通过人类与大语言模型协作进行上下文自适应需求缺陷预测

Max Unterbusch paluno – 鲁尔软件技术研究所 杜伊斯堡-埃森大学 德国埃森
max.unterbusch@uni-due.de

Andreas Vogelsang paluno – 鲁尔软件技术研究所 杜伊斯堡-埃森大学 德国埃森
andreas.vogelsang@uni-due.de

arXiv:2601.01952v1 [cs.SE] 5 Jan 2026

摘要

传统的自动化需求评估通常依赖通用模式作为缺陷性的替代指标，并通过基于规则的启发式方法或在大型标注数据集上训练的机器学习分类器来实现。然而，“缺陷”的定义本质上取决于具体情境，并因项目、领域和利益相关者的理解而异。本文提出一种人类与大语言模型协作（HLC）方法，将缺陷预测视为自适应过程，而非静态分类任务。HLC 在反馈循环中利用大语言模型思维链推理：用户在查看预测结果及其解释的同时对其进行验证，而这些已验证样例则通过少样本学习自适应地指导后续预测。我们使用 QuRE 基准中的弱词坏味道对该方法进行评估；该基准包含 1,266 条经过标注的梅赛德斯-奔驰需求。结果表明，HLC 能够有效适应已验证样例的提供，仅需 20 个已验证样例便可迅速提升性能。纳入经过验证的解释（而不仅仅是标签），使 HLC 在保持高召回率的同时，大幅优于标准少样本提示和微调后的 BERT 模型。这些结果凸显了大语言模型的上下文学习和思维链学习能力如何助力自适应分类方法摆脱一刀切模型，并为能够从利益相关者反馈中持续学习的工具创造机遇。

CCS 概念

• **软件及其工程** → **需求分析**；软件缺陷分析。

关键词

大语言模型、需求工程、质量、人在回路

ACM 参考格式：

Max Unterbusch 和 Andreas Vogelsang。2026年。通过人类与大语言模型协作进行上下文自适应需求缺陷预测。载于*2026*年 *IEEE/ACM* 第*48*届国际软件工程会议（*ICSE-NIER '26*），*2026*年*4*月*12*日至*18*日，巴西里约热内卢。ACM，美国纽约州纽约市，5页。

1 引言

未被发现的缺陷，例如自然语言需求中的歧义，可能会在下游软件工程任务中演变为代价高昂的问题，并危及项目成功 [7]。为应对这一风险，需求工程（RE）领域开发了自动化方法，利用通用模式作为缺陷性的代理指标来检测需求异味。这些方法依赖基于规则的方法 [4, 8, 18] 这些方法需要复杂的、由人工设计的启发式方法，或基于机器学习/深度学习的方法 [2, 10, 19]，后者依赖经过标注且规模足够大的数据集，而这类数据集十分稀缺 [9]。这两种方法的开发成本都很高，且缺乏灵活性，因为它们局限于特定类型的缺陷和自然语言。除了这些实际局限之外，从将需求质量视为使用质量的角度来看，现有方法在概念上也存在缺陷：需求“好”或“坏”，取决于它们对下游软件工程活动的支持程度 [6],，而这取决于具体情境。

我们的想法是，通过添加情境化缺陷预测层来改进现有的异味检测。该层在自适应反馈循环中结合上下文学习与思维链（思维链）推理，利用人类与大语言模型协作。大语言模型会为每项缺陷预测生成推理句子，用户可以连同预测结果一起接受或拒绝这些句子，从而逐步积累已验证样例及其相关推理依据。通过基于相似度的示例选择，最相关的过往样例可为未来的预测提供指导，使系统能够持续适应特定的开发情境。得益于大语言模型的预训练知识，即使没有可用样例，该流程也可以从零样本学习开始。

为了评估这一方法，我们利用最近发布的 QuRE 基准 [5], 针对弱词坏味道开展了初步实证调查。该数据集包含汽车制造商梅赛德斯-奔驰的需求，其中包含弱词，并由内部测试工程师标注为有缺陷或无缺陷。弱词缺陷性的分类颇具挑战，因为这需要语义理解（示例见表1）。我们模拟了该方法在示例池逐渐扩大的情况下的运行过程，并调查大语言模型能否根据用户反馈调整缺陷预测结果。

我们的初步结果表明，即使在数据极度匮乏的情况下，该方法也能有效适应特定语境下对缺陷性的解读。我们发现，将经过验证的解释与标签结合起来至关重要：仅使用 20 个示例的 HLC，其表现就显著优于不进行推理的标准少样本提示，以及使用 320 个示例微调的 BERT 模型。

本文提出一种范式转变：摒弃僵化、千篇一律的方法，转而采用明确纳入情境因素和利益相关方主导判断的方法。我们预计，这种人类与大语言模型协作流程将拓展至软件工程领域其他依赖情境的任务，例如代码评审；在这些任务中，“正确性”往往取决于利益相关方的视角和情境因素。这一范式挑战了这样一种普遍假设：软件工程自动化工具需要大量预先标注的数据集以及普遍认同的真实标签；相反，它表明

被 *ICSE-NIER '26* 接收，巴西里约热内卢
©2026年作者版权所有。

有效的质量保证可以通过利益相关方的渐进式反馈实现。这为未来研究能够适应情境的软件工程工具开辟了方向，这类工具能够从组织中学习并不断演进。

2 背景与相关工作

需求质量保证研究主要沿着两个方向展开：（i）使用受控语言，从设计上预防缺陷；（ii）针对不受约束的自然语言需求采用验证方法 [8]。在后者领域，研究人员将模式定义为下游软件工程任务中潜在问题的通用替代指标[4], ，即所谓的需求异味，并提出了自动检测这些异味的方法。

基于规则的方法 [4, 8, 18] 通过词性标注、词典和句法分析来检测此类异味。被动语态和弱词等基础异味易于实现，但通常会过度判定缺陷性，因为在实践中，大多数情况在具体语境中并无危害 [11]。使用定义更严格的模式（如模糊代词和主观语言）的复杂异味，则更难实现，也难以通过手工编写的规则加以维护。多项研究采用机器学习/深度学习方法来检测更具体的缺陷类型，如照应歧义或并列歧义 [2, 10, 19]。这些统计方法能更好地考虑语境，但缺乏灵活性，且需要规模足够大的标注数据集，而此类数据集十分稀缺 [9]。

近期研究开始探索使用大语言模型进行需求质量评估，针对无歧义性、一致性或 ISO 29148 特性等抽象质量维度 [12–14]。然而，这些研究通常提示大语言模型给出整体判断，结果往往参差不齐 [3, 16]，而且为大语言模型提供的指导也很有限。与我们的工作最为接近的是 Bashir 等人 [1]，他们尝试了用于歧义检测的少样本提示策略，并评估大语言模型能否事后解释其预测结果，发现这类方法在提供解释方面效果良好。相比之下，我们的方法会在给出判定之前生成推理句，将其用作解释以收集用户反馈，并将反馈加入示范示例中，以指导后续预测。

3 人类与大语言模型协作方法

我们的人类与大语言模型协作（HLC）方法（图1）运行——通过基于简单模式构建方法，在上下文中预测其缺陷性，从而彻底避免冷启动问题。从没有任何标注数据开始，我们先采用零样本思维链提示。针对识别出的模式（例如，一个弱词），大语言模型会先生成一句推理陈述，再判断是否存在缺陷。用户会明确验证每项发现，并将思维链推理作为解释。用户可以更正推理、标签，或两者兼而有之。已验证样例（需求、识别出的模式、推理、标签）会存入一个池中，以支持少样本提示。由于冗余样例会带来收益递减、近因偏差以及上下文窗口限制，将所有可用样例都用作示例并不切实际。因此，对于每条输入需求，都会分别通过嵌入相似度检索示例，以提高提供相关样例的可能性。这样，大语言模型便能依据相似的历史样例及其明确的推理依据作出判断，使预测结果符合利益相关者对质量的特定情境解读。

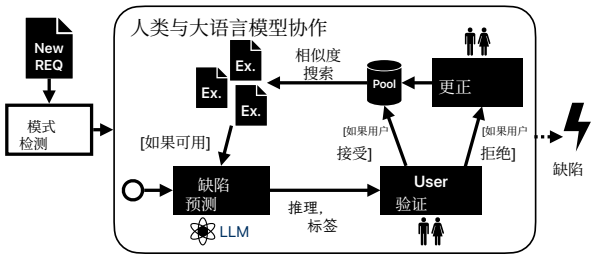


图1：人类与大语言模型协作方法

与先前工作的区别：复杂的缺陷预测方法依赖复杂规则或机器学习/深度学习方法，这些方法假定存在普适性真值，并且在部署静态分类器之前需要手工制定规则或大量标注数据集。相比之下，我们利用简单的高召回率模式；由于这些模式过于宽泛，传统方法往往会产生过多假阳性。HLC 通过灵活、适应上下文的人类与大语言模型协作，区分有缺陷和无缺陷的情形，从而解决这一问题。与标准的基于大语言模型的少样本方法不同，HLC 从零开始构建示例池，其中包含经验证的解释，用于指导未来的预测；而这类解释通常并不存在，除非由人工编写。

与主动学习（AL）从标注数据开始，并通过迭代查询样例来重新训练不同，我们的验证循环就是其运行模式：每个预测结果都会经过验证，反馈通过上下文学习融入其中，而思维链则提供了固有的可解释性。假设召回率足够高，人类主要通过以最少的工作量纠正预测结果来筛除良性案例。

其创新之处在于，借助人类与大语言模型协作，通过推理不断构建经过验证的样例，从而摆脱预先标注的数据集。我们随后将这一理念扩展到其他软件工程自动化任务，并探讨工具与流程集成如何为其提供支持。

4 研究设计

在这项初步研究中，我们的目标是估计 HLC 方法在模拟使用场景中的表现。我们以预测弱词的缺陷性这一初始案例作为性能基准，并考虑以下研究问题（RQ）：**HLC 在弱词缺陷预测中的有效性如何?**

为回答这一研究问题，我们采用来自梅赛德斯-奔驰的行业需求基准数据集，这些需求由其内部测试工程师标注了弱词的缺陷性。在实验设置中，我们使用带标签的数据作为用户反馈的替代，以模拟反馈循环，并测试经验证样本池大小不断增加时的效果。作为基线方法，我们将 HLC 方法与以下两种方法进行比较：(i) 不使用思维链的大语言模型，以评估推理对性能的影响；(ii) 对 BERT 模型进行微调，以代表预先训练分类器这一既有做法。

4.1 研究对象

研究数据：我们采用 QuRE 基准 [5], ，其中包含 2,111 条独特的梅赛德斯-奔驰需求（示例见表1）。每条需求至少包含公司目录中的 23 个弱词之一。该数据集针对弱词缺陷性进行了标注

，最多由三名公司内部测试人员完成，并针对难以判断的案例协商确定标签。虽然我们没有关于标注者间一致性的信息，但该数据集使我们能够评估此方法对梅赛德斯-奔驰质量保证实践的适应程度。

数据集准备：我们对包含多个弱词的需求进行了去重。如果某项需求同时包含缺陷弱词和非缺陷弱词，我们保留缺陷实例，因为缺陷类是少数类。由于数据集中弱词和缺陷的原始分布未知 [5], ，我们对非缺陷类进行了下采样，得到一个包含 1,266 个实例的平衡数据集（每类 633 个实例）。

抽样策略： 我们设计了一种嵌套抽样策略，用于跟踪随着向同一示例池添加更多数据，性能如何变化。我们将数据集随机划分为三个折（每折 422 个实例，并按标签分层）。在每个折内，我们递归抽取大小分别为 320、160、80、40 和 20 的分层子集，确保每个较小的池都包含在下一个较大的池中。每个池都在一个单独的交叉分配折上进行评估，因此示例池中的实例不会出现在其评估集中。抽样策略的可视化见我们的在线材料[2]。

语言模型： 由于比较大语言模型并非本文的核心，我们使用 gpt-4.1-mini（gpt- 4.1-mini-2025-04-14）进行了所有实验。我们选择该模型是因为它成本效益高，且 MMLU 表现出色[1], ，优于先前需求工程歧义研究中表现最佳的大语言模型 [1]。

4.2 方法实现

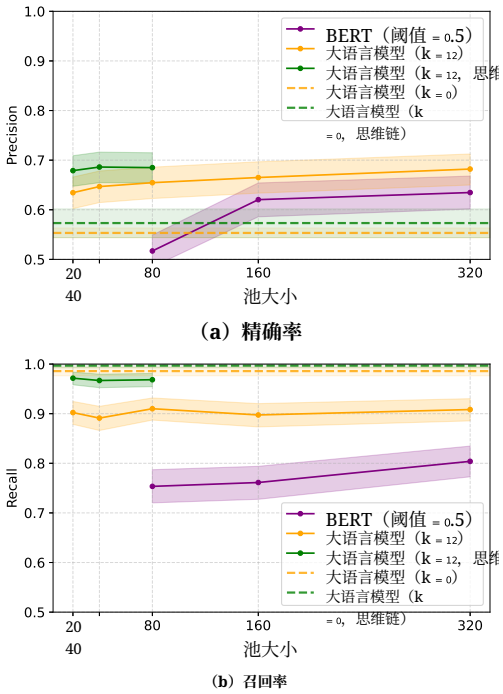
我们将弱词的缺陷预测定义为二元分类任务，模型接收一个输入元组 $X = r, w,$ ，其中 r 是自然语言需求， w 是其中包含的弱词，而 $Y =$ $y \in$ 的目标是赋予其名义标签 {defect, not defect}。

提示与示例整合： 系统提示（见在线材料[2]）简要定义了弱词，并要求大语言模型判断 w 是否会使 r 产生歧义。示例以需求、 + 弱词及相应推理、 + 标签的输入输出对形式附加。在思维链变体中，明确指示大语言模型在作出预测前先生成一句推理说明。新实例作为用户提示提供，并遵循相同的输入结构。

示例选择：我们使用 text-embedding-3-small 模型[3] byOpenAI 对所有需求进行了预嵌入。我们尝试了$k \in \{0, 12\}$ 个示例： $k = 0$ 表示零样本（没有可用示例），而 $k = 12$ 则分别选取与目标需求的余弦相似度最高的六个缺陷示例和非缺陷示例。为了利用大语言模型的近因偏差，我们将示例排序，使最相似的示例位于最后 [15]。

推理示例：为了模拟一个由用户验证的推理池，我们为大小为80的池及其子集中的所有示例生成了推理句。与HLC一致，解释首先由大语言模型生成；但不同于实际的协作方法，我们以真实标签为条件进行生成，随后亲自审查每条解释。在审查过程中，我们确保每条推理依据都与标签一致，并在需要时进行小幅修改。虽然在我们的HLC 方法中，推理

[1]https://openai.com/index/gpt-4-1/ [2]https://doi.org/10 30244633 3 6084 9. /m.figshare.
https://platform.openai.com/docs/models/text-embedding-3-small



（a）精确率

（b）召回率

图2：随着池大小增加的分类性能（通过自助法重采样获得的95%置信区间，N=10,000）

标签由实际利益相关者审查，但我们为全部240项需求找到了合理的推理（摘录见表1）。

微调： 我们将基于大语言模型的方法与微调较小的仅编码器模型（BERT-base-cased）进行比较，该模型曾代表需求分类领域的最先进水平 [17]。为确保与会标出弱词的基于大语言模型的方法公平比较，我们使用特殊标记标注弱词边界。 [CLS]表示通过前馈层和Softmax激活用于二分类。我们在不同示例池上分别微调模型 ≥ 80，因为较小的池不足以获得可靠结果。完整细节见我们的在线材料[2]。

5 研究结果与讨论

我们使用精确率、召回率和F1分数评估性能，并将缺陷视为正例。由于漏掉真实缺陷的代价高于将无害案例误报为缺陷，因此召回率尤为重要；不过，原始数据集分布未知 [5]，所以我们报告未加权的F1分数。表2和图2汇总了所有池大小下的结果。置信区间通过对拼接后的预测结果（每种配置1,266个）进行自助法重采样（10,000次迭代）获得，同时反映池内不确定性和池间变异性。

在没有任何已验证样例（$k = 0$）的情况下，基于大语言模型的方法召回率（> 0.98）较高，但精确率（$\approx 0.55 - 0.57$）有限。在这种设置下，思维链推理并未显著提升性能，但能提供解释——在结构上

表1：需求示例及推理解释（弱词以粗体标记）

ID	需求	缺陷	解释
255	TCU 通过 CAN 和 LIN 以冗余方式连接至 ORC，以执行在**特定**碰撞等级下的自动紧急呼叫。	yes	“特定”一词用于描述哪些碰撞等级会触发自动紧急呼叫，但并未定义具体的碰撞等级，因此无法明确哪些碰撞等级应触发紧急呼叫。
92	对于后排座椅娱乐系统（RSU 或平板电脑），系统应播放警报，并将其发送到选定的乘员。	no	“appropriate”一词指与所选乘员相对应的音频输出；结合上下文可知，这里指的是具体选定的乘员，如需求中所述。

表2：实验结果

池大小	方法	CoT	k	精确率	召回率	F1
–	GPT	Yes	0	0.573	0.997	0.728
	GPT	No	0	0.553	0.986	0.709
20	GPT	Yes	12	0.679	0.972	0.799
	GPT	No	12	0.634	0.902	0.745
40	GPT	Yes	12	0.686	0.967	0.803
	GPT	No	12	0.647	0.891	0.750
80	GPT	Yes	12	0.685	0.968	0.802
	GPT	No	12	0.655	0.910	0.761
	BERT	–	–	0.517	0.754	0.613
160	GPT	No	12	0.665	0.897	0.764
	BERT	–	–	0.620	0.761	0.684
320	GPT	No	12	0.682	0.908	0.779
	BERT	–	–	0.635	0.804	0.709

与我们预先生成的解释相似。这些解释有助于引出用户反馈，而这正是构想中的协作循环所需要的。

仅使用 20 个已验证样例（每类 10 个）时，性能便显著提升。思维链少样本提示（$k = 12$）将精确率提高至 ≈ 0.70，同时保持较高的召回率（≈ 0.97）；而非思维链变体的召回率（≈ 0.90）有所下降，精确率（≈ 0.63）也较低。

将示例池扩大到 20 个以上，只能带来有限的精确率提升（非思维链变体从 20 个样例增加到 320 个时提升 +4.8 个百分点），而召回率没有改善。这表明不断扩大池规模会产生边际效益递减。实际应用中存在一种权衡：强制持续进行推理更正虽能缓慢提高精确率，却会增加用户的工作量；而限制池规模增长虽能减少开销，却会留下更多假阳性待处理。

在最多 320 个示例上微调 BERT，其精确率（≈ 0.64）和召回率（≈ 0.80）均低于仅使用 20 个样例的 HLC 方法。其 F1分数甚至低于零样本思维链提示。虽然调整阈值可以提升实际所需的召回率，但精确率会随之下降。

对 RQ 的回答。随着从零样本过渡到少样本预测，HLC 能有效调整弱词的缺陷预测，仅用 20 个样例便实现了快速的性能提升。在样例中加入经验证的解释，而不仅仅是标签，使 HLC 的表现优于缺少此类推理依据的标准提示方法。不应将这些结果过度解读为端到端性能估计，因为 QuRE 基准有意对具有挑战性的案例进行了过采样。

6 未来计划

我们的目标是推动 HLC 范式从当前的概念验证阶段，发展为面向情境自适应软件工程的综合框架自动化。我们的研究议程涵盖三个互补方向：工具开发与评估、拓展至更广泛的软件工程任务，以及完善底层方法。

由于 HLC 依赖人在回路反馈，实际应用需要借助工具支持来最大限度地减少验证工作。为此，我们开发了*Requirely*，这款原型工具的设计基于我们的初步研究发现。Requirely 允许用户灵活配置弱词以外的检查器，提供具有解释的上下文自适应需求缺陷预测，并在富文本编辑器中提供自动生成的改进建议（请在我们的在线材料中观看演示视频[2]）。

情境内评估：我们计划在真实的开发情境中与实际利益相关者一起评估 Re- quirely 等工具，以衡量其在现实环境中的表现，确定哪些质量缺陷能够有效检测、哪些无法有效检测，并了解将基于 HLC 的工具集成到软件工程流程中的组织影响。我们还将调查用户对 HLC 与静态分类器的看法，考察其可用性和接受度。

扩展至其他软件工程任务：我们预计，HLC 范式可从需求质量保证扩展到软件工程中其他依赖上下文的分类任务，在这些任务中，由利益相关者主导的判断可能比一刀切的解决方案更受青睐。具体而言，我们计划将 HLC 应用于代码审查，因为代码质量同样因利益相关者而异，并且依赖上下文。与需求异味类似，代码异味也可作为 HLC 的切入点，并产生可采取行动的结果。将 HLC 集成到代码审查工作流（例如 GitHub 拉取请求）中，有望使其逐步适应审查者偏好和团队标准，加快常规质量检查，同时让人类专家腾出精力处理其他任务。

方法改进： 我们当前的研究聚焦于采用简单的基于模式的切入点进行二分类。我们计划探索 HLC 是否也能直接处理未过滤工件。我们还将研究如何将 HLC 从分类任务拓展到生成任务，例如为需求或代码提供改进建议。将实际采用的改进反馈到示例池中，系统便能逐渐学会生成更有帮助且符合上下文的建议。此外，我们还将探讨调整精确率与召回率权衡的方法。

致谢

由德国研究基金会（DFG，德国研究基金会）资助——项目编号：566352773。我们感谢 Alexander Korn、Mersedeh Sadeghi 和 Henning Femmer 提供的宝贵意见。

参考文献

[1] Sarmad Bashir、Alessio Ferrari、Muhammad Abbas Khan、Per Erik Strandberg、Zulqarnain Haider、Mehrdad Saadatmand 和 Markus Bohlin。2025。利用大语言模型进行需求歧义检测与解释：一项工业研究。载于第*41*届国际软件维护与演化会议。IEEE，奥克兰，新西兰，620–631。[2] Saad Ezzini、Sallam Abualhaija、Chetan Arora 和 Mehrdad Sabetzadeh。2022。需求中照应歧义的自动化处理：一项多方案研究。载于第*44*届国际软件工程会议论文集。ACM，美国宾夕法尼亚州匹兹堡，187 199 10 1145 3510003 3510157 – 。doi: 。/ 。[3] Alessandro Fantechi、Stefania Gnesi、Giuseppe Lami 和 Alessandro Maccari。2003。语言学技术在用例分析中的应用。《需求工程》8, 3 2003 161 170 10 1007 00766 003 0174 0 ( ), – 。doi: 。/s - - -[4] Henning Femmer、Daniel Méndez Fernández、Stefan Wagner 和 Sebastian Eder。2017。利用需求异味快速进行质量保证。《系统与软件杂志》123 2017 190 213 10 1016 2016 02 047 ( ), – 。doi: 。/j.jss. . .[5] Henning Femmer、Frank Houdek、Max Unterbusch 和 Andreas Vogelsang。2025。QuRE 描述与比较分析：一个新的工业需求质量数据集。载于*2025*年*IEEE*第*33*届国际需求工程会议研讨会（*REW*）。IEEE，西班牙瓦伦西亚，23–29。[6] Henning Femmer 和 Andreas Vogelsang。2019。需求质量就是使用质量。《*IEEE*软件》36，3（2019），83–91。doi:10.1109/MS.2018.110161823[7] D Méndez Fernández、Stefan Wagner、Marcos Kalinowski、Michael Felderer、Priscilla Mafra、Antonio Vetrò、Tayana Conte、M-T Christiansson、Des Greer、Casper Lassenius 等。2017。为需求工程中的痛点命名：实践中的当代问题、原因与影响。《实证软件工程》22，5（2017年10月），2298–2338。doi:10.1007/s10664-016-9451-7[8] Alessio Ferrari、Gloria Gori、Benedetta Rosadini、Iacopo Trotta、Stefano Bacherini、Alessandro Fantechi 和 Stefania Gnesi。2018。利用自然语言处理模式检测需求缺陷：铁路领域的一项工业实践。《实证软件工程》23，6（2018），3684–3733。doi:10.1007/s10664-018-9596-7[9] J ulian Frattini、Lloyd Montgomery、Jannik Fischbach、Michael Unterkalmsteiner、Daniel Mendez 和 Davide Fucci。2022。面向文本需求的质量因素实时可扩展本体。载于*2022*年*IEEE*第*30*届国际需求工程会议（*RE*）。IEEE，澳大利亚墨尔本，274–280。doi:10.1109/ RE54965.2022.00041[10] Mohammad Kasra Habib、Stefan Wagner 和 Daniel Graziotin。2021。利用深度学习检测需求异味：经验、挑战与未来工作。载于*2021*年*IEEE*第*29*届国际需求工程会议研讨会（*REW*）。IEEE，美国印第安纳州圣母大学，153–156。doi:10.1109/REW53955.2021. 00027[11] Jennifer Krisch 和 Frank Houdek。2015。被动语态不佳与弱词之谜：汽车行业的一项实证调查。载于*2015*年*IEEE*第*23*届国际需求工程会议（*RE*）。IEEE，加拿大安大略省渥太华，344 351 10 1109 2015 7320451 – 。doi: 。/RE. 。[12] Madhava Krishna、Bhagesh Gaur、Arsh Verma 和 Pankaj Jalote。2024。在软件需求规格说明中使用大语言模型：一项实证评估。载于*2024*年*IEEE*第*32*届国际需求工程会议（*RE*）。IEEE，冰岛雷克雅未克，475–483。doi:10.1109/RE59067.2024.00056[13] Sebastian Lubos、Alexander Felfernig、Thi Ngoc Trang Tran、Damian Garber、Merfat El Mansi、Seda Polat Erdeniz 和 Viet-Man Le。2024。利用大语言模型进行软件需求质量保证。载于*2024*年*IEEE*第*32*届国际需求工程会议（*RE*）。IEEE，冰岛雷克雅未克，389–397。doi:10.1109/RE59067.2024.00046[14] Taslim Mahbub、Dana Dghaym、Aadhith Shankarnarayanan、Taufiq Syed、Salsabeel Shapsough 和 Imran Zualkernan。2024。GPT-4 能否帮助检测需求分析中的歧义、不一致和不完整性？一项综合案例研究。《*IEEE Access*》12（2024），171972–171992。doi:10.1109/ACCESS.2024.3464242[15] Alexander Peysakhovich 和 Adam Lerer。2023。注意力排序可抵御长上下文语言模型中的近因偏差。arXiv:2310.01427 [cs] doi:10. 48550/arXiv.2310.01427[16] Daniel Seifert、Lisa Jöckel、Adam Trendowicz、Marcus Ciolkowski、Thorsten Honroth 和 Andreas Jedlitschka。2024。大语言模型（LLM）能否与人类需求评审人员一较高下？——需求文档检查实验的复现。载于以产品为中心的软件过程改进，Dietmar Pfahl、Javier Gonzalez Huerta、Jil Klünder 和 Hina Anwar（编）。第15452卷。Springer Nature Switzerland，查姆，27–42。doi:10.1007/978-3- 031-78386-9 3_[17] Max Unterbusch、Mersedeh Sadeghi、Jannik Fischbach、Martin Obaidi 和 Andreas Vogelsang。2023。应用评价中的解释需求：分类体系与自动检测。载于*2023*年*IEEE*第*31*届国际需求工程会议研讨会（*REW*）。IEEE，德国汉诺威，102–111。doi:10.1109/REW57809.2023.00024[18] Alvaro Veizaga、Seung Yeob Shin 和 Lionel C. Briand。2024。自然语言需求中的异味自动检测与推荐。《*IEEE*软件工程汇刊》50，4（2024），695–720。doi:10.1109/TSE.2024.3361033

[19] Hui Yang、Alistair Willis、Anne De Roeck 和 Bashar Nuseibeh。2010。自然语言需求中有害协调歧义的自动检测。载于*IEEE/ACM* 自动化软件工程国际会议论文集。ACM，比利时安特卫普，53–62。doi:10.1145/1858996.1859007

收到日期：2025年9月30日；接受日期：2025年12月1日