在中间 (LITM) 重排器来改进 RAG.

多样性重排器使用案例库的向量化工具对检索结果进行向量化, 计算检索案例之间的相似度, 然后依据这些相似度关系进行重新排序和选择, 以增强最终案例集的多样性. 其工作原理是, 多样化的文档集可以提高 LLM 生成更广泛、更深入的答案的能力. 多样性重排器使用以下算法过程处理检索到的案例集.

(1) 使用句向量化工具计算每个案例和查询的嵌入.

(2) 选择语义上与查询最接近的案例作为第 1 个选定的案例.

(3) 对于每个剩余的案例, 计算其与已选择案例的平均相似度.

(4) 选择与已选案例平均相似度最不高的案例.

(5) 此选择过程持续进行, 直到选择所有案例, 最终得到一个案例列表, 其排序方式为对整体多样性贡献最大的案例到贡献最小的案例.

另外, 由于 LLM 更加重视上下文窗口的开头和结尾, LITM 重排器交替将最佳匹配案例放在列表的两端, 使 LLM 能够轻松访问和使用它们. 以一个简单的情况作为示例, 假如重排前列表由 1–10 的单个数字按升序排列, 则经过 LITM 重排后的列表为: [1 3 5 7 9 10 8 6 4 2].

4.2 面向非功能需求生成的提示工程模块

在 RAG 框架中, 增强阶段的核心任务是将检索的知识有效整合到生成过程中. 传统 RAG 方法通常直接将检索结果拼接到提示中, 但面对宽泛或细节不足的提示时, 大模型往往产生普适性输出——这类输出虽适用于多种场景, 却难以满足特定应用的最优需求. 提示工程则通过系统化设计和优化输入提示, 引导大语言模型生成具有高准确性、相关性和连贯性的响应. 相比之下, 精确且详细的提示能够显著降低模型的不确定性, 引导其生成更符合特定场景需求的内容.

本文认为, 软件工程等专业领域需要更精细的知识融合策略, 而提示工程作为连接检索与生成的关键桥梁, 其设计质量直接决定了检索知识的利用效率和最终内容的生成质量. 基于此, 本节提出了一套面向非功能需求生成的专业化提示工程方案, 通过在 LLM 提示工程策略的基础上设计精细提示模板, 实现检索的 FR-NFR 知识向有效提示信息转化, 同时确保生成需求的专业性和完整性. 该方案支持动态嵌入外部知识, 为特定领域内的需求生成提供更为可靠的框架支持.

在设计提示模板时, 首要目标是融入软件工程领域知识, 确保生成内容的专业性, 这要求模板能够有效引导大语言模型运用软件质量标准和需求工程最佳实践, 将专业知识转化为具体的非功能需求描述. 其次, 模板需要构建结构化的分析框架, 提升需求获取的系统性, 通过设定清晰的分析步骤和思考路径, 引导模型全面考虑功能场景特征、质量属性关联以及历史经验参考. 此外, 模板中还需要融入情感激励元素, 通过建立积极的交互氛围提升模型的生成质量. 具体而言, 在提示语中加入对模型专业能力的肯定性表述, 强调任务的重要性和影响力, 并通过设定明确的奖励机制来激励模型产出更高质量的结果. 这种情感激励策略能够有效调动模型的潜力, 使其在需求生成过程中表现出更强的主动性和创造性. 最后是规范输出格式, 保证生成需求的可测试性, 这意味着模板必须明确要求输出包含具体的质量指标和验证方法, 而不是停留在抽象的质量描述层面.

因此, 提示模板由 5 个部分构成: 角色定位、任务背景、分析步骤、输出格式和情感激励.

(1) 角色定位

研究表明, 明确的专家角色定位有助于模型生成更专业和规范的需求描述[41]. 本文采用了专家角色定位策略 (system role definition), 通过设定模型扮演资深需求工程师的角色来增强其专业输出能力. 这种角色定位不同于简单的指令设定, 而是通过详细描述专业背景、工作经验和核心能力, 使模型能够更好地理解和执行专业任务, 该部分的核心表述为: “你是一位资深的需求分析师, 具有 10 年以上软件需求工程经验, 精通 ISO/IEC 25010 软件标准”. 此处特别强调了 ISO/IEC 25010 标准的应用, 这一选择基于两个考虑: 首先, 该标准是软件质量领域的权威规

范, 涵盖了完整的质量特性体系; 其次, 通过明确的标准引用, 可以约束模型在规范的质量框架内进行推理.

(2) 任务背景

在任务背景部分, 通过将检索到的相似案例嵌入提示中, 实现了动态提示构造与少样本学习 (few-shot learning) 提示策略. Brown 等人[42]的研究证实, 少量但相关的示例能够有效引导大语言模型进行任务适配. 本文通过检索模块获取的相关案例不仅作为参考样例, 更重要的是作为隐性知识的载体, 帮助模型理解功能需求与非功能需求之间的映射关系. 这种动态知识注入的方式较传统的静态示例更具适应性, 能够根据输入需求的特点提供最相关的参考信息. 任务背景部分核心表述如下.

“基于你的专业知识, 结合一些历史项目经验, 为以下功能需求设计非功能需求:
[新功能需求]: (输入的功能需求描述)
[历史案例]: (检索到的案例列表)”

---

(3) 分析步骤

Wei 等人[43]的研究表明, 通过明晰的步骤引导可以帮助大语言模型形成更系统的推理过程, 采用结构化的任务分解能够帮助大语言模型更好地处理复杂问题. 分析步骤的设计采用了任务分解 (task decomposition) 和思维链 (chain-of-thought) 相结合的方法. 本文设计的分析步骤如下所示.

“请按以下步骤进行分析和生成:
1. 功能场景分析——分析功能特征和使用场景;
2. 质量属性分析——基于 ISO/IEC 25010 评估相关质量属性 (性能效率、可用性、可靠性、安全性等);
3. 经验参考与创新——参考历史经验并结合创新思考;
4. 需求生成——输出规范、可测试的非功能需求.”

---

(4) 输出格式

为了确保生成结果的实用性, 模板在输出部分采用了明确的输出约束, 这样做能够显著提升大语言模型生成内容的规范性和一致性, 输出格式的设计为: “按照列表形式输出非功能需求描述, 确保每条需求具体可度量, 不输出任何其他内容”. 这种设计限制了大语言模型的生成内容范围, 能够使得生成内容更专注于本文所需要的非功能需求, 也更简洁. 除此之外, 这样设计的另一目的是使得输出文本中不包含冗余的文字, 以便于后续通过不同评价指标对其生成质量进行有效评估, 避免无关内容影响评估结果.

(5) 情感激励

情感激励部分的设计旨在通过建立积极的交互氛围, 提升模型的生成质量. 通过在大语言模型的提示中加入情感激励元素, 可以增强模型的责任感和主动性, 促使其在生成非功能需求时更加认真和细致. 情感激励部分的核心表述为: “请记住, 你的分析结果对该软件系统的成败至关重要, 请务必认真分析”, 这种表述不仅强调了任务的重要性, 还通过赋予模型一定的责任感, 激励其生成更高质量的非功能需求. 具体而言, 情感激励的设计围绕两个核心要素展开, 其一, 任务重要性的强调是通过明确阐述分析结果对软件系统成败的深远影响来实现, 这有助于模型深刻认识到任务的紧迫性和重要性, 并在生成过程中分配更多的“注意力”资源. 其二, 责任感的建立则依赖于诸如“请务必认真分析”等措辞的运用, 这些表达方式赋予模型更高的责任感, 促使其在生成过程中表现出更好的谨慎性和细致性, 以避免产生模糊或不准确的需求描述.

综合以上 5 个部分的提示模块, 完整的提示模板如下所示.

“你是一位资深的需求分析师, 具有 10 年以上软件需求工程经验, 精通 ISO/IEC 25010 软件标准. 基于你的专业知识, 结合一些历史项目经验, 为以下功能需求设计非功能需求:
[新功能需求]: (输入的功能需求描述)
[历史案例]: (检索到的案例列表)

请按以下步骤进行分析和生成:

1. 功能场景分析——分析功能特征和使用场景;
2. 质量属性分析——基于 ISO/EC 25010 评估相关质量属性 (性能效率、可用性、可靠性、安全性等);
3. 经验参考与创新——参考历史经验并结合创新思考;
4. 需求生成——输出规范、可测试的非功能需求.

按照列表形式输出非功能需求描述, 确保每条需求具体可度量, 不输出任何其他内容.

请记住, 你的分析结果对该软件系统的成败至关重要, 请务必认真分析!”

4.3 基于大语言模型的需求生成模块

在完成案例检索和提示模板构建后, 生成阶段的核心任务是利用大语言模型生成高质量的非功能需求, 与传统 RAG 方法主要关注事实性知识的整合不同, 此处模型需要在参考历史案例的基础上, 结合新功能需求的特点, 生成既专业又符合实际的非功能需求描述.

基于大语言模型的需求生成模块由 4 个主要组件构成: 输入处理组件、大语言模型生成组件、输出后处理组件和质量评估反馈组件, 这些组件协同工作, 确保生成的非功能需求能够满足软件工程领域的实际需求.

(1) 输入处理组件: 该组件负责接收来自案例检索模块的输入数据, 包括输入的功能需求和检索到的相似历史案例. 输入处理组件会对这些数据进行预处理, 具体包括: 数据格式标准化, 将功能需求和历史案例统一转换为大语言模型可识别的结构化格式; 上下文构建, 将功能需求与检索到的案例按照预定义的提示模板进行组织, 形成完整的输入序列, 确保其符合大语言模型的输入格式要求.

(2) 大语言模型生成组件: 该组件是模块的核心, 负责根据输入的功能需求和检索到的历史案例, 生成非功能需求. 通过结合提示工程策略, 该组件能够引导大语言模型生成高质量的需求描述.

(3) 输出后处理组件: 该组件对生成的非功能需求进行后处理, 具体处理流程包括: 首先, 执行冗余信息清理, 采用正则表达式匹配和文本分析技术, 自动识别并去除大语言模型生成内容中与核心需求描述无关的冗余信息: ① 引导性词汇过滤: 基于预定义的引导性词汇库 (如“基于分析”“综上所述”“总结如下”等), 使用正则表达式模式匹配识别并删除模型生成的过渡性表述. ② 重复表述检测: 采用基于编辑距离的文本相似度算法, 计算生成内容中句子间的相似度, 当相似度超过 0.8 阈值时自动去除重复句子, 保留语义表达更完整的版本. ③ 格式标记清理: 通过正则表达式匹配识别 Markdown 格式标记 (如“**”“##”等)、编号标识符和其他非需求描述的格式化内容并予以删除. 其次, 进行格式规范化处理, 将清理后的需求文本按照预定义的软件需求规范模板进行结构化整理, 确保最终输出的非功能需求能够符合后续不同评估方法的输入标准.

(4) 质量评估反馈组件: 该组件负责对生成的非功能需求进行质量评估, 采用专家评分方式进行多维度评估, 指标包括完整性、准确性、可测试性和一致性. 为确保评估的一致性和可操作性, 本文构建了快速评估机制: 由 2 名软件工程专家组成评估小组, 具体评分标准见第 5.1 节. 本文将各维度的质量阈值设定为 4.0 分 (5 分制), 作为“可接受”质量的最低标准, 约对应 75% 分位数, 既保证了质量要求, 又避免了过于严格导致的效率损失. 当完整性不足时, 温度系数增加 0.1 (最大不超过 0.9), 促进生成内容的多样性和覆盖面; 当准确性不足时, 温度系数降低 0.1, 惩罚系数增加 0.1, 增强术语使用的准确性; 当可测试性不足时, 在提示模板中动态增加“请提供具体的量化指标和验证方法”引导语, 并适当降低温度系数 0.05; 当一致性不足时, 重新执行检索步骤, 增加检索案例数量 k 值 (如从 5 增加到 7), 提供更丰富的上下文参考. 该反馈机制最多执行 3 次迭代, 确保在有限的计算资源下获得满足质量要求的非功能需求. 单个需求的专家评估平均耗时约 3 min, 包括约 1 min 的需求理解和 2 min 的多维度评分. 考虑到评估成本, 本文进一步通过实验探讨迭代次数的选择以尽可能减少人工干预. 在 200 个随机选取的测试样本上, 统计了不同迭代次数下的质量达标情况: 绝大部分样本首次生成即可达标, 第 1 次生成后达标率为 85%, 即绝大部分的样本无需人工干预便已达标; 第 2 次迭代后累计达标率上升至 91%; 第 3 次迭代后达标率为 96%. 继续增加迭代次数, 第 4 次、第 5 次的边际收益分别仅为 2% 和 1%, 但会使计算成本增加 40% 以上. 因此, 如果使

用者对达标率的要求不高 (85% 以下), 则可以不执行反馈机制以节省评估成本以及尽可能快地生成结果; 而如果对质量要求较高, 3 次迭代可以在质量保证与计算效率间达到最优平衡.

在使用 LLM 进行内容生成时模型的选择直接影响生成需求的质量, 本文在第 5 节中通过实验对比分析不同大语言模型的表现, 实验结果显示, 综合而言 Claude-3.5 的生成质量最高, 最适合用于本算法的生成核心, 能够生成相对而言最专业、最全面的非功能需求. Claude-3.5 的表现如此出色可能是因为该模型在预训练阶段已经接触了大量软件工程相关的文本, 具备了一定的领域知识基础; 其次, 模型具有较强的上下文理解能力, 能够准确把握检索增强提供的案例信息; 最后, 模型的生成能力在保持创造性的同时, 也能很好地遵循给定的输出约束.

为使模型在非功能需求生成任务上达到最佳效果, 本文对关键参数进行系统性优化, 其中最为关键的是温度 (temperature) 与惩罚 (penalty) 参数的调节. 通过对温度和惩罚参数这两个系数在 [0.1, 0.9] 区间内的梯度实验发现, 温度系数对生成内容的随机性和多样性影响显著, 当温度系数较高时, 生成结果的内容多样性增加, 容易生成高质量内容. 然而, 过高的随机性可能会引入一些不必要的噪声, 尤其是当惩罚系数取值过高时, 模型容易生成脱离上下文或缺乏准确性的内容. 因此, 将温度系数设置为 0.9, 惩罚系数设置为 0.5 能够在保证需求描述专业性的同时, 产生足够的表达变化. 这种参数配置在保证输出连贯性的同时, 能够有效抑制模型生成冗余或矛盾的需求描述.

5 实验分析

5.1 实验设计

为了验证本文方法的优势, 本文在第 3 节中构建的功能需求-非功能需求结构化关联数据集上进行实验. 在实验设计上, 将数据集按照 8:2 的比例随机采样划分为历史案例库和测试数据集, 也就是总计共 3 000 个功能需求-非功能需求对作为训练集, 剩下的 856 对作为测试集, 以实现训练和测试的有效分离. 历史案例库用于支持方法的检索模块, 而测试数据集则用于评估生成模型的实际效果. 由于是通过随机采样进行划分, 可以认为训练集和测试集中的应用领域与非功能需求类别的分布相同. 同时, 选择了以下 3 个对比基线方法.

(1) 人工设计: 采用传统的需求分析方法, 由经验丰富的需求工程师根据功能需求手动设计非功能需求, 代表传统专家经验下的需求获取效果.

(2) 推荐算法: 基于功能需求与非功能需求之间的历史关联关系, 采用基于内容或协同过滤的推荐算法生成非功能需求, 反映传统机器学习方法的性能表现.

(3) GPT-3.5: 以当前主流的大语言模型 ChatGPT 为核心, 通过提示工程直接生成非功能需求, 代表当前主流生成式 AI 的基础性能.

通过与这些基线方法的系统性对比, 本文旨在验证所提方法在需求完整性、准确性和一致性等关键维度的优势. 为了全面评估非功能需求生成方法的有效性, 本文设计了专家评分和自动评分两类评估指标, 以期多维度验证方法的实用性与优越性.

在专家评分环节, 本文邀请了 10 位具有硕士及以上学位的软件工程专家, 他们均具备丰富的需求工程理论知识和 5 年以上的工程实践经验, 对生成的非功能需求进行多维度评分. 评分标准采用李克特五点量表[44], 得分从最低 1 分到最高 5 分, 具体评分规则如表 2 所示. 为了保障评分的标准与一致性, 在专家评分前进行统一培训, 确保每一位专家对评分标准和评分规则都有细致了解, 并且每一份需求由 3 名专家独立评估, 当一份需求的最高得分与最低得分差异大于 1 分时需要组织专家讨论, 以达成共识. 最后, 计算每一份需求得分的平均分, 并统计所有需求得分的平均分作为整体性能的专家评估得分.

在自动化评分环节, 评价指标分为两个类别: 一种是评估生成的非功能需求与数据集中的真实非功能需求的相似度, 本文选择 BLEU[45], ROUGE[46], METEOR[47]和 BERTScore[48]作为这一类的评价指标.

由于数据集中的真实非功能需求往往也并不全面和标准, 因此另一种评价指标着眼于评估生成的非功能需求的专业性和多样性, 分别用专业性指标和多样性指标来评估. 其中专业性指标 *Term* 评估生成文本中关键的领域术语与可度量的数值指标占比, 用于评判生成文本的专业性, 其计算公式为:

$$Term = \frac{n_t + n_n}{N_w} \tag{4}$$

其中, n_t 表示不同领域术语数量, 通过本文构建的领域术语表进行匹配确定, 该领域术语表由专家手动构建包含 8 个类别共 541 条术语, n_n 表示数值指标数量, 通过正则匹配数值符号确定, N_w 表示生成文本的单词总数. 多样性指标 *Category* 评估生成文本是否涵盖预定义的非功能需求分类 (如性能、安全性、可用性等):

$$Category = \frac{n_c}{N_c} \tag{5}$$

其中, n_c 表示覆盖的分类数, 通过上文提到的领域术语表确定, 即如果匹配到了术语表中一个类别的任一术语, 则认为覆盖了该类, N_c 表示总分类数, 值为 8.

表 2　专家评分标准及评分规则

评估维度	分值	评分标准
需求完整性	1	严重缺失必要的 NFR 要素, 大部分约束条件和质量属性未涉及
	2	缺少多个重要的 NFR 要素, 质量属性描述不完整
	3	基本覆盖主要的 NFR 要素, 但部分约束条件或质量属性描述不足
	4	覆盖大部分必要的 NFR 要素, 仅个别次要属性描述不够完整
	5	完全覆盖所有必要的 NFR 要素, 包含详细的约束条件和质量属性
准确性	1	严重偏离软件工程规范, 专业术语使用混乱或错误
	2	多处不符合软件工程规范, 术语使用不准确
	3	部分符合软件工程规范, 存在少量专业术语使用不当
	4	基本符合软件工程规范, 专业术语使用恰当, 个别表达不够严谨
	5	完全符合软件工程规范, 专业术语使用准确, 表达严谨
可测试性	1	几乎没有定量指标, 要求难以验证和度量
	2	较少要求有定量指标, 大部分指标难以验证
	3	部分要求有定量指标, 部分指标描述模糊
	4	大部分要求有定量指标, 少数指标需要进一步细化
	5	包含明确的定量指标, 所有要求都可被验证和度量
一致性	1	与 FR 严重不一致, 存在明显的逻辑矛盾
	2	与 FR 存在多处不一致, 影响需求理解
	3	与 FR 存在部分不一致, 但不影响整体理解
	4	与 FR 基本保持逻辑一致, 存在极少数不明显的不一致
	5	与 FR 完全保持逻辑一致, 约束条件相互补充
可实现性	1	在现有技术条件下难以实现, 或实现成本过高
	2	大部分要求实现难度大, 需要突破性技术支持
	3	部分要求可实现, 部分要求需要较高技术成本
	4	大部分要求可实现, 少数要求实现难度较大
	5	在现有技术条件下完全可实现, 实现成本合理
明确性	1	表述混乱, 充满歧义, 难以理解
	2	表述不够清晰, 存在较多模糊或歧义的描述
	3	表述基本清晰, 存在少量模糊或歧义的描述
	4	表述较为清晰, 极少存在可能引起歧义的描述
	5	表述清晰准确, 完全无歧义, 易于理解
相关性	1	与系统领域特征几乎无关, 约束条件不适用
	2	与系统领域特征关联度低, 多数约束条件相关性差
	3	基本符合系统领域特征, 部分约束条件相关性不强
	4	大部分符合系统领域特征, 个别约束条件相关性较弱
	5	完全符合系统领域特征, 约束条件高度相关

表 2　专家评分标准及评分规则 (续)

评估维度	分值	评分标准
可用性	1	需求描述完全定制化, 无法在其他系统中复用
	2	需求描述高度定制化, 难以在其他系统中复用
	3	需求描述部分可复用, 需要适度调整才能应用于类似系统
	4	需求描述较为通用, 经细微调整后可应用于类似系统
	5	需求描述具有通用性, 可直接应用于类似系统

鉴于大语言模型生成内容具有固有的随机性特征, 本文对后续所有基于大型语言模型的实验任务均采用 3 次重复实验取算术平均值的方案. 此外, 考虑到高级别大型语言模型的计算资源开销, 除非另有明确说明, 本节中的实验均采用 GPT-3.5 作为基础大模型进行测试.

5.2 实验结果

基于第 5.1 节实验设计, 本节进行本文方法与基线方法在构建数据集上的对比实验与结果评估. 专家评分结果汇总于表 3. 可以看出, 本文非功能需求生成方法在所有维度上均取得了最高得分, 平均得分达到 4.67 分, 显著高于基线方法. 与人工设计方法相比, 本文方法在“完整性”和“可重用性”维度上优势尤为明显, 分别提升了 0.39 分和 0.48 分. 这表明, 通过检索增强生成机制和精心设计的提示工程, 本文方法能够生成更全面且具有更高复用价值的非功能需求. 人工设计虽然在“准确性”方面表现较好 (4.45 分), 但在时效性和覆盖广度上存在一定局限. 推荐算法的平均得分仅为 3.69 分, 在所有方法中表现最弱, 尤其在可测试性 (3.60 分) 和“明确性”(3.70 分) 方面存在显著不足. 此结果揭示了传统推荐算法在处理高语义复杂度任务时的固有局限性, 难以充分利用上下文语义信息, 导致生成的需求缺乏具体测试标准和明确的实现细节. GPT-3.5 直接生成的非功能需求在“准确性”和“一致性”维度表现相对较好 (分别为 4.20 分和 4.05 分), 但在“需求完整性” (4.12 分) 和“可测试性” (3.92 分) 方面仍有明显差距. 这一现象反映了大语言模型虽具有强大的语言生成能力, 但缺乏领域特定的结构化知识引导, 难以产生全面且高质量的专业需求描述. 相比之下, 本文方法通过结合历史案例检索和领域知识提示, 有效解决了上述问题. 在“准确性” (4.75 分) 和“需求完整性” (4.71 分) 两个关键维度上取得了最高分, 显示了检索增强生成 (RAG) 策略在提升生成内容与输入功能需求相关性方面的显著优势. 此外, “可实现性” (4.60 分) 和“可测试性”(4.63 分) 的高分也表明, 生成的需求在实际工程实现和测试环节具有较强的可操作性.

表 3　专家评分结果

方法	完整性	准确性	可测试性	一致性	可实现性	明确性	相关性	可重用性	平均得分
人工设计	4.32	4.45	4.21	4.38	4.19	4.25	4.1	4.22	4.26
推荐算法	3.78	3.85	3.60	3.72	3.58	3.70	3.65	3.69	3.69
GPT-3.5	4.12	4.20	3.92	4.05	3.89	4.01	3.98	4.03	4.02
本文方法	**4.71**	**4.75**	**4.63**	**4.68**	**4.60**	**4.66**	**4.69**	**4.70**	**4.67**

注: 粗体表示该指标下的最优值

除了专家评分之外, 还对本次实验结果进行了自动化评价指标计算, 表 4 展示了本文方法与基线方法在自动化评分指标上的对比结果.

表 4　自动化指标评分结果, 粗体表示该指标下的最优值

方法	BLEU	ROUGE	METEOR	BERTScore	*Term*	*Category*
人工设计	0.482	0.656	0.683	0.914	0.021	0.283
推荐算法	0.084	0.305	0.205	0.873	0.002	0.125
GPT-3.5	0.022	0.199	0.192	0.851	0.059	**0.375**
本文方法	**0.649**	**0.804**	**0.826**	**0.952**	**0.064**	0.292

从文本相似度评估指标来看, 本文方法在各项指标上均表现优异. 在 BLEU 评分上, 本文方法达到 0.649, 远高于其他基线方法; 在 ROUGE 和 METEOR 指标上也呈现相似趋势, 分别达到 0.804 和 0.826, 同样大幅领先于基线

方法. 结果表明, 通过检索增强机制引入历史案例能够有效引导大语言模型生成与标准答案在词序、语法结构和语义表达上更为接近的非功能需求. BERTScore 指标的优异表现 (0.952) 进一步验证了本文方法在语义理解和表达方面的突出能力. 在专业性评估维度, 本文方法的优势体现了检索增强策略在保持领域专业性方面的效果, 通过从历史案例中提取相关专业术语和表达方式, 能够生成更符合软件工程领域规范的需求描述. 在衡量专业性的指标方面, 基于大语言模型的方法普遍优于传统方法, 这一结果反映了大语言模型在专业术语使用方面的内在优势. 值得注意的是, 在多样性指标上, GPT-3.5 以 0.375 的得分表现最佳, 本文方法的 0.292 次之, 而推荐算法的 0.125 表现最弱. 深入分析后本文认为 GPT-3.5 由于其生成机制的多样性, 能够覆盖更广泛的需求类别, 但可能牺牲了与标准答案的一致性; 相比之下, 本文方法通过检索机制引入的历史案例, 虽然提升了生成内容的准确性和专业性, 但在一定程度上限制了生成内容的发散性. 这表明在需求类别的覆盖广度上, 本文方法仍存在进一步优化的空间.

为了更直观地展示本文方法在 NFR 生成完整性方面的优势, 本节选取一个典型的功能需求进行案例分析, 对比不同方法生成的 NFR 在质量属性覆盖方面的差异. 输入功能需求: “The system should support user login via username and password”, 中文: “系统应支持用户通过用户名和密码进行登录”. 表 5 展示了 3 种不同方法针对该功能需求生成的 NFR 结果及其质量属性覆盖分析.

表 5　不同方法的非功能需求生成结果及 *Category* 指标对比

方法	生成的非功能需求	覆盖的质量属性	*Category*
推荐算法	The product shall allow for customization of start page and views preferences	可用性	0.125
GPT-3.5	- The system should have a response time of less than 2 seconds for user login. - User passwords must be encrypted and stored securely in the database. - The login page should have a user-friendly interface with clear instructions for users. - The system should have a backup and recovery mechanism in place to ensure login data is not lost. …	性能效率、安全性、可靠性、可用性	0.5
本文方法	- The system shall comply with corporate user interface standards for login screens - The login page shall be intuitive with 95% of users able to successfully log in on first attempt without training - The maximum response time for login authentication shall not exceed 3 seconds - The system shall support concurrent login of at least 1 000 users - The login functionality shall comply with GDPR and relevant data protection regulations - The login logs should be fully documented for audit purposes …	性能效率、安全性、可靠性、可用性、可维护性	0.625

注: … 表示篇幅限制, 剩余文本不展示

从表 5 中可以看出, 推荐算法表现最弱, 仅覆盖 1 个质量属性, 难以指导实际开发和测试工作. GPT-3.5 展现了较好的多样性, 覆盖 4 个质量属性, 而本文方法通过检索增强机制, 不仅覆盖了 5 个质量属性, 还在每个属性下提供了具体、可测试的需求描述. 特别是在安全性方面, 不仅考虑了数据传输加密, 还包含了具体的数据保护规定; 在可维护性方面, 增加了日志审计需求, 体现了系统运维的考虑. 这一案例分析表明, 本文方法通过引入历史案例知识和结构化提示工程, 能够有效提升 NFR 生成的完整性, 为实际软件项目提供更全面的质量保障指导.

为深入分析模型在不同种类非功能需求生成方面的能力差异, 本文按照 ISO/IEC 25010 标准的 8 个主要质量特征对生成结果进行了分类统计. 表 6 展示了本文方法在不同 NFR 类别上的性能表现统计.

从分类分析结果可以看出, 模型在不同 NFR 类别上的生成能力存在显著差异. 性能效率类需求的生成效果最佳 (专家评分 4.82), 这可能是因为性能需求在训练数据中出现频率最高, 且具有相对标准化的表达模式. 安全性和可用性需求也表现良好, 分别获得 4.75 和 4.71 的专家评分. 相比之下, 可移植性和兼容性需求的生成效果相对较弱, 专家评分分别为 4.29 和 4.38, 这反映了这些需求类别在实际项目中出现频率较低, 且表达方式更加多样化, 增加了自动生成的难度. 覆盖率数据显示, 性能效率类需求的覆盖率达到 63.4%, 而可移植性需求仅为 6.8%, 这进一步证实了模型对常见需求类别的处理能力更强.

综合专家评分和自动化评分结果分析, 本文方法通过检索增强机制有效地融合了历史案例知识与大语言模型

的生成能力, 在保证专业性的同时显著提高了生成质量. 与传统的推荐算法和直接使用大语言模型的方法相比, 本方法充分发挥了两种技术的互补优势: 历史案例提供了领域相关的结构化知识支持, 而大语言模型则贡献了强大的语言理解和生成能力. 这种结合不仅提升了生成内容的质量, 还有效解决了领域适应性问题, 为非功能需求自动化获取提供了一种实用且可靠的解决方案.

表 6　不同 NFR 类别的生成性能对比

NFR类别	专家评分	BLEU	ROUGE	覆盖率 (%)
性能效率	4.82	0.721	0.856	63.4
安全性	4.75	0.689	0.823	61.8
可靠性	4.63	0.654	0.798	38.9
可用性	4.71	0.703	0.834	39.6
可维护性	4.51	0.598	0.751	22.7
兼容性	4.38	0.542	0.712	8.4
功能适用性	4.67	0.675	0.807	15.2
可移植性	4.29	0.498	0.668	6.8

5.3 消融实验

为了深入验证本文提出方法中各核心组件的必要性和有效性, 本节设计了一系列消融实验, 主要分为两个部分: 第 1 部分是模块消融, 主要目的是验证本文所设计的方法架构的有效性与紧凑性, 通过逐一移除或替换关键模块来评估其对整体性能的影响; 第 2 部分是参数敏感性分析, 从检索、增强和生成这 3 个关键环节出发, 通过调整不同的超参数来探究系统性能对参数选择的敏感程度.

5.3.1 模块消融

表 7 列出了模块消融的实验结果, 在所有指标上, 完整模型的表现显著优于消融后的各对比实验, 这充分验证了系统整体架构的高效协作性和各模块的必要性.

表 7　模块消融实验结果

模型配置	BLEU	ROUGE	METEOR	BERTScore	*Term*	*Category*
本文方法	**0.649**	**0.804**	**0.826**	**0.952**	**0.064**	**0.291 667**
移除案例检索模块	0.027	0.207	0.194	0.852	0.061	0.25
移除提示工程模块	0.619	0.797	0.840	0.950	0.063	0.25
移除需求生成模块	0.196	0.412	0.293	0.894	0.019	0.25
移除多样性重排模块	0.497	0.695	0.757	0.937	0.058	0.25
移除LITM重排模块	0.569	0.790	0.825	0.950	0.057	0.25

注: 粗体表示该指标下的最优值

针对具体模块的消融结果, 可以得出以下分析.

(1) 案例检索模块的重要性: 移除案例检索模块后, BLEU 分数从 0.649 骤降至 0.027, ROUGE 和 METEOR 指标也有显著下降. 这表明, 历史案例作为上下文支持对于生成高质量非功能需求至关重要, 该模块通过提供语义相关的历史知识, 显著提升了生成内容与标准答案的一致性.

(2) 提示工程模块的贡献: 移除提示工程模块后, 虽然 METEOR 指标略有上升 (0.840), 但其他指标均有不同程度的下降. 这说明结构化提示模板在引导大语言模型生成领域适配的高质量内容方面发挥了重要作用, 特别是在保持内容与功能需求一致性方面.

(3) 需求生成模块的核心作用: 移除需求生成模块后, 模型将直接使用检索到的案例作为模型输出, 本文模型也就退化成普通的基于相似度计算的推荐算法, 因此导致所有指标显著下降, 尤其是 *Term* 指标从 0.064 降至 0.019, 表明大语言模型在融合历史案例知识并生成专业、准确的非功能需求方面具有不可替代的作用.

(4) 重排模块的优化效果: 移除多样性重排模块和 LITM 重排模块均导致性能下降, 特别是多样性重排模块的移除对 BLEU 和 ROUGE 指标影响更大, 说明这些重排步骤在优化生成内容质量、多样性和相关性方面都具有重

要作用.

从表 7 结果可以看到相对于其他指标, *Category* 非常稳定, 除了完整的本文方法外, 其余实验的结果均为 0.25. 在对每一次实验结果的具体 *Category* 覆盖情况进行分析之后发现, 所有实验都覆盖了性能 Performance 和安全 Security 两类. 我们认为是因为这两类是最基础最常见也是最容易被覆盖到的非功能需求类别, 导致实验结果中 *Category* 指标相对稳定.

5.3.2 参数敏感性分析

在非功能需求生成的整个流程中, 检索步骤是实现上下文信息引入的关键环节, 本文设计了 3 种不同的检索方式进行对比实验来分析检索策略对系统性能的影响, 分别为随机检索 (Random)、密集向量检索 (Dense) 和稀疏向量检索 (Sparse). 表 8 展示了不同检索方式的性能对比结果. 实验结果表明, 密集向量检索是支持高质量非功能需求生成的最佳选择, 在所有相似度指标 (BLEU、ROUGE、METEOR、BERTScore) 上, 密集向量检索均取得了最优成绩, 同时在专业性指标 (*Term*) 上也表现最佳. 这充分验证了基于深度语义的检索方法在捕获需求间复杂语义关系方面的优越性.

表 8 检索方法对比实验结果

方法	BLEU	ROUGE	METEOR	BERTScore	*Term*	*Category*
Random	0.050	0.200	0.261	0.845	0.046	**0.563**
Dense	**0.649**	**0.804**	**0.826**	**0.952**	**0.064**	0.292
Sparse	0.317	0.438	0.461	0.894	0.038	0.250

注: 粗体表示该指标下的最优值

检索步骤中的相似历史案例数量 k 是影响生成结果质量的关键参数, 为探讨其对系统性能的影响, 本实验设置 k 的取值范围为 1–19, 逐步增加每次检索返回的案例数量, 并分析生成结果在不同指标上的表现. 图 5 显示了不同 k 值下的实验结果, 直观可见 k 值的变化对生成质量具有显著影响. 在小范围案例 (如 k=1) 中, 生成结果较为准确但缺乏多样性, 这意味着单一案例能够为生成提供高相关性语义信息, 但上下文的单一性限制了生成结果的多样性和领域知识覆盖. 在中等范围案例 (如 k=9, 11) 时, 系统表现出最佳的平衡性, 因此适量的案例能够提供丰富的上下文信息, 有助于生成结果兼具准确性和多样性. 而在大范围案例 (如 k≥15) 中, 由于过多的历史案例引入了冗余信息, 导致上下文不够聚焦, 从而影响了生成结果的准确性.

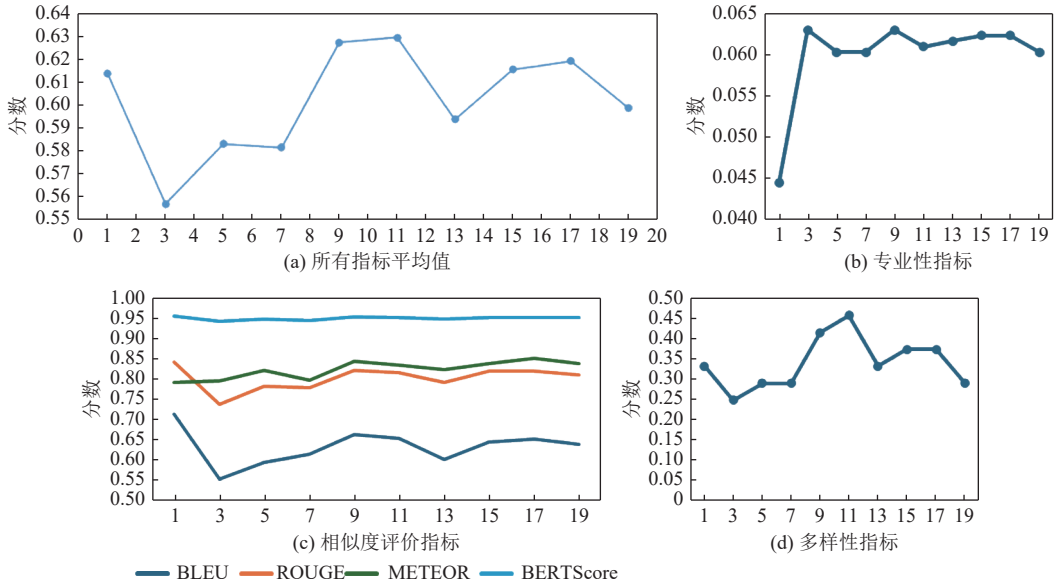

图 5 不同 k 取值的实验结果

值得注意的是, 当 k=1 时, 多样性指标达到相对高值, 这是因为单一历史案例的约束相对较弱, 使得大语言模型更多地依赖其内在的生成能力, 产生跨越多个质量属性类别的多样化内容, 但这种多样性往往缺乏针对性和深度; 而当 k=3, 5, 7 时, 多个相似案例的存在形成了更强的语义约束, 引导模型生成更加聚焦和一致的需求描述, 虽然提高了生成内容的准确性和相关性, 但在一定程度上限制了跨类别的多样性表达; 随着 k 值进一步增大至 9, 11 时, 足够数量的多样化历史案例重新为模型提供了丰富的参考模式, 使多样性指标再次提升. 这一现象反映了检索增强生成中“约束与创新”之间的动态平衡: 适度的案例约束能够提升生成质量, 但过度约束可能限制内容的多样性, 而充分的案例覆盖则能够在保证质量的同时恢复生成的多样性.

为评估提示模板对生成质量的影响, 本文设计了一个系统性实验, 采用 GPT-3.5、GPT-4 和 Claude-3.5 这 3 种主流大语言模型作为生成核心, 并构建了 4 种复杂度递增的提示模板. Prompt1 仅包含基本任务背景和输出格式约束; Prompt2 在此基础上引入角色定位以增强模型任务理解; Prompt3 进一步整合思维链引导模型进行步骤化推理; Prompt4 则通过增加情感激励元素形成完整提示框架. 这种从简到繁的模板设计使我们能够有效评估不同提示元素对生成质量的贡献度及其在不同模型间的普适性. 实验结果如表 9 所示.

表 9　3 种大模型在不同提示模板下的实验结果

大模型	提示模板	专家评分	BLEU	BERTScore	*Term*	*Category*
GPT-3.5	Prompt1	4.325	**0.768**	**0.963**	0.052	0.250
	Prompt2	4.513	0.731	0.960	0.058	0.250
	Prompt3	4.605	0.728	0.959	0.056	0.250
	Prompt4	**4.678**	0.649	0.952	**0.064**	**0.292**
GPT-4	Prompt1	4.457	0.495	0.935	0.064	0.250
	Prompt2	4.577	**0.513**	**0.937**	0.066	0.250
	Prompt3	4.691	0.451	0.928	0.073	0.375
	Prompt4	**4.744**	0.354	0.911	**0.098**	**0.375**
Claude-3.5	Prompt1	4.389	**0.694**	**0.958**	0.058	0.250
	Prompt2	4.593	0.631	0.951	0.066	0.250
	Prompt3	4.765	0.425	0.921	0.082	0.292
	Prompt4	**4.831**	0.184	0.879	**0.163**	**0.750**

注: 粗体表示该指标下此大模型的最优值

随着提示模板复杂度的增加, 专家评分、专业性指标和多样性指标显著增加, 而相似度指标逐步下降. 由此分析, 复杂提示模板能更全面地引导模型生成高质量、专业性更强、内容范围更广的内容, 而简单的提示模板更容易引导模型生成更加简单且与标准答案更接近的内容. 因此可以得出结论, 简单提示更能保证生成内容的相关度, 而复杂提示则更能提升生成内容的深度与广度. 此外, 不同大模型对提示复杂度的响应程度也存在明显差异: GPT-3.5 对提示变化的敏感性相对较低, 各指标变化较为平缓; GPT-4 表现出中等的响应度; 而 Claude-3.5 对提示变化最为敏感, 尤其在专业性和多样性方面的提升最为显著. 这种差异可能与模型的参数规模、训练方法和优化目标有关. 总体而言, Prompt4 虽然在相似度指标上表现较弱, 但在专家评分、专业性和多样性等更贴近实际应用需求的指标上表现最佳, 特别是与 Claude-3.5 模型的组合效果尤为突出. 在实际应用中, 应根据具体任务需求选择适合的提示模板和模型, 以达到最佳的生成效果.

在生成阶段, 为了研究不同大模型在非功能需求生成任务中的性能差异, 本实验选择了 5 种主流大语言模型, 包括 GPT-3.5、GPT-4、Claude-3、Claude-3.5 和 Gemini-2.0, 分别作为需求生成模块的核心组件. 图 6 以柱状图形式展示了 5 种大模型在各评估指标上的表现, 并附加了专家评分结果. 从实验结果来看, 5 种大模型均能胜任非功能需求生成任务, 但在不同性能指标上各有侧重. 具体而言, 相似度表现较高的模型 (如 GPT-3.5) 在专业性和多样性方面略显不足; 而专业性和多样性表现突出的模型 (如 Claude-3.5 和 Gemini-2.0) 虽然在相似度指标上稍逊一筹, 但在专业性和多样性方面表现突出, 这种趋势在专家评分中得到进一步验证. 这种现象可能源于高级大模型更强的语义理解和创新能力, 能够基于输入生成更丰富、多样的内容, 而非简单复制或遵循固定模式. 在实际应用

中, 模型的选择应根据任务的具体需求进行权衡: 若更注重生成内容与输入需求的高一致性, 可以选择相似度表现更高的模型; 而若更注重生成内容的专业性和多样性, 则更复杂、更具领域知识适配能力的模型将是更优的选择.

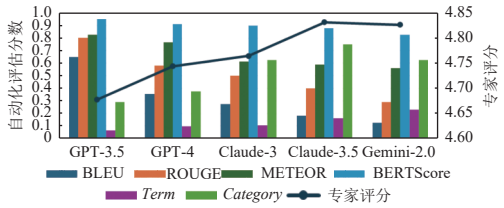

图 6　不同大模型的生成结果对比图

为了进一步探讨大语言模型生成质量对关键参数变化的敏感性, 本文选择了两个重要参数 (温度系数和惩罚系数) 并分别在 5 个不同取值下 (0.1、0.3、0.5、0.7、0.9) 进行交叉组合, 共构成 25 组实验条件, 以分析参数对模型性能的影响规律. 图 7 热力图形式展示了实验结果, 图中数据为所有评估指标的平均值.

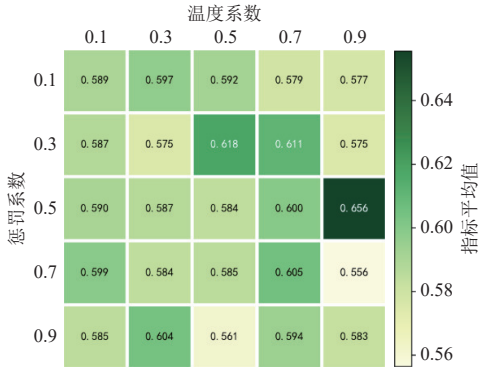

图 7　温度系数与惩罚系数取值组合实验结果热力图

从实验结果可以观察到, 温度系数和惩罚系数的不同取值对模型性能的贡献呈现出一定的非线性特征和交互性. 惩罚系数主要用于减少生成内容的重复性, 较小值时内容流畅但缺乏多样性, 过高值时重复性低但可能脱离上下文, 中等值时效果最佳. 温度系数直接调节生成内容的随机性, 值越高, 生成的内容越多样化但可能缺乏稳定性, 中等范围时能保持语义相关性和丰富性. 与此同时, 图 7 中颜色最深区域出现在高温度系数 (0.9) 和中等惩罚系数 (0.5) 的组合位置, 表明这两个参数存在显著的交互效应. 这一组合使模型能够在保持内容多样性的同时, 有效控制重复和冗余, 达到生成质量的最佳平衡. 因此, 综合实验结果和分析, 模型的生成质量对参数变化高度敏感, 且最佳性能的实现依赖于温度系数与惩罚系数的合理协同调节. 具体而言, 温度系数取值为 0.9, 惩罚系数取值为 0.5 时, 生成结果的综合质量达到最优, 这一结论为后续实现系统的参数调优提供了重要参考, 有助于在实际应用中快速找到最优参数组合, 提升非功能需求生成的效率和质量.

6 总　结

本文构建了包含 22 647 对 FR-NFR 关联关系的结构化关联数据集, 提出了一种结合大语言模型与检索增强生成技术的非功能需求自动生成方法, 设计了由语义案例检索、提示工程和模型生成这 3 个核心模块组成的自动化框架. 实验结果表明, 该方法在专家评分和自动评分指标上均优于现有方法, 证明了 RAG 技术在提升非功能需求生成质量方面的有效性. 未来我们将解决非功能需求的层次性问题, 实现从局部到系统级的非功能需求协同生成; 增强模型在数据稀缺领域的泛化能力; 提高生成内容的可信度与时效性; 改进模型输出的可解释性与透明度. 通过持续改进, 该方法有望为软件工程实践提供更加高效可靠的需求获取支持.

References

[1] Biolchini J, Mian PG, Natali ACC, Travassos GH. Systematic review in software engineering. Technical Report, ES 679/05, COPPE/UFRJ, 2005.

[2] Macaulay LA. Requirements Engineering. London: Springer, 1996. [doi: 10.1007/978-1-4471-1005-7]

[3] Grady JO. System Requirements Analysis. Amsterdam: Elsevier, 2006. [doi: 10.1016/B978-0-12-088514-5.X5000-0]

[4] Chung L, Nixon BA, Yu E, Mylopoulos J. Non-Functional Requirements in Software Engineering. New York: Springer, 2000. [doi: 10.1007/978-1-4615-5269-7]

[5] Ameller D, Ayala C, Cabot J, Franch X. How do software architects consider non-functional requirements: An exploratory study. In: Proc. of the 20th IEEE Int'l Requirements Engineering Conf. Chicago: IEEE, 2012. 41–50. [doi: 10.1109/RE.2012.6345838]

[6] Ullah S, Iqbal M, Khan AM. A survey on issues in non-functional requirements elicitation. In: Proc. of the 2011 Int'l Conf. on Computer Networks and Information Technology. Abbottabad: IEEE, 2011. 333–340. [doi: 10.1109/ICCNIT.2011.6020890]

[7] Kurtanović Z, Maalej W. Automatically classifying functional and non-functional requirements using supervised machine learning. In: Proc. of the 25th IEEE Int'l Requirements Engineering Conf. Lisbon: IEEE, 2017. 490–495. [doi: 10.1109/RE.2017.82]

[8] Rahman K, Ghani A, Misra S, Rahman AU. A deep learning framework for non-functional requirement classification. Scientific Reports, 2024, 14(1): 3216. [doi: 10.1038/s41598-024-52802-0]

[9] Umar MA, Lano K. Advances in automated support for requirements engineering: A systematic literature review. Requirements Engineering, 2024, 29(2): 177–207. [doi: 10.1007/s00766-023-00411-0]

[10] Zhao WX, Zhou K, Li JY, Tang TY, Wang XL, Hou YP, Min YQ, Zhang BC, Zhang JJ, Dong ZC, Du YF, Yang C, Chen YS, Chen ZP, Jiang JH, Ren RY, Li YF, Tang XY, Liu ZK, Liu PY, Nie JY, Wen JR. A survey of large language models. arXiv:2303.18223, 2023.

[11] Zhang Y, Li YF, Cui LY, Cai D, Liu LM, Fu TC, Huang XT, Zhao EB, Zhang Y, Chen YL, Wang LY, Luu AT, Bi W, Shi F, Shi SM. Siren's song in the AI ocean: A survey on hallucination in large language models. Computational Linguistics, 2025: 1–46. [doi: 10.1162/COLI.a.16]

[12] Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, Küttler H, Lewis M, Yih WT, Rocktäschel T, Riedel S, Kiela D. Retrieval-augmented generation for knowledge-intensive NLP Tasks. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 793.

[13] Alsaqaf W, Daneva M, Wieringa R. Agile quality requirements engineering challenges: First results from a case study. In: Proc. of the 2017 ACM/IEEE Int'l Symp. on Empirical Software Engineering and Measurement. Toronto: IEEE, 2017. 454–459. [doi: 10.1109/ESEM.2017.61]

[14] Macasaet R, Chung L, Garrido JL, Noguera M, Rodríguez ML. An agile requirements elicitation approach based on NFRs and business process models for micro-businesses. In: Proc. of the 12th Int'l Conf. on Product Focused Software Development and Process Improvement. Torre Canne Brindisi: Association for Computing Machinery, 2011. 50–56. [doi: 10.1145/2181101.2181114]

[15] Younas M, Jawawi DNA, Ghani I, Kazmi R. Non-functional requirements elicitation guideline for agile methods. Journal of Telecommunication, Electronic and Computer Engineering, 2017, 9(3-4): 137–142.

[16] Käpyaho M, Kauppinen M. Agile requirements engineering with prototyping: A case study. In: Proc. of the 23rd IEEE Int'l Requirements Engineering Conf. Ottawa: IEEE, 2015. 334–343. [doi: 10.1109/RE.2015.7320450]

[17] Kopczyńska S, Ochodek M, Nawrocki J. On importance of non-functional requirements in agile software projects—A survey. In: Jarzabek S, Poniszewska-Marańda A, Madeyski L, eds. Integrating Research and Practice in Software Engineering. Cham: Springer, 2020. 145–158. [doi: 10.1007/978-3-030-26574-8_11]

[18] Nawrocki J, Ochodek M, Jurkiewicz J, Kopczyńska S, Alchimowicz B. Agile requirements engineering: A research perspective. In: Proc. of the 40th Int'l Conf. on Current Trends in Theory and Practice of Computer Science. Nový Smokovec: Springer, 2014. 40–51. [doi: 10.1007/978-3-319-04298-5_5]

[19] de Gooijer T. Discover quality requirements with the mini-QAW. In: Proc. of the 2017 IEEE Int'l Conf. on Software Architecture Workshops. Gothenburg: IEEE, 2017. 196–198. [doi: 10.1109/ICSAW.2017.52]

[20] Kopczyńska S, Nawrocki J. Using non-functional requirements templates for elicitation: A case study. In: Proc. of the 4th IEEE Int'l Workshop on Requirements Patterns. Karlskrona: IEEE, 2014. 47–54. [doi: 10.1109/RePa.2014.6894844]

[21] Dragicevic S, Celar S, Novak L. Use of method for elicitation, documentation, and validation of software user requirements (MEDoV) in agile software development projects. In: Proc. of the 6th Int'l Conf. on Computational Intelligence, Communication Systems and Networks. Tetova: IEEE, 2014. 65–70. [doi: 10.1109/CICSyN.2014.27]

[22] Maiti RR, Mitropoulos FJ. Capturing, eliciting, and prioritizing (CEP) NFRs in agile software engineering. In: Proc. of the 2017 SoutheastCon. Concord: IEEE, 2017. 1–7. [doi: 10.1109/SECON.2017.7925365]

[23] Dongmo C. A review of non-functional requirements analysis throughout the SDLC. Computers, 2024, 13(12): 308. [doi: 10.3390/computers13120308]

[24] Rashwan A, Ormandjieva O, Witte R. Ontology-based classification of non-functional requirements in software specifications: A new corpus and SVM-based classifier. In: Proc. of the 37th Annual IEEE Computer Software and Applications Conf. Kyoto: IEEE, 2013. 381–386. [doi: 10.1109/COMPSAC.2013.64]

[25] Kaur K, Kaur P. BERT-CNN: Improving BERT for requirements classification using CNN. Procedia Computer Science, 2023, 218: 2604–2611. [doi: 10.1016/j.procs.2023.01.234]

[26] Rahman MA, Haque MA, Tawhid MNA, Siddik MS. Classifying non-functional requirements using RNN variants for quality software development. In: Proc. of the 3rd ACM SIGSOFT Int'l Workshop on Machine Learning Techniques for Software Quality Evaluation. Tallinn: ACM, 2019. 25–30. [doi: 10.1145/3340482.3342745]

[27] Yahya AE, Gharbi A, Yafooz WMS, Al-Dhaqm A. A novel hybrid deep learning model for detecting and classifying non-functional requirements of mobile apps issues. Electronics, 2023, 12(5): 1258. [doi: 10.3390/electronics12051258]

[28] Mylopoulos J, Chung L, Yu E. From object-oriented to goal-oriented requirements analysis. Communications of the ACM, 1999, 42(1): 31–37. [doi: 10.1145/291469.293165]

[29] Rahman MM, Ripon S. Elicitation and modeling non-functional requirements—A POS case study. arXiv:1403.1936, 2014.

[30] Ramos F, Costa A, Perkusich M, Almeida H, Perkusich A. A non-functional requirements recommendation system for scrum-based projects. In: Proc. of the 30th Int'l Conf. on Software Engineering and Knowledge Engineering. Redwood City: KSI Research Inc. and Knowledge Systems Institute Graduate School, 2018. 149–148. [doi: 10.18293/SEKE2018-107]

[31] Pohl K. Requirements Engineering: An Overview. Aachen: RWTH, Fachgruppe Informatik, 1996.

[32] Glinz M. On non-functional requirements. In: Proc. of the 15th IEEE Int'l Requirements Engineering Conf. Delhi: IEEE, 2007. 21–26. [doi: 10.1109/RE.2007.45]

[33] IEEE. 830-1998—IEEE Recommended practice for software requirements specifications. IEEE, 1998. 1–40. [doi: 10.1109/IEEESTD.1998.88286]

[34] ISO. ISO/IEC 25010:2011 Systems and software engineering—Systems and software quality requirements and evaluation (SQuaRE)—System and software quality models. Geneva: Int'l Organization for Standardization, ISO, 2011.

[35] Lazaridou A, Kuncoro A, Gribovskaya E, Agrawal D, Liška A, Terzi T, Gimenez M, de Masson d'Autume C, Kocisky T, Ruder S, Yogatama D, Cao K, Young S, Blunsom P. Mind the gap: Assessing temporal generalization in neural language models. In: Proc. of the 35th Int'l Conf. on Neural Information Processing Systems. Curran Associates Inc., 2021. 2247.

[36] Cuskley C, Woods R, Flaherty M. The limitations of large language models for understanding human language and cognition. Open Mind, 2024, 8: 1058–1083. [doi: 10.1162/opmi_a_00160]

[37] Huang L, Yu WJ, Ma WT, Zhong WH, Feng ZY, Wang HT, Chen QL, Peng WH, Feng XC, Qin B, Liu T. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. ACM Trans. on Information Systems, 2025, 43(2): 42. [doi: 10.1145/3703155]

[38] Fleiss JL. Measuring nominal scale agreement among many raters. Psychological Bulletin, 1971, 76(5): 378–382. [doi: 10.1037/h0031619]

[39] Aamodt A, Plaza E. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 1994, 7(1): 39–59. [doi: 10.3233/AIC-1994-7104]

[40] Liu NF, Lin K, Hewitt J, Paranjape A, Bevilacqua M, Petroni F, Liang P. Lost in the middle: How language models use long contexts. Trans. of the Association for Computational Linguistics, 2024, 12: 157–173. [doi: 10.1162/tacl_a_00638]

[41] Shanahan M, McDonell K, Reynolds L. Role play with large language models. Nature, 2023, 623(7987): 493–498. [doi: 10.1038/s41586-023-06647-8]

[42] Brown TB, Mann B, Ryder N, *et al*. Language models are few-shot learners. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 159.

[43] Wei J, Wang XZ, Schuurmans D, Bosma M, Ichter B, Xia F, Chi EH, Le QV, Zhou D. Chain-of-thought prompting elicits reasoning in large language models. In: Proc. of the 36th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2022. 1800.

[44] Likert R. A technique for the measurement of attitudes. Archives of Psychology, 1932, 22(140): 5–55.

[45] Papineni K, Roukos S, Ward T, Zhu WJ. BLEU: A method for automatic evaluation of machine translation. In: Proc. of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia: ACL, 2002. 311–318. [doi: 10.3115/1073083.1073135]

[46] Lin CY. ROUGE: A package for automatic evaluation of summaries. In: Proc. of the 2024 Text Summarization Branches Out. Barcelona:

ACL, 2004. 74–81.

[47] Lavie A, Agarwal A. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In: Proc. of the 2nd Workshop on Statistical Machine Translation. Prague: ACL, 2007. 228–231.

[48] Zhang TY, Kishore V, Wu F, Weinberger KQ, Artzi Y. BERTScore: Evaluating text generation with BERT. In: Proc. of the 8th Int'l Conf. on Learning Representations. Addis Ababa: OpenReview.net, 2020.

作者简介

欧阳柳波, 博士, 副教授, 博士生导师, CCF 高级会员, 主要研究领域为软件需求工程, 知识工程, 自然语言处理.

叶巧莹, 硕士生, CCF 学生会员, 主要研究领域为软件需求工程, 自然语言处理.

孟心如, 硕士生, CCF 学生会员, 主要研究领域为软件需求工程, 自然语言处理.

杜漫茹, 硕士生, 主要研究领域为软件需求工程, 自然语言处理.