自动化软件工程中的大语言模型提示工程报告：基于当前实践与期望的指南

Alexander Korn
alexander.korn
@uni-due.de 杜伊斯堡-埃森大学 德国埃森

Lea Zaruchas
科隆大学 德国科隆

Chetan Arora
chetan.arora
@monash.edu 莫纳什大学 澳大利亚墨尔本

Andreas Metzger
andreas.metzger@uni-due.de
杜伊斯堡-埃森大学 德国埃森

Sven Smolka
sven.smolka@uni-due.de 杜伊斯堡-埃森大学 德国埃森

Fanyu Wang
fanyu.
wang@monash.edu 莫纳什大学 澳大利亚墨尔本

Andreas Vogelsang
andreas.vogelsang@uni-due.de
杜伊斯堡-埃森大学 德国埃森

arXiv:2601.01954v1 [cs.SE] 5 Jan 2026

摘要

大语言模型，尤其是 GPT 等仅解码器生成模型，正越来越多地用于自动化软件工程任务。这些模型主要通过自然语言提示进行引导，因此提示工程是影响系统性能和行为的关键因素。尽管这类模型在软件工程研究中的作用日益凸显，但与提示相关的决策却很少以系统、透明的方式记录，从而阻碍了不同研究之间的可复现性和可比性。为弥补这一缺口，我们开展了一项两阶段实证研究。首先，我们分析了自 2022 年以来发表于三大软件工程会议的近 300 篇论文，以评估当前对提示设计、测试和优化的报告情况。其次，我们调查了来自这些会议的 105 名程序委员会成员，以了解他们对大语言模型驱动研究中提示报告的期望。根据研究发现，我们制定了一套结构化指南，将报告要素分为必不可少、值得采用和极佳三类。结果显示，当前实践与审稿人期望之间存在显著偏差，尤其体现在版本披露、提示依据说明和效度威胁方面。我们提出的指南旨在推动基于大语言模型的软件工程研究提升透明度、可复现性和方法论严谨性。

计算机分类系统概念

• **软件及其工程**；•**通用与参考** →计算标准、征求意见稿和指南；调查与综述；

允许免费制作本作品的全部或部分内容的数字版或纸质版副本，供个人或课堂使用，但不得为营利或商业目的制作或分发副本，且副本首页必须附上本声明和完整引文。对于本作品中由作者以外的其他人拥有版权的组成部分，必须尊重其版权。注明出处后可以摘录。以其他方式复制、再版、发布到服务器或分发给他人，均须事先获得明确许可和/或支付费用。请发送邮件至 permissions@acm.org 申请许可。*FORGE '26*，巴西里约热内卢© 2026年 版权归所有者/作者所有。出版权已授权给 ACM。ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM https://doi.org/XXXXXXX.XXXXXXX

关键词

大语言模型、指南、提示、软件工程研究、调查

ACM 引用格式：

Alexander Korn、Lea Zaruchas、Chetan Arora、Andreas Metzger、SvenSmolka、Fanyu Wang 和 Andreas Vogelsang。2026年。《自动化软件工程中的大语言模型提示工程报告：基于当前实践与期望的指南》。载于《第三届*ACM*人工智能基础模型与软件工程国际会议论文集（*FORGE '26*）》。ACM，纽约，纽约州，美国，11 页。https://doi.org/XXXXXXX.XXXXXXX

1 引言

自动化软件工程旨在应用计算方法和工具，实现软件工程生命周期中各项活动的自动化。大语言模型（LLM），尤其是 GPT-3 [2]，等仅解码器生成模型，已迅速改变软件工程（SE）任务的自动化方式 [4]。这些模型既能流畅地使用自然语言“交流”，也能使用多种编程语言，因此适用于多种软件工程任务，例如代码合成、测试生成、缺陷修复、需求分析等 [9]。大语言模型主要由文本提示引导。提示，即提供经过精心设计的文本指令，已成为引导模型行为的主要接口，因此提示工程（*PE*）是影响大语言模型性能的关键因素 [8, 12, 13]。

近年来，仅解码器大语言模型在软件工程研究中的应用稳步增长。正如本文后文将展示的，我们发现，自 2022 年以来，软件工程领域三大顶级会议发表的论文中，近 300 篇利用此类模型自动化多种软件工程任务。然而，尽管这类模型已得到广泛应用，提示词如何构建、优化和评估，却鲜少以系统或透明的方式报告。

尽管提示在决定模型行为方面起着核心作用，但当前研究实践在如何记录和论证提示词方面缺乏一致性，而其他软件工程实验产物通常都有相应要求。目前尚无关于提示词设计、测试或优化的既定报告标准。因此，与提示词相关的决策往往报告不足或完全省略，从而限制了可复现性，

降低了研究之间的可比性，并最终阻碍了这一快速发展的领域取得进展。

鉴于基于大语言模型的软件工程研究具有新颖性且不断演进，仅凭专家意见自上而下地制定标准还为时过早。相反，我们认为，报告指南应源于对现有实践的观察以及研究社群自身的期望。要了解目前提示是如何报告的，以及应当如何报告，需要开展实证调查，并以文献分析和研究人员的见解为基础。

为满足这一需求，我们开展了一项多阶段实证研究，并围绕以下研究问题展开：

研究问题1：研究人员目前如何在软件工程研究论文中报告提示？我们分析了自2022年以来发表于ICSE、FSE和ASE的近300篇论文，以评估作者如何报告提示设计、验证和优化。

研究问题2：软件工程研究人员对软件工程研究论文中的提示创建、评估和报告有何期望？我们对上述会议的105名程序委员会（PC）成员进行了调查，以了解他们对提示报告实践的期望。

研究问题3：当前状况与期望的一致性如何？我们将明确表达的期望提炼为一份指南，并将其与当前的报告实践进行了比较。

本研究作出以下贡献：

（1）对近300篇软件工程研究论文中提示设计、测试和优化的当前报告方式进行分类和频率分析（研究问题1）。（2）从软件工程审稿人那里通过实证得出一组报告期望（研究问题2）。（3）通过比较分析揭示当前实践与社区期望之间的差距和一致之处（研究问题3）。（4）提出一份以实证证据为基础的指南，以提升基于大语言模型的软件工程研究的透明性、可复现性和可比性。

通过综合当前实践和审稿人期望，本研究旨在提高软件工程领域基于大语言模型的研究的方法学标准，并为未来工作提供更加透明且可复现的基础。

数据可用性

我们在研究中使用的所有数据以及用于分析这些数据的代码，均已在我们的复现包中提供[1]。

2 相关工作

近期的系统性文献综述（SLR）分析了大语言模型在自动化软件工程任务中的应用，旨在确定所使用的大语言模型、数据收集与准备方法、提示工程策略，以及优化和评估大语言模型性能的技术。此类系统性文献综述中，有几篇聚焦于特定的软件工程任务，例如需求工程 [5], 代码生成 [17], 程序修复 [18], 和软件测试 [16]。相比之下，Hou 等人 [4] 开展了一项涵盖整个软件开发生命周期的系统性文献综述。

尽管上述出版物为了解大语言模型在自动化软件工程中的使用方式提供了重要见解，但它们并未明确说明软件工程研究中应如何报告提示。

Trinkenreich 等人 [15] 呼吁软件工程研究社区制定报告指南，以确保持续保持研究的严谨性并发挥影响。具体而言，他们倡导在使用大语言模型时进行透明报告，包括明确说明模型及版本、提示策略以及人工监督机制。然而，他们的工作并未进一步提出具体且可操作的指南。

Baltes 等人 [1]提出了一套关于在软件工程中使用和评估大语言模型的指南。这些指南是在 2024 年国际软件工程研究网络（ISERN）会议上的专家讨论基础上制定的，遵循自上而下、以共识为导向的过程，涵盖工具架构、评估指标、基线和基准测试等广泛主题。具体而言，在提示报告方面，他们提出了八项建议，其中五项标记为必须 ，三项标记为 应当，内容涉及完整披露提示词（包括结构和格式）、说明提示设计的理由、提供复用文档、输入处理以及共享交互日志。这些指南已发布在一个开源代码库中[2]。

我们的工作通过实证评估软件工程研究中提示报告的当前状况，并了解程序委员会成员的期望，补充了这项工作。Baltes 等人的指南体现了专家共识，而我们的指南则自下而上地从近 300 篇软件工程研究论文中观察到的报告实践中提炼而成，并通过对顶级软件工程会议程序委员会成员开展的调查进行了验证。

此外，我们认为研究发现为社区持续推进实证研究实践标准化的工作作出了宝贵贡献。具体而言，ACM SIGSOFT 实证标准[3]目前为多种实证方法的研究设计与报告提供了结构化指南，但尚未纳入专门针对基于大语言模型的研究的标准。通过提供以实证为依据、针对具体任务的提示报告见解，我们旨在弥补这一空白，并支持制定未来的标准，以确保在软件工程中透明且可复现地使用大语言模型。

3 提示报告现状（研究问题1）

本研究问题旨在分析近期提出大语言模型驱动的软件工程方法的研究论文如何描述和评估提示。

为考察软件工程研究目前如何报告提示，我们对近期发表于顶级会议的论文进行了系统综述。此次综述旨在评估涉及大语言模型的实证研究中提示报告实践的覆盖范围、一致性和详尽程度。

我们选择了文献综述方法，以客观、全面地了解当前的实践现状。系统综述是一种成熟的研究证据综合方法，尤其适用于识别特定技术或工件（如提示词）在已发表的研究中的使用和报告方式所呈现的趋势、空白和差异 [6]。

[1]https://doi.org/10.5281/zenodo.16101751

[2]https://llm-guidelines.org/ [3] https://www2.sigsoft.org/EmpiricalStandards/

3.1 研究设计

我们通过系统性文献综述回答研究问题1（RQ1） [7]。通过考察顶级软件工程会议发表的论文，我们旨在识别提示文档中的常见实践、所提供的详细程度，以及报告中用于创建和评估提示词的技术。研究结果将突出潜在的报告空白，评估研究之间的不一致之处，并提供有关提示工程（PE）目前在软件工程研究中的应用方式的信息。本综述将作为理解提示报告现状的基础，并有助于为未来的研究建立最佳实践。

论文筛选：我们首先收集了自2022年以来发表于CORE排名[4]中三大软件工程会议的所有论文，以此开展论文筛选流程：IEEE/ACM国际软件工程会议（ICSE'22–ICSE'25）、ACM国际软件工程基础会议（FSE'22–FSE'24），以及IEEE/ACM自动化软件工程国际会议（ASE'22–ASE'24）。[5] 我们将2022年选为起始年份，是基于这样一种假设：在ChatGPT于2021年12月发布之前，仅解码器大语言模型并不常见。

我们将范围限定在这些会议，以聚焦于通常体现最高方法学标准和最新研究实践的发表渠道。为保持语料库的一致性，我们排除了期刊论文，因为与篇幅受限的会议论文相比，期刊论文篇幅更长、格式不同，可能导致报告行为存在显著差异。此外，鉴于期刊论文的评审周期较长，许多文章在我们开展分析时可能尚未具有相关性。

我们分三个步骤筛选这些论文，以确保数据集中只保留与本研究相关的论文。

(1) 按文档长度筛选：我们只考虑全文论文。页数少于 7 页的论文将被排除，因为短篇论文可能没有充分阐述方法、进行初步评估，或因篇幅限制而有所取舍。

(2) 按关键词筛选：为筛除不相关的论文，我们定义了一组预期会出现在所有相关论文中的关键词。未包含以下任何关键词的论文将被排除：大语言模型、大语言模型、大语言模型、生成式人工智能、生成式人工智能、*OpenAI*、*GPT*、*ChatGPT*、*Llama*、*Claude*、提示词、提示、提示。

(3) 基于大语言模型的筛选：最后，我们使用不同的大语言模型进一步筛选论文。我们提示大语言模型仅纳入满足以下条件的论文：a) 主要关注自动化软件工程任务；b) 使用生成式大语言模型（即仅解码器模型）；c) 开展原始研究（即不包括荟萃分析、文献综述等）。完整提示词如下。

系统提示词：基于大语言模型的筛选

你是一名开展文献综述的研究人员。我们将提供多篇学术论文的全文。你的任务是根据以下严格标准，判断是否应纳入每篇论文：

纳入标准：

- 该研究必须以自动化软件工程（SE）任务为主要关注点。
- 该研究必须使用生成式大语言模型（仅解码器架构，例如 GPT 模型）。 • 该研究必须是原始研究（即提出、评估或实现某种方法）。必须排除荟萃分析、文献评审和系统综述。

说明：

• 仅根据论文内容作出判断。 • 如果论文未明确满足所有标准，则将其排除。 • 只能回复一个单词："include"或"exclude"。

- 请勿提供解释、理由或其他文本。

示例：
{四个示例，包括论文标题、一句话摘要，以及是否纳入或排除该论文}

我们测试了不同的提示词，以进行基于大语言模型的筛选，采用的提示工程技术包括角色提示、零样本提示和少样本提示。在多个模型上测试了各种提示词后，我们根据最终提示词能否通过正确纳入符合纳入标准的论文来实现最高召回率，选定了最终提示词。我们使用相同的纳入标准，对 ICSE'22 至 ICSE'24 的所有论文进行人工筛选，以评估这一方法；这些标准也通过提示词提供给大语言模型。随后，我们将基于大语言模型的筛选结果与人工筛选结果进行了比较。

在最终筛选过程中，我们使用了三种不同的大语言模型（*gpt- 4.1-mini-2025-04-14*、*deepseek-v3-0324*和*ge mini-2.5-flash-preview- 05-20*），并分别通过相应的应用程序接口访问这些模型。所有模型的温度均设为 1.0。我们还测试了较低的温度设置，但并未因此获得更好的结果。我们没有配置其他设置。选择这三种模型的原因是：（a）在与其他模型的比较测试中，它们的性能最佳；（b）在受测模型中，它们的成本效益最高；（c）它们由不同的提供商提供。我们认为这一点有助于缓解潜在的特定提供商偏差，从而提升该方法的可信度。这些模型的精确率和召回率指标见表 1。

表现最佳的模型 *gpt-4.1-mini-2025-04-14*，召回率达到了 84.83 %。在最终筛选步骤中，只要三个模型中任意一个建议纳入某篇论文，我们就将其纳入。由于预筛选大幅减少了人工工作量，我们认为这种策略所达到的 98.28 % 综合召回率已足够。尽管这可能降低精确率，该方法仍优先保持较高的召回率，因为之后仍会进行人工数据提取，从而能够在后续阶段识别并排除任何假阳性。

表 2 概述了各筛选步骤前后论文的数量。

[4]https://portal.core.edu.au/conf-ranks/ [5]FSE'25 的论文集出版时间距离投稿截止日期太近，因此未能纳入。

表1：基于大语言模型的过滤性能

模型	精确率	召回率
gpt-4.1-mini-2025-04-14	68.75 %	94.83 %
deepseek-v3-0324	88.46 %	79.31 %
gemini-2.5-flash-preview-05-20	86.54%	77.59%
综合结果（≥ 1 共 3 项）	**67.86%**	**98.28%**

数据提取： 选定相关论文后，我们开始进行数据提取。我们创建了一份包含 11 个问题的提取表，这些问题列于表 3。提取表中详细说明了各个问题，并明确说明了如何作答；该提取表是我们复现包的一部分。

问题 E3–E5 和 E7–E9 为封闭式问题，只能回答 是、否或部分。只有当论文中使用了多个提示或大语言模型，但无法对所有提示或大语言模型给出一致答案时，才允许选择部分。问题 E1 为自由文本问题，而问题 E2、E6、E10 和 E11 则为开放文本问题，其答案受限于由本文作者在提取过程中制定的预定义答案集。问题 E10 仅包含软件开发生命周期（*SDLC*）阶段，而问题 E11 最初采用了从 Hou 等人 [4], 提取的所有任务，以便与其研究进行可比性比较。

为确保本文六位作者在数据提取过程中保持一致，我们进行了三轮提取，前两轮作为测试轮次。第一轮中，每位作者从 10 篇论文中提取数据。论文的分配存在重叠：每篇论文由两位作者评审，且每位作者所负责的论文集均与另外两位作者的论文集重叠。提取完成后，我们手动检查了差异，讨论了对提取表的误解，并完善了问题以提高其清晰度。

完成这些初步改进后，我们进行了第二轮筛选；每位作者再次收到 10 篇论文，分配方式与第一轮类似，存在重叠。数据再次由各自独立提取，并讨论了所有遗留的疑问，以最终确定提取表，并统一所有人对问题的理解，确保数据提取的一致性。

在最后一轮提取中，我们将筛选步骤后剩余的所有论文（332 篇；参见表 2）平均分配给各位作者。这次，各位作者评审的论文子集互不重叠。尽管选择这种方式是出于时间效率考虑，但鉴于此前两轮有助于统一我们对提取流程的理解，我们确信这种做法是有效的。此外，两位作者还对其他评审者随机抽取的 5 篇论文进行了交叉验证，以发现任何遗留的系统性不一致。

3.2 研究结果

这些结果呈现了从最终筛选出的 286 篇论文中提取的数据（见表 2）。图1概述了封闭式问题的结果。下文还将结合这些结果，报告开放式问题的结果。

大语言模型使用： *E1*（使用了大语言模型）： 大多数论文（92.31 %）提到了研究中使用的大语言模型名称。39.16 % 的论文将参数数量包含在名称中（例如，*llama3 70*1643 *b*）。仅有 . % 的论文注明了确切版本。我们没有将 *gpt-3.5* 等标签视为版本说明，因为 GPT 和其他模型在不同主版本之间可能存在显著差异，实际上相当于不同的模型。相反，我们将特定快照或日期视为版本标识（例如，*0125* 或*2024-05-13*）。使用最多的模型包括 *gpt-3.5-turbo* （63 次）、*gpt-4* （61 次）、*codellama* （36 次）、*gpt-3.5* （27 次）和*text-davinci-003* （12 次）。

E2（配置参数）： 在所有论文中，69.93% 报告了至少一个配置参数。报告频率最高的参数包括温度（131 次）、输出令牌上限（33 次）、top-*p* 值（29 次）、提示迭代次数（24 次）和输入令牌上限（23 次）。

提示描述与设计： *E3*、*E4*、*E5*（提示文档）： 在大多数论文中，作者会全部或部分描述所使用的提示及其结构（75.17 %；是 + 部分描述）。超过半数的论文中，作者甚至逐字提供了所使用的提示（69.58 %）。一些作者详细描述了提示，但未列出确切措辞，因此 E4 题的肯定回答比例更高。约半数论文（58.74 %）专门说明了提示构建的理由，即解释为何以特定方式创建提示。

E6（提示工程技术）： 在 62.24 % 的论文中，作者报告了所使用的提示工程技术。报告最频繁的提示工程技术包括少样本提示 （62 次）、思维链提示 （53 次）、零样本提示 （49 次）、上下文学习 （27 次）和检索增强生成 （19 次）。所有论文共提及 50 种不同的提示工程技术。完整清单收录于我们的复现包中。需要注意的是，我们仅在作者明确将某种方法称为提示工程技术时才将其提取出来；也就是说，我们没有自行识别提示工程技术（例如，通过分析提示）。

提示测试的论文中，*E7*（提示确定）： 仅有 46.5 % 的论文中，作者明确说明了他们在研究过程中对所使用的提示词进行了优化或调优。在其余 53.5 % 的论文中，没有迹象表明作者在研究过程中对提示词进行了优化。自动提示调优技术（例如自我优化）仅被少数论文采用（4.9 % 的论文）。

E8（提示比较）： 44.06 % 的论文明确描述了不同的提示变体，并提供了其性能结果。

提示作为效度威胁：*E9*：仅有 23.43 % 的论文明确将提示作为效度威胁的一部分进行报告。在这些论文中，作者通常会提到，结果可能会受到提示词的构成和措辞影响。重新措辞或优化提示词可能会改变结果。

3.3 效度威胁

构念效度：我们的分析仅关注来自顶级三大软件工程会议（ICSE、FSE、ASE）的论文，可能无法充分代表更广泛的软件工程研究领域。尽管这些会议

表 2：论文筛选流程

	ICSE				FSE			ASE			Sum
	2022	2023	2024	2025	2022	2023	2024	2022	2023	2024	
已发表论文数量	197	211	237	246	186	205	121	213	209	265	**2,090**
全文论文数量（≥ 7 页）	197	207	234	246	130	161	121	116	145	174	**1,731**
含关键词的全文论文数量	24	44	94	152	19	55	61	23	74	138	**684**
经过基于大语言模型的筛选后的全文论文数量	5	13	40	97	7	20	42	7	34	67	**332**
人工分析后的论文最终数量	**4**	**11**	**38**	**91**	**3**	**10**	**33**	**3**	**22**	71	**286**

表3：数据提取表。C = 封闭式问题（是/否/部分），O =开放式问题

ID	问题	C/O
1. 大语言模型使用		
E1	使用了哪些大语言模型?	O
E2	大语言模型的哪些配置参数是否报告?	O
2. 提示描述与设计		
E3	是否逐字提供了完整的提示词? 论文是否逐字提供了完整的提示词，例如以清单形式提供？允许使用占位符和模板。	C
E4	是否解释了提示词及其结构? 论文是否解释了提示词及其结构，例如是否在逐字呈现之外的正文中对其进行描述？	C
E5	作者是否说明了他们为何以这种方式构造提示词? 选择某种提示工程技术的依据是什么？技巧、结构、措辞、上下文等？	C
E6	报告了哪些提示工程技术?	O
3. 提示测试与评估		
E7	论文是否提及任何形式的提示测试或调优（例如提示优化），以提升大语言模型的性能? 此问题关注论文是否提及提示词的测试或调优，而无需提供具体细节。	C
E8	论文是否报告了多种提示词变体的结果? 此问题考察论文是否明确描述提示词的变化，并详细说明这些变化的差异，同时呈现这些变化对应的结果。	C
4. 效度威胁		
E9	提示是否被视为效度威胁?	C
5. 软件工程任务		
E10	大语言模型用于哪些任务类别?	O
E11	大语言模型用于哪些任务?	O

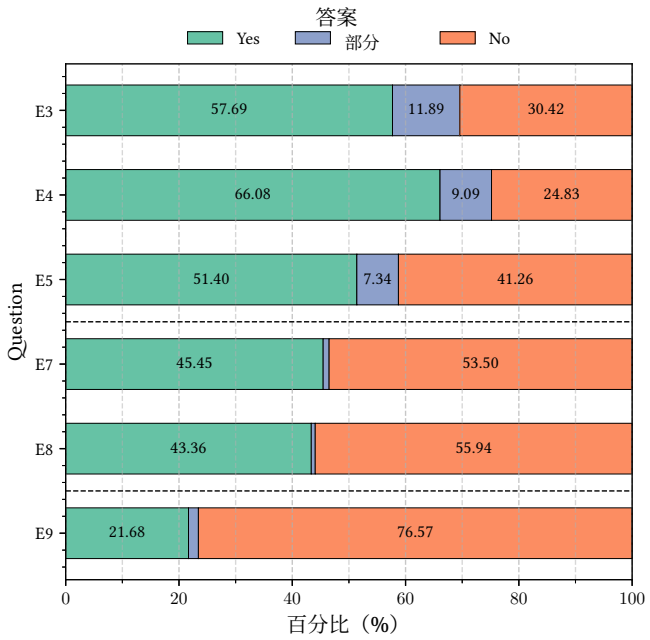

图1：条形图展示了以是、否或部分作答的封闭式提取问题的百分比（参见表3）。

具有很高的筛选标准和影响力，但重要的基于大语言模型的研究也可能出现在其他场合，例如专业会议或期刊。此外，发表偏倚可能会影响我们的结果：成功应用提示的论文可能更容易被接收和发表，从而可能导致我们对实际做法的认识出现偏差。此外，我们的语料库涵盖了截至2025年年中发表的论文。由于会议投稿时间线的影响，其中许多论文很可能不晚于2024年年中撰写。因此，我们的发现可能滞后于当前做法或提示报告方面的最新趋势。

内部效度： 尽管采用了系统化流程，论文筛选和数据提取仍存在偏倚风险。虽然我们使用基于关键词和大语言模型辅助筛选的方法来识别相关论文，但仍有可能无意中排除了一些相关研究。我们测试了几个用于自动筛选论文的提示词，最终选定了一个召回率较高的提示词。不过，进一步的提示调优或许能提高召回率。

此外，尽管我们采用了一种信息提取方法，只允许从可扩展的预定义列表中进行选择，但对报告实践的解读仍可能涉及主观判断，尤其是在边界情况或细节含糊不清时。我们通过研究人员之间交叉核对编码来减轻这一问题；然而，主观性无法被完全消除。

外部效度： 我们的研究结果可能无法推广到所有涉及大语言模型的软件工程研究，尤其是在工业界或非学术环境中，因为这些场景中的提示工程实践和文档规范可能存在显著差异。此外，其他使用大语言模型的研究社群（如自然语言处理、人机交互或教育领域）可能遵循不同的报告标准。因此，尽管我们的研究结果以软件工程社群为基础，但所提出的指南可能无法直接适用于其他领域。最后，随着大语言模型生态系统快速发展，工具、应用程序接口和社区规范不断变化，我们的结果可能会逐渐过时，从而限制分析的长期适用性。

软件工程研究人员的4项期望（RQ2）

为调查软件工程研究人员的期望，我们将软件工程会议程序委员会的成员作为调查对象。我们设计并开展了一项在线调查，以了解他们认为提示工程哪些方面必须报告。我们采用基于问卷的方法，通过在线调查工具收集广泛且多样化的回答，以期提高代表性。调查已被广泛确立为收集描述性和回顾性见解的有效手段，能够为“针对特定方法、工具或技术的最新进展综述”提供有价值的信息 [10]。

如下文所述，这项调查经过精心设计，旨在提升易用性和参与者投入度。参与者按顺序浏览各个页面，确保调查格式清晰、结构合理且易于导航。虽然在线调查确实要求参与者具备一定的技术能力，但我们预计目标人群，即软件工程研究人员，会具备所需技能。

4.1 研究设计

参与者抽样： 我们选取了2022年至2024年间 ICSE、ASE 和 FSE 的程序委员会成员，潜在参与者总数为612名。鉴于这些会议具有国际化的范围和性质，样本涵盖了来自不同国籍、背景以及软件工程研究领域的研究人员。我们通过电子邮件联系参与者，邀请他们参与本研究。

调查设计：本次调查的设计遵循了 Kitchenham等人 [7], 和 Punter等人 [10]所述的原则。完整的调查内容，包括所有问题，均收录于我们的复现包中。下文将更详细地介绍调查的关键方面。在避免屏幕上因问题过多而显得拥挤的同时，我们控制了页面数量，以降低参与者随着时间推移而注意力分散的风险。调查共包含五个页面，每页有 2–4 个问题，另有一个页面包含反馈文本框。完成调查大约需要 10 分钟。

调查中既包含封闭式问题，也包含开放式问题。封闭式问题用于收集清晰且可量化的数据 [10]。这类问题能够提供直接明了的答案（例如是或否），从而简化作答和后续分析。相比之下，开放式问题安排在调查末尾，旨在深入了解参与者回答背后的其他背景因素。这些问题对于验证通过封闭式问题获得的数据，以及收集无法通过其他方式获取的信息至关重要。我们努力尽量减少顺序效应等偏差，例如某个问题的回答可能影响后续问题的答案。问题的顺序经过了精心设计，以减轻这一影响。

为鼓励参与者如实作答，我们向所有参与者保证其匿名性。我们未收集任何个人身份信息。调查开放了30天。为尽可能提高回应率，三周后我们发送了一封提醒邮件，邀请尚未完成调查的人员参与。

问卷内容：最终问卷包含五个内容部分和一个反馈部分。这些部分共包含17个问题，其中包括4个开放式问题（含1个最后的反馈问题）和13个封闭式问题。表4列出了所有部分及其对应的问题。对于封闭式问题，我们采用了以下量表，该量表与 ACM SIGSOFT《软件工程实证标准》 [11]中使用的量表相似：

- 必不可少：为满足对清晰度、可复现性或严谨性的要求，论文必须包含的必要要素。
- 值得采用：建议包含的要素，有助于提升质量，但并非严格必需。
- 极佳：可纳入的高级要素，能够超越通常的预期，并显著提升论文的整体质量。
- 不推荐：不应纳入论文的要素，因为它无法提升质量，还可能超出预期范围、引入歧义或造成其他不良影响。

我们在问卷开头和每个部分的顶部都列出了所有回答选项的定义，方便参与者随时查阅。

4.2 研究结果

我们联系了612位曾任程序委员会成员，其中105人（17.16 %）作出了回答。共有92份回答完整，所有封闭式问题均已作答。这相当于15.03 %的回应率，显著高于基于问卷的软件工程调查通常可达到的预期回应率5 % [14]。完整数据集可在我们的复现包中获取。

在92份完整回答中，59位参与者（64.13 %）表示熟悉大语言模型，另有31位（33.7%）表示有一定了解。仅有2位参与者（2.17 %）表示完全不了解大语言模型，因此未纳入结果。

当被问及评审涉及大语言模型的研究论文的经验时，大多数参与者（70 人，占 76.09%）表示经常评审此类论文（每年超过 5 篇），而 19 人（占 20.65%）表示偶尔评审（每

表4：调查问卷

ID	问题
1. 一般信息	
S1	您在软件工程领域的主要专长是什么?
S2	您对在软件工程研究中使用大语言模型有多熟悉?
S3	您评审涉及大语言模型的研究论文的频率如何?
2. 大语言模型使用	
S4	作者会说明所使用的大语言模型。
S5	作者会准确说明所使用的大语言模型版本。
S6	作者使用不同的大语言模型并比较结果。
3. 提示词使用	
S7	作者描述用于完成任务的提示。
S8	作者提供所用提示的确切内容。
S9	作者解释为何选择特定的提示词结构或措辞。
S10	作者使用并提及提示工程技术来创建提示。
4. 提示词测试与迭代	
S11	作者报告了他们如何改进/迭代提示，以提升性能。
S12	作者应用自动化提示调优技术来优化化他们的提示词。
S13	作者测试了多种提示变体，并报告了结果。
S14	作者将提示词的使用作为效度威胁的一部分进行讨论，或将其列为潜在局限性。
5. 提示中被忽视的方面	
S15	您认为提示词使用或文档方面是否有任何内容经常在研究论文中被忽视?
S16	您是否还想补充其他方面或意见关于提示词使用和文档?
6. 反馈	
S17	您是否对以下内容有任何评论、建议或反馈这项调查?

年 1–4 篇）。仅有 3 位参与者（占 3.26%）表示迄今从未评审过此类论文。为确保数据集反映具有相关经验的参与者，其回答未纳入分析。在这些参与者中，有 2 人与表示对大语言模型完全不了解的参与者为同一批人，因此共有 89 份回答用于分析。所有封闭式问题的结果见图2。

大语言模型使用：大多数参与者（89 人中的 87 人；97.75%）同意，说明所使用的大语言模型是必不可少的，其中多数参与者（82.02%）强调了明确指定确切版本的重要性。一些受访者也在自由文本回答中强调了这一点，尤其是考虑到大语言模型快速发展的现状（例如："提示的有效性可能取决于大语言模型的参数，

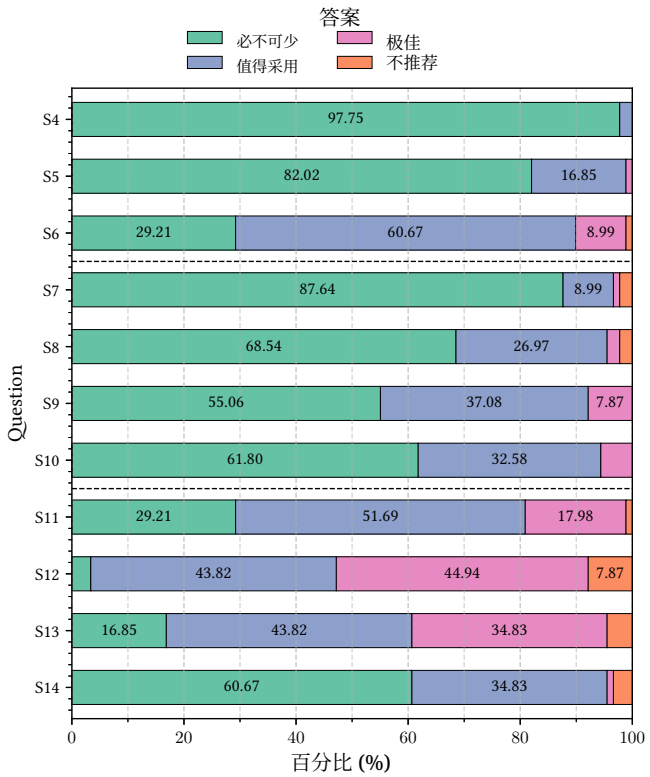

图2：条形图展示了封闭式调查问题的回答百分比（参见表4），回答选项为必不可少、值得采用、极佳**或**不推荐。

而这些参数变化很快。"、"大语言模型不断发展，因此提示工程技术也在不断变化。"）。此外，60.67%的参与者认为，比较不同大语言模型之间的结果值得采用，而29.21%的人认为这是必不可少的。

提示词使用：大多数参与者（87.65%）支持描述提示词，且多数人（68.54%）认为纳入完整提示词必不可少。提供精确提示词是最常被提及的关注点，在自由文本答案栏的多条回答中均有出现（例如："我很少看到有人使用精确提示词。我想这可以理解，毕竟有页数限制，但还是令人失望。"）。超过半数参与者（55.06%）表示，充分说明提示词构建方式的理由必不可少。此外，61.8%的参与者认为采用并报告提示工程技术必不可少，而32.58%认为值得采用。

提示词测试与迭代：本节所涉及的方面通常被视为有助于提升研究质量的要素，而非严格必需的内容。约半数参与者（51.69 %）认为，改进或迭代提示值得采用，另有 29.21 % 认为这必不可少。然而，在自由文本栏中，有几位受访者认为，作者往往没有说明提示词是如何开发、调优或选择的（例如："具体提示词和提示调优策略的细节确实

经常缺失。”，需要先进行某种预实验，才能找到合适的提示词。”）。可以接受的，而非如实反映他们通常的评审实践。

提示调优被43.82 %的受访者评为值得采用，44.94 %的受访者评为极佳。此问题收到的不推荐 回答数量也最多（7；7.87 %）。在评论中，参与者对应用提示调优时，提示针对特定大语言模型过拟合的风险表示担忧（例如：“提示针对特定大语言模型过拟合的问题正逐渐变得严重。”）。多数人认为使用多个提示变体有益，但并非必不可少；43.82 %认为这必不可少，34.83 %认为这极佳。

此外，大多数参与者（60.67 %）认为，承认与提示相关的效度威胁必不可少，而34.83 %认为值得采用。

提示中被忽视的方面：在这一开放式回答栏中，参与者指出需要更清晰地记录大语言模型设置和配置。五位参与者强调，应在报告提示和数据集的同时报告大语言模型参数（例如温度和核采样参数），因为这些因素都会影响模型行为，并且对可复现性很重要。六位参与者指出，缺少模型选择依据。此外，四位参与者指出，缺乏有关计算成本的信息，包括运行大语言模型所需的资源消耗，以及使用商业API产生的费用。

总体而言，研究强调了可复现性的重要性。一位参与者建议，由于担忧商业模型长期可用性，不能完全依赖它们。另一位参与者认为，提示创建应被视为一种程序开发形式，并指出从设计到开发和测试的整个生命周期都应得到系统记录。

4.3 效度威胁

构念效度：构念效度面临的一项主要威胁是参与者可能会误解调查条目。为减少歧义，我们使用了标准的软件工程术语，并明确界定了评分量表（即，必不可少、值得采用、极佳、不建议）。不过，个人对这些类别的理解差异仍可能影响回答的一致性。为减少措辞偏差，我们将条目表述为中性陈述（例如，“作者注明所使用的大语言模型”），而非带有价值判断的问题（例如，“是否必须注明大语言模型？”）。尽管做出了这些努力，条目的表述方式仍可能存在细微偏差，从而影响回答。四点评分量表的粒度也限制了表达能力：一些参与者可能发现，难以在这些固定类别中充分表达细致的观点。最后，报告条目本身的选择也可能引入偏差，因为该列表源自我们的文献综述（第 3 节），可能未涵盖参与者认为相关的所有维度。

内部效度： 调查本身缺乏互动性，因此限制了澄清或追问的机会。与访谈不同，我们无法深入探究含糊或相互矛盾的答案。为支持定量分析并确保与文献综述保持一致，我们接受了这一限制，将其视为一种权衡。此外，可能存在自我报告偏差，因为参与者的回答方式或许是他们认为在社交或学术上

外部效度：我们的调查面向曾在顶级软件工程会议程序委员会任职的研究人员。虽然这一群体适合用于评估审稿人期望，但他们的看法未必能充分代表积极使用大语言模型的开发者、从业者或其他领域的研究人员。学术视角可能会使结果偏向于基于科学透明度、而非工业实用主义的期望。此外，无响应偏差也是一个值得关注的问题：对大语言模型或提示工程特别感兴趣的参与者可能更愿意回答。这可能会过分强调提示词报告的重要性。不过，从开放式回答和多样化的批评意见来看，我们观察到基于大语言模型的研究的支持者和怀疑者均参与其中，这表明调查涵盖了多种观点。

5 当前状况与期望的契合度（研究问题3）

针对研究问题3，我们将提取出的实践与调查中评估的期望进行比较。为此，我们首先根据调查回答制定一份指南，然后将其与文献评审中确定的报告实践进行比较。

5.1 指南推导

我们通过分析调查中收集到的不同报告要素的重要性感知，推导出一份指南。我们使用统计方法分析调查条目之间响应模式的差异。首先，我们使用弗里德曼检验（一种用于重复测量的非参数综合检验）检验调查条目的响应模式之间是否存在统计学上的显著差异。弗里德曼检验结果高度显著（$p < 0.000001$ with $\alpha = 0.05$），表明有强有力的证据显示各项标准的回答存在差异。因此，我们拒绝所有标准具有相同响应分布的零假设。为确定这些差异，我们对所有标准配对进行了威尔科克森符号秩检验，作为事后检验。未经调整的 p 值通过邦费罗尼校正法进行了校正。

图 3 展示了调查条目（S4– S14）响应模式之间显著差异的基于图的表示。箭头表示响应条目之间具有统计学显著性的成对差异（威尔科克森符号秩检验，经过邦费罗尼校正，$\alpha = 0.05$）。

基于这项分析，我们归纳出三组条目，并将其分别归类为必不可少、值得采用和极佳 类别。此分类与 ACM SIGSOFT 实证标准 [11]一致。为制定指南，我们遵循 Baltes 等人 [1] 的思路，并按照 RFC 2119[6]的建议使用必须、应该和可以 。我们根据条目的响应秩中位数，以及通过事后比较识别出的统计显著差异，将条目划分为必不可少、值得采用和极佳三组。

我们的最终指南见表 5，根据重要性分为三组。该表概述了关键报告

[6]https://www.rfc-editor.org/rfc/rfc2119

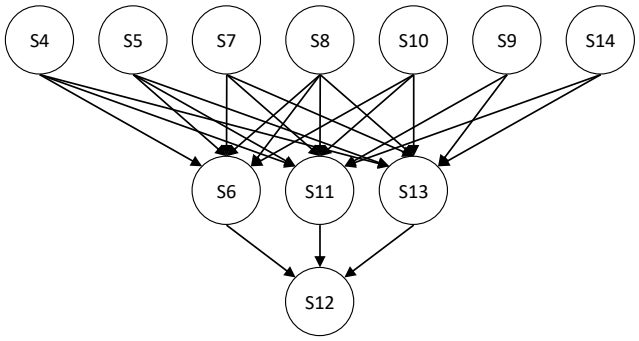

图3：调查条目（S4–S14）及其响应模式。从节点 *A* 指向 *B* 的箭头表示，*A* 的响应中位数显著高于 *B*。

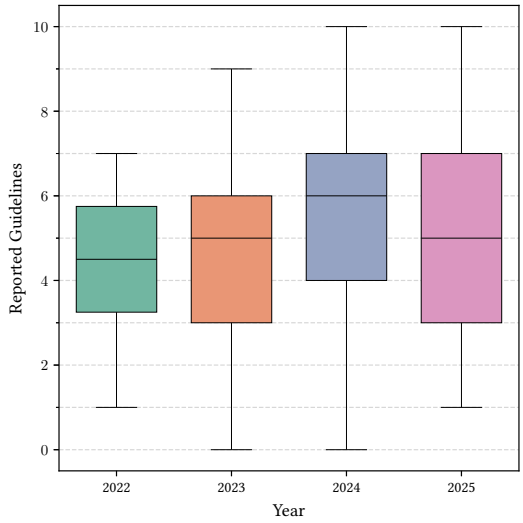

图4：箱线图展示了按提取论文的年份统计的每篇论文所报告的指南数量。

要素及其分类和文献报道频率，以专家判断为依据，并将其与实证证据进行比较。列“程序委员会成员一致度”中的数值对应于将该条目归入相应组别的受访者比例（例如，44.94 % 的受访者将使用自动化提示调优技术归为极佳）。

为评估报告实践是否随时间推移有所改善，我们分析了各发表年份的指南遵循度趋势。图 4 展示了各年份每篇论文遵循的指南条目数量分布。数据显示，指南遵循度略有上升趋势，但中位数仍然不高，在 11 个可能条目中为 4.5 至 6 项。差异也相当显著：一些论文几乎遵循了所有建议的实践，但仍有相当数量的论文仅报告三项或更少，凸显出报告标准仍缺乏一致性。

5.2 实践讨论

期望与实践的量化比较：结果揭示了社区期望与当前报告实践之间存在明显差距。尤其是两项指南条目，差距特别大。首先，尽管超过 80% 的调查受访者认为报告所使用的大语言模型的确切版本是必不可少的实践，但只有 16.4% 的受分析论文报告了这一信息。这一差距可能源于作者对此认识不足，例如，一些作者可能没有意识到，即使版本名称相同，GPT-4 等模型也会定期更新；也可能反映出一种看法，即这些版本差异微不足道。

其次，只有约20%的论文将提示视为一种潜在的效度威胁，而超过60%的受访者认为这一讨论必不可少。这种不匹配可能归因于提示在软件工程研究中的相对新颖性；在这一领域，关于效度威胁和缓解策略的社区规范尚未确立。

遵循值得采用和卓越条目的论文比例总体上与审稿人期望相符，但对必不可少条目的遵循情况仍不一致。只有 51–66% 的论文报告了大多数必不可少条目，这表明仍有很大的改进空间。

解读与启示：论文遵循所提指南条目的频率与审稿人对报告这些条目的重视程度之间存在差异，可能有多种原因。一个直接的解释是，主要软件工程会议设定了严格的页数限制，这往往迫使作者省略被认为不太重要的方法细节。人们尚未就提示词究竟属于方法的一部分，还是仅仅属于实验设置达成社区共识，这可能会进一步放大这一影响。许多研究人员仍将提示措辞视为实现细节，而不是会影响结果和可复现性的方法论决策。

审稿人偏见可能会进一步强化这一循环。例如，如果只有约三分之二的审稿人认为必须包含确切的提示词，其余三分之一的审稿人可能会在评审时忽略这一遗漏。因此，作者收到的反馈并不一致，并且可能会在后续投稿中降低对提示报告的重视程度。随着时间推移，评审标准的这种差异可能导致报告实践变得更加简略，即使大多数审稿人在理念上都重视透明度。

同样重要的是，在理解这些不匹配为何发生的同时，还要认识到它们会如何影响已发表研究的可复现性和方法论严谨性。鉴于当前大语言模型快速演进，在实验中省略所用模型的确切版本，已经严重限制了可复现性，因为同一模型的不同更新版本可能产生明显不同的输出。同样，未报告确切的提示词也可能完全妨碍复现，因为后续研究人员无法重现最初生成结果的交互过程。

方法论严谨性还取决于对实验决策进行透明论证。当作者既不说明为何选择特定提示词，评审也不一致地要求提供此类解释时，研究的概念基础就会变得更加薄弱。大多数论文都没有将提示作为潜在效度威胁，这一事实表明，提示仍未被普遍

表5：报告大语言模型提示工程指南

指南	论文中遵循该指南的程序委员会成员的论文中	一致
必不可少		
作者必须注明所使用的大语言模型（例如，*GPT-4*、*Llama3*、*Claude Opus4*）。	92.31%	97.75%
作者必须准确注明所使用的大语言模型版本（例如，*GPT-4 2024-08-06*）。	16.43 %	82.02 %
作者必须逐字提供所使用的确切提示词。他们可以使用模板缩短提示词。	57.69 %	68.54 %
作者必须描述所使用的提示词及其结构。	66.08 %	87.64 %
作者必须说明选择特定提示词结构或措辞的理由。	51.40 %	55.06 %
作者必须提及所使用的所有提示工程技术（例如：少样本提示、思维链）。	62.24 %	61.80 %
作者必须将其对提示的使用作为效度威胁的一部分进行讨论。	21.68 %	60.67 %
值得采用		
作者应 使用不同的大语言模型，并比较结果。	56.99 %	60.67 %
作者应 报告他们如何改进/迭代提示，以提升性能。	45.45 %	51.69 %
作者应 测试多种提示变体并报告结果。	43.36 %	43.82 %
极佳		
作者可以 应用自动化提示调优技术。	4.90 %	44.94%

视为一种如果不谨慎应用就可能使结果产生偏差的方法学因素。因此，解决这些报告遗漏对于确保可信、可复现且具有理论基础的基于大语言模型的软件工程研究至关重要。

为弥合这些差距，我们建议作者采用提示词报告模板，以便在不超出页数限制的情况下，纳入更长的提示词和详细说明。作者还应批判性地评估哪些大语言模型特定信息（例如模型版本、系统设置或微调参数）对于最大限度地提高可复现性和方法论严谨性是必要的。评审人员则可以考虑使用检查清单，以确保在评估过程中此类信息的完整性。将这些以实证为基础的标准直接纳入审稿表，既能简化评审流程，又能强化一致的预期。

6 结论与未来方向

我们方法的一般局限性： 尽管我们的指南建立在系统性文献综述和对经验丰富的软件工程研究人员的调查基础上，但该方法仍存在若干应予以承认的一般局限性。

首先，大语言模型驱动的软件工程领域中的预期和实践正在迅速演变。随着提示技术、大语言模型能力和社区规范不断发展，所提出指南的部分内容可能会过时或需要修订。

其次，我们的方法可能会引入描述性与规范性之间的偏差。文献综述反映的是作者选择报告的内容，而这些内容未必总是符合最佳实践。同样，调查回答体现的是参与者认为应当报告的内容，这些看法可能受到个人经验、规范或接触情况的影响，而非基于对报告有效性的实证验证。

第三，针对多样化的使用场景，该指南可能存在过度概括的问题。不同软件工程任务（例如，代码生成与测试合成）、大语言模型配置（例如，微调模型与零样本 API）以及研究类型的提示策略和文档需求各不相同 [1]。一套统一的建议可能无法充分考虑这些因任务和背景而异的差异。一些受访者也指出了这一点（例如，“对我来说，这很大程度上取决于研究问题。如果提示词是实验的核心，就必须披露。”）

最后，拟议的指南尚未经过实证验证，因而其实际影响仍不明确。尽管我们认为该指南能够提升可复现性和透明度，但仍需通过后续研究评估，采用该指南是否能切实提升审稿质量、可复现性或研究清晰度。

这些局限性表明，我们应将该指南视为一个以实证为基础的起点，而非固定不变或普遍适用的标准，并随着领域的发展不断重新审视和完善。与现有证据的关系：我们的指南与 Baltes 等人提出的 [1] 在若干关键方面有所重叠，例如强调报告确切提示词、描述其结构以及记录提示工程技术。我们在此基础上进一步提供了定量数据，说明这些做法的实际遵循频率及审稿人的期望程度，从而为确定优先次序提供了实证依据。重要的是，我们并不声称要取代或凌驾于 Baltes 等人提出的更广泛的大语言模型指南之上。相反，我们将本研究视为一项互补性工作，专门关注软件工程研究中的提示报告，并对其更广泛提案中与提示相关的关键方面进行实证验证。Baltes 等人提出的是由专家主导的愿景，而我们的工作旨在以当前实践和审稿人期望为这一愿景提供依据。

未来工作： 我们的研究为未来研究和社区参与开辟了多个方向。一个很有前景的方向是，将现有指南扩展到新兴的“提示软件” [3] 和“AI 软件”[7]，范式，在这些范式中，提示会成为嵌入软件系统中的持久性产物。在这些情境下，记录提示不仅对研究可复现性至关重要

[7] https://conf.researchr.org/home/aiware-2025

对长期可维护性、调试和管理技术债务也至关重要。除了提示工程之外，上下文工程作为一种在大语言模型有限的上下文窗口内管理信息的技术，也日益重要。未来的指南可以纳入关于上下文分段、词元压缩和检索增强生成的文档实践；这些做法在复杂的大语言模型驱动的系统中正变得愈发重要。

另一个重要方向是对我们的报告指南进行实证验证。尽管我们的调查和文献分析支持其相关性，但未来的工作可以评估其对实际研究质量的影响（例如，通过衡量可复现性、同行评审评分或实验设计清晰度方面的改进）。此外，尽管我们的研究聚焦于软件工程领域，但其他领域也采用了类似的提示实践，例如人机交互、数据科学和自然语言处理。研究这些指南如何迁移到其他研究社群，或如何根据其他研究社群的需求进行调整，将有助于确保其更广泛的适用性。

为了促进实际采用，可以开发工具支持以及面向作者和评审者的支持机制。这可能包括提示报告模板、自动化检查清单，或与工件评估流程的集成。鉴于大语言模型研究的创新步伐，我们还设想将这些指南作为一项持续更新的资源进行维护，使其能够随着新兴工具、提示策略和社区规范的发展而不断演进。

最后，我们希望将研究发现贡献给更广泛的社群标准化工作。特别是，我们认为可以与 Baltes 等人开展互补合作。 [1]，他们自上而下制定的指南涵盖了范围更广的大语言模型相关研究实践。我们的实证性自下而上的结果可提供有价值的对照和验证。我们还计划将研究发现提交给 ACM SIGSOFT 实证标准，[8]，目前这些标准尚未针对大语言模型驱动的研究提供指导。通过为这些社群推动的倡议作出贡献，我们希望推动软件工程及其他领域形成一致、高质量的实践，以记录和评估大语言模型的使用情况。

致谢

我们使用生成式人工智能（例如，Gemini 2.5 和 ChatGPT）来辅助并验证数据提取和筛选，并改进文本。

参考文献

[1] Sebastian Baltes、Florian Angermeir、Chetan Arora、Marvin Muñoz Barón、Chun- yang Chen、Lukas Böhme、Fabio Calefato、Neil Ernst、Davide Falessi、Brian Fitzgerald、Davide Fucci、Marcos Kalinowski、Stefano Lambiase、Daniel Russo、Mircea Lungu、Lutz Prechelt、Paul Ralph、Rijnard van Tonder、Christoph Treude 和 Stefan Wagner。2025年。《软件工程中涉及大语言模型的实证研究指南》。arXiv:2508.15503 [cs.SE] https: //arxiv.org/abs/2508.15503[2] Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell、Sandhini Agarwal、Ariel Herbert-Voss、Gretchen Krueger、Tom Henighan、Rewon Child、Aditya Ramesh、Daniel Ziegler、Jeffrey Wu、Clemens Winter、Chris Hesse、Mark Chen、Eric Sigler、Mateusz Litwin、Scott Gray、Benjamin Chess、Jack Clark、Christopher Berner、Sam McCandlish、Alec Radford、Ilya Sutskever 和 Dario Amodei。2020年。语言模型是少样本学习者。载于神经信息处理系统进展，第33卷。Curran Associates, Inc.，1877–1901。

[3] Zhenpeng Chen、Chong Wang、Weisong Sun、Guang Yang、Xuanzhe Liu、Jie M. Zhang 和 Yang Liu。2025年。提示软件工程：面向大语言模型提示开发的软件工程。doi:10.48550/ARXIV.2503.02400[4] Xinyi Hou、Yanjie Zhao、Yue Liu、Zhou Yang、Kailong Wang、Li Li、Xiapu Luo、David Lo、John Grundy 和 Haoyu Wang。2024年。面向软件工程的大语言模型：系统性文献综述。*ACM*软件工程与方法论汇刊 33, 8 2024 220 1 220 79 10 1145 3695988（12月），：–：。doi：。/[5] Kaicheng Huang、Fanyu Wang、Yutan Huang 和 Chetan Arora。2025年。需求工程中的提示工程：文献综述与路线图。arXiv: 2507 07682 [ ] 2507 07682 . cs.SE https://arxiv.org/abs/ 。[6] Barbara Kitchenham、Stuart Charters 等。《开展软件工程系统性文献综述的指南》。2007（2007）。[7] Barbara A. Kitchenham 和 Shari Lawrence Pfleeger。2002年。调查研究原则（第二部分）：调查设计。*SIGSOFT*软件工程通讯27，1（2002年1月），18–20。doi:10.1145/566493.566495[8] Pengfei Liu、Weizhe Yuan、Jinlan Fu、Zhengbao Jiang、Hiroaki Hayashi 和 Graham Neubig。2023年。预训练、提示与预测：自然语言处理领域提示方法的系统性调查。*ACM*计算调查55，9（2023 195 1 195 35 10 1145 3560815 年1月），：–：。doi：。/ Anh[9] Nguyen-Duc、Beatriz Cabrero-Daniel、Adam Przybylek、Chetan Arora、Dron Khanna、Tomas Herda、Usman Rafiq、Jorge Melegati、Eduardo Guerra、Kai-Kristian Kemell、Mika Saari、Zheying Zhang、Huy Le、Tho Quan 和 Pekka Abrahamsson。2023年。软件工程中的生成式人工智能——研究议程。doi: 10 48550 2310 18648 ./ARXIV. 。T. Punter、M. Ciolkowski、B.[10] Freimut 和 I. John。2003年。在软件工程中开展在线调查。载于国际实证软件工程研讨会（*ISESE*）。IEEE，80–88。doi:10.1109/isese.2003.1237967[11] Paul Ralph、Nauman bin Ali、Sebastian Baltes、Domenico Bianculli、Jessica Diaz、Yvonne Dittrich、Neil Ernst、Michael Felderer、Robert Feldt、Antonio Filieri、Breno Bernard Nicolau de França、Carlo Alberto Furia、Greg Gay、Nicolas Gold、Daniel Graziotin、Pinjia He、Rashina Hoda、Natalia Juristo、Barbara Kitchen- ham、Valentina Lenarduzzi、Jorge Martínez、Jorge Melegati、Daniel Mendez、Tim Menzies、Jefferson Molleri、Dietmar Pfahl、Romain Robbes、Daniel Russo、Nyyti Saarimäki、Federica Sarro、Davide Taibi、Janet Siegmund、Diomidis Spinel- lis、Miroslaw Staron、Klaas Stol、Margaret-Anne Storey、Davide Taibi、Damian Tamburri、Marco Torchiano、Christoph Treude、Burak Turhan、Xiaofeng Wang 和 Sira Vegas。2021年。软件工程研究实证标准。arXiv:2010.03525 [cs]doi:10.48550/arXiv.2010.03525[12] Pranab Sahoo、Ayush Kumar Singh、Sriparna Saha、Vinija Jain、Samrat Mondal 和 Aman Chadha。2024年。大语言模型提示工程的系统性调查：技术与应用。doi:10.48550/ARXIV.2402.07927[13] Jiho Shin、Clark Tang、Tahmineh Mohati、Maleknaz Nayebi、Song Wang 和 Hadi Hemmati。2025年。提示工程还是微调：对大语言模型代码能力的实证评估。载于*IEEE/ACM*第*22*届软件仓库挖掘国际会议（*MSR*）。IEEE， 490 502 10 1109 66628 2025 00082 –。doi：。/msr。[14] Janice Singer 、 Susan E Sim 和 Timothy C Lethbridge。2008。用于现场研究的软件工程数据收集。载于高级实证软件工程指南。Springer，9–34。[15] Bianca Trinkenreich、Fabio Calefato、Geir Hanssen、Kelly Blincoe、Marcos Kali- nowski、Mauro Pezzè、Paolo Tell 和 Margaret-Anne Storey。2025年。登上列车，否则将被留在车站：使用大语言模型开展软件工程研究。doi:10.48550/ARXIV.2506.12691[16] Junjie Wang、Yuchao Huang、Chunyang Chen、Zhe Liu、Song Wang 和 Qing Wang。2024年。使用大语言模型进行软件测试：调查、现状与展望。*IEEE*软件工程汇刊50，4（2024），911–936。doi:10.1109/TSE.2024. 3368208[17] Daoguang Zan、Bei Chen、Fengji Zhang、Dianjie Lu、Bingchao Wu、Bei Guan、Wang Yongji 和 Jian-Guang Lou。2023年。大语言模型与自然语言到代码相遇：一项调查。载于第*61*届计算语言学协会年会（第*1*卷：长篇论文）。计算语言学协会。doi:10.18653/ v 1 2023 411 / .acl-long。Quanjun Zhang、Chunrong Fang、Yuxiang Ma、Weisong Sun 和 Zhenyu Chen。2024年。基于学习的自动程序修复调查。*ACM*软件工程与方法论汇刊33，2（2024），55:1–55:69。[18] doi:10.1145/3631974

[8]https://www2.sigsoft.org/EmpiricalStandards/