面向功能安全软件需求提取的模型驱动提示词生成与优化方法

邵志钧[1)] 吴 际[1)] 曹鸿宇[1)] 王彦伟[1)] 孙 青[1)] 杨海燕[1)]
高艳华[2)] 徐 健[2)]

[1)](北京航空航天大学计算机学院 北京 100191)
[2)](北京控制与电子技术研究所 北京 100038)

摘 要 提取功能安全需求(Functional Safety Requirement)是安全关键软件(Safety-Critical Software)开发的重要步骤,对软件功能安全性有着重要的影响。一直以来,安全需求提取由系统工程师和软件工程师以协作方式人工完成,高度依赖于他们所掌握的领域知识和经验,面临着遗漏的风险,同时周期也比较长。因此,降低安全需求提取的遗漏风险和提高提取效率是一个重要的研究问题。本文提出一种模型驱动的大语言模型(Large Language Model,LLM)提示词生成与优化方法,分析安全需求提取所依赖的领域知识形成元模型,并建立提示词框架指导提示词的构建,在此基础上通过自动生成的零样本提示词引导LLM提取文本的安全需求特征,然后据此从历史安全需求中为少样本提示词选取相似案例,引导大语言模型结合领域知识和历史经验进行安全需求提取。本文在三个安全关键领域的需求案例上对本文所提方法的安全需求提取性能与现有需求提取方法和主流检索增强生成(Retrieval Augmented Generation,RAG)方法进行了对比评估和消融实验,探究了基础LLM、示例数量、相似度案例选择策略、应用场景、方法设计对本文方法的性能影响,并进一步开展案例研究探究了本文方法的实践效果。结果表明,本文方法可以有效向LLM提供安全需求方面的领域知识与经验,从而获得更优的安全需求提取效果,在所选的三个中文LLM上相比于现有基于LLM的需求提取方法,$F1$值提升最高可达25.75%,相比于基于TF-IDF和语义相似度的RAG方法在提取系统功能和软件安全信息上有更好的效果,$F1$值提升最高分别可达5.18%和6.14%。

关键词 安全关键软件;安全需求提取;模型驱动工程;提示工程;大语言模型
中图法分类号 TP311 **DOI号** 10.11897/SP.J.1016.2025.02752

Model-Driven Prompt Generation and Optimization Method for Functional Safety Software Requirements Extraction

SHAO Zhi-Jun[1)] WU Ji[1)] CAO Hong-Yu[1)] WANG Yan-Wei[1)] SUN Qing[1)]
YANG Hai-Yan[1)] GAO Yan-Hua[2)] XU Jian[2)]

[1)](School of Computer Science and Engineering, Beihang University, Beijing 100191)
[2)](Beijing Institute of Control and Electronic Technology, Beijing 100038)

Abstract Functional safety requirements extraction is a crucial and foundational step in the development lifecycle of safety-critical software systems, directly influencing the functional safety and overall reliability of such systems. Functional safety requirements serve as the cornerstone for designing, implementing, and verifying software components that must operate safely under

收稿日期:2025-01-17;在线发布日期:2025-07-30。**邵志钧**,博士研究生,主要研究领域为软件工程、安全关键软件建模。E-mail:zjshao@buaa.edu.cn。**吴 际(通信作者)**,博士,副教授,主要研究领域为安全关键软件建模、安全关键软件验证与测试。E-mail:wuji@buaa.edu.cn。**曹鸿宇**,硕士研究生,主要研究领域为软件工程、软件建模与验证。**王彦伟**,硕士研究生,主要研究领域为软件工程、软件测试。**孙 青**,博士,副教授,主要研究领域为软件工程、个性化学习辅助技术。**杨海燕**,硕士,讲师,主要研究领域为软件工程。**高艳华**,硕士,研究员,主要研究领域为软件工程、软件安全性。**徐 健**,硕士,研究员,主要研究领域为安全关键嵌入式软件、软件安全性。

all conditions, especially in domains where failures can lead to catastrophic consequences, such as automotive, aerospace, and medical devices. Traditionally, the extraction of safety requirements has been a labor-intensive, manual process carried out collaboratively by system engineers and software engineers. This approach heavily depends on the engineers' domain-specific knowledge and accumulated experience, making it not only time-consuming but also prone to the risk of omissions, which can undermine the quality of the final requirements. Given these challenges, reducing the risk of omissions and enhancing the efficiency and comprehensiveness of the safety requirement extraction process have become pressing and significant research topics. In recent years, Large Language Models (LLMs) have demonstrated strong capabilities in natural language understanding and generation, suggesting new opportunities for automating safety requirements extraction. However, leveraging LLMs effectively in this context requires addressing the challenge of adequately infusing them with domain-specific knowledge and practical experience, especially given the nuanced and complex nature of safety requirements. To address these issues, this paper proposes a novel, model-driven approach for the generation and optimization of prompts for LLMs, specifically tailored to the extraction of functional safety requirements. By metamodeling the domain knowledge necessary for safety requirements extraction, this method established prompt frameworks to guide the construction of prompt, and uses LLMs to extract the safety requirement features of natural language text with automatically generated zero-shot prompts. Based on the extracted safety requirements features, this method identifies similar cases from historical safety requirements extraction results to generate few-shot prompts, which guides LLMs to combine domain knowledge and historical experience for safety requirements extraction. This paper conducts a comprehensive evaluation of the safety requirements extraction performance of the proposed method in comparison with existing requirements extraction methods and mainstream Retrieval Augmented Generation (RAG) methods on three cases from safety-critical domain requirements practice, and explores the impacts of the underlying LLM, the number of examples included in the few-shot prompt, the example selection strategy, the application scenarios on the performance of the proposed method. This paper also conducts ablation experiments to investigate the impact of method design on performance of the proposed method, and further conducts a case study to explore the practice effectiveness of the method. The results show that the proposed method can effectively provide domain knowledge and experience to LLMs, thus obtaining better safety requirements extraction results, with $F1$-score enhancements of up to 25.75% on the three selected Chinese LLMs compared to the existing LLM-based requirements extraction method. When compared with TF-IDF and semantic similarity-based RAG methods, the proposed method has better results in extracting system functionality and software safety information, with $F1$-score enhancements of up to 5.18% and 6.14%, respectively. These results demonstrate that our approach can effectively bridge the gap between domain knowledge and LLM capabilities, offering a promising direction for automating and improving safety requirement engineering in safety-critical software domains.

Keywords　safety-critical software; safety requirements extraction; model-driven engineering; prompt engineering; large language model

1 引 言

安全关键系统(Safety-Critical System)中软件的非预期行为可能会导致系统失效,对环境、财产和人员造成严重损害[1-2]。因此,包括航空航天、汽车在内的安全关键领域,要求软件系统具备完善的安全性设计以确保系统的安全运行。安全性评估标准ARP4761A[3]和 MIL-STD-882E[4]明确提出,功能安全需要全面考虑潜在的风险,并开展针对性的安全需求分析,即根据历史安全信息以及已知的系统设计信息,对软件和系统存在的安全风险进行全面识别并指定相应的安全需求[5]。对于规模庞大、结构复杂的安全关键系统来说,其安全风险涉及大量相互关联的影响因素,例如异质组件交互、复杂的时序关系等。因此,软件安全需求分析需要基于历史和已有系统资料中包括系统架构、系统功能在内的关键信息进行分析,实践中这些信息的识别和整理通常依赖于系统工程师和软件工程师的经验与协作。

当前工业实践中依照的 DO-178C[6]、ARP4761A[3]等相关标准,以及广泛应用的 PHA(Preliminary Hazard Analysis, PHA)[7]、FMEA(Failure Mode and Effect Analysis, FMEA)[8]、FTA(Fault Tree Analysis, FTA)[9]等软件安全需求分析方法,缺乏对提取安全需求分析所需信息的系统指导,因此实践中需要工程师阅读文档并结合自身的领域知识和经验对安全需求信息进行梳理和提取。为降低安全需求提取的难度,软件工程领域提出了一系列基于用例模板[10-11]、半形式化模型[12-14]的安全需求提取方法,这些方法使用受限自然语言模板和图形化模型引导用户捕捉关键的安全相关信息,但仍然需要用户对相关文档进行阅读并手动分析和识别相关信息,面对规模庞大的系统时,仍然存在遗漏的问题。近年来,随着机器学习和自然语言处理技术的发展,一批针对一般软件需求和软件信息安全(Security)需求的自动化提取方法被提出[15-17]。但由于功能安全需求的语料数据极其稀缺,导致难以应用机器学习方法来自动识别和提取安全需求[18-19]。总的来说,安全需求提取目前依赖人工阅读梳理,提取质量与工程师的领域知识和经验高度相关。因此,实践中容易出现耗时长且存在遗漏安全需求的风险。如何降低安全需求遗漏风险并提高提取过程的效率是一个亟待解决的问题。

最近,以 ChatGPT[20]、LLaMA[21]、Qwen[22]为代表的一系列大语言模型(Large Language Model, LLM)发展迅猛。LLM 基于海量语料进行预训练,有很强的自然语言处理能力和丰富的人类知识,同时具有开箱即用的特点[23],为安全需求提取任务提供了自动化的解决机会。现有很多研究通过提示学习的方式指导大语言模型转变为领域专家,承担测试[24]、建模[25-27]等工作。作为 LLM 的重要输入,提示词质量对 LLM 完成任务的性能有很大的影响[23]。实践中,要构造高质量提示词需要对 LLM 和任务本身都有深入理解,并在构造过程中多次迭代,这提高了将 LLM 应用于各类任务中的难度[28]。对于诸如软件缺陷复现[29]、领域建模[23,25]、安全需求提取等高度依赖领域知识与经验的任务,由于 LLM 缺乏特定领域的相关知识[30-31],通常需要使用如少样本提示[20]、思维链提示[32]等方法来构造更有针对性的提示词,通过示例将任务思路、领域知识等信息提供给 LLM 以更好地完成任务[23]。因此,对于安全需求提取任务来说,如何将提示工程和安全关键软件两方面知识相结合以构造高质量提示词是个关键问题。

为此,本文提出一种面向功能安全软件需求提取的模型驱动提示词生成与优化方法,基于表示为元模型的安全需求提取所涉及的相关概念和提示工程领域知识,通过提示词框架捕捉相关信息生成提示词以指导 LLM 进行安全需求提取。该方法首先通过零样本提示词引导 LLM 获取安全需求特征,然后据此从历史安全需求中识别相似案例为少样本提示词选取切合的示例,引导 LLM 结合领域知识与历史经验完成软件安全需求的提取。该方法降低了对使用者具备安全关键软件领域知识的要求,因而也降低了因工程师缺乏相应领域知识而遗漏安全需求的风险,在保证安全需求提取质量的同时提高了需求提取的效率。

为验证方法效果,本文整理了包含 3 个安全关键领域需求案例的数据集,基于数据集在 3 个基础 LLM 上对本文方法的安全需求提取效果和性能影响因素进行探究,并与当前先进的需求信息提取方法和主流检索增强生成(Retrieval Augmented Generation, RAG)方法进行了对比实验。实验结果显示本文方法能够有效提取文本中的安全需求信息,且在提取功能和安全方面信息有着更优的性能。本文进一步针对 AFTI/F-16 飞行控制系统开展案例研究,通过定性分析探究了本文方法所提取的安全

需求信息与工程师实践关注点的一致程度,证明了本文方法在实践中的可行性。

本文的主要贡献包括 4 个方面:

(1)针对安全需求提取主要依赖于手工方式的问题,提出了一个基于 LLM 的自动化方法,自动从自然语言需求文本中提取安全需求信息,降低了人力投入;

(2)针对安全需求提取极大依赖于领域知识的问题,提出了一种模型驱动的提示词自动构造方法,将提示工程领域知识和安全需求相关概念整合为安全需求提取提示词元模型,并在此基础上使用提示词框架建模和生成提示词,降低了安全需求遗漏的风险;

(3)选择合适的示例是构造高质量少样本提示词的关键,本文提出了一个基于安全需求特征相似度的历史安全需求案例检索方法,可以为少样本提示词筛选安全特征角度的相似示例,使其具有更好的引导效果;

(4)整理了包含 3 个安全关键领域的中文需求数据集①,包含 42 条涵盖了系统功能、系统架构、安全性方面信息的需求文本,每条需求均由领域专家人工对其中的安全需求信息进行了识别标注,可开放给其他研究使用。

本文其余部分组织如下:第 2 节介绍本文方法的预备知识;第 3 节详细介绍本文提出的面向功能安全软件需求提取的模型驱动提示词生成与优化方法;第 4 节介绍本文所整理数据集以及评估实验设计;第 5 节对评估实验结果进行分析和讨论;第 6 节介绍案例研究设计及结果;第 7 节讨论本文的有效性分析和局限性;第 8 节对相关研究进行总结;第 9 节对全文进行总结并介绍未来工作。

2　预备知识

本节从安全系统工程、软件安全需求、LLM、提示工程以及元模型五个方面介绍预备知识。

2.1　安全系统工程

安全系统工程作为系统工程的子领域[33],应用系统工程原理、标准和技术识别危害,并消除危害或降低危害的影响[4]。安全系统工程通过危害识别、风险评估、安全性设计技术,识别、分析、评价潜在危险,并对系统设计等进行调整,从而实现安全目标,保证系统在生命周期内的安全运行[2]。

当前航空、汽车等安全关键系统逐渐由硬件主导转向软件主导[2]。此类系统中的软件以传感器数据作为输入进行计算,基于计算结果控制作动器与外界进行交互[34]。在该过程中,软件的非预期行为可能会通过作动器等设备间接导致危险。因此,软件安全是当前此类系统安全工程任务的重点[1-2]。

2.2　软件安全需求

软件安全需求是软件安全设计的重要依据,通常在安全系统工程初期被识别[35]。本文关注于功能安全软件需求,不同于一般软件需求关注于定义系统必须满足的功能和约束[36],软件安全需求关注于对软件行为进行约束,从而确保系统和软件不会出现不安全行为而导致严重后果[37]。

软件安全需求与一般软件需求在获取方式上也存在差异。一般软件需求获取通常需要开发人员与利益相关者通过访谈、讨论等方式进行沟通从而确定软件所需满足的功能和特性[38-39]。反之,软件安全需求关注于识别不期望系统出现的行为,同时限制或消除危害影响[37],其获取过程需要从系统架构、系统功能需求、系统危险分析等信息中识别组件、接口、任务等信息,并在此基础上依据相关标准,以人工方式结合历史经验识别潜在失效,分析软件安全需求以降低危害发生的可能性或限制危害的影响。基于以上分析,软件安全需求的具体获取流程可被归纳为图 1。本文关注于图中的第 1 阶段,即从输入信息中系统提取安全需求相关信息从而支撑后续的安全需求分析任务。

2.3　大语言模型

LLM 作为预训练语言模型(Pre-trained Language Model, PLM)[40],其兴起最早可以追溯到 Transformer 模型[41]的提出,Transformer 模型凭借其自注意力机制和强大的并行计算能力,可以有效处理输入序列中的长距离依赖关系,从而捕捉文本中复杂的语义关系,为之后 BERT[42]、GPT[20]②③等 PLM 的提出提供了基础。凭借着强大的自然语言处理能力,PLM 逐渐成为自然语言处理(Natural Language Processing, NLP)任务的主流方法。LLM 在 PLM 基础上进一步扩大了训练语料数据规模及模型的参数规模[43-44],显著提升了自然语言处理能力。

近期,以 ChatGPT[20]和 LLaMA 系列[21]模型

① Chinese safety requirement elicitation dataset, https://zenodo.org/ records/14409179 2024,12,12

② Language Models are Unsupervised Multitask Learners, https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf 2024,12,12

③ Improving Language Understanding by Generative Pre-Training, https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf 2024,12,12

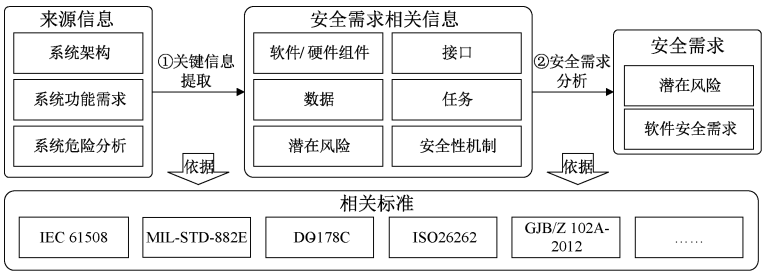


图 1 软件安全需求获取任务流程示意图

为代表的生成式 LLM 进一步提升了性能，拥有强大的自然语言理解和生成能力，通过提示词实现与用户的实时交互，已开始在代码生成[45]、软件建模[23,25-27]等一系列软件工程任务得到了应用。

2.4 提示工程

LLM 在特定任务上的性能与提示词质量高度相关。通过构造和优化提示词，在不调整模型参数和重新训练模型的前提下，实现对 LLM 的微调，从而优化其任务性能，这种微调方式被称为提示工程。

目前有多种提示工程方法[46]，广泛使用的主要有零样本（Zero-shot）提示、少样本（Few-shot）提示[20]和思维链（Chain-of-Thought，CoT）提示[32]。其中，零样本提示仅向 LLM 提供任务的文本描述，引导模型使用其内在的知识来完成任务；少样本提示则向 LLM 提供带有标注的示例，增强模型对给定任务的理解，这种方法可以提高 LLM 解决复杂问题的能力；CoT 提示则用于解决步骤繁多的复杂任务，将问题按推理步骤分解为多个子问题，使 LLM 学习问题的求解过程来解决复杂问题。

2.5 元模型

在软件工程中，元模型（Metamodel）是模型的更高层次抽象[47]，通常被用于定义领域模型的约束、规则和概念及之间的关系[48]。在相关研究和实践中，元模型也常被用于建模领域概念及关系，为后续任务提供概念框架[10-11,49]。从本文角度来说，在安全需求提取任务上工程师所掌握的领域知识因人而异，而 LLM 又缺乏特定领域的知识，因此本文使用元模型固化安全需求提取任务 LLM 所需的领域知识，从而为指导 LLM 进行安全需求提取提供提示词构建支持。

3 面向安全需求提取的提示词生成与优化

本节面向安全需求提取任务，提出模型驱动的提示词生成与优化方法。首先介绍方法的总体框架，然后对面向安全需求提取任务的提示词生成方法和基于安全需求特征的 RAG 方法进行介绍。

3.1 方法框架概述

安全需求提取任务对工程师的领域知识和经验都提出了很高的要求，否则就容易出现遗漏的问题，本文提出一种模型驱动的安全需求提取提示词生成与优化方法，方法关注于将领域知识和经验结合到提示词中，引导 LLM 系统地从包含系统架构信息、系统功能需求、系统危险分析在内的自然语言文本输入信息中提取组件、接口、数据等安全需求信息。方法包括面向安全需求提取任务的提示词生成、基于安全需求特征检索增强生成的安全需求提取两个关键步骤，整体流程如图 2 所示。

首先，本文对安全关键软件安全概念进行分析建立安全概念元模型，安全概念元模型关注于覆盖安全需求提取任务中需要关注的信息，并系统刻画信息之间存在的关联关系。进一步，为有效指导 LLM 提取安全需求信息，基于提示工程经验和安全概念元模型，针对安全需求提取提示词中所需包含的信息建立概念提取提示词元模型，通过将该元模型与安全概念元模型整合为安全需求提取提示词元模型并建立提示词框架，为安全需求提取提示词生成奠定基础。

其次，为了向 LLM 提供安全需求提取任务所需的领域知识，设计了一种基于安全需求特征相似度的 RAG 方法，通过检索历史案例为少样本提示词选取切合的示例以提升 LLM 的安全需求提取性能。方法通过 LLM 获取目标需求文本的安全需求特征，计算目标需求文本和历史案例在安全需求特征上的相似度，筛选与目标需求文本安全特征最为相似的历史安全需求，将历史安全需求及其来源需求文本作为少样本提示词示例，指导 LLM 进行安全需求提取，从而得到高质量的安全需求信息。

当前在提示工程领域，已有多个研究关注于优化少样本提示和 CoT 提示的示例选取来提升 LLM 的性能[50]，但都是关注于逻辑推理问题，不涉及特

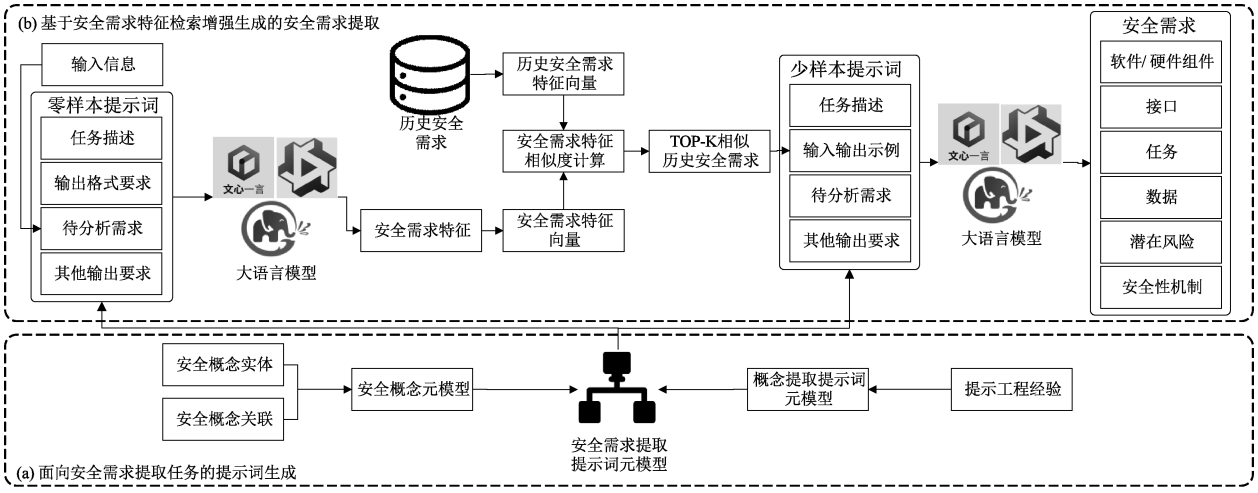


图 2　面向功能安全软件需求提取的模型驱动提示词生成与优化方法研究框架

定领域的知识。而对安全关键软件来说,安全需求涉及功能、架构、安全等多方面的知识。因此,本文需要解决如何给 LLM 提供所需的领域知识与经验来提取相应的安全需求信息。本文首先使用零样本提示指导 LLM 提取安全需求特征,进而据此筛选相似度高的历史安全需求及其来源文本作为示例来构造少样本提示。该方法避免了无关示例对 LLM 造成的误导,同时通过自动化的提示词生成和案例筛选也降低了使用 LLM 进行安全需求提取的难度。

3.2　面向安全需求提取任务的提示词生成

本节对安全关键软件的安全领域概念分析、安全需求提取提示词元模型、安全需求提取提示词的建模和生成方法进行介绍。

3.2.1　安全概念分析

提取和分析软件安全需求通常需要以系统信息为输入[6,37]并围绕功能需求开展[51],而系统功能需求通常基于系统架构定义。因此,本文从系统架构和系统功能两个角度识别软件安全需求。

系统架构定义了系统组件之间的结构与关系。Leveson 指出分析软件需要结合系统上下文开展[52],包括计算机硬件和系统内的其他组件[37],同时组件间交互所使用的硬件接口也是安全关键系统安全分析的重要关注[53]。因此,本文在系统架构角度识别得到软件组件、硬件组件、组件、硬件接口四个关键概念,其中组件包括软件组件和硬件组件两类。

系统功能明确了系统所需完成的任务[37,53],在安全关键系统中功能通常涉及交互过程中组件的行为和数据交互[51],因此,本文系统功能角度关注于任务和数据。

Leveson 提出软件安全需要从系统层面出发,明确系统中的潜在故障并提供消除或控制危险的安全性机制[37]。因此,本文在软件安全需求角度关注于故障和安全性机制两个概念。

基于以上分析,本文从系统架构、系统功能、软件安全需求三个角度共梳理得到八个重要的相关概念。由于安全需求提取是安全关键系统开发的关键步骤,需要严格依照相关标准开展。为保证所识别概念符合相关标准,本文进一步针对通用标准(IEC 61508[54])和来自航空航天(DO-178C[6]、NASA-STD-8719.13C[55]、ARP4761A[3])、汽车(ISO26262[56])、装备(MIL-STD-882E[4])三个安全关键领域的相关标准进行分析调研,确认所识别概念也是相关标准中安全需求的关注重点。

基于系统架构、系统功能、软件安全需求三个方面梳理得到的八个概念,本文建立了如图 3 所示的安全概念领域模型,该模型依照统一建模语言(Unified Modeling Language,UML)类图规范构建。模型中包含系统架构、系统功能、软件安全需求三类概念,其中系统架构定义了系统功能的设计结构,安全需求识别了系统架构和功能层面潜在的故障,并定义了相应的安全性机制以应对故障的出现。

系统架构从系统中组件交互的角度捕捉了软硬件实体以及信息交互所使用的硬件接口。其中,组件可分为软件组件和硬件组件两类,软件组件运行于硬件组件之上,通过计算实现对硬件组件行为的控制,而硬件组件则为软件组件提供其正常交互运行所需的硬件接口,硬件组件之间的通信需要通过硬件接口进行。在安全关键系统中,功能通常以任

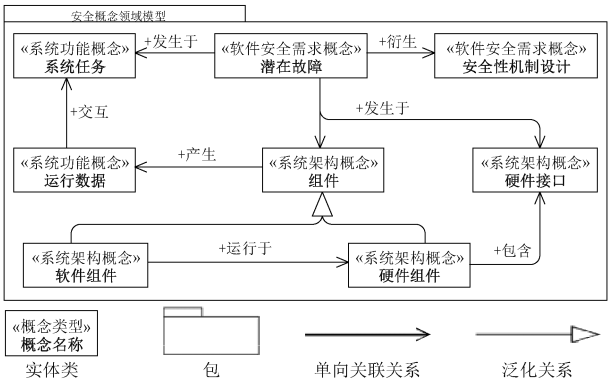


图 3 安全关键软件安全概念领域模型图

务的形式来表达,定义了系统组件在运行过程中需完成的一系列行为。任务的功能实现依赖于数据计算和交互,数据由组件在运行过程中产生。

安全需求一般围绕识别出的潜在故障来获取,包含故障及其处理机制[10],故障是系统运行过程中出现的状态错误。针对所识别的潜在故障,需要相应的安全机制来降低故障的发生概率或消除故障对系统发挥正常功能的影响。不失一般性,本文主要从系统任务、软硬件组件、硬件接口中识别潜在故障,进而提取相应的安全需求。

3.2.2 安全需求提取提示词元模型设计

如 2.2 节所述,软件安全需求获取与一般软件需求获取在目标和流程上的差异也导致二者在应用 LLM 完成任务时的提示词存在差异。现有将 LLM 应用于一般软件需求获取的研究,通常以用户对系统的期望作为 LLM 输入,以满足期望的软件需求为输出,因此提示词设计关注于引导 LLM 对用户期望进行泛化或细化以提炼生成相应的需求[27,57]。相比之下,本文所关注的安全需求提取任务需要从文本中提取隐含的安全需求信息,其难点在于难以确定安全需求分析需要哪些关键信息,以及如何从文本中准确地提取,这也是该任务依赖于工程师的领域知识和经验的原因所在。由于 LLM 缺乏相应的领域知识和经验,需要通过提示词来明确所需提取的关键信息并向 LLM 提供历史提取经验,从而使 LLM 满足安全需求提取任务的领域知识和经验要求。

通过提示词向 LLM 提供领域知识和经验同时涉及安全关键系统和提示工程两方面知识,本文使用元模型对两方面知识进行整合,设计了安全需求提取提示词元模型。该元模型基于现有提示工程方法的实践经验[20,23],捕捉了提示工程和安全需求中的关键概念,分离了提示工程和领域概念的关注点,保证安全需求提取任务系统性和全面性的同时降低了构造提示词的难度,元模型如图 4 所示,该模型依照 UML 类图规范构建。

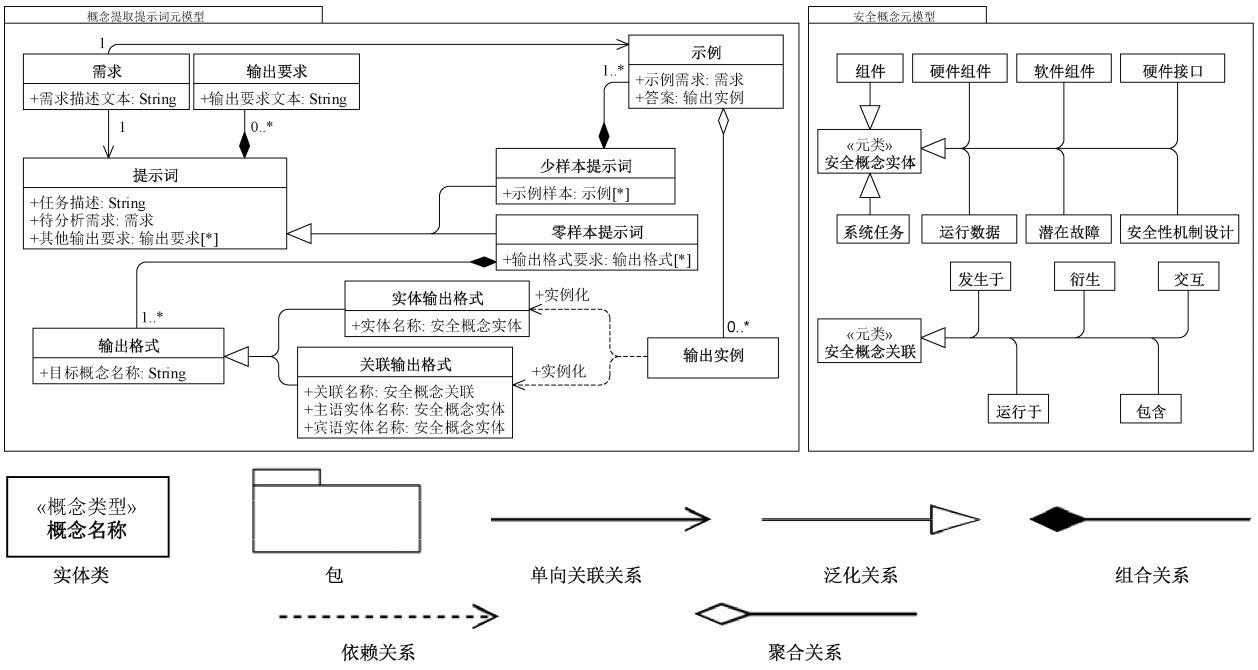


图 4 安全需求提取提示词元模型图

安全需求提取提示词元模型由两部分组成:概念提取提示词元模型、安全概念元模型。其中,安全概念元模型用于明确所需提取的关键信息,元模型定义了领域模型所要表达的实体和关系,包含对安

全概念实体和安全概念关联两种元类的定义。安全概念实体定义了领域模型中的概念实体,安全概念关联则定义了实体之间的关联关系。结合图 3 中识别的领域概念,系统任务、潜在故障、组件等概念扩展自安全概念实体元类,概念间的衍生、交互等关系则扩展自安全概念关联元类。

基于 3.2.1 节的分析,本文将所需提取的关键安全需求要素和关系定义为图 3 的领域模型,通过引导 LLM 识别安全概念模型中的要素和关系,从而形成系统的安全需求。因此该任务可被看作是自然语言实体关联识别任务,这也是当前需求获取工作的普遍做法[15,17]。本文借鉴这种做法,对当前常用的零样本提示和少样本提示方法进行建模,基于安全概念模型语法设计概念提取提示词元模型。其中,提示词类来自于对零样本提示和少样本提示共有结构的抽象,包括任务描述、待分析需求、其他输出要求三个部分。任务描述用于明确 LLM 所需完成的任务;待分析需求向 LLM 提供待分析的文本;其他输出要求则明确了 LLM 在任务过程和输出上的约束,单个提示词可以添加多个输出要求。

零样本提示词类在提示词类的基础上增加了输出格式要求属性,单个零样本提示词可根据所需获取目标信息的数量添加多个输出格式约束,并可根据捕捉领域概念类型的不同,进一步被区分为实体输出格式和关联输出格式,分别用于捕捉领域概念元模型中的实体和关联概念。

少样本提示词类同样继承自提示词类,增加了示例样本属性。单个少样本提示词可根据需要提供一个或多个示例,每个示例包含一个输入任务文本描述和多个输出实例,输出实例需要按照实体或关联的输出格式给出,因此是实体输出格式和关联输出格式两个类的实例。

在安全需求提取提示词元模型中,概念提取提示词元模型用于捕捉安全需求中的实体要素和关联概念。因此,概念提取提示词元模型可针对安全概念领域模型生成相应的安全需求提取提示词,通过输出格式明确所需捕捉的安全信息,并通过示例提供安全需求提取的领域知识和经验。这种设计将提示词和领域分析解耦,降低了提示词构造的复杂度,提高了提示词设计的可扩展性,并且方便替换领域模型中的相关要素。

3.2.3　安全需求提取提示词建模

安全需求提取提示词元模型捕捉了安全需求提取过程中提示工程和安全需求两个角度的关键概念,在此基础上本文采用提示词框架捕捉元模型中定义的信息,实现对零样本和少样本提示词的建模,具体方法如下:

(1)零样本提示词建模。零样本提示词在安全需求提取提示词元模型中由提示词类和零样本提示词类定义,可被形式化定义为

$$P_{zero}=\{t_{zero},r,F,c\} \tag{1}$$

$$F=\{(s_1,o_1),(s_2,o_2),\cdots,(s_n,o_n)\} \tag{2}$$

其中,t_{zero} 为零样本提示词任务描述,对应于图 4 元模型中提示词类的任务描述,用于定义 LLM 的任务内容;r 为待分析需求文本,对应于提示词类的待分析需求属性;F 表示包括实体和关联在内的各类安全需求信息的输出格式要求,对应于零样本提示词类的输出格式要求,由一系列安全需求信息名称和相应的输出格式构成,分别由 s_i 和 o_i 表示,具体如公式(2);c 表示零样本提示词的其他输出要求,对应于提示词类的其他输出要求属性。本文使用零样本提示词框架引导用户针对上述信息进行建模,框架及所捕捉提示词信息与元模型的对应关系见表 1。

表 1　安全需求提取零样本提示词框架及对应元模型元素

提示词结构	对应元模型元素
任务描述	提示词::任务描述
输出格式要求	零样本提示词::输出格式要求
待分析需求	提示词::待分析需求
其他输出要求	提示词::其他输出要求

(2)少样本提示词建模。少样本提示词在安全需求提示词元模型中由提示词类和少样本提示词类定义,可被形式化定义为

$$P_{few}=\{t_{few},E_1,E_2,\cdots,E_n,r,c\} \tag{3}$$

$$E=\{hr,(s_1,A_1),(s_2,A_2),\cdots,(s_n,A_n)\} \tag{4}$$

$$A=\{a_i,\cdots,a_m\} \tag{5}$$

其中,t_{few} 是少样本提示词的任务描述,对应于提示词类的任务描述;E_i 表示输入输出示例,对应于少样本提示词类的示例样本属性,通过一组示例向 LLM 提供领域知识和历史安全需求提取经验,根据元模型中的定义,少样本提示词可以包含多个示例,每个示例 E_i 由一个历史需求案例文本 hr 和包含多个输出实例的答案组成,单个输出实例包含历史安全需求信息特征 s_i、由人工获取的安全需求信息集合 A_i 两部分,a_i 表示某个具体的安全需求信息条目,具体如公式(4)(5),在本文方法中输入输出示例由方法自动检索得到;待分析需求 r 和其他输出要求 c 则与零样本提示词定义相同。本文使用少样本提示词框架引导用户针对上述信息进行建模,框架

及所捕捉提示词信息与元模型的对应关系见表 2。

表 2 安全需求提取少样本提示词框架及对应元模型元素

提示词结构	对应元模型元素
任务描述	提示词::任务描述
输入输出示例	少样本提示词::示例样本
待分析需求	提示词::待分析需求
其他输出要求	提示词::其他输出要求

用户通过使用以上提示词框架可以对零样本和少样本提示词建模，框架所捕捉信息可以实例化为安全需求提取提示词模型，用于后续的安全需求提取提示词生成。

3.2.4 安全需求提取提示词生成

使用提示词框架建模得到的安全需求提取提示词模型包含了构建安全需求提取提示词所需的任务描述、输出要求、目标概念等信息，基于此可以实现零样本提示词和少样本提示词的自动生成，具体流程如算法 1 所示。

算法 1. 安全需求提取提示词生成算法

输入：

安全需求提取提示词模型 *Model*

输出：

安全需求提取提示词文本 *promptString*

1. *promptString* = String
2. *prompt* = *Model*.*get*(提示词) //从模型中获取提示词类实例
3. *promptString*.*add*(*prompt*::任务描述)
4. IF *prompt*::*type* == 零样本提示词 DO:
5. FOR 实体 in *Model*.*get*(实体输出格式) DO:
6. *promptString*.*add*(*format*(实体::实体名称))
7. END FOR
8. FOR 关联 in *Model*.*get*(关联输出格式) DO:
9. *promptString*.*add*(*format*(关联::主语实体名称，关联::宾语实体名称))
10. END FOR
11. END IF
12. ELSE IF *prompt*::*type* == 少样本提示词 DO:
13. FOR 示例 in *Model*.*get*(示例) DO:
14. *promptString*.*add*(示例::示例需求)
15. *promptString*.*add*(示例::答案)
16. END FOR
17. END IF
18. *promptString*.*add*(提示词::待分析需求)
19. FOR 输出要求 in 提示词::输出要求 DO:
20. *promptString*.*add*(输出要求)
21. END FOR
22. RETURN *promptString*

算法 1 的输入为针对单个提示词建模的提示词模型，即基于 3.2.3 节中提示词框架所捕捉信息建立的模型实例。算法的第 2～3 行首先对模型进行解析并从中提取提示词类实例，将提示词类实例中的任务描述加入到提示词文本中。若建模的提示词是零样本提示词，则通过算法的第 4～11 行进行零样本提示词构建，构建零样本提示词需要将安全概念实体和关联按照输出格式加入到提示词中，从而明确 LLM 所需提取的信息。其中 5～7 行将领域模型中所有的安全概念实体按照规定的输出格式加入到提示词中，8～10 行则将领域模型中的安全概念关联按照输出格式要求加入到提示词中，从而完成零样本提示词的构建。若提示词为少样本提示词，则通过算法的第 12～17 行进行少样本提示词构建，构建过程中按序将示例以示例需求和答案的组合添加到提示词中，其中答案在元模型中是实体输出格式和关联输出格式的实例，即答案的格式需要符合实体和关联的输出格式。算法的第 18 行将待分析需求追加到提示词中，明确安全需求提取任务的目标需求文本。算法的第 19～21 行将 LLM 的输出约束逐个添加到提示词中，指导 LLM 输出符合要求的信息。

3.3 基于检索增强生成的安全需求提取

由于零样本提示无法向 LLM 提供领域知识和经验，效果通常弱于少样本提示。而如何选择有针对性的示例则是构造少样本提示词需要解决的关键问题。本文将零样本提示与少样本提示相结合，首先基于零样本提示词来获取安全特征，即图 3 领域模型中的安全需求相关信息，以此为基础检索具有相似安全特征的历史安全需求案例，从而为少样本提示词从安全特征角度选取切合的示例完成安全需求提取。

3.3.1 基于零样本提示的安全特征提取

在本文方法中，零样本提示词用于指导 LLM 在不结合领域知识与历史经验的情况下提取安全特征，从而为少样本提示从安全特征角度识别相似示例。该过程可被形式化定义为

$$f_{LLM}(P_{zero})=\{(s_1,A_1),(s_2,A_2),\cdots,(s_n,A_n)\} \tag{6}$$

其中，P_{zero} 为提取过程中所使用的零样本提示词，其具体含义如公式(1)所示；s_i 为输出的第 i 类安全需求信息名称；A_i 为 LLM 获取的第 i 类安全需求信息集合。相应的，a_i 表示某个具体的安全需求信息条目(见公式(5))。

根据该任务特征,本文使用零样本提示词框架对安全特征提取零样本提示词建模,所建模零样本提示词如表 3 所示。

表 3　安全特征提取零样本提示词建模

提示词结构	提示词内容
任务描述	提取出目标案例中的明确指出的信息。待分析需求中明确指出的信息通过以下格式给出:
输出格式要求	＜安全需求信息名称＞:＜输出格式＞
待分析需求	＜待分析需求文本＞
其他输出要求	仅从案例中提取信息,案例中不包含的无需进行联想生成归纳,若为无则输出'无'。

该提示词通过任务描述部分明确 LLM 的任务是提取安全特征,并按照给定格式输出;同时通过其他输出要求约束 LLM 的联想和生成能力,以避免 LLM 在需求提取过程引入待分析需求以外的信息,或输出不符合格式要求的信息。

提示词中的输出格式要求则由一系列安全需求信息名称和相应的输出格式构成(见表 4),本文基于图 3 中安全概念领域模型结构,从系统架构、系统功能、软件安全需求三个角度识别了相应的实体和实体间关系,并设计了相应的输出格式。实体输出格式要求 LLM 输出实体名称;关联则需要输出存在相应关联的两个实体,并通过"—"相连。为保证输出的相对独立,不同的实体和关系之间使用分号进行分割,并使用井字号"#"来代表一类安全需求信息的结尾。完整零样本提示词建模需要将表 4 中的安全需求信息名称和输出格式按表 3 定义的结构按顺序全部添加到提示词。

表 4　待提取的安全需求信息及输出格式要求

安全需求类型	安全需求角度	安全需求信息名称	输出格式
实体	系统架构要素	软件组件	软件组件 1;软件组件 2;…… #
		硬件组件	硬件组件 1;硬件组件 2;…… #
		硬件接口	硬件接口 1;硬件接口 2;…… #
	系统功能要素	运行数据	数据 1;数据 2;…… #
		系统任务	任务 1;任务 2;…… #
	软件安全需求要素	潜在故障	故障 1;故障 2;…… #
		安全性机制设计	安全性设计 1;安全性设计 2;…… #
关联	系统架构关联	软件硬件运行关系	软件组件－硬件组件;…… #
		接口硬件隶属关系	硬件接口－硬件组件;…… #
	系统功能关联	组件所产生数据关系	软件组件/硬件组件－数据;…… #
		数据所涉及任务	数据－任务;…… #
	软件安全需求关联	故障发生位置	故障－软件组件/硬件组件/硬件接口/数据;…… #
		可能发生故障的功能	故障－任务;…… #
		故障所对应安全性机制	故障－安全性机制设计;…… #

图 5 提供了一个由算法 1 在此基础上生成的完整零样本提示词示例。

```
任务:提取出目标案例中的明确指出的信息。目标案例中明确指出的信息通过以下格式给出:

软件组件:软件组件1;软件组件2;…… #
硬件组件:硬件组件1;硬件组件2;…… #
硬件接口:硬件接口1;硬件接口2;…… #
运行数据:数据1;数据2;…… #
系统任务:任务1;任务2;…… #
潜在故障:故障1;故障2;…… #
安全性机制设计:安全性设计1;安全性设计2;…… #
软件硬件运行关系:软件组件-硬件组件;…… #
接口组件隶属关系:硬件接口-硬件组件;…… #
组件所产生数据关系:软件组件/硬件组件-数据;…… #
数据所涉及任务:数据-任务;…… #
故障发生位置:故障-软件组件/硬件组件/硬件接口/数据;…… #
可能发生该故障的功能:故障-任务;…… #
故障所对应安全性机制:故障-安全性机制设计;…… #

目标案例:AFTI/F-16数字飞行控制系统基本上是具有三条独立电子通路的三余度系统。主要电子组件是三个相同的数字飞行控制计算机。三余度飞机运动传感器(速率陀螺和加速度计)向飞行控制计算机提供飞机状态反馈信息。四余度驾驶员侧杆和方向舵脚蹬力传感器接收驾驶员指令,并将这些输入信号转换为模拟电信号传输到三余度飞行控制计算机进行处理。这些信号被控制律用来计算舵面偏转指令。三余度数字飞行控制系统能提供两次飞行控制计算机故障后的双故障-工作能力。

仅从案例中提取信息,案例中不包含的无需进行联想生成归纳,若为无则输出'无'。
```

图 5　零样本提示词示例

3.3.2　相似安全需求示例选取

现有研究主要基于语义相似度选择相似的示例来构造少样本提示词[58],为 LLM 提供相关知识来提升其在指定任务上的性能。然而,正如前文所述,安全需求提取需要结合系统架构、系统功能和软件安全需求多方面领域知识,从需求文本中识别软件安全需求的相关信息,因此常用的基于语义相似度的方法难以从安全角度匹配合适的示例向 LLM 提供必要的领域知识。为有效从安全角度检索到相关的历史案例,本文在图 3 中的领域概念模型的指导下捕捉安全特征,使用基于安全特征的 RAG 方法,从零样本提示的结果中提取表 4 所示的安全需求特征并向量化。表 4 给出的安全需求信息包含系统架构、系统功能、软件安全 3 个安全需求角度共 14 个安全需求信息特征维度,捕捉了图 3 所定义的安全概念及概念间关系,为保证示例在安全层面提供尽可能相似的信息,可通过文本中包含的各类安全信息数量来判断两个文本中安全需求特征的相似程

度。首先从需求文本中按照表 4 定义的各类安全需求信息来提取特征，进而形成安全需求向量：

$$H=[l_i,\cdots,l_n] \tag{7}$$

其中，$n=14$，分别对应于表 4 中的 14 个安全需求信息维度，l_i 为每个维度所包含的安全需求信息条目数量，即公式(5)中 a 的数量，从而形成一条需求文本所对应的安全需求信息特征向量。

安全需求向量有 14 个维度，对应系统架构、系统功能、软件安全 3 个角度的实体概念及其关联，由于关联描述了两个实体间的关联关系，因此与实体概念存在很强的相关性，难以保证不同维度间的独立性，可能导致部分特征在相似性计算中被重复考虑，影响相似度结果的准确性。为解决此问题，本文采用了主成分分析(Principal Component Analysis，PCA)对数据进行降维，将原始的 14 维数据转换为 7 个独立的主成分，通过消除变量间相关性[59]以避免特征被重复考虑。

经过 PCA 降维后，本文通过余弦相似度[60]，如公式(8)所示，来筛选高相似度的历史安全需求作为少样本提示案例。

$$\cos\boldsymbol{X}\cdot\boldsymbol{Y}=\frac{\boldsymbol{X}\cdot\boldsymbol{Y}}{\|\boldsymbol{X}\|\cdot\|\boldsymbol{Y}\|}=\frac{\sum_{i=1}^{n}(x_i\times y_i)}{\sqrt{\sum_{i=1}^{n}(x_i)^2}\times\sqrt{\sum_{i=1}^{n}(y_i)^2}} \tag{8}$$

其中，$\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 分别是 LLM 产生的安全需求特征向量和历史安全需求的特征向量，余弦值接近 1，则两个向量的夹角越接近 0 度，表明两个向量越相似。n 表示向量的特征数量。

算法 2. 相似安全需求识别算法

输入：

安全需求特征文本列表 *answerList*

历史安全需求特征向量库 *histBase*

输出相似历史安全需求数量参数 *k*

输出：

相似需求列表 *similarList*

```
answerVec, cosineList, similarList = [], [], []
FOR answerText in answerList DO:
answerItems = Itemized(answerText)  //各类安全需求信息条目化
answerVec.add(count(answerItems))  //安全需求信息条目数量作为特征加入特征向量
END FOR
answerVec = PCA(answerVec)  //安全需求特征向量PCA 降维
FOR histCase in histBase DO:
similarity = cosine(answerVec, histCase[Vec])  //计算当前安全需求特征向量与历史安全需求的余弦相似度
CosineList.add({histCase[ID]: similarity})  //记录当前安全需求特征向量与每一个历史安全需求的余弦相似度
END FOR
sortedCosineList = sortBySimilarity(CosineList, DESC)// 降序排序与所有历史安全需求的余弦相似度
i = 0
WHILE i < k DO:
similarReq = histBase[sortedCosineList[i][ID]][Req]  //记录来源需求文本
similarResult = histBase[sortedCosineList[i][ID]][Result] //记录历史安全需求分析结果
similarList.add({similarReq: similarResult})  //向示例列表加入历史安全需求分析结果及其来源需求分析文本
i += 1
END WHILE
RETURN similarList
```

算法 2 的输入为按照表 4 中安全需求信息输出格式进行描述的安全需求文本 *answerList*，其中 14 个维度分别对应 14 类安全需求信息的输出文本，字典 *histBase* 包含所有历史安全需求提取结果、对应来源文本以及降维后的安全需求特征向量，参数 k 为输出相似历史安全需求的数量。算法的第 2～5 行首先将每一个维度的需求文本根据输出格式进行条目化，并识别每个维度对应的条目数量，从而进行安全特征向量化。算法的第 6 行将向量化后的安全特征进行 PCA 降维。

算法的第 7～10 行计算初步安全需求特征向量 *answerVec* 与所有历史安全需求特征向量之间余弦相似度，并以历史安全需求序号 *histCase*[*ID*]为索引，添加到余弦相似度序列 *cosineList* 中。算法的第 11 行对余弦相似度序列 *cosineList* 进行降序排列。算法的第 12-18 行从降序排列后的余弦相似度列表 *sortedCosineList* 中选出相似度最高的 k 个历史安全分析结果，并将历史安全分析结果的需求和来源文本逐个添加到相似需求列表 *similarList* 中。

3.3.3 基于少样本提示词的安全需求提取

在本文方法中，少样本提示词需要结合 3.3.2 节筛选得到的相似示例完成安全需求提取。该过程可被形式化为：

$$f_{LLM}(P_{few})=\{(s_1,A_1),(s_2,A_2),\cdots,(s_n,A_n)\} \tag{9}$$

其中,P_{few} 表示指导 LLM 进行安全需求提取所使用的少样本提示词,其具体含义见公式(3),A_i 和 s_i 的含义同公式(5)(6)。

本文使用少样本提示词框架对安全需求提取少样本提示词建模,所建模少样本提示词如表 5 所示。该提示词通过任务描述强调了 LLM 需要按照示例中的输入和输出关系,结合其中包含的领域知识和经验,来对待分析需求进行安全需求提取。并在输入输出示例部分向 LLM 提供筛选得到的相似历史安全需求和相应的需求案例文本,从而向 LLM 提供领域知识和历史安全需求提取经验。

表 5　安全需求提取少样本提示词建模

提示词结构	提示词内容
任务描述	请参照以下示例及示例提取结果的输入输出关系,使用此思路,提取出目标案例中包含的信息
输入输出示例	[<示例需求> <示例需求安全分析结果>]
待分析需求	<待分析需求文本>
其他输出要求	仅从案例中提取存在的信息,案例中不包含的无需进行联想生成归纳,若为无则输出'无'

图 6 提供了一个由算法 1 在此基础上生成的完整少样本提示词示例。

任务:请参照以下示例及示例提取结果的输入输出关系,使用此思路,提取出目标案例中的明确指出的信息。

示例1:AFTI/F-16飞行控制系统中,飞行控制计算机会以飞行员通过飞行控制板、侧杆、油门等人工控制飞机飞行姿态产生的输入信号,以及各类传感器对外界物理数据的采集和对飞机各个设备运行状态的监控作为输入进行计算,并将计算结果输出到复合舵机进行作动,从而实现对飞机飞行姿态的调节。该系统作为安全关键系统,传感器、飞行控制计算机和作动装置均有多余度设计,从而保证系统的可靠性和安全性。
示例提取结果1:软件组件:无 #
硬件组件:飞行控制板;侧杆;油门;各类传感器;复合舵机;飞行控制计算机;作动装置 #
硬件接口:无 #
运行数据:输入信号;外界物理数据;各个设备运行状态 #
软件硬件运行关系:无 #
接口组件隶属关系:无 #
组件所产生数据关系:飞行控制板-输入信号;侧杆-输入信号;油门-输入信号;各类传感器-外界物理数据;各类传感器-各个设备运行状态;飞行控制计算机-计算结果 #
系统任务:飞机飞行姿态计算;复合舵机作动;飞机飞行姿态调节 #
数据所涉及任务:输入信号-飞机飞行姿态计算;外界物理数据-飞机飞行姿态计算;各个设备运行状态-飞机飞行姿态计算 #
潜在故障:无 #
故障发生位置:无 #
可能发生该故障的功能:无 #
安全性机制设计:传感器多余度设计;飞行控制计算机多余度设计;作动装置多余度设计 #
故障所对应安全性机制:无 #

目标案例:AFTI/F-16采用侧杆控制器及脚蹬实现操纵。四余度的力变换器将驾驶员的操纵力转换为模拟电信号,经过模/数变换器输入飞行控制主计算机,通过与其他信号进行综合计算,按指定算法产生控制指令,并通过控制相应复合舵机从而实现对飞机姿态的控制。

仅从案例中提取存在的信息,案例中不包含的无需进行联想生成归纳,若为无则输出'无'。

图 6　少样本提示词示例

4　实验设计

本节介绍数据集及其构建过程,以及评估实验设计,包括评估指标、实验设置和研究问题。

4.1　基准数据集

PROMISE 数据集①和 PURE 数据集[61]是当前需求工程领域的主流数据集,其中 PROMISE 数据集包含来自 15 个项目的 625 条条目化需求,均标注具体的需求类别,因此常被用于需求分类任务[62-63]。PURE 数据集则包含有 79 个完整的需求文档,当前被广泛应用于需求建模[64]、需求分类[65]等任务。然而以上数据集主要关注于软件的需求信息,缺少系统架构等安全需求分析所必需的系统信息。同时,当前需求数据集以英文为主,无法满足中文需求工程任务的需要,因此本文任务无法应用于以上数据集。

为解决此问题,本文收集整理了 3 个来自不同安全关键领域的具体案例,分别为 AFTI/F-16 验证机飞行控制系统(Flight Control System,简称 FCS)[66-68]、车用自动巡航控制系统(Cruise Control System,简称 CCS)[69-70]、机载自动油门系统(Automatic Throttle Control,简称 ATC)[66]。案例均由领域专家依照安全关键软件实践中的需求要求编制形成需求文档,需求文档包含案例的系统架构、功能需求、安全性设计等信息。考虑到当前 LLM 对长文本的处理能力有限,同时为保证每一条需求文本包含完整的安全上下文信息,本文将 3 个案例的需求文档按自然段进行拆分,形成最终的数据集,其基本信息如表 6 所示。数据集中共包含 42 条具体需求,FCS、CCS、ATC 案例分别包含 15、14、13 条需求,需求平均长度为 174 字。根据需求涵盖的信息进行分类,在合计 42 条需求中,35 条需求包含系统架构信息,29 条需求包含功能需求信息,18 条需求包含安全性设计信息,数据集中大部分需求涵盖一个方面以上的信息。

在针对数据集中的 42 条具体需求进行安全需求信息标注过程中,本文组织 2 位来自北京控制与电子研究所的领域专家,分别独立从 42 条需求文本中识别图 3 安全需求领域模型中的硬件组件、故障等实体和软件硬件运行关系、故障发生位置等实体间关系。在数据标注开始前事先根据标注标准进行培训,之后由两名专家独立进行标注,两名专家标注结果中一致的内容将被保留,存在分歧的部分被进一步讨论修改,标注结果由第一作者进行审查,并按照表 4 的输出格式进行整理,作为安全需求提取的基准。

相比于现有的需求数据集,本文数据集为中文,包含了 3 个来自于不同安全关键领域的需求案例,

① nfr, https://zenodo.org/records/268542 2024,12,1

即航空装备领域、车载电子领域和民用航空领域，且均从安全需求提取角度进行了整理，标注了安全需求相关的实体及其关系，保证了系统架构、软件功能等信息的完整性。

表 6 安全关键软件需求数据集基本信息

案例	案例领域	需求包含字符数量			涵盖对应类型信息需求数量			案例包含需求数量
		最小	最大	平均	系统架构	功能需求	安全性设计	
FCS	航空装备	92	367	165	14	10	10	15
CCS	车载电子	77	376	172	11	8	3	14
ATC	民用航空	89	293	186	10	11	5	13
完整数据集		77	376	174	35	29	18	42

4.2 评估指标

对安全需求提取过程来说，提取结果的准确性和全面性同样重要，识别错误和缺漏均有可能为系统带来未知的风险，因此本文选择 $F1$ 值作为 LLM 安全需求提取性能的综合评估指标。$F1$ 值综合考虑了准确率（$Precision$）和召回率（$Recall$），是二者的调和平均值，其中准确率评估 LLM 提取安全需求的准确性，即在 LLM 提取的所有安全需求中，基准安全需求所占比例；召回率评估 LLM 提取安全需求的全面性，即 LLM 正确提取的安全需求占目标需求所包含所有基准安全需求的比例。准确率、召回率和 $F1$ 的计算由公式(10)、(11)和(12)定义：

$$Precision=\frac{TP}{TP+FP} \tag{10}$$

$$Recall=\frac{TP}{TP+FN} \tag{11}$$

$$F1=\frac{2\cdot Precision\cdot Recall}{Precision+Recall} \tag{12}$$

其中，TP 为 LLM 提取的基准安全需求数量，FP 为 LLM 提取的非基准安全需求数量，FN 为 LLM 未提取到的基准安全需求数量。

由于安全需求的提取结果因人而异，因此基准安全需求集合不是标准答案。在评估 LLM 安全需求提取能力时，本文首先将 LLM 提取的安全需求与领域专家的分析结果进行文本比对，识别 LLM 正确提取且表述一致的情况，之后由两位领域专家进一步进行审查以检查与基准安全需求一致但表述有差异的识别结果，并将其标注为与基准安全需求一致，从而形成最终的评估结果。

4.3 基线方法

考虑到当前 LLM 在需求工程的 NLP 任务中表现出了优秀的能力和灵活性，且先前并无与本文方法目标一致的其他方法，本文选择基于 LLM 的通用需求提取方法作为基线方法进行对比，从而评估本文方法相比于现有需求提取方法的效果。

另一方面，本文方法使用 LLM 提取目标需求的安全需求特征，并在此基础上检索相似示例构造少样本提示词，进行安全需求提取。作为一种 RAG 方法，其核心差异在于示例检索策略，因此本文选择基于主流示例选取策略 RAG 方法所构建的少样本提示作为基线方法进行对比。

基于以上分析，本文从两个角度选择基线方法：(1)目前公开发表最优的基于 LLM 的通用需求提取方法，对比基线确定为 zsl-ner 方法[71]；(2)基于目前主流示例检索策略 RAG 方法构建的少样本提示，对比基线确定为基于词频－逆向文件频率（Term Frequency-Inverse Document Frequency，TF-IDF）示例选取策略、基于语义相似度示例检索策略。

①zsl-ner 方法[71]。该方法是目前性能最优的需求提取方法，它基于 LLM 来识别需求中的命名实体，相比于基于机器学习的需求提取方法，该方法不仅在需求信息的识别任务上取得了最优的效果，而且可以根据需要提取指定的目标需求信息，同时也可以针对中文文本进行需求提取任务。

②基于 TF-IDF[72] 的示例选取策略。TF-IDF 结合了词频和逆文档频率衡量词语在语料库中的重要性，使用 TF-IDF 评估文本相似度从而选取相似案例是当前 RAG 实践的主流策略[73]。实验中，本文将整个数据集作为语料库，并使用中文分词广泛应用的 jieba 分词器①对需求文本进行分词。通过计算目标需求与数据集中其他需求在 TF-IDF 向量上的余弦相似度，选择相似度最高的需求文本及安全分析结果作为示例形成少样本提示。

③基于语义相似度的示例选取策略，这是当前 RAG 实践中主要应用的策略，通常通过文本的语义相似度来识别相关文本信息，广泛使用基于 BERT 的模型作为检索器[74]。因此，本文选择谷歌公司的 BERT-base-chinese 预训练语言模型[42]作为检索器计算需求文本之间的语义相似度，并选择语义相似度最高的需求文本及安全分析结果作为示例形成少

① jieba，https://github.com/fxsjy/jieba 2025,4,10

样本提示,从而与本文方法进行对比。BERT-base-chinese 模型包含 110 万个参数,可以有效地捕捉中文文本中的语义信息。

4.4　基础 LLM

由于本文关注于中文语境下的安全需求提取,为了分析不同 LLM 对于安全需求提取任务的完成情况,本文实验采用了 ERNIE-4.0-8K、GLM-3-Turbo、Qwen-Turbo 三个中文 LLM。

4.5　研究问题与实验设置

本文共设置 2 个研究问题(Research Question,RQ),分别探究本文方法所提取的安全需求质量和方法效果的影响因素:

研究问题 1:本文方法指导 LLM 提取的安全需求质量如何?

基于不同方法提取得到的安全需求质量,对比分析本文方法和基线方法在安全需求提取任务上的应用效果。

研究问题 2:哪些因素会对本文方法提取的安全需求质量产生影响?

该研究问题探究基础 LLM、示例数量、相似度案例选择策略、应用领域、方法设计五个方面对本文方法性能的影响。具体 RQ 如下:

(1)RQ2.1:基础 LLM 对本文方法提取的安全需求质量有何影响?

不同 LLM 在自然语言理解、领域知识掌握等方面的性能差异,可能会对下游任务性能造成影响。本 RQ 通过对比分析本文方法框架下所选三个 LLM 的安全需求提取任务性能,探究基础 LLM 对安全需求提取质量的影响。

(2)RQ2.2:示例数量对本文方法提取的安全需求质量有何影响?

现有研究表明,少样本提示中示例数量过少可能导致 LLM 无法学习到充分的知识,而示例数量过多则可能对 LLM 造成干扰[75]。因此,针对所选择的三种 LLM,对比分析了不同示例数量少样本提示下,LLM 安全需求提取任务的完成效果。

(3)RQ2.3:相似度案例选择策略对本文方法提取的安全需求质量有何影响?

本文采用 RAG 来选择更有针对性的历史安全需求案例,从而增强 LLM 的需求提取能力。一般认为,相似度越高的案例,增强效果越好。但也有研究表明低相似度案例同样可以增强 LLM 的任务性能[76]。为探究不同的安全需求特征相似度选择策略对本文方法效果的影响,该 RQ 将历史安全需求案例按与目标需求的安全特征相似度排名,把历史安全需求案例按相似度由高到低均匀划分为五组,并从每一组选择相似度最高的案例来构造提示词,形成五个相似度选择策略,分别为 Hst_Strategy(最高相似度选择策略)、H_Strategy(高相似度选择策略)、M_Strategy(中等相似度选择策略)、L_Strategy(低相似度选择策略)和 Lst_Strategy(最低相似度选择策略),从而可以对比分析不同选择策略对安全需求提取效果的影响。

(4)RQ2.4:本文方法在不同应用领域下的安全需求提取性能如何?

本文方法通过向 LLM 提供领域知识从而提升 LLM 的安全需求提取性能,而不同应用领域的知识并不通用,这可能对本文方法在不同安全关键领域上的性能和适用性产生影响。因此该研究问题围绕航空装备、车载电子、民用航空三个不同安全关键领域,对本文方法在不同应用领域的安全需求提取性能进行分析。

(5)RQ2.5:本文方法中基于零样本提示的安全特征提取设计和 PCA 降维设计对本文方法提取的安全需求质量有何影响?

本文方法使用零样本提示提取安全特征,并据此为少样本提示选择相似示例,同时应用 PCA 将安全需求特征向量降维,以避免特征被重复考虑从而影响示例选取结果。该研究问题通过消融实验探究基于零样本提示的安全特征提取设计和 PCA 降维设计的有效性。由于安全特征是本文方法为少样本提示选取相似示例的核心依据,针对基于零样本提示的安全特征提取设计的消融实验将使用随机示例选择策略,从数据集中随机抽取需求文本及其安全需求信息提取结果,作为示例构造少样本提示词进行安全需求提取,重复进行三次实验,最终结果取平均值。PCA 降维设计的消融实验则不会在相似示例选取的过程中对安全特征向量进行降维。

5　实验结果与分析

按照第 4 节中的实验设计,本节针对实验结果展开分析。

5.1　研究问题 1:本文方法指导 LLM 提取的安全需求质量如何?

本文方法与基线方法在三个 LLM 上的安全需求提取性能如表 7 所示。

(1)对比当前最优的基于 LLM 的需求提取方法:

表 7 基线方法和本文方法的安全需求提取性能(*F*1)

安全需求特征角度	基础 LLM	基线方法					本文方法	
		zsl-ner	基于主流示例检索策略的 RAG 方法					
			基于 TF-IDF		基于语义相似度			
			1-shot	2-shot	1-shot	2-shot	1-shot	2-shot
系统架构	ERNIE-4.0-8K	64.31	73.92	76.12	75.17	**76.92**	63.08	69.08
	GLM-3-Turbo	46.41	55.24	54.93	60.66	**61.35**	53.26	55.10
	Qwen-Turbo	8.99	41.90	44.63	**53.61**	40.76	43.69	41.16
系统功能	ERNIE-4.0-8K	28.66	46.04	46.74	50.50	43.69	44.59	**52.98**
	GLM-3-Turbo	19.50	33.24	31.07	35.90	36.87	33.78	**37.23**
	Qwen-Turbo	6.42	30.09	29.73	16.75	27.47	29.90	**35.27**
软件安全	ERNIE-4.0-8K	26.79	42.65	48.67	43.05	42.01	47.00	**50.75**
	GLM-3-Turbo	15.07	34.69	34.95	23.53	35.65	18.35	**41.79**
	Qwen-Turbo	7.84	13.49	19.61	8.86	15.98	15.42	**20.20**
整体性能	ERNIE-4.0-8K	42.19	55.30	57.02	58.33	55.34	51.72	**58.53**
	GLM-3-Turbo	29.10	41.83	39.82	43.66	**45.39**	39.02	44.73
	Qwen-Turbo	7.55	30.43	33.15	23.27	29.21	30.73	**33.30**

相比于 zsl-ner 基线方法,本文方法的整体性能在所有 LLM 均有显著的提升,提升分别达到了 16.34%、15.63%和 25.75%。从安全需求特征角度来看,zsl-ner 在系统架构特征维度上,使用 ERNIE-4.0-8K 和 GLM-3-Turbo 模型取得了接近本文方法的性能,仅有 4.77%和 8.69%的差距;而在系统功能和软件安全特征维度上,zsl-ner 在三个 LLM 上与本方法的差距均在 15%以上。

zsl-ner 方法作为一种基于 LLM 的零样本提示方法在需求命名实体识别任务上的 *F*1 值可以达到 98%[71],但在本文实验中其性能则与本文方法存在显著的差距,这表明传统基于命名实体识别的需求提取方法无法有效识别安全需求。同时,虽然 zsl-ner 可以在系统架构方面达到与本文方法相近的水平,但是在系统功能和软件安全角度则由于无法向 LLM 提供领域知识而导致性能差距较大。可见,识别系统功能和软件安全相关信息需要提供更多的领域知识支撑。

综上,本文方法相比于当前最优的 zsl-ner 方法,可以向 LLM 提供更有针对性的领域知识,从而在安全需求提取任务上取得了显著的性能提升。

(2)对比基于目前主流示例检索策略 RAG 方法所构建的少样本提示:

两种目前主流示例检索策略 RAG 方法所构建的少样本提示中,基于 TF-IDF 和语义相似度策略方法在性能存在小幅差异。从整体性能来看,本文方法相比于基于 TF-IDF 和语义相似度策略基线方法,本文方法在 GLM-3-Turbo 模型上的性能相比于最优的语义相似度策略方法有 0.66%的下降,但 ERNIE-4.0-8K 和 Qwen-Turbo 两个模型上相比于次优方法分别有 0.5%和 0.15%的提升,可见本文方法在整体性能上与语义相似度策略相近,并略有优势。

进一步从安全需求特征角度来看,本文方法在识别系统架构信息上的性能相比于基于 TF-IDF 和基于语义相似度方法较弱。而在提取系统功能方面,本文方法相比于次优方法在三个 LLM 上的 *F*1 值则分别提升了 2.48%、0.36%和 5.18%。在软件安全上,本文方法相比于次优方法则分别有 2.08%、6.14%和 0.59%的提升。可见本文方法相比于基于 TF-IDF 和语义相似度的 RAG 方法,在安全需求提取任务上的提升主要体现在系统功能和软件安全两个方面,而在系统架构方面则相对较弱。这主要是因为在需求文本中系统架构信息通常语义清晰且表达明确,而系统功能和软件安全信息则较为抽象,无法仅依靠语义进行识别,需要结合领域经验提取。因为基于 TF-IDF 和语义相似度的 RAG 方法分别倾向于选择存在更多相同词语和语义相似的示例,这两类在提取系统架构信息上的提升更为显著。相比之下,本文方法从安全特征角度识别的相似案例则可以为 LLM 提供更加切合的领域知识和经验,在提取系统功能和软件安全信息上具有更好的性能。

综上,本文方法的安全需求提取性能略优于基于语义相似度的 RAG 方法。具体优势体现在系统功能和软件安全信息的提取能力上,但是相比于基于文本特征的 TF-IDF 和语义相似度检索策略来说,在提取系统架构方面的性能有所不足。

由 RQ1 结果可见,现有通用的需求信息提取方法并不适用于安全需求信息的提取,该类方法无法

向 LLM 提供针对性领域知识。本文方法以及基于 TF-IDF 和语义相似度的 RAG 方法,则可以通过选取相似示例向 LLM 提供领域知识而有效提升 LLM 的安全需求提取性能。可见相似案例可以向 LLM 提供针对性的领域知识和经验,从而提升安全需求提取任务性能。而本文方法与基于 TF-IDF 和语义相似度的 RAG 方法在提取不同角度安全需求信息上的性能差异则说明,不同 RAG 方法的示例选取策略可以向 LLM 提供不同的领域知识,进而加强 LLM 在提取特定安全需求信息上的性能。如 Firesmith 所述[77],安全需求涉及功能、接口、数据等多个方面,包含很多不同的类别。而提取安全需求中不同类别的信息,亦有着不同的领域知识需要。因此,如何针对目标案例特征向 LLM 提供充分多样的领域知识,是未来进一步提升 LLM 的安全需求提取性能的关键。

5.2　研究问题 2:哪些因素会对本文方法提取的安全需求质量产生影响?

(1)RQ2.1:基础 LLM 的不同对本文方法提取的安全需求质量有何影响?

由表 7 可见,本文方法在使用 ERNIE-4.0-8K 模型时,本文方法在所有安全需求特征维度上的安全需求提取性能上都取得了最优。使用 GLM-3-Turbo 时的性能相比于 ERNIE-4.0-8K 有一定差距,但比使用 Qwen-Turbo 的性能更优。造成这一结果的主要原因在于 LLM 本身的性能差异,然而提取安全需求信息是一个综合性任务,LLM 的自然语言理解与生成能力、知识掌握、示例学习能力等方面均会对其性能产生影响。

从差距情况来看,本文方法使用 Qwen-Turbo 模型时与使用 ERNIE-4.0-8K 模型的差距可达 25.23%。从安全需求特征角度来看,三个 LLM 在软件安全上的性能差距最大,达到了 30%以上,而在提取系统架构和系统功能上的性能差距则在 20%左右。综上,基础 LLM 性能对本文方法的影响体现在所有三个维度信息的提取上,其中软件安全方面的影响最大。

综上,本文方法在综合性能更强的基础 LLM 上获得的性能提升更大,该结果与已有将 LLM 应用于软件工程任务的研究一致[23,64],说明基础 LLM 本身性能对包括安全需求提取在内的下游任务性能有很大影响,基础 LLM 在自然语言处理能力、知识掌握、示例学习等方面的性能提升都会进一步提升下游任务的性能。考虑到本文方法在不同 LLM 上的适用性,该结果也意味着方法在未来更先进 LLM 上的提升潜力。

(2)RQ2.2:示例数量对本文方法提取的安全需求质量有何影响?

本文方法在包含不同示例数量少样本提示词下的性能如图 7 所示。其中,ERNIE-4.0-8K 和 GLM-3-Turbo 两个模型在少样本提示词中包含 2 个示例时取得最优效果,而 Qwen-Turbo 则在包含 3 个示例时取得最优效果。三个 LLM 的最优性能相比于仅包含 1 个示例的少样本提示来说,均有高于 5%的性能提升。可见,在一定示例数量范围内,包含更多示例的少样本提示可以提升 LLM 的任务性能,这与其他相关研究的观察相一致。

从图 7 中也可以看出,三个 LLM 分别在 2-shot 和 3-shot 达到性能峰值之后,均随着少样本提示包含的示例数量增加出现了性能下降的趋势。通过对结果的审查发现,部分提取到的安全需求中包含了与任务和待分析需求无关的示例信息。由此推测,示例数量过多可能会引入无关信息干扰 LLM 对任务的理解,从而影响 LLM 的任务性能。

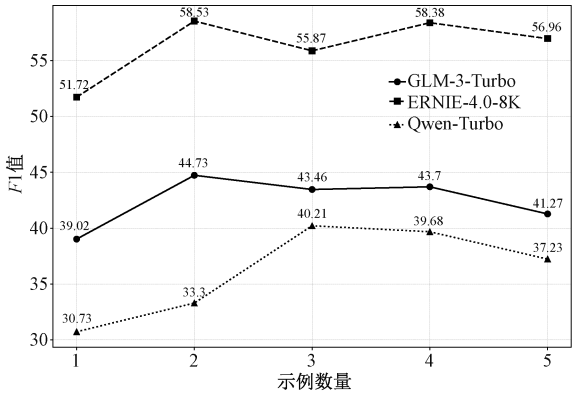


图 7　不同少样本提示示例数量下本文方法性能折线图

综上所述,少样本提示中包含的示例数量会对本文方法的安全需求提取性能造成影响,在本文所选的三个 LLM 上,少样本提示词中包含 2～3 个示例时性能最佳,之后随着示例数量的增加反而会造成安全需求提取性能的下降。该结果与现有研究一致,即示例达到一定数量后,少样本提示中更多的示例数量难以提升甚至会降低 LLM 的任务性能[75]。该结果说明,在使用本文方法进行安全需求提取时,选取少量的高质量示例即可获得更优的效果,同时少量的示例也意味着更低的成本。该结论可为后续本文方法应用提供示例数量选取的依据。

(3)RQ2.3:相似度案例选择策略对本文方法提取的安全需求质量有何影响?

通过应用相似度示例的五个选择策略，LLM提取安全需求的性能如表 8 所示。可以发现，基于 Hst_Strategy 构建的 1-shot 提示词在 GLM-3-Turbo 和 Qwen-Turbo 上，以及 2-shot 提示词在三个模型上均有着最好的性能，1-shot 提示词在 ERNIE-4.0-8K 上的性能略低于最优性能，但也非常接近。

其中，ERNIE-4.0-8K 上基于 Lst_Strategy 构建的 1-shot 提示词性能优于 Hst_Strategy，在 ERNIE-4.0-8K 和 Qwen-Turbo 上基于 Lst_Strategy 构建的 2-shot 提示词也表现出了优于 M_Strategy 的性能。Lst_Strategy 策略选择的相似度最低示例相比于 Hst_Strategy 选择的相似度最高示例，与目标需求在安全需求特征上的差异最大。本文结合现有研究结果分析认为，这是因为低相似度示例向 LLM 提供了与目标需求安全需求特征差异较大的信息[75]，提升了 LLM 的泛化能力，从而提高了 LLM 安全需求提取的性能。

表 8 不同相似度选择策略所获得的安全需求提取性能（*F*1）

示例数量	使用 LLM	Hst_Strategy	H_Strategy	M_Strategy	L_Strategy	Lst_Strategy
1-shot	ERNIE-4.0-8K	51.72	50.36	49.17	48.85	**52.13**
	GLM-3-Turbo	**39.02**	37.99	38.76	37.42	38.18
	Qwen-Turbo	**30.73**	30.04	28.33	29.70	27.23
2-shot	ERNIE-4.0-8K	**58.53**	54.72	53.80	52.28	55.10
	GLM-3-Turbo	**44.73**	40.81	39.12	39.16	38.81
	Qwen-Turbo	**33.30**	31.32	28.01	28.64	29.61

综上所述，高相似度案例可以提升 LLM 的安全需求提取效果，但 LLM 的安全需求提取效果并不完全随示例的相似度降低而降低，低相似度案例可能会向 LLM 提供更为多样的知识，从而通过提升 LLM 的泛化能力提高 LLM 的安全需求提取性能。该结果与现有研究的结论一致[75]。意味着在本文所提出的安全需求相似度示例选择策略下，可以通过选择最相似示例提升 LLM 的任务能力，或选择最不相似示例提升 LLM 的泛化能力，从而达到提升 LLM 的任务性能的目的。该结果也显示，不同相似度示例组合可能通过平衡 LLM 的任务能力和泛化能力，进一步提升 LLM 的安全需求提取任务性能。由于示例组合策略并非本文研究范畴，本文不做深入。

(4)RQ2.4：本文方法在不同应用领域下的安全需求提取性能如何？

本文针对方法在数据集中不同领域案例下的性能表现进行了方差分析(Analysis of Variance，简称 ANOVA)，该方法被广泛用于比较三个或以上组别之间的均值差异。因此本文使用 ANOVA 分析探究本文方法在三个 LLM 上不同应用场景下性能的差异显著性，分析结果如表 9。

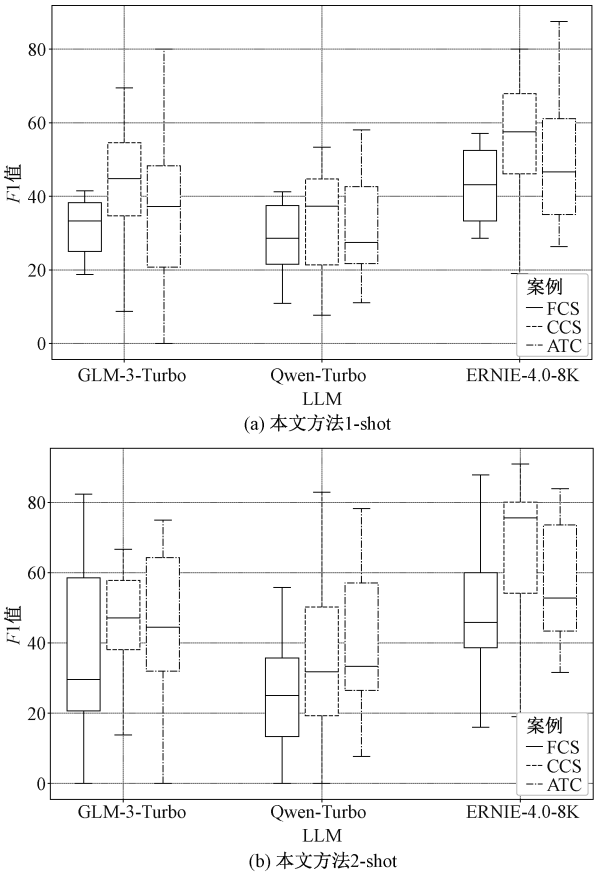
表 9 本文方法在不同应用场景 ANOVA 分析结果（*p* 值）

LLM	1-shot	2-shot
ERNIE-4.0-8K	0.549	0.063
GLM-3-Turbo	0.497	0.704
Qwen-Turbo	0.546	0.169

由表 7 可见，本文方法构造的 1-shot 和 2-shot 提示词在三个 LLM 上不同的应用场景下均没有显著的性能差异。这是因为本文方法构建的安全需求领域模型只涵盖了通用的安全关键软件领域概念而没有引入特定领域下的概念。

图 8 以箱型图形式展示了本文方法在三个应用场景下的性能，其中图 8(a)和 8(b)分别展示了使用 1-shot 和 2-shot 提示词的本文方法在三个应用场景下的性能。其中，来自车载电子领域的 CCS 案例的上四分位数、中位数、下四分位数在大多数情况下都是较高的，可见方法在车载电子领域的安全需求提取性能更为优秀。而来自航空装备领域的 FCS 案例在所有 LLM 的 1-shot 和 2-shot 方法上，都有着较低的上四分位数、中位数，说明本文方法在此类案例上的性能较弱。推测这是由 LLM 在不同领域的知识差异所导致，因为航空装备领域公开信息相较于车载电子领域更少，LLM 训练语料中两个领域的信息存在规模差异导致了 LLM 在不同应用领域上的性能差异。

综上，在不同应用领域上本文方法性能存在差异，但差异并不显著。差异具体体现在方法在公开信息更多的安全关键领域上有更好的应用效果，而在公开信息较少的领域则效果相对较弱。该结果表明在将本文方法迁移至其他领域时，需要注意 LLM 自身的领域知识水平对方法性能的影响。同时，由于本文关注于安全关键系统中的通用概念，在将本文方法应用于新领域时，仍需参照领域的权威标准



(a) 本文方法1-shot

(b) 本文方法2-shot

图 8　不同应用场景下安全需求提取性能箱型图

捕捉相关概念并建立领域概念模型,从而保证方法在目标领域的适用性和有效性。本文方法领域元模型与提示词元模型分离的设计,也在保证了将本文方法应用于不同安全关键领域时的良好扩展性。

(5)RQ2.5:本文方法中基于零样本提示的安全特征提取设计和 PCA 降维设计对本文方法提取的安全需求质量有何影响?

消融实验结果如表 10 所示。从方法整体性能的角度来说,基于零样本提示的安全特征提取设计和 PCA 降维设计对本文方法在所有基础 LLM 上对安全需求提取性能均有积极作用,最高提升分别可达 11.57%和 8.53%。且在 2-shot 上的性能提升比 1-shot 更为显著,意味着基于零样本提示的安全特征提取设计和 PCA 降维设计均可引导本文方法为少样本提示选择更优的示例,从而获得安全需求提取性能的提升。

从基础 LLM 角度来说,基于零样本提示的安全特征提取设计和 PCA 降维设计在 ERNIE-4.0-8K 上的提升较 GLM-3-Turbo 和 Qwen-Turbo 来说更为显著,意味着包含基于零样本提示的安全特征提取设计和 PCA 降维设计的本文方法在性能更优的基础 LLM 上可以获得更多的提升。

表 10　消融实验结果(*F1*)

基础 LLM	示例数量	方法设置	安全需求特征维度			整体性能
			系统架构	系统功能	软件安全	
ERNIE-4.0-8K	1-shot	本文方法	63.08	44.59	47.00	51.72
		w/o 零样本提示设计	62.15(−0.93)	33.70(−10.89)	21.21(−25.79)	41.99(−9.73)
		w/o PCA 降维设计	70.13(+7.05)	37.34(−7.25)	31.09(−15.91)	48.48(−3.24)
	2-shot	本文方法	69.08	52.98	50.75	58.53
		w/o 零样本提示设计	66.67(−2.41)	38.05(−14.93)	27.75(−23.00)	46.96(−11.57)
		w/o PCA 降维设计	72.55(+3.47)	39.04(−13.94)	34.66(−16.09)	50.00(−8.53)
GLM-3-Turbo	1-shot	本文方法	53.26	33.78	18.35	39.02
		w/o 零样本提示设计	55.95(+2.69)	25.49(−8.29)	16.86(−1.49)	36.67(−2.35)
		w/o PCA 降维设计	56.65(+3.39)	30.67(−3.11)	16.52(−1.83)	38.94(−0.08)
	2-shot	本文方法	55.10	37.23	41.79	44.73
		w/o 零样本提示设计	55.67(+0.57)	22.94(−14.29)	21.51(−20.28)	35.44(−9.29)
		w/o PCA 降维设计	56.84(+1.74)	32.66(−4.57)	26.94(−14.85)	41.34(−3.39)
Qwen-Turbo	1-shot	本文方法	43.69	29.90	15.42	30.73
		w/o 零样本提示设计	39.26(−4.43)	25.42(−4.48)	6.23(−9.19)	25.85(−4.88)
		w/o PCA 降维设计	44.57(+0.88)	25.79(−4.11)	10.56(−4.86)	28.71(−2.02)
	2-shot	本文方法	41.16	35.27	20.20	33.30
		w/o 零样本提示设计	32.81(−8.35)	28.03(−7.24)	8.89(−11.31)	25.21(−8.09)
		w/o PCA 降维设计	47.82(+6.86)	25.18(−10.09)	17.32(−2.88)	31.25(−2.05)

从特征维度角度来说,基于零样本提示的安全特征提取设计在提取系统架构信息软件安全信息上的提升较少,而在提取系统架构、系统功能信息上的提升更为显著。该结论与 RQ1 结论一致,可见该设计使本文方法能够从安全特征角度识别相似案例,通过少样本提示为 LLM 从安全角度提供更为切合的领域知识。

同样在特征维度,PCA 设计消融实验结果显

示,本文方法在提取系统架构方面安全需求信息上普遍存在性能下降情况,而在系统功能和软件安全信息上则出现了显著的性能提升。这是因为 PCA 降维后的特征向量保存了更多功能和安全方面的信息,同时弱化了系统架构方面的信息,使得在选择示例时更多地考虑了系统功能和软件安全信息的相似性。

综上,本文方法中基于零样本提示的安全特征提取设计和 PCA 降维设计对于安全需求提取性能存在积极影响。其中二者均在提取系统功能和软件安全维度信息上对方法有着显著的提升。该结果证明了基于零样本提示的安全特征提取设计和 PCA 降维设计的有效性。

6 案例研究

本节基于数据集中的 FCS 案例开展案例研究,从工程师的安全性实践角度来评估本文方法所提取的安全需求信息,进一步探究本文方法的实用性。

FCS 案例整理自 AFTI/F-16 电传操纵验证机飞行控制系统的公开资料,飞行控制系统以飞行员的操控指令、传感器所收集的外界物理信息和飞机各组件运行状态为输入,计算实现对飞机的飞行姿态控制。案例中共包含有 15 条具体需求,包括对系统架构、系统功能、软件安全方面信息的描述。该案例来自于真实工业实践,具有较为复杂的需求描述,且包含大量安全需求分析所需考虑的信息,因此基于该案例开展案例研究在一定程度上可以代表工业实践中设计的关注要点。

6.1 案例研究过程

案例研究基于先前实验结果开展,基础模型为 ERNIE-4.0-8K 模型,使用 2-shot 少样本提示词和 Hst_Strategy 相似度示例选择策略。

所有 15 条需求的安全需求信息提取结果分别由四名飞行控制系统软件工程师独立评估,四名工程师均具有安全需求分析经验,以及至少 2 年的飞控系统软件开发和需求分析经验。工程师需要结合自身领域知识和经验并依据其所关注的安全需求信息,识别提取结果中正确、错误、遗漏的安全需求信息,之后由第一作者将四位工程师的识别结果合并为最终结果,并以此为基准评估本文方法捕捉工程师所关注的安全需求信息能力,从而探究方法的实用性。在此基准之上,使用准确率、召回率、$F1$ 值三个指标对本文方法进行综合评估,指标的计算方法见公式(10)、(11)和(12)。

6.2 案例研究结果

表 11 展示了本文方法从 FCS 案例中提取得到的安全需求信息数量以及捕捉工程师所关注安全需求信息的效果。本文方法共从 FCS 案例中捕捉得到 256 条安全需求信息,其中系统架构信息、系统功能信息、软件安全信息分别捕捉得到 113 条、105 条、38 条。

由评估结果可见,本文所提取的安全需求信息在系统架构、系统功能、软件安全三个方面上的准确率均达到了 90%以上,说明本文方法所提取的安全需求信息几乎都符合工程师的关注要点,可见本文方法可以有效捕捉工程师所关注的安全需求信息。

表 11 案例研究评估结果

安全需求特征维度	LLM 提取安全需求信息数量	本文方法效果		
		Precision	*Recall*	*F1*
系统架构	113	94.69	79.58	86.48
系统功能	105	98.10	72.41	83.32
软件安全	38	97.37	57.58	72.36
合计	256	96.48	72.52	82.80

相比之下,本文方法在召回率上的性能则较弱,三个方面信息的召回率均在 80%以下,说明本文方法在识别工程师所关注安全需求信息的全面性上存在局限,存在遗漏信息的情况。其中遗漏最为严重的是软件安全信息,这与 RQ1 的结果一致。然而尽管如此,本文方法仍然能够捕捉到工程师所关注的大部分安全需求信息,这表明其安全需求提取实践中的应用潜力。

通过对比案例研究与 RQ1 的 $F1$ 结果发现,案例研究的评估结果更佳,造成该差异的原因在于评估基准的差异。通过进一步检查工程师的评估结果,发现工程师更关注方法所提取信息的正确性,对于输出格式正确性任务和安全性机制的划分关注较少,这是导致该差异的主要原因。

通过分析工程师的评估结果,本文进一步对方法提取结果的错误和遗漏原因进行探究。其中错误主要体现在两方面:LLM 领域知识的不足而导致的安全需求信息分类错误,例如将“作动器接口装置”划分为硬件组件而非接口;以及上下文信息理解不足而导致的关系提取错误。

遗漏问题则主要体现在两个方面:LLM 泛化能力不足而导致的隐含信息遗漏、LLM 领域知识不足而导致的遗漏。以图 9 中展示的需求提取结果为例,通常来说工程师或 LLM 会结合上下文中的飞

行控制计算机推测出运行于其上的飞行控制软件,然而本文方法在提示词中明确要求 LLM 仅关注于案例中包含的信息,限制了 LLM 的泛化能力,虽然显著减轻了 LLM 幻觉对提取结果的影响,但也导致方法依据上下文推理隐含信息的能力受限。由此可见,虽然 LLM 的泛化能力可能向提取结果中引入幻觉影响,但安全需求提取仍然需要泛化能力来保证提取结果的全面性,因此如何有效地利用 LLM 的泛化能力是未来研究的关键。另一方面,由于 LLM 自身领域知识不足,方法未能识别侧杆、油门、作动装置为硬件组件,同时未能将计算结果识别为运行数据。针对 LLM 领域知识不足的问题,正如 RQ1 和 RQ2.1 的所述,未来可以通过进一步探究如何向 LLM 提供更为充分多样的领域知识,或使用综合性能更加强大的 LLM 来得以解决。

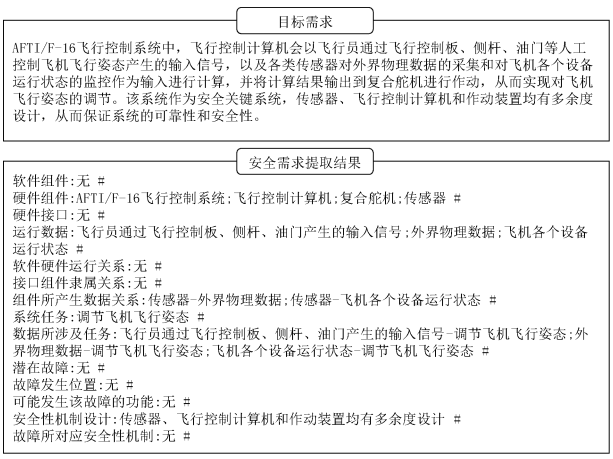


图 9　案例研究结果

综上,虽然案例研究结果显示本文方法仍然存在安全需求信息提取错误和遗漏的风险,但从准确性和全面性角度来说与工程师的实践关注要点与具有很高的一致性。虽然案例研究结果表明了本文方法的应用潜力,但考虑到工业实践对提取结果正确性和全面性的严苛要求,以及 LLM 领域知识不足等问题的影响,本文方法目前并不具备取代工程师独立完成任务的能力,仅可作为辅助手段协助工程师进行安全需求提取。正如其他基于 LLM 的软件工程任务解决方案[27],在将其输出结果应用于下游任务前,工程师的人工检查和修正是不可或缺的。

7　讨　论

本节对实验结果的实践启示和局限性进行讨论,并进一步讨论本文实验的有效性威胁。

7.1　有效性分析

本节对本文实验的有效性威胁从结构有效性、内部有效性、外部有效性三个角度展开讨论,并进一步讨论了本研究的实践意义和局限性。

7.1.1　结构有效性

为减轻结构有效性威胁。本文整理了三个不同领域的安全关键系统案例组成数据集,并在其上开展实验。同时,本文使用 $F1$ 指标评估方法提取安全需求信息的性能,从而综合考虑所提取信息的准确性和全面性。

7.1.2　内部有效性

内部有效性威胁主要来自于在评估流程中的人工参与。为了尽可能减轻内部有效性威胁,本文首先将 LLM 提取得到的安全需求与领域专家的分析结果进行文本比对,识别 LLM 正确获取且表述一致的情况,之后由两位领域专家进一步进行审查以检查内容与基准安全需求一致但表述有差异的识别结果,并将其标注为与基准安全需求一致,从而形成最终的评估结果,保证评估结果的正确性。

7.1.3　外部有效性

本文外部有效性威胁主要在于两点:(1)本文实验在多大程度上可以代表现实工业实践?(2)本文实验结果是否可以推广到其他安全关键系统领域?

对于第一个问题,本文实验所使用案例基于公开信息由工程师按当前工业领域需求文档要求编写而成。虽然规模存在差异,但尽可能保证了本文案例在文本特点上与当前现实工业案例的一致。在实验过程中,本文按照自然段对文档拆分开展实验,考虑到 LLM 的上下文长度限制,现实实践中通常也需要将文档拆分后进行提取。因此本文实验案例可以代表现实工业案例。

对于第二个问题,本文实验选择的航空装备、车载电子、民用航空三个领域案例虽然无法完全代表所有安全关键领域,但是本文方法所提取的安全需求信息整理自通用标准以及来自航空航天、汽车、装备领域所依照的权威标准。虽然不同安全关键领域存在不同的关注点,但是本文整理的通用信息仍然能覆盖其中的大部分关注点。因此本文实验结果对其他安全关键系统领域仍然具有参考价值。

7.2　局限性

本文的局限性主要体现在以下方面:

(1)有限的案例规模。虽然本文数据集中所包含的案例均整理自真实的具体安全关键系统案例,但受限于有限的公开资料,数据集中的案例与真实

安全关键系统需求相比规模有限。但由于本文案例均依照现实工业需求文档编写要求进行编写,且在文档的拆分过程中也保证了与现实工业实践的一致,该局限并不会对本文方法在现实工业案例上的应用情况产生影响。未来也将向数据集中追加更大规模的案例。

(2)通用的安全需求概念。为保证本文在不同安全关键领域上的通用性,本文参照相关文献和标准识别了的通用安全需求概念并建立领域模型,为安全需求提取提供指导。这意味着本文方法无法直接被用于捕捉通用概念之外的信息,如有需要则需对领域模型进行调整补充。

(3)未对方法的效率提升效果作进一步探究。鉴于当前安全需求提取实践依赖人工,本文方法作为自动化方法能否通过辅助工程师提取安全需求从而提升任务效率是一个重要的问题。然而本文方法关注于探究 LLM 在安全需求提取任务中的应用潜力,实践中的效率提升并非本文的当前核心关注点,未来会进一步探究。

8 相关工作

本文工作关注于通过自动生成的提示词与 LLM 交互从而实现安全需求的自动提取,因此,本文从安全需求提取、提示词自动生成和检索增强生成三个方面对当前的一些代表性工作进行总结。

8.1 安全需求提取

现实工业实践中广泛使用的包括 PHA[7]、FMEA[8]等安全需求获取方法,均需要工程师人工提取历史系统和目标系统中的安全需求相关信息,并在此基础上完成通用软件安全需求裁剪和特定软件安全需求获取工作[5,78]。相关标准和方法中均没有明确规定所需提取的信息和具体流程,因此该任务对工程师的分析经验和系统理解都提出了很高的要求,具有很大的难度。因此出现了一系列基于模板的安全需求提取方法,如 Safety-RUCM[10]、RM-RNL[79]等方法,这些方法通过用例模板引导工程师捕捉提取关键的安全需求信息,降低安全需求提取难度,同时通过受限自然语言降低安全需求描述的二义性,但是此类方法仍然对使用者的系统理解有着很高的要求,且当系统规模提升时,这些方法通常会带来庞大的工作量。

近年来,一系列基于机器学习和自然语言处理的自动化需求提取方法被提出。此类方法大多关注于一般软件需求的提取任务。例如 Haris 等人[80]将需求描述的语法特征整理为需求样板,使用 Spacy NLP 基于需求样板从需求规格说明书中提取需求描述语句。Jin 等人[81]关注嵌入式系统的问题框架建模问题,使用 BERT 模型从需求文本中提取实体和关系,并基于这些信息构造问题框架模型。

亦有部分方法关注于系统的信息安全(Security)需求信息提取,例如李广龙等人[17]提出了一种从英文自然语言描述中获取安全需求的方法,该方法使用基于深度学习的安全目标多标签分类模型识别需求语句的安全目标,同时使用 NLP 方法从需求文本中提取实体、实体关系,根据安全目标使用提取信息匹配安全需求模板,从而实现安全需求的自动获取。Riaz 等人[15]使用 k-NN 模型识别文本中的安全相关句子及其安全目标,并提供上下文特定的模板以辅助工程师捕捉句子中的安全需求信息。

随着 LLM 的快速发展,其强大的自然语言处理能力让需求提取领域得到了新的发展。比如 Das 等人[71]针对从需求中识别命名实体的问题,提出了一种名为 zsl-ner 的基于 GPT-3 模型的零样本提示方法,该方法可以识别需求中的实体并进行分类,相比于先前基于机器学习和神经网络方法有大幅的性能提升。Jin 等人[65]基于嵌入相似度选取与目标需求文本相似的示例,构造少样本提示从信息物理系统(Cyber-Physical System,CPS)需求文本中提取问题框架建模所需的实体和关联关系。

由于 LLM 具有丰富的人类知识,目前也有相关研究探究了 LLM 在安全分析任务上的应用。比如 Nouri 等人[82-83]针对危害分析与风险评估(Hazard Analysis and Risk Assessment,HARA)任务提出了一种基于提示词的流水线原型,并在工业环境下探究了流水线效率和 LLM 在安全分析任务上的局限。Xia 等人[84]使用 LLM 进行风险分析和文本生成辅助人工进行 FMEA 分析。Qi 等人[85]和 Diemert 等人[86]则分别针对 LLM 在系统理论过程分析(System Theoretic Process Analysis,STPA)和危害分析中的辅助作用进行探究。这些方法关注于应用 LLM 的泛化能力来直接根据已有信息生成安全分析结果。

综上所述,当前安全领域中对 LLM 的应用主要关注于利用 LLM 泛化能力直接进行安全需求分析,而 LLM 的幻觉影响则会对其分析结果合理性造成影响。而本文关注的安全需求提取任务当前仍然缺乏有效的自动化方法支持安全(Safety)需求提

取,同时缺少相关数据集用于模型训练,导致当前安全需求获取任务的门槛高、工作量大的问题仍然无法解决。因此,本文针对当前现状提出一种基于LLM无需训练的提示学习方法用于安全需求提取任务。

8.2　提示词自动生成

提示词指定了LLM所需完成的任务,其质量是LLM性能的重要影响因素[27]。手动设计的高质量提示词通常需要对LLM和任务领域有深入的了解,并通过大量的试错迭代。为降低提示词设计过程中的成本,当前有大量研究关注于提示词自动生成任务,根据生成方法不同,该方面研究可以分为两大类别:基于模板的提示词生成和基于生成式模型的提示词生成。

(1)基于模板的提示词生成

当前有大量研究关注提示词构造技巧,提出了包括少样本提示、思维链提示等可以显著提升LLM性能的提示方法。这些提示方法通常有着固定的结构,对其中部分的细节进行修改即可产生不同的提示词,处理不同的下游任务。基于模板的提示词生成方法正是来自于这种思想,将各类提示方法抽象为提示词模板,通过模板捕捉提示词中的可变信息,从而自动生成高质量提示词。例如Brown等人[20]指出了零样本提示和少样本提示中的基本结构,为之后的提示词模板构建提供基础。White等人[87]归纳整理了常用的提示技巧,并形成提示词模式目录,提高了提示词的可重用性和不同领域的适用性。Sorensen等人[88]提出一种无需标注示例和直接访问模型的提示词模板选择方法,该方法从一组提示词模板中选择使输入和模型输出之间互信息最大的模板,进行提示词构建。Clariso等人[89]针对包括大语言模型、文本生成音乐模型、文本生成图像模型等在内的生成式人工智能模型提示词特征进行分析并建立了提示词元模型,并基于元模型设计了一种领域特定语言(Domain Specific Language,DSL)用于构建提示词,该DSL支持对各类模型的提示词进行描述,并且可以实现不同LLM之间的迁移。

此类方法大大降低了构造高质量提示词的门槛和工作量,减少了构造高质量提示词的迭代试错。此类方法仍然需要人工介入,因此方法具有很高的灵活性,所生成的提示词可被用于处理不同的下游任务,但基于模板构造得到的提示词在框架层面不会发生变化,意味着生成的提示词可能无法在其所支持的所有下游任务上达到最优性能。

(2) 基于生成式模型的提示词生成

基于生成式模型的提示词生成方法主要依赖于LLM的自然语言生成能力,引导LLM生成高质量的提示词。例如Zhou等人[28]使用GPT-3等黑盒LLM根据输入输出示例逆向生成候选指令,并将候选指令结合测试样例作为提示词输入给LLM,根据LLM输出的正确情况得到高质量的候选指令,并使用迭代蒙特卡洛搜索方法,驱使LLM生成更多相似的指令变体并找到最优的提示词指令。但由于该方法仍然是在离散空间中搜索,意味着搜索到的指令仍然可能并非最佳指令。针对这个问题,Chen等人[90]提出一种名为InstructZero的基于贝叶斯优化的LLM指令优化方法,该方法使用LLM将软提示转化为指令,输入到LLM进行评估,评估结果通过贝叶斯优化得到新的软提示,通过反复迭代生成最优的任务指令。Pryzant等人[74]则利用LLM来发现其生成的提示词中的缺陷,并使用LLM识别的缺陷对所生成提示词进行优化生成新的提示词,通过不断迭代来生成最优提示词。

以上文章均关注于零样本提示,对于少样本提示和思维链提示来说,LLM的性能还会受到示例的影响。针对这个问题,Zhang等人[50]认为示例的多样性对LLM性能存在影响,并提出了一种Auto-CoT的思维链提示自动生成方法,该方法将数据集中的问题聚类为多个簇,并从中选择最具代表性的问题,通过告知LLM逐步思考的方式生成问题的推理链,作为示例来构造相应的提示词。

基于生成式模型的方法相比于基于模板的方法,可以进一步在指令和层面找到更优的提示,但此类方法均需要与LLM进行多次交互,相比于基于模板的方法来说有着更高的使用成本。

综上所述,考虑到安全需求提取任务需要大量相关领域知识支撑,而当前基于模板的提示词生成方法大多从提示方法的角度关注于简单任务的提示词的生成,基于生成式模型的方法则主要关注于简单任务的指令优化,这些方法由于缺乏对特定领域知识的考虑,而难以应用于安全需求提取任务。因此,本文首先结合安全关键软件领域知识,设计专门的安全需求提取提示词生成方法。

8.3　检索增强生成

当前LLM在自然语言理解与生成能力上展现出了强大的能力,可以生成媲美人类的流畅文本,但其所生成的文本经常与期望或事实存在很大的差距,甚至提供虚假信息,这也被称为幻觉[91]。RAG

通过结合信息检索和文本生成技术，从外部知识库中检索与用户查询内容相关的信息，与用户的原始问题结合形成一个全面的提示词，引导 LLM 生成符合预期且正确的答案，从而避免 LLM 幻觉[92]。

从 RAG 的框架结构角度来进行划分，RAG 可被分为 Naive RAG[92]、Advanced RAG[93]、Modular RAG[94-95]三类。其中，Naive RAG 仅包含索引、检索、生成三个步骤，是最简单的 RAG 方案。然而，该方案可能因检索质量低等问题导致生成回答质量较低。因此 Advanced RAG 针对检索阶段进行了优化，此类方法显著提升了检索内容的质量及其与问题的相关度，从而提升了此类方法生成回答的质量。然而以上两类 RAG 均为链式结构，灵活性较低，Modular RAG 则可以通过组合多种模块使方法更加灵活应对各类需求。

9 总结与展望

全面准确的安全需求提取是安全关键软件安全性的重要保证，但随着软件规模和复杂度的提升，安全需求提取任务的工作量和对领域知识及经验的要求也随之提升。针对这一现状，本文提出了一种面向功能安全需求提取的模型驱动提示词生成与优化方法，将安全需求提取所涉及的相关领域知识表示为元模型以自动生成和优化提示词，引导 LLM 结合领域知识与历史经验完成软件安全需求提取任务。通过使用来自航空装备、民用航空和车载电子领域的需求案例对本文方法指导 LLM 完成安全需求提取任务的效果进行探究发现，本文方法可以有效向 LLM 提供领域知识与经验，从而在提取安全需求信息上获得性能提升。同时讨论了基础 LLM、少样本提示包含的示例数量、相似度案例选择策略、应用场景、方法设计对本文方法的性能影响，并进一步开展案例研究对本文方法的适用性进行探究，为本文方法的应用提供了重要的参考。

本文认为，经典的模型驱动方法与提示工程的结合，可以更好地将 LLM 的自然语言处理能力和丰富的知识应用于需求工程任务。因此，本文未来的工作主要是两个方面：(1)尝试将模型驱动的提示词生成迁移到通用的需求提取任务；(2)探索如何从提示工程的角度提高需求提取任务的性能。

参 考 文 献

[1] Martins L E G, Gorschek T. Requirements engineering for safety-critical systems: A systematic literature review. Information and Software Technology, 2016, 75: 71-89

[2] Leveson N G. Software safety in embedded computer systems. Communications of the ACM, 1991, 34(2): 34-46

[3] S-18 Aircraft and Sys Dev and Safety Assessment Committee. Guidelines for conducting the safety assessment process on civil aircraft, systems, and equipment. ARP4761A. Warrendale, USA, 2023

[4] Department of Defense. Department of Defense standard practice: system safety. MIL-STD-882E. Washington, USA, 2012

[5] Huang Z Q, Xu B F, Kan S L, et al. Survey on embedded software safety analysis standards, methods and tools for airborne system. Journal of Software, 2014, 25(2): 200-218 (in Chinese)
(黄志球，徐丙凤，阚双龙等. 嵌入式机载软件安全性分析标准、方法及工具研究综述. 软件学报，2014，25(2)：200-218)

[6] RTCA Special Committee 205, EUROCAE Working Group 71. Software considerations in airborne systems and equipment certification. DO-178C. Washington, USA, 2011

[7] Flaus J-M. Preliminary hazard analysis//Flaus J-M. Risk Analysis: Socio-technical and Industrial Systems. Hoboken, USA: John Wiley & Sons, Ltd, 2013: 151-178

[8] Goddard P L. Software FMEA techniques//2000 Annual Reliability and Maintainability Symposium. Los Angeles, USA, 2000: 118-123

[9] Lee W S, Grosh D L, Tillman F A, et al. Fault tree analysis, methods, and applications ? A review. IEEE Transactions on Reliability, 1985, R-34(3): 194-203

[10] Wu X, Liu C, Xia Q. Safety requirements modeling based on rucm//2014 IEEE Computers, Communications and IT Applications Conference. Beijing, China, 2014: 217-222

[11] Zhang H, Yue T, Ali S, et al. A restricted natural language based use case modeling methodology for real-time systems//2017 IEEE/ACM 9th International Workshop on Modelling in Software Engineering (MiSE). Buenos Aires, Argentina, 2017: 5-11

[12] Liu J, Wang H, Zheng W. A safety modelling method for high-speed train control systems based on uml extension//2020 Chinese Automation Congress (CAC). Shanghai, China, 2020: 317-321

[13] Wang H, Zhong D, Zhao T, et al. Integrating model checking with sysml in complex system safety analysis. IEEE Access, 2019, 7: 16561-16571

[14] Xiao M, Dong Y, Gou Q, et al. Architecture-level particular risk modeling and analysis for a cyber-physical system with aadl. Frontiers of Information Technology & Electronic Engineering, 2020, 21(11): 1607-1625

[15] Riaz M, King J, Slankas J, et al. Hidden in plain sight: Automatically identifying security requirements from natural language artifacts//IEEE 22nd International Requirements Engineering Conference (RE). Karlskrona, Sweden, 2014: 183-192

[16] Hibshi H, Jones S T, Breaux T D. A systemic approach for natural language scenario elicitation of security requirements. IEEE Transactions on Dependable and Secure Computing, 2022, 19(6): 3579-3591

[17] Li G L, Shen G H, Huang Z Q, et al. Security requirements elicitation method for natural language requirements. Journal of Chinese Computer Systems. 2023, 44(12): 2646-2655 (in Chinese)
(李广龙, 沈国华, 黄志球等. 一种面向自然语言需求的安全需求获取方法研究. 小型微型计算机系统, 2023, 44(12): 2646-2655.)

[18] Ali N, Hussain M, Hong J E. SafeSoCPS: A composite safety analysis approach for system of cyber-physical systems. Sensors, 2022, 22(12): 4474

[19] Cheligeer C, Huang J, Wu G, et al. Machine learning in requirements elicitation: A literature review. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2022, 36: e32

[20] Brown T B, Mann B, Ryder N, et al. Language models are few-shot learners//Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS). Virtual, 2020: 1877-1901

[21] Touvron H, Lavril T, Izacard G, et al. LLaMA: Open and efficient foundation language models. arXiv:2302.13971, 2023

[22] Bai J, Bai S, Chu Y, et al. Qwen technical report. arXiv: 2309.16609, 2023

[23] Chen K, Yang Y, Chen B, et al. Automated domain modeling with large language models: a comparative study//ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS). Västerås, Sweden, 2023: 162-172

[24] Wang J, Huang Y, Chen C, et al. Software testing with large language models: survey, landscape, and vision. arXiv: 2307.07221, 2024

[25] Chaaben M B, Burgueño L, Sahraoui H. Towards using few-shot prompt learning for automating model completion//2023 IEEE/ACM 45th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER). Melbourne, Australia, 2023: 7-12

[26] Bajaj D, Goel A, Gupta SC, et al. MUCE: A multilingual use case model extractor using gpt-3. International Journal of Information Technology, 2022, 14(3): 1543-1554

[27] Jin D M, Jin Z, Chen X H, et al. ChatModeler: A human-machine collaborative and iterative requirements elicitation and modeling approach via large language models. Journal of Computer Research and Development, 2024, 61(2): 338-350. (in Chinese)
(靳东明, 金芝, 陈小红等. ChatModeler:基于大语言模型的人机协作迭代式需求获取和建模方法. 计算机研究与发展, 2024, 61(2): 338-350.)

[28] Zhou Y, Muresanu A I, Han Z, et al. Large language models are human-level prompt engineers. arXiv:2211.01910, 2023

[29] Feng S, Chen C. Prompting is all you need: Automated android bug replay with large language models//Proceedings of the IEEE/ACM 46th International Conference on Software Engineering (ICSE). Lisbon, Portugal, 2024: 1-13

[30] Qu C, Dai S, Wei X, et al. Tool learning with large language models: a survey. arXiv:2405.17935, 2024

[31] Zhao W, Zhou K, Li J, et al. A survey of large language models. arXiv:2303.18223

[32] Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models//Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS). New Orleans, USA, 2024: 24824-24837

[33] Leveson N G. Systems theory and its relationship to safety//Engineering a Safer World: Systems Thinking Applied to Safety. Cambridge, USA: MIT Press, 2012: 61-72

[34] Hu L, Xie N, Kuang Z, et al. Review of cyber-physical system architecture//2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops. Shenzhen, China, 2012: 25-30

[35] Dezfuli H, Benjamin A, Everett C, et al. NASA system safety handbook: system safety framework and concepts for implementation-volume 1. Washington D. C., USA: National Aeronautics and Space Administration, Technical Report: NASA/SP-2010-580, 2011

[36] Software Engineering Standards Committee of the IEEE Computer Society. IEEE recommended practice for software requirements specifications. IEEE Std 830-1998. New York, USA, 1998

[37] Leveson N G. Software safety: why, what, and how. ACM Computing Surveys, 1986, 18(2): 125-163

[38] Goguen J A, Linde C. Techniques for requirements elicitation//Proceedings of the IEEE International Symposium on Requirements Engineering. San Diego, USA, 1993: 152-164

[39] Pacheco C, García I, Reyes M. Requirements elicitation techniques: a systematic literature review based on the maturity of the techniques. IET Software, 2018, 12(4): 365-378

[40] Wang H, Li J, Wu H, et al. Pre-trained language models and their applications. Engineering, 2023, 25: 51-65

[41] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need//Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS). Long Beach, USA, 2017: 6000-6010

[42] Devlin J, Chang M-W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT). Minneapolis, USA, 2019: 4171-4186

[43] Fan A, Gokkaya B, Harman M, et al. Large language models for software engineering: Survey and open problems//IEEE/ACM International Conference on Software Engineer-

ing: Future of Software Engineering (ICSE-FoSE). Melbourne, Australia, 2023: 31-53

[44] Hou X, Zhao Y, Liu Y, et al. Large language models for software engineering: a systematic literature review. arXiv: 2308.10620, 2024

[45] Jiang J, Wang F, Shen J, et al. A survey on large language models for code generation. arXiv:2406.00515, 2024

[46] Sahoo P, Singh A K, Saha S, et al. A systematic survey of prompt engineering in large language models: techniques and applications. arXiv:2402.07927, 2024

[47] Shukla S. Metamodeling: What is it good for? IEEE Design & Test of Computers, 2009, 26(3): 96-96

[48] Fatehah M, Mezhuyev V, Al-Emran M. A systematic review of metamodelling in software engineering. Recent Advances in Intelligent Systems and Smart Applications. Cham, Switzerland: Springer International Publishing, 2021: 3-27

[49] Yue T, Ali S, Zhang M. RTCM: A natural language based, automated, and practical test case generation framework// Proceedings of the 2015 International Symposium on Software Testing and Analysis (ISSTA). Baltimore, USA, 2015: 397-408

[50] Zhang Z, Zhang A, Li M, et al. Automatic chain of thought prompting in large language models. arXiv:2210.03493, 2022

[51] Allenby K, Kelly T. Deriving safety requirements using scenarios//Proceedings of the Fifth IEEE International Symposium on Requirements Engineering. Toronto, Canada, 2001: 228-235

[52] Leveson N G. Safety as a system property. Communications of the ACM, 1995, 38(11): 146

[53] Firesmith D. A taxonomy of safety-related requirements// Proceedings of the Workshop on Requirements for High Assurance Systems (RHAS'04). Kyoto, Japan, 2004: 11

[54] International Electrotechnical Commission. Functional safety of electrical/electronic/programmable electronic safety-related systems-Part 1: General requirements. IEC 61508-1. Geneva, Switzerland, 2010

[55] NASA Office of Safety and Mission Assurance. NASA Software Safety Standard. NASA-STD-8719.13C. Washington, USA, 2013

[56] ISO technical committees. Road vehicles-Functional safety-Part 2: Management of functional safety. ISO 26262-2. Geneva, Switzerland, 2018

[57] Arora C, Grundy J, Abdelrazek M. Advancing requirements engineering through generative Ai: Assessing the role of llms. Generative AI for Effective Software Development. Cham, Switzerland: Springer Nature Switzerland, 2024: 129-148

[58] An S, Zhou B, Lin Z, et al. Skill-based few-shot selection for in-context learning//Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP). Singapore, 2023: 13472-13492

[59] Jolliffe I. Principal component analysis. 2nd Edition. New York, USA: Springer New York, 2002

[60] Wang Y Y, Zhang M, Yang JR, et al. Research on Fairness in Deep Learning Models. Journal of Software, 2023, 34(9): 4037-4055 (in Chinese)
(王昱颖, 张敏, 杨晶然等. 深度学习模型中的公平性研究. 软件学报, 2023, 34(9): 4037-405.)

[61] Ferrari A, Spagnolo G O, Gnesi S. PURE: A dataset of public requirements documents//IEEE 25th International Requirements Engineering Conference (RE). Lisbon, Portugal, 2017: 502-505

[62] Hey T, Keim J, Koziolek A, et al. NoRBERT: Ttransfer learning for requirements classification//IEEE 28th International Requirements Engineering Conference (RE). Zurich, Switzerland, 2020: 169-179

[63] Luo X, Xue Y, Xing Z, et al. PRCBERT: Prompt learning for requirement classification using bert-based pretrained language models//Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering (ASE). Rochester, USA, 2023: 1-13

[64] Jin D, Zhao S, Jin Z, et al. An evaluation of requirements modeling for cyber-physical systems via llms. arXiv:2408.02450

[65] Khayashi F, Jamasb B, Akbari R, et al. Deep learning methods for software requirement classification: A performance study on the PURE dataset. arXiv:2211.05286

[66] Gao J Y. Aircraft fly-by-wire system and active control technology. Beijing: Beihang University Press, 2005 (in Chinese)
(高金源. 飞机电传操纵系统与主动控制技术. 北京:北京航空航天大学出版社, 2005.)

[67] Barfield A, Van VB, Anderson D. AFTI/F-16 advanced multimode control system design for task-tailored operations//Proceedings of the Aircraft Systems and Technology Conference. Dayton, USA, 1981: 1707

[68] Mackall D A. AFTI/F-16 digital flight control system experience. Washington, USA: National Aeronautics and Space Administration, Technical Report: 19840012524, 1983

[69] Yadav A K, Szpytko J. Safety problems in vehicles with adaptive cruise control system. Journal of KONBiN, 2017, 42(1): 389-398

[70] Yang Y. The research of an AADL-based approach to automotive real-time embedded system modeling[Master's Thesis]. Hunan University, Changsha, China, 2012 (in Chinese)
(杨阳. 基于 AADL 的车用嵌入式实时系统建模方法研究[硕士学位论文]. 湖南大学,长沙,中国, 2012.)

[71] Das S, Deb N, Cortesi A, et al. Zero-shot learning for named entity recognition in software specification documents//IEEE 31st International Requirements Engineering Conference (RE). Hannover, Germany, 2023: 100-110

[72] Sparck Jones K. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, 1972, 28(1): 11-21

[73] Zhao P, Zhang H, Yu Q, et al. Retrieval-augmented generation for ai-generated content: A survey. arXiv:2402.19473

[74] Pryzant R, Iter D, Li J, et al. Automatic prompt optimization with "gradient descent" and beam search. arXiv:2305.03495, 2023

[75] Pecher B, Srba I, Bielikova M, et al. Automatic combination of sample selection strategies for few-shot learning. arXiv:2402.03038

[76] Liu J, Shen D, Zhang Y, et al. What makes good in-context examples for gpt-3? arXiv:2101.06804

[77] Firesmith D. Engineering safety requirements, safety constraints, and safety-critical requirements. Journal of Object Technology, 2004, 3: 27-42

[78] Xu X, Bao X, Lu M, et al. A study and application on airborne software safety requirements elicitation//Proceedings of the 2011 9th International Conference on Reliability, Maintainability and Safety. Guiyang, China, 2011: 710-716

[79] Wang F, Yang Z B, Huang Z Q, et al. An approach to generate the traceability between restricted natural language requirements and aadl models. IEEE Transactions on Reliability, 2020, 69(1): 154-173

[80] Haris M S, Kurniawan T A. Automated requirement sentences extraction from software requirement specification document//Proceedings of the 5th International Conference on Sustainable Information Engineering and Technology. New York, USA, 2021: 142-147

[81] Jin D, Wang C, Jin Z. Automating extraction of problem diagrams from natural language requirement documents//IEEE 31st International Requirements Engineering Conference Workshops (REW). Hannover, Germany, 2023: 199-204

[82] Nouri A, Cabrero-Daniel B, Törner F, et al. Engineering safety requirements for autonomous driving with large language models//IEEE 32nd International Requirements Engineering Conference (RE). Reykjavik, Iceland, 2024: 218-228

[83] Nouri A, Cabrero-Daniel B, Torner F, et al. Welcome your new ai teammate: On safety analysis by leashing large language models//Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI. Lisbon, Portugal, 2024: 172-177

[84] Xia Y, Jazdi N, Weyrich M. Enhance fmea with large language models for assisted risk management in technical processes and products//IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA). Padova, Italy, 2024: 1-4

[85] Qi Y, Zhao X, Khastgir S, et al. Safety analysis in the era of large language models: A case study of stpa using chatgpt. arXiv:2304.01246

[86] Diemert S, Weber J H. Can large language models assist in hazard analysis? arXiv:2303.15473

[87] White J, Fu Q, Hays S, et al. A prompt pattern catalog to enhance prompt engineering with ChatGPT. arXiv:2302.11382, 2023

[88] Sorensen T, Robinson J, Rytting C, et al. An information-theoretic approach to prompt engineering without ground truth labels//Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL). Dublin, Ireland, 2022: 819-862

[89] Clarisó R, Cabot J. Model-driven prompt engineering//ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS). Västerås, Sweden, 2023: 47-54

[90] Chen L, Chen J, Goldstein T, et al. InstructZero: Efficient instruction optimization for black-box large language models. arXiv:2306.03082, 2023

[91] Huang L, Yu W, Ma W, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. arXiv:2311.05232, 2023

[92] Gao Y, Xiong Y, Gao X, et al. Retrieval-augmented generation for large language models: A survey. arXiv:2312.10997, 2024

[93] Ma X, Gong Y, He P, et al. Query rewriting for retrieval-augmented large language models. arXiv:2305.14283, 2023

[94] Peng W, Li G, Jiang Y, et al. Large language model based long-tail query rewriting in taobao search. arXiv:2311.03758, 2024

[95] Shao Z, Gong Y, Shen Y, et al. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. arXiv:2305.15294, 2023

SHAO Zhi-Jun, Ph. D. candidate. His main research interests include software engineering and safety-critical software modeling.

WU Ji, Ph. D., associate professor. His main research interests include safety-critical software modeling, verification and testing.

CAO Hong-Yu, M. S. candidate. Her main research interests include software engineering, software modeling and verification.

WANG Yan-Wei, M. S. candidate. His main research interests include software engineering and software testing.

SUN Qing, Ph. D., associate professor. Her main research interests include software engineering and personalized learning.

YANG Hai-Yan, M. S., lecturer. Her main research interest is software engineering.

GAO Yan-Hua, M. S., researcher. Her main research interests include software engineering and software safety.

XU Jian, M. S., researcher. His main research interests include safety-critical embedded software and software safety.

Background

When conducting software safety requirements analysis, it is necessary to comprehensively consider information from various aspects, including system architecture and system functionality. The process of extraction of this information is still done manually with the collaboration of system engineers and software engineers, which results in a significant investment of manpower and the risk of omission of safety requirements. How to reduce the risk of omission of safety requirements and improve the efficiency of the extraction process is an urgent problem to be solved.

Recently, a series of Large Language Models (LLMs), represented by ChatGPT, LLaMA, and Qwen, have developed rapidly. LLMs are pre-trained on massive corpora and have excellent natural language understanding abilities and rich human knowledge. They also have the characteristic of being ready to use without the need for additional training, providing automated solutions for software engineering tasks that lack public data, such as safety requirements extraction.

This paper proposes a model-driven prompt generation and optimization method for functional safety requirements extraction. The method extracts and represents relevant domain knowledge involved in safety requirements extraction as a meta-model to automatically generate and optimize prompts. The method first automatically generates zero-shot prompts to guide the LLM to obtain safety requirements features from target requirements. Then, it identifies similar cases from historical safety requirement based on the features to construct few-shot prompts, guiding the LLM to combine domain knowledge and historical experience to complete the software safety requirements extractions task.

This method reduces the demand for users to have domain knowledge of safety-critical software, and thus reduces the risk of omission of safety requirements due to engineers' lack of domain knowledge, and improves the efficiency of extraction process while ensuring the quality of safety requirements.