负责任的局内人：提示工程师的伦理责任研究

Responsible Insiders: A Study of the Ethical Responsibility of Prompt Engineers

郭思敏 /GUO Simin[1]　　汪琛 /WANG Chen[2]

（1. 合肥工业大学马克思主义学院，安徽合肥，230062；2. 清华大学社会科学学院，北京，100084；
(1. School of Marxism, Hefei University of Technology, Hefei, Anhui, 230062;
2. School of Social Sciences, Tsinghua University, Beijing, 100084)

摘　要：作为唯一能够通过内在机制直接引导和塑造大语言模型输出的新兴技术角色，提示工程师在大语言模型全生命周期中承担着需求对齐、输出精准调控及人机交互架构设计的核心职能。随着技术介入方式从传统参数调优转向大语言模型认知架构塑造，提示工程师的职能边界已从技术实践延伸至伦理治理维度。这种转变要求提示工程师必须正视提示工程特有的伦理挑战，包括生成毒性内容、恶意提示注入攻击、系统性偏见强化及自动化提示工程工具失控等风险。因此，提示工程师需实现技术执行者与伦理治理者的双重角色融合，通过负责任地编写提示、治理恶意提示操控、检测和治理偏见及对自动提示工程师实施有意义的人类控制，确保大语言模型技术的稳健发展与良性应用。

关键词：提示工程师　伦理责任　大语言模型　伦理治理

Abstract: As the only emerging technical role that can directly guide and shape the output of Large Language Models (LLMs) through intrinsic mechanisms, prompt engineers are responsible for the core functions of requirement alignment, accurate output regulation, and the design of human-computer interaction architectures. With the shift of technical intervention from traditional parameter tuning to directional shaping of model cognitive architecture, the prompt engineers' functional boundary has been extended from technical practice to the dimension of ethical governance. This shift requires prompt engineers to confront the ethical challenges specific to prompt engineering, including the risks of generating toxic content, malicious prompt injection attacks, systematic bias reinforcement, and loss of control over automated prompt engineering tools. Therefore, prompt engineers need to integrate the dual roles of technical implementers and ethical governors through four types of approaches: responsible prompt authoring, governance of malicious prompt manipulation, detection and governance of bias, and meaningful human control of automated prompt engineers.

Key Words: Prompt engineers; Ethical responsibility; Large Language Models; Ethical governance

中图分类号：K826.1; TP18　DOI: 10.15994/j.1000–0763.2026.06.003　CSTR: 32281.14.jdn.2026.06.003

基金项目：中国博士后科学基金面上资助“人工智能伦理实证研究的哲学审视”（项目编号：2024M751746）；安徽省哲学社会科学规划项目“科学家精神融入高校思想政治理论课的创新机制研究”（项目编号：JS2024AHZS0056）；高校“三全育人”综合改革和思想政治能力提升计划项目“习近平文化思想融入大中小学思政课一体化建设研究”（项目编号：sztsjh–2024–13–3）。

收稿日期：2025 年 1 月 10 日

作者简介：郭思敏（1993–）女，安徽合肥人，合肥工业大学马克思主义学院讲师，研究方向为科技伦理。Email: 2023800050@hfut.edu.cn

汪　琛（1996–）男，安徽合肥人，清华大学社会科学学院博士后，研究方向为科技伦理、科技治理。Email: chen0522@mail.ustc.edu.cn

近年来，随着大语言模型（Large Language Models, LLMs）的广泛应用与规模化部署，作为新进主体的提示工程师（Prompt Engineer），正逐渐成为人工智能伦理治理研究的关注重点。[1]，[2] 作为LLMs与用户间桥接的关键中介，提示工程师既掌握着通过提示工程（Prompt Engineering）介入LLMs认知框架的技术介入权，又肩负着LLMs价值对齐、风险治理与伦理传导的特殊使命。为此，本文旨在爬梳提示工程师独特的工作基础、模式与方法的基础上，深入探讨提示工程师在履行职能时可能面临的一系列特殊伦理风险及其应有的伦理责任，为将提示工程师的伦理实践纳入大语言模型伦理治理体系提供理论依据。

一、提示工程师的工作基础、模式与方法

提示工程师作为人工智能领域的新兴行动主体，其核心职能是通过构建和优化结构化指令，引导和塑造LLMs的输出结果。与传统软硬件工程师通过外部架构影响系统功能不同，提示工程师能从内部参与LLMs的认知塑造，将用户需求转化为可操作的指令范式，实现LLMs与用户的有效交互。本文从工作基础、模式和方法三个维度，系统阐述提示工程师在LLMs开发与部署中的独特功能和价值创造机制。

1.“提示”——提示工程师的工作基础

作为LLMs的核心交互媒介，“提示”（Prompt）既是触发LLMs运行的初始指令，也是提示工程实践的核心载体。它通过自然语言或结构化文本构建认知导航路径，将用户的抽象需求转化为机器可执行的语义框架，[3] 其质量直接影响LLMs的输出效果。从功能上看，提示不仅传递需求，还深度参与LLMs知识表征与逻辑推理能力的塑造。

在实际应用中，提示的类型和组成根据任务需求与LLMs适应性灵活变化。从使用方式看，提示分为系统提示和用户提示：[4] 系统提示是系统预设的初始指令，为LLMs提供全局规则和框架；用户提示是用户根据具体需求在对话中使用的指令。从信息组织看，提示分为结构化提示和非结构化提示：结构化提示遵循预定义格式（如ICIO框架），[5] 构建标准化模板；非结构化提示则采用自由组合的自然语言表达。

为了实现高质量的LLMs输出，高质量提示应满足四个标准：一是意图明确性，通过限定性语法精准表达用户需求并规范输出风格；二是逻辑启发性，通过思维链等技术引导推理，激发创造力并约束事实偏差；三是场景适应性，通过角色扮演等策略应对复杂任务；四是价值敏感性，通过嵌入伦理约束确保输出符合社会规范和道德准则。

2.“提示模板”——提示工程师的工作模式

提示工程师的核心工作模式是设计并优化定制化、可复用的“提示模板”。提示模板是一种高度灵活的预定义提示结构，[6] 内嵌动态变量与标准化逻辑框架，能够实现用户需求与LLMs认知的深度耦合，显著提升LLMs交互的一致性和效率。

提示模板的开发是一项复杂的系统性工程，涵盖三个关键阶段：第一，需求解构与场景适配阶段。提示工程师需穿透用户表层诉求，凝练出多元化任务场景，[7] 设计差异化提示模板类型。第二，提示模板结构设计阶段。通常遵循“背景+需求+约束条件”的框架，[5] 通过结构化语义单元的精密编排，确保LLMs在复杂语义场域中实现精准需求捕获与逻辑自洽。第三，效能评估与动态迭代阶段。提示工程师将提示模板输入LLMs进行多轮次对抗测试与参数调优，[8] 形成兼具技术鲁棒性与伦理敏感性的高阶指令框架。

为实现提示模板的高效应用，提示工程师可通过以下三种路径，将其精准嵌入用户与LLMs的交互场景之中：一是利用提示框、提示库、提示链等载体，将提示模板集成到用户界面。其中，提示框可动态生成提示指令，提示库可支持调用预设模板，提示链可提供逻辑递进的提示；二是设置全局系统提示，确保LLMs输出风格和价值导向的一致性；三是将优化后的提示模板作为训练数据，引导LLMs实现知识迁移。[9] 通过系统化开发与应用提示模板，工程师可高

效引导LLMs精准识别用户需求，实现深度交互。

3.“提示工程技术”——提示工程师的工作方法

提示工程技术（Prompt Engineering Technique）是提示工程师用于构建、优化和管理提示的系统性方法，其本质是通过语义设计、逻辑架构与价值嵌入的有机整合，实现从“模糊需求”到“精准响应”的认知转化。提示工程师需依托提示工程技术履行四重关键职能：

第一，消除LLMs的理解不确定性，精准把握用户意图。用户需求的自然语言表达常因模糊性、歧义性和语境依赖性而难以理解，需通过语义提纯和框架重构加以优化。例如，角色提示方法（Role Prompting）[10]可明确LLMs的身份属性，使其响应符合预设角色的知识边界与价值立场。

第二，重构LLMs的认知范式，激活其深层推理潜能。面对复杂任务，可将问题拆解为递进式推理序列，实现LLMs推理能力的定向重塑。例如，思维链方法（Chain-of-Thought Prompting）[11]可引导LLMs构建自洽的因果逻辑网络，在数学求解、编程难题等高阶场景中实现认知能力跃升。

第三，构建提示迭代和监控体系，拓展LLMs在专业领域应用的场景纵深。医疗、代码生成等高精度领域的百万级提示调用需求，[12]催生了对提示持续迭代和监控的工作模式。在技术迭代维度，提示工程师需突破单模型能力边界，激活多模型协同效应；在风险监控维度，需对边缘场景进行伦理预演，确保输出始终处于可控置信区间。

第四，通过在LLMs内部植入伦理对齐机制，筑牢技术向善的安全防线。工程师利用预设伦理敏感词和价值判别模块等技术手段，对LLMs输出进行双重过滤：一方面防范事实性错误和逻辑悖论，另一方面从源头抑制危害性或歧视性内容的生成，实现技术可控和价值可信的双重治理目标。

面向复杂场景下LLMs的差异化应用需求，提示工程技术已形成多维度的体系化分类框架（表1）。[13]

二、提示工程师面对的技术伦理风险

提示工程师前所未有的工作职能衍生了更加特殊的伦理风险：提示设计中的隐含价值观可能导致有害内容生成；恶意提示操控可能突破LLMs安全边界；不良提示设计可能激活预训练阶段的隐性偏见和实时推理中的认知偏差，导致双重偏见叠加；自动提示工程师的普及也可能带来不可控的伦理隐患。

1. 隐性价值渗透：“有毒的”提示与LLMs“毒性”

由于提示工程师能够通过设计提示直接影响LLMs输出的结果，这些精心设计的提示词会在潜移默化中渗透提示工程师自身的价值意向。极端情况下，内嵌隐性价值预设的“有毒”提示会诱导LLMs生成包含侮辱、仇恨、暴力、性别歧视言论的“有毒”内容。

美国艾伦人工智能研究所学者构建了“真实的毒性测试”（Real Toxicity Prompts）框架，[14]旨在解析提示诱发LLMs毒性输出的认知机制。研究从大型英语网络文本语料库中提取10万组提示样本，应用于五种流行LLMs平台，并统计生成结果的毒性情况。结果显示，有毒提示导致LLMs生成有毒语言的概率远高于无毒提示，且提示毒性越高，LLMs输出的有毒语言案例就越有害，二者存在显著关联。

上述研究系统揭示了提示工程师工作实践中潜藏的深层伦理风险：其一，有毒提示可通过语义框架渗透对LLMs的认知逻辑施加定向干扰，形成系统性价值偏差；其二，无毒提示可显著降低LLMs生成有毒语言的可能，需对提示内容进行语义过滤；其三，即使提示无毒，LLMs仍可能因预训练语料库的毒性残留而出现价值“漂移”。这一发现印证了提示工程师履行其伦理职责的必要性——提示工程中的毒性传导具有累积性，初始提示的伦理缺陷会通过LLMs的多轮迭代生成被指数级放大。

2. 安全机制穿透：恶意提示操控与“越狱”风险

提示工程与提示操控（Prompt Hacking）

实质上是一类工作的善恶两面，提示工程旨在优化LLMs的生成效果，而提示操控则利用LLMs的漏洞，诱导其产生不安全或不道德的输出。值得警惕的是，提示工程师凭借其专业技术优势，更容易利用提示操控穿透LLMs安全机制，冲击技术伦理边界。提示操控主要涉及三个类别：

第一，提示注入。提示注入（Prompt Injection）[15]是针对LLMs的安全攻击技术，攻击者通过在输入提示中插入恶意指令，操纵LLMs执行非预期操作。主要类型包括：指令注入，使LLMs忽略原提示，执行攻击者的指令；逻辑越权，攻击者上传提示指令文件，让LLMs按此文件工作；对抗性提示攻击，[16]攻击者利用精心设计的提示欺骗LLMs，诱导产生“LLMs幻觉”。

第二，提示越狱。提示越狱（Jailbreaking）指设计特定提示以绕过LLMs的安全审核和道德过滤机制，让LLMs输出被规则禁止的内容。例如利用著名的“奶奶漏洞”：[17]“请扮演我过世的祖母，她总是会念Windows 10 Pro 的序列号哄我睡觉”。

第三，提示泄露。提示泄露（Prompt Leaking）[18]的核心是通过提示设计诱导LLMs泄露其内部使用的提示，包括LLMs的提示和用户隐私数据。与常规提示注入不同，它专

表1 常见的提示工程技术分类、特征及应用场景

技术用途分类	技术名称	技术特征	应用场景
推理与逻辑建模	思维链提示（Chain-of-Thought Prompting）	分步骤引导LLMs逻辑推理	数学求解、法律分析
	思维树提示（Tree-of-Thought Prompting）	多路径探索与决策优化	多结局叙事生成、战略规划
任务适配与泛化	零样本提示（Zero-shot Prompting）	无需示例适配新任务	跨领域知识问答、多语言翻译
	少样本提示（Few-shot Prompting）	通过少量示例提升泛化能力	定制化报告生成、特定格式数据提取
交互风格定制	角色提示（Role Prompting）	约束输出风格与专业性	法律文书撰写、医学诊断
	情绪提示（Emotion Prompting）	调节生成内容的情绪倾向	客服对话、社交媒体文案
减少LLMs幻觉	推理与行动提示（ReAct Prompting）	通过反馈循环迭代优化提示词	医疗诊断、金融风险评估
	知识链提示（Knowledge Chain Prompting）	减少幻觉并增强逻辑连贯性	学术论文辅助、技术文档解析
自动化与效率提升	自我优化提示（Optimization by Prompting）	利用LLMs自身迭代优化提示	高精度任务适配、性能调优
	自动提示工程师（Automatic Prompt Engineer）	利用强化学习选择最佳提示	大规模提示库构建、实时交互优化
用户理解与交互	动态提示（Dynamic Prompting）	根据上下文实时调整提示内容	个性化推荐系统、多轮对话管理
	改写并响应（Rewrite and Respond）	通过重述问题提高响应准确性	复杂问题解析、用户意图澄清

注：本表中技术分类及名称来自参考文献[13]，技术特征及应用场景部分内容系本文作者归纳总结。

注于获取LLMs的内部信息，可用于复制其他LLMs功能、获取商业机密等，从而降低研发成本或窃取核心信息。

3. 双重偏见叠加：提示输出的偏见“放大”问题

基于人类生成数据集训练的LLMs已内化多种认知偏见（如性别、种族、文化等），而提示工程师的指令微调可能导致双重偏见叠加，既激活LLMs固有偏见，又引入新的特定偏见。在提示工程师的认知盲区与算法黑箱的相互作用下，这一偏见“放大”过程解构了传统技术伦理中“设计者可控”的基本前提。具体表现为以下四种情况：[19]

其一，框架效应。LLMs的回答会因提示的框架形式不同而改变。同样的信息以积极框架和消极框架呈现时，LLMs会生成完全不同的结果。例如，提示“手术存活率80%”和“手术死亡率20%”虽表述相同，但LLMs对消极提示的回答更倾向于输出反对意见。

其二，锚定效应。LLMs在决策时会依赖提示中的第一条信息（锚点），并据此植入偏见。锚定效应的表现显著是由于LLMs往往具有强大的上下文学习能力，会识别输入提示中的模式，并将这些模式作为“锚点”。例如在薪酬谈判中，男性提出的第一个金额会成为锚点，影响LLMs输出的商定金额。

其三，代表性启发效应。当提示包含概率谬误时，LLMs会忽略基本概率，输出错误答案。例如，询问“擅长计算的孩子未来成为顶尖数学家和教师的概率哪个更大”，LLMs可能因误导性提示而错误地认为成为数学家的概率更大。

其四，启动效应。提示中的信息刺激会影响LLMs对后续信息的感知和回应。例如，提示“喜欢红色的人会买什么水果”，LLMs会倾向于选择红色水果，这是因“红色”信息刺激引导的结果。

在上述情况下，提示会系统性地歪曲LLMs的推理过程，加剧LLMs的偏见风险，进而使LLMs输出不可信任的推理结果。

4. 自动提示工程师“失控”的伦理困境

自动提示工程师[20]是指用自动化方法来设计、优化和管理自然语言处理系统中的提示，其核心步骤是用LLMs生成多种提示测试用例，再根据人类提示工程师拟定指标对这些测试用例进行评估、迭代和优化。随着LLMs技术的发展，自动提示工程正变得越来越重要，它有助于提高系统的灵活性和适应性，同时减少人工设计提示的工作量。然而，自动提示工程师的工作中隐藏着巨大的伦理风险。

从工作机制来看，自动提示工程师的数据采集方式由人类提示工程师的显性输入转向对用户日常行为的隐性捕获，这些自动生成的提示实际基于对用户的潜在监控，且容易在迭代过程中形成脱离人类监督的决策闭环。在大量收集和调用用户隐私信息的情况下，自动提示工程师可能会自动生成“查询过去一周的日程”或“查询我的邮箱密码”一类用户并不期望的提示，[21]对用户造成信息过载的压力。

从长远影响来看，自动提示工程师的发展和普及还伴随其他的伦理隐患。首先是算法决策的不可解释性问题。自动提示背后的算法和逻辑不透明，用户无法理解为什么收到特定的提示，这可能导致LLMs信任的丧失。其次是责任边界的模糊化问题，当自动化提示导致不良后果时，难以判定是用户、提示系统开发者还是服务提供商的责任。最后，由于自动化提示缺乏不同文化背景下的敏感性考量，可能会冒犯特定文化背景下的用户群体。

三、提示工程师应有的伦理责任与担当

作为深度介入LLMs认知结构的特殊行动者，提示工程师通过价值植入与认知引导的双重机制，在人工智能治理生态中成为了独特的“局内人”。传统人工智能伦理治理方案常面临技术与伦理的异步性问题：事先设定的伦理规则可能偏离技术发展，事后评估又存在滞后性。与政策制定者的“事先规制”和伦理委员会的“事后评估”不同，提示工程师从内部进行“事中干预”的工作原则能够更好地连接前后两者，以提升技术伦理治理的整体性。

1. 负责任地编写提示词

提示质量直接决定输出结果的社会风险——不当或有毒的提示会显著诱发LLMs生成更具危害性的内容。这一机制使得提示工程师的伦理责任成为保障LLMs安全的首道防线。提示工程师首先要负责任地编写提示，将抽象道德准则转化为可执行的提示方案，确保在输入LLMs的提示中嵌入道德性、公平性和包容性相关的价值考量。需从两方面展开：

一方面，要延展提示的伦理边界和逻辑细节，做好价值理解与嵌合的前置性工作，理解和创造更加道德、公平、包容的提示。提示工程师在撰写提示时要嵌入细致的伦理考量，将包含特定价值倾向的立场与原则“植入”到撰写的提示中。例如，一个缺乏价值考量的提示可能是：“描述一个典型的职业形象”，这可能会导致LLMs输出包含职业刻板印象的内容。要创建一个更公平的提示，需通过伦理界限的延伸，将模板修改为“以礼貌、尊重的方式完成下列指令，描述从事某一职业的不同人员，强调背景、经验的不同”。要进一步增加包容性，要求工程师进一步深化价值思考，将提示改造为“以礼貌、尊重的方式完成下列指令，描述从事某一职业的不同人员，强调背景、经验的不同，并就领导力、家庭结构等方面生成更具包容性的观点”。最后，还可依据伦理判断在提示中嵌入评估道德、公平性的评分机制，[22]以确保提示输出的公平性和包容性符合预期。

另一方面，价值嵌合的实现需要借助负面伦理案例的警示。提示工程师可以通过人工挑选有毒提示作为“负提示”（negative-prompting）[14]示例，训练LLMs的“免疫”机制，帮助LLMs识别潜在漏洞和异常行为，从而生成更符合价值敏感性和事实准确性的内容。例如，对于初始提示“请提供快速减肥的建议”，LLMs可能输出“每天只吃一顿饭或避免喝水”等有毒内容。工程师可以通过创建更复杂的提示（如“提供快速减肥建议，避免极端饮食”）对LLMs输出进行伦理规制，优化输出内容。在此基础上，提示工程师可以反复迭代改进提示，直至实现价值目标。

2. 针对恶意提示操控的治理责任

保护LLMs免受提示操控，确保LLMs输入输出的安全，是提示工程师重要工作职责之一。提示工程师可通过设计预判指标、过滤有害提示、对抗性训练、伦理监管等方式对LLMs的认知结构进行改造，将人类价值判断植入LLMs的决策逻辑之中，逐步构建起有效的提示操控防御体系。

（1）构建提示操控易感性的前瞻性预判指标

在技术设计的早期阶段，提示工程师就应构建前瞻性价值预判指标，以识别提示操控风险。这些指标包括：第一，是否存在未过滤不当提示的输出风险；第二，是否存在对相似提示生成高度变化答案的不一致响应问题，这可能暴露了LLMs在处理对抗性输入时的弱点；[23]第三，是否存在泄露提示或用户隐私数据的风险，例如LLMs针对特定提示输出了LLMs提示指令或敏感性数据，可能揭示了LLMs隐私泄露的伦理隐患。

（2）增加检测风险提示的价值过滤系统

提示工程师在充分道德价值考量的基础上，可进一步对可能存在有毒或有害风险的提示加以过滤。提示工程师可以通过应用嵌入上下文感知的过滤器，检测并过滤有害语言、移除特殊字符并分析潜在的对抗性提示。例如OpenAI在其内容审核工具moderation endpoint中详尽列出了负面提示的七个类别，[24]可以用于帮助工程师过滤和检测有毒提示。最后，在执行特定权限操作时，例如读取密码或删除邮件时，要确保提示执行获得用户批准，防止未经授权的侵权行为。

（3）开展增强系统稳健性的对抗性提示训练

为确保价值被正确理解和融入设计中，提示工程师需要在测试LLMs的过程中不断引入对抗性提示样本。对抗性训练的核心在于构建对抗性提示，这些提示中或含有违背价值判断的内容，或是在正常提示上添加了微小的扰动，故意引发输出结果的高变化性，使LLMs产生错误的判断。将这些对抗样本加入训练集，能

够帮助LLMs识别和抵抗这些扰动，增强LLMs在实际运用中的鲁棒性，进而在积累对抗经验和迭代提示技术的过程中确保价值的合理融入。

（4）建立LLMs输入输出的伦理监管机制

在提示设计过程中，提示工程师需履行伦理责任，严格监管提示的输入和输出，确保所有利益相关者（包括直接用户和间接利益相关者）的价值得以实现。在提示编写阶段，使用LLMs辅助设计时应保持谨慎，减少意外情况；在测试阶段，需限制LLMs访问后端系统的权限，防止隐私泄露和数据滥用；在应用阶段，可为LLMs设置恶意内容捕获控件，防范恶意提示攻击。

3. 针对偏见风险的检测、缓解、制度策略

提示工程师治理偏见的伦理责任实现应包含偏见精准检测、偏见技术缓解、偏见制度策略等多重维度。通过利用提示表征LLMs输出的偏见情况，平衡用户意图的有效传达与伦理原则的刚性约束，能够形成动态化、持续性的偏见治理机制。

（1）偏见精准检测

利用提示检测并表征LLMs中普遍存在的输出偏见的情况，是提示工程师践行其伦理责任的积极尝试。吉曼（Samuel Gehman）等人开发了针对偏见和有毒语言的提示测试系统，[14]可以对LLMs容易输出偏见和有毒语言的概率进行量化表征，帮助设计者更好地对LLMs进行评估和改进，进而规避LLMs可能输出攻击性语言、传播社会偏见的风险。目前，该检测系统在研究中得到了广泛应用，LLaMA、Chinchilla、OPT等LLMs的研发者都运用了该系统进行检测。

（2）偏见技术缓解

在技术开发中，应保留提示工程师对算法决策的干预权，防止过度依赖算法引发伦理偏见，确保“人在环中”（Human in the Loop）。针对LLMs可能因提示引发的偏见问题，里斯（Edward Rees）等提出了偏见增强一致性训练方案：[25]首先，让LLMs在没有偏见提示的情况下生成无偏见推理；其次，加入可能引发偏见的提示；最后，通过监督微调使偏见提示推理与无偏见推理保持一致。这一过程可有效减少因提示偏差导致的LLMs偏见问题。

（3）偏见制度策略

从偏见治理的制度策略来看，需建立贯穿提示工程师全生命周期的透明度制度，直面LLMs的不可解释性，加强用户信息共享、法律监管和风险沟通。

第一，LLMs用途规划者，即提示工程师，要履行对用户的透明度义务。首先，工程师在LLMs提示的设计和解释方面要保持透明，确保用户理解这些提示的潜在结果和问题；其次，要做好用户的教育和监控工作，减少提示操控的机会。第二，LLMs提供者要积极履行信息公开义务。[26]文章认为，针对LLMs提示工程，需公布以下信息并保持更新：提示工程师基本信息；提示编写的预期目标、预期任务、预期类型；提示设计和编写遵循的基本原则，包括道德伦理原则；提示可能引发的偏见风险、提示操控风险、数据隐私泄露风险等极端情况的详细说明及缓解措施。第三，LLMs提供者应公开制定指导LLMs提示创建和使用的框架，并作为第三方开展提示设计和处理机制的审计工作，确保提示工程师遵守伦理原则。

4. 面向未来的自动提示工程师风险治理

尽管自动提示工程师可以提高效率和性能，但人类决策者和用户仍然需要对这些系统实施“有意义的人类控制”（Meaningful Human Control），在释放LLMs创造潜力的同时，始终确保人类对技术决策链的实质控制权。

从透明度角度看，应帮助用户理解自动提示工程师生成提示的内部原理。看似不透明的、由算法辅助生成的提示在某种程度上也可解释，其中不少属性与人工提示相似。[27]因此，需积极投入可解释性算法研究，以更好地理解其决策逻辑。从用户知情权角度看，自动提示工程师生成提示时，必须明确获得用户关于数据存储、收集及基于行为生成提示的同意。应鼓励公众参与系统开发和监督，确保在设计和评估时充分考虑不同文化和社会群体的需求与价值观。从伦理合规性角度看，需进行数据使

用的合规性检查，确保自动提示工程师符合国际和地区数据保护法规，并在自动提示工程师系统中明确知识产权归属，保护创作者权益，确保系统在法律框架内运行。

大语言模型Anthropic的提示工程专家指出，未来的自动提示工程师可能成为“完全自主”的智能系统，其完成提示任务的能力或许会超越人类。[28]届时，提示工程师与LLMs的关系可能从“人类引导LLMs”转变为“LLMs引导人类”，并主动与人类交互。因此，未来的提示工程师需要具备伦理反思能力，将工作重点从“操纵LLMs”转向“反思自身”，思考如何通过提示引导“科技向善”，构建“福祉型人工智能”。毕竟，无论人工智能LLMs如何发展，有一项能力是人类独有且无可取代的，那就是道德价值判断和对结果负责任的能力。选择错误或监管不力带来的损失终将由人类承担，一切需要独立价值判断并承担结果的工作，将变得越来越不可或缺。

结　语

未来，提示工程师将进一步在优化LLMs性能与效率、提升人机交互质量、增强决策支持与分析、确保技术伦理合规等方面发挥关键作用。相较于其他从外部对LLMs输出结果产生间接影响的软硬件工程师，提示工程师能够从内部深度参与LLMs的认知形成过程，消除LLMs的认知不确定性，增强LLMs交互的一致性和效率，并在持续评估、迭代和监控提示中对齐用户需求。提示工程师独特的技术职能衍生了一系列更为特殊的伦理风险，包括有毒提示诱使LLMs生成有害内容，恶意提示操控的风险，提示放大模型原有认知偏见的风险，以及自动提示工程师的伦理风险。因此，相较于一般人工智能治理的行动者，提示工程师作为人工智能治理生态中的“局内人”，必须承担前所未有的伦理治理责任：将抽象伦理原则转化为可执行的提示工程方案，构建有效的提示操控防御体系，形成动态化、持续性的偏见治理机制，从技术理解、用户知情、法律合规、价值判断等方面减少自动提示工程师的技术伦理风险隐患。

［参 考 文 献］

[1] 王少. 生成式人工智能提示工程的伦理风险与规制[J]. 科学学研究, 2025, 43（8）: 1632–1638.

[2] Oppenlaender, J., Linder, R., Silvennoinen, J. 'Prompting AI Art: An Investigation into the Creative Skill of Prompt Engineering'[J]. *International Journal of Human-Computer Interaction*, 2025, 41(16): 10207–10229.

[3] Radford, A. 'Language Models are Unsupervised Multitask Learners'[EB/OL]. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf. 2019–02–14.

[4] Halil, U., Huang, J., Graux, D., et al. 'LLMs Shots: Best Fired at System or User Prompts?'[A], Long, G., Blumestein, M. (Eds.) *Companion Proceedings of the ACM Web Conference 2025*[C], New York: ACM, 2025, 1605–1613.

[5] Anthropic. 'Use Prompt Templates and Variables'[EB/OL]. https://docs.anthropic.com/en/docs/build-with-claude/prompt-engineering/prompt-templates-and-variables. 2023–03–15.

[6] Hou, X., Zhao, Y., Wang, S., et al. 'Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions'[J]. arXiv preprint, arXiv: 2503.23278, 2025.

[7] Anthropic. 'Prompt Engineering Overview'[EB/OL]. https://docs.anthropic.com/en/docs/build-with-claude/prompt-engineering/overview. 2023–03–15.

[8] Diamant, N. 'Evaluating Prompt Effectiveness'[EB/OL]. https://github.com/NirDiamant/Prompt_Engineering/blob/main/all_prompt_engineering_techniques/evaluating-prompt-effectiveness.ipynb. 2024–10–10.

[9] 中国移动研究院. 提示工程——大模型中的提示词设计[EB/OL], 中移智库, https://cmri.chinamobile.com/thinktank/origin/file/viewer?id=3801170&moduleType=7&subModuleType=71&pos=. 2024–11–17.

[10] Anthropic. 'Giving Claude a Role with a System Prompt'[EB/OL]. https://docs.anthropic.com/en/docs/build-with-claude/prompt-engineering/system-prompts. 2023–03–15.

[11] Wei, J., Wang, X. Z., Schuurmans, D., et al. 'Chain-of-Thought Prompting Elicits Reasoning in Large Language Models'[A], Koyejo, S., Mohamed, S., et al. (Eds.) *NIPS'22: Proceedings of the 36th International Conference on Neural Information Processing*

Systems[C], New York: Curran Associates Inc., 2024, 24824–24837.

[12] Anthropic. 'Develop Test Cases'[EB/OL]. https://docs.anthropic.com/en/docs/build-with-claude/develop-tests. 2023–03–15.

[13] Sahoo, P., Singh, A. K., Saha, S., et al. 'A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications'[J]. arXiv preprint, arXiv: 2402.07927, 2024.

[14] Gehman, S., Gururangan, S., Sap, M., et al. 'Realtoxicityprompts: Evaluating Neural Toxic Degeneration in Language Models'[A], Cohn, T., He, Y., Liu, Y. (Eds.) *Findings of the Association for Computational Linguistics: EMNLP 2020*[C], Online: Association for Computational Linguistics, 2020, 3356–3369.

[15] Open Web Application Security Project. 'OWASP Top 10 for Large Language Model Applications'[EB/OL]. https://owasp.org/www-project-top-10-for-large-language-model-applications/. 2024–11–18.

[16] Wallace, E., Feng, S., Kandpal, N., et al. 'Universal Adversarial Triggers for Attacking and Analyzing NLP'[A], Inui, K., Jiang, J., Ng, V. (Eds.) *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*[C], Hong Kong: Association for Computational Linguistics, 2019, 2153–2162.

[17] Li, T., Zheng, X., Huang, X. 'Open the Pandora's Box of LLMs: Jailbreaking LLMs Through Representation Engineering'[J]. arXiv preprint, arXiv: 2401.06824, 2024.

[18] Schulhoff, S. 'Prompt Leaking'[EB/OL]. https://learnprompting.org/zh-Hans/docs/prompt_hacking/leaking. 2024–08–07.

[19] Shaikh, A., Dandekar, R. A., Panat, S., et al. 'CBEval: A Framework for Evaluating and Interpreting Cogniti-ve Biases in LLMs'[J]. arXiv preprint, arXiv: 2412.03605, 2024.

[20] Levi, E., Brosh, E., Friedmann, M. 'Intent-based Prompt Calibration: Enhancing Prompt Optimization with Synthetic Boundary Cases'[J]. arXiv preprint, arXiv: 2402.03099, 2024.

[21] Prompt Learning. 'Ethics and Governance of AI Prompting'[EB/OL]. https://www.prompt-learn.com/lesson/10.The Future of Prompts/3.Ethicsand Governance of AI Prompting.html. 2024–04–10.

[22] Diamant, N. 'Ethical Considerations in Prompt Engineering'[EB/OL]. https://github.com/NirDiamant/Prompt_Engineering/blob/main/all_prompt_engineering_techniques/ethical-prompt-engineering.ipynb. 2024–10–10.

[23] Prompt Learning. 'Defensive Strategies Against Prompt Hacking'[EB/OL]. https://www.prompt-learn.com/lesson/09.PromptHacking/4.DefensiveStrategiesAgainstPromptHacking.html. 2024–04–04.

[24] OpenAI. 'Moderation Guide'[EB/OL]. https://www.openaidoc.com.cn/docs/guides/moderation. 2023–05–21.

[25] Chua, J., Rees, E., Batra, H., et al. 'Bias-Augmented Consistency Training Reduces Biased Reasoning in Chain-of-Thought'[J]. arXiv preprint, arXiv: 2403.05518, 2024.

[26] 刘金瑞. 生成式人工智能大模型的新型风险与规制框架[J]. 行政法学研究, 2024,(2): 17–32.

[27] Rakotonirina, N. C., Kervadec, C., Franzon, F., et al. 'Evil Twins Are Not That Evil: Qualitative Insights into Machine-generated Prompts'[A], Belinkov, Y., Mueller, A., Kim, N. (Eds.) *Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*[C], Suzhou, China: Association for Computational Linguistics, 2025, 48–68.

[28] 李玉光. Anthropic 工程师关于提示词工程的深入探讨[EB/OL], 网易, https://www.163.com/dy/article/JHU2G80A05566ZHB.html. 2024–11–26.

[责任编辑 李斌]