【LLM学习】Applied LLMs：LLMs构建应用程序的实践经验总结（2）运营策略

方方 方方的算法花园 2024年06月16日 22:44 浙江

“本文主要针对《What We’ve Learned From A Year of Building with LLMs》文章进行了翻译和总结，原文是一个非常实用的指南，介绍如何利用LLMs构建成功的产品，文章内容比较长，我会分成战术应用、运营、战略三篇文章进行解读。”

原文地址：https://applied-llms.org/（发布时间：June 8, 2024）

作者：Eugene Yan、Bryan Bischof、Charles Frye、Hamel Husain、Jason Liu、Shreya Shankar

【LLM学习】Applied LLMs：LLMs构建应用程序的实践经验总结（1）战术应用

本文专注于运营策略方面，涉及产品推出的日常管理和团队构建，提供高效团队运作的实践指南。这一部分适用于希望可持续、可靠部署产品的技术领导者。

01 数据

正如高品质的食材是制作美味佳肴的关键，输入数据的质量同样对机器学习系统的表现起着决定性作用。输出数据则扮演着反馈机制的角色，是评估产品是否正常运作的关键指标。

因此，研究人员对数据的关注度极高，他们不仅会定期审视系统的输入和输出，还会深入分析数据的分布特性、识别极端情况，并探究模型的潜在局限。这种对数据的细致审视是确保机器学习系统高效、准确运行的基础。

1.1 检查开发与生产之间的数据偏移

在传统的机器学习流程中，"训练 - 服务偏移"是一个常见的问题源，它发生在训练数据与模型实际应用中的数据不一致的情况下。尽管大语言模型（LLM）可以在不涉及训练集的情况下使用，但开发与生产之间的数据偏差仍然可能发生。为了确保生产中的准确度，开发阶段测试的数据必须与系统实际使用中的数据保持一致。

大语言模型的数据偏移类型：

1. **结构性偏移：** 这种偏移涉及到数据格式上的不一致，例如JSON字典与列表类型值的差异、大小写不统一、拼写错误或句子不完整等。由于LLM对特定数据格式高度敏感，这些细微变化可能导致模型表现不稳定。
2. **基于内容的偏移：** 这种偏移涉及到数据的意义或上下文的差异。

偏移检测方法：

- **定期监测：** 定期检测LLM的输入与输出间的偏移，通过监测简单的指标，如输入输出的长度或特定格式要求（例如JSON或XML），来直接监测变化。
- **高级漂移检测：** 使用聚类技术对输入输出对的嵌入进行分析，以识别主题漂移，这可能表明用户正在探索模型未覆盖的新领域。

- **保留数据集的更新：** 确保使用的保留数据集是最新的，并能反映用户最近的交互方式。这包括在保留数据中反映生产中常见的输入错误。
- **输出质量评估：** 定期进行模型输出的质量评估，俗称“氛围审查”，以确保结果符合预期并满足用户需求。
- **非确定性偏移检查：** 通过对测试集中每个输入重复运行多次并分析所有输出，可以更好地捕捉偶尔出现的异常。

1.2 定期审查LLM的输入输出样本以优化性能

大语言模型（LLM）是不断发展的系统，虽然在零样本场景下表现出色，但其失败模式难以预测。因此，定期检查实际生产中的输入输出样本对于深入理解LLM的表现至关重要。

现场实物检查（Genchi Genbutsu）： 从实际生产环境中获取的输入输出对提供了不可替代的应用反馈。这种现场实物检查是评估LLM实际表现的关键。

评价标准漂移（Criteria Drift）： 随着开发者与数据的交互，对优质与劣质输出的认识可能会发生变化，这一现象称为评价标准漂移。因此，预先设定的评估标准可能不全面，需要定期更新。

循环评估与标准调整： 在开发过程中，可能需要更新提示语以提高优质回应的概率并降低劣质回应的出现。这一循环评估与标准调整的过程是必要的，因为直接观察输出前很难准确预测LLM的行为或用户偏好。

日志样本的每日检查： 通过每日检查LLM的输入和输出日志样本，我们可以迅速识别并适应新出现的模式或失败模式。一旦发现问题，应立即进行断言或评估处理。

评价标准的及时更新： 任何对失败模式定义的更新都应及时反映在评价标准中，以确保评估的准确性。

情况检查与团队推广： 这些“情况检查”是判定输出质量的重要信号，应通过编码和断言来具体操作。此外，这种做法应在团队中推广，例如在日常轮值中增加对输入和输出的审核或标注任务。

02 模型应用

通过LLM的 API，我们能够依赖少数供应商提供的智能服务。这虽然为我们带来便利，同时也意味着我们需要在性能、响应时间、数据处理能力和成本等多方面进行权衡。随着每个月都有新的、更优秀的模型推出，我们必须随时准备升级我们的产品，从旧模型过渡到新模型。在这一节中，我们将分享在使用那些无法完全控制、无法自托管和管理的技术时的心得体会。

2.1 优化LLM输出以适应结构化集成

在众多实际应用场景中，大语言模型（LLM）的输出需要以机器可读的格式服务于下游应用程序：

- **Rechat系统**：一个房地产客户关系管理（CRM）系统，它需要前端能够渲染的结构化响应。
- **Boba工具**：一个产品策略创意生成工具，它需要包含标题、摘要、可行性评分和时间框架的结构化输出。

- **LinkedIn应用**：展示了如何利用LLM生成YAML格式的输出，以决定使用哪些技能，并提供执行这些技能所需的参数。

这种模式体现了Postel法则（即“在接受时宽容，在发送时严格”）的应用，预计这种模式因其高效性将具有持久的生命力。

指导LLM输出结构化数据的工具：

- **Instructor**：适用于使用LLM API（如Anthropic或OpenAI）的情况，Instructor是输出结构化数据的行业标准工具。
- **Outlines**：适用于使用自托管模型（如Huggingface）的情况，Outlines提供了类似的指导功能。

2.2 跨模型迁移提示指令的挑战

在大语言模型（LLM）的应用中，精心构建的提示指令可能在某些模型上表现出色，而在其他模型上效果不佳。这种情况常出现在更换模型提供商或升级同一模型的不同版本时。

案例分析：

- **Voiceflow的迁移经验：** 在从gpt-3.5-turbo-0301迁移到gpt-3.5-turbo-1106的过程中，Voiceflow发现其意图分类任务的准确率下降了10%。这一发现得益于他们进行的评估工作。
- **GoDaddy的性能观察：** GoDaddy注意到，在升级到版本1106后，gpt-3.5-turbo与gpt-4之间的性能差异有所减少，这可能让一些期待gpt-4保持领先优势的人感到失望。

迁移提示指令的注意事项：

- 迁移提示指令到不同模型时，所需投入的时间和努力可能远超简单的API端点更换。
- 我们不应自动期望同一提示指令在不同模型中能够产生相同或更优的效果。
- 拥有可靠的自动化评测系统至关重要，它可以帮助我们在迁移前后测量任务性能，从而减少人工验证的工作量。

2.3 固定模型版本以维护稳定性

在机器学习流程中，"改变任何事物都将改变一切"的原则至关重要。这一点在依赖可能悄无声息发生变化的外部组件，如大语言模型（LLM）时，尤为突出。

固定模型版本的益处：

- **供应商支持：** 许多模型供应商提供了固定特定版本模型的选项，例如gpt-4-turbo-1106。这使我们能够使用特定版本的模型权重，确保其不会发生改变。
- **生产环境的稳定性：** 在生产环境中固定模型版本可以防止模型行为出现不预期的变化，减少因模型更换引起的客户投诉，如过度冗长的输出或其他未预见的失误。

影子流程的实施：

- **影子环境：** 维护一个镜像生产环境的影子流程，但使用最新的模型版本，可以安全地进行新版本的测试和尝试。
- **逐步验证：** 通过影子流程，我们可以验证新模型输出的稳定性和品质，确保在更新生产环境前，新版本的表现符合预期。

2.4 精选小型模型以优化任务性能

在开发新应用时，我们常被诱惑去选择最强大的模型。然而，一旦验证了任务的可行性，探索更小型的模型可能会带来意外的效益。

小型模型的优势：

- **速度与成本：** 小型模型通常响应更快，运行成本更低。
- **性能提升技术：** 通过思考链路、多样本提示和上下文学习等技术，小型模型有可能实现与大型模型相媲美的性能

微调和工作流设计：

- **微调：** 对特定任务进行微调可以显著提升小型模型的性能。
- **工作流优化：** 精心设计的工作流可以使小型模型的表现甚至超越大型模型，同时实现更快的响应和更低的成本。

实例与展望：

- **Haiku与十样本提示：** 例如，Haiku模型结合十样本提示在某些情况下能胜过无样本的Opus和GPT-4。
- **未来趋势：** 预计未来将出现更多通过优化小型模型来平衡输出质量、响应速度和成本的工作流程设计。

分类任务的小型模型示例：

- **DistilBERT：** 轻量级模型DistilBERT（67M参数）已经是一个强大的基准。
- **DistilBART：** 在开源数据上微调的DistilBART（400M参数），在识别虚假内容的任务上展现了0.84的ROC-AUC效率，远超大多数大型模型，同时大幅降低了延迟和成本。

03 产品

尽管新技术开辟了前所未有的可能性，优秀产品的构建原则却历久弥新。因此，即使我们首次面对新问题，也无需在产品设计上重新起步。坚实的产品基础能让我们在开发大语言模型应用时，为服务对象创造真正的价值。

3.1 设计师在AI产品设计中的早期与深入参与

设计师的早期参与对产品构建和呈现方式至关重要。设计师的角色远不止美化产品，他们通过深度挖掘和优化用户体验，推动产品创新，甚至可能需要颠覆传统规则和模式。

设计师的专长与贡献：

- **用户需求转化：** 设计师擅长将用户需求转化为具体形态，探索更有效的问题解决方案，为AI技术的应用提供更多机会。
- **任务导向设计：** 构建AI产品应聚焦于用户需要完成的任务，而非单纯侧重技术本身。

设计思维的应用：

- **需求分析：** 深入思考产品如何满足用户需求，例如考虑是否适合采用聊天机器人、自动填充功能，或是否需要开发全新的方法。

- **设计模式审视：** 思考现有设计模式如何服务于用户需求，并探索它们在AI产品中的适用性。

设计师带来的价值：设计师为团队带来的不仅是美学视角，更重要的是他们的设计思维，这种思维能够促进团队从用户的角度出发，创造出更贴合用户需求的AI产品。

3.2 设计UX以实现人在环中（HITL）

在用户体验（UX）设计中融入人在环中（Human-in-the-Loop, HITL）的概念，是获取高质量数据标注的有效途径。这种方法不仅能够即时优化输出结果，还能收集数据以提升模型性能。

电商平台的HITL设计示例：

- **用户初步分类与LLM核查：** 用户挑选商品类别，LLM定期检查并纠正分类错误。
- **LLM后台自动分类：** 用户无需选择类别，LLM在后台自动分类，但可能存在错误。
- **LLM实时推荐与用户调整：** LLM向用户推荐商品类别，用户确认并根据需要调整。

这三种方法展示了LLM在用户体验中的不同作用，从初步分类任务到核查工具，再到实时推荐系统。

HITL的优势：

- **减轻用户负担：** 第三种方法平衡了用户的认知负担，避免了必须熟悉分类体系的需求。
- **确保用户控制权：** 用户可以审核并修改LLM的建议，确保控制权在自己手中。
- **形成反馈循环：** 合适的建议被采纳，不恰当的建议被修正，形成自然模型改进的反馈循环。

HITL在不同应用场景的实践：

- **编程助手：** 用户可以完全采纳、稍作修改后采纳或忽略建议。
- **Midjourney：** 用户可以选择升级下载、修改图片或生成新的图片。
- **聊天机器人：** 用户可以通过点赞、点踩或选择重新生成回答来提供反馈。

反馈的类型：

- **直接反馈：** 用户应产品请求提供的信息，如点赞和点踩。
- **间接反馈：** 通过用户互动行为获得的信息，如编程助手和Midjourney的使用情况。

3.3 优先排序产品要求的关键步骤

在将演示产品投入使用的过程中，我们必须仔细考虑并优先排序以下关键要求：

1. **可靠性：** 确保99.9%的在线时间，并保证输出的结构化。
2. **无害性：** 避免生成攻击性、不适宜工作场所或其他有害内容。
3. **事实一致性：** 保持对提供背景信息的忠诚度，不编造内容。
4. **有用性：** 满足用户的需求和请求。
5. **可扩展性：** 符合响应时间协议，支持所需的吞吐量。
6. **成本：** 考虑到有限的预算限制。
7. **合规性：** 包括安全、隐私、公平性、GDPR、DMA等其他法规和标准。

优先排序的必要性：尝试一次性解决所有这些要求可能导致产品无法及时推出。因此，我们必须果断地确定哪些要求是基本的，哪些是次要的。基本要求（如可靠性和无害性）是产品能够

正常运行或市场化的前提。

找到最受欢迎的基本产品：关键在于识别并集中资源开发最受欢迎的基本产品功能，这些功能是用户最看重的。

接受并优化首个版本：我们必须接受并认识到首个版本不会完美，应该尽快将其推向市场，并根据用户反馈进行调整。

3.4 用例驱动的风险承受能力调整

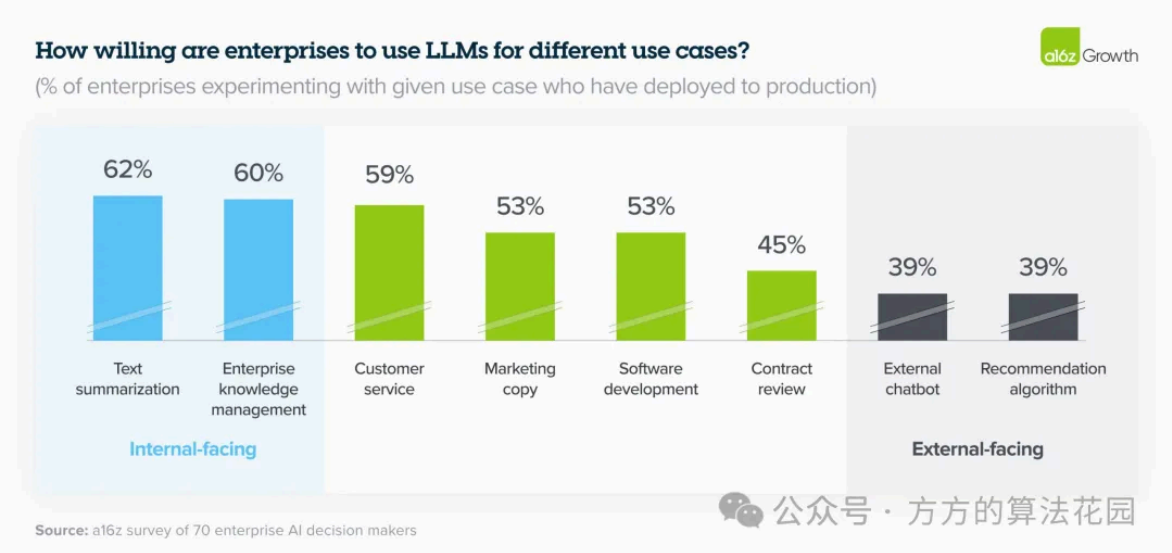
在选择语言模型和确定应用程序审查标准时，必须基于具体用例和目标受众来调整风险承受能力。

高风险用例的严格标准：在提供医疗或财务咨询等面向客户的聊天机器人中，我们必须设定极高的安全和准确性门槛。这是因为任何错误或失误都可能导致严重的后果，并严重损害用户对品牌的信任。

低风险用例的灵活标准：对于一些关键性较低的应用，例如推荐系统或内部使用的内容分类和总结工具，过于严苛的要求可能会不必要地拖慢开发进程，并不会带来相应的价值增加。

企业内部应用的快速进展：正如最新的a16z报告所指出的，许多企业在推进内部大语言模型应用的速度超过了外部应用。通过在内部试验AI技术，企业不仅可以快速实现价值，还能在一个更加可控的环境中学习如何管理风险。

逐步扩展到客户面向应用：随着对风险管理的信心增强，企业可以逐步将应用扩展到面向客户的场景，从而在确保安全和准确性的同时，实现更广泛的市场应用。



企业在内部和外部用例中使用大语言模型的比例 (来源：a16z 报告)

04 团队及其角色

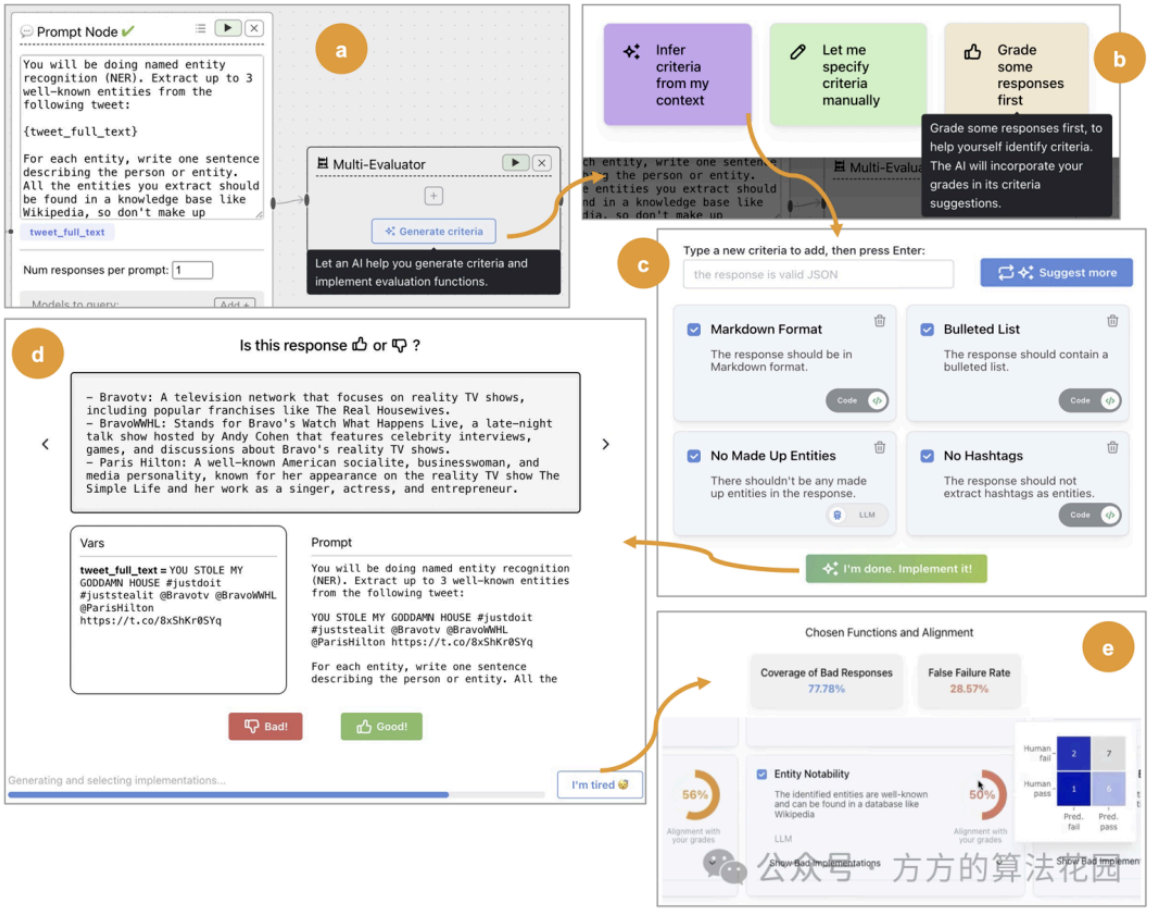
虽然定义任何一个职位都不是件容易的事，但为这个新兴领域撰写职位描述尤其具有挑战性。我们不会采用交叉职称的维恩图或是职位描述的建议。相反，我们将确认 AI 工程师这一新兴角色，并探讨其定位。同时，我们还会讨论团队其他成员的职责分配方式。

4.1 注重过程而非工具

在面对像大语言模型 (LLM) 这样的新兴技术时，软件工程师往往更倾向于关注工具。这种偏好导致我们忽略了工具应解决的问题和相关过程，进而引入了无意的复杂性，严重影响了团队的长期效率。

“Fuck You, Show Me The Prompt.”（https://hamel.dev/blog/posts/prompt/）讨论了一些工具如何自动生成大语言模型的提示。文章正确地指出，不先理解解决问题的方法或流程就急于使用这些工具的。

此外，这些工具往往规定不明确。比如，目前市场上涌现出许多评估大语言模型的工具，它们提供了一套“即开即用的大语言模型评估工具”，包括对毒性、简洁度、语调等的评估。但我们看到，许多团队在没有深入考虑自己业务领域的特定风险时就匆忙采用这些工具。与之形成鲜明对比的是 EvalGen，它专注于教育用户如何参与到每一个步骤中，从定义评估标准到标记数据，再到评估验证，以此来创建符合行业特点的评估流程。软件将用户引导完成以下工作流程：



Shankar, S., et al. (2024). 谁来验证验证者？将大语言模型辅助的评估与人类偏好相对齐（ https://arxiv.org/abs/2404.12272）

EvalGen 引导用户实施一套最佳实践，以制定有效的大语言模型评估：

（1）定义特定领域的测试，这些测试根据提示自动形成，并可以通过代码断言或使用大语言模型作为评判。
（2）强调与人类判断的一致性，确保测试能准确反映出设定的标准。
（3）随着系统（如提示）的更新，持续改进这些测试。

EvalGen 为开发者提供了一个关于评估构建过程的心理模型，不局限于任何特定工具。我们发现，在向 AI 工程师提供这种背景知识后，他们通常会选择更简洁的工具或自己开发工具。

LLM 包含的组件众多，这里不可能详尽列举。但 AI 工程师在开始使用这些工具之前，了解这些过程是非常重要的。

4.2 持续进行新实验以推动机器学习产品发展

实验在产品开发中的作用：机器学习产品的发展与实验密切相关，这不仅包括A/B测试或随机对照试验，更涉及到不断尝试修改系统的最小组件并进行离线评估。

评估对实验的促进作用：评估的重要性不仅在于其可信度和可靠性，更在于它能够加速迭代过程，从而快速优化系统。

实验成本的降低：当前实验成本极低，使得用不同方法解决相同问题成为常态。数据收集和模型训练的成本已大幅降低，提示工程的成本主要为人力成本。

团队的提示工程能力：确保团队成员掌握提示工程的基本知识，以激励他们进行实验，产生创新想法。

实验的多重目的：实验不仅是探索新领域的手段，也是充分利用现有资源的方式。对于新任务的初步版本，鼓励团队成员尝试不同方法或寻找更快的解决方案。

探索高效的提示技术：利用连锁思维或少样本等高效提示技术来提升任务品质，不要因工具限制而阻碍实验。

工具的重建与优化：如果现有工具限制了实验，不要犹豫重建工具或寻求更优秀的替代品。

产品规划中的实验与评估：在产品或项目规划时，预留足够时间建立评估系统和进行多次实验。在规格书中加入明确的评估标准，并在项目路线图中合理安排实验和评估时间。

4.3 普及AI技术使用，实现团队成员能力提升

AI技术的普及目标：随着生成式AI的广泛应用，我们期望团队中的每个成员，不仅限于专家，都能理解并掌握这项技术。直接使用大语言模型（LLMs）是培养对AI工作原理感知的有效途径。

大语言模型的易用性：LLMs的使用门槛低，即使不具备编程技能，也能通过优化提示和评估来提升性能。

教育的重要性：教育是帮助团队成员掌握AI技术的关键。可以从提示工程的基础开始，学习如何使用n-shot提示和思维链（CoT）技术来调整模型输出。

技术性讲解的价值：熟悉这些技术的人能够更深入地理解LLMs的自回归输出生成方式，包括输入和输出token的处理差异及其对响应时间的影响。

提供实践机会：为团队成员提供实际操作和探索AI技术的机会，例如举办编程马拉松，这不仅能激发创新思维，还能加速技术掌握和应用。

编程马拉松的成效：尽管投入时间看似成本高，但编程马拉松能够带来意想不到的成果。一些团队通过这种方式显著缩短了项目开发周期，有的则催生了创新的用户体验设计。

创新与加速实现：一些团队通过编程马拉松快速实现了原计划多年的发展规划，而其他团队则探索出利用LLMs能力创新的设计方案，这些已成为未来重点项目。

4.4 构建AI产品：超越"仅有AI工程"的误区

对AI工程师角色的误解：随着新职位的出现，人们往往对这些角色的能力有过高的期望。例如，数据科学家和机器学习工程师（MLE）在早期被认为能够独立胜任数据驱动项目，但很快发现他们需要与软件和数据工程师合作。

AI工程师的新兴角色：当前，AI工程师这一新兴角色也面临着类似的误解。一些团队错误地认为拥有AI工程师就足够了，而忽视了构建机器学习或AI产品需要广泛的专业角色。

多角色合作的必要性：在AI产品开发过程中，我们发现企业常常落入"只要有AI工程就行"的误区，导致产品难以从演示阶段向广泛应用扩展。有效的评估技能，与机器学习工程师的传统优势相符，是构建产品过程中的关键环节。

角色类型及其必要时机：

1. **产品构建阶段：** 需要AI工程师进行快速原型设计和产品迭代，但并非始终必需。
2. **系统基础建设：** 根据数据类型和规模，可能需要部署平台和/或数据工程师，建立数据查询及分析系统。
3. **AI系统优化：** 设计评估指标、建立评估体系、进行实验、优化信息检索、调试随机系统等，这些工作适合机器学习工程师（AI工程师也能胜任）。
4. **领域专家的参与：** 在小型公司，由创始团队担任；在大公司，由产品经理承担。领域专家的参与对于确保产品符合实际需求至关重要。

招聘和职责安排：明智地安排人员的招聘和职责顺序对于资源的有效利用和团队稳定至关重要。不适当的招聘时机或错误的建设顺序可能导致资源浪费和员工流失。

#LLM学习 12

#LLM学习 · 目录

上一篇
【LLM学习】Applied LLMs：LLMs构建应用程序的实践经验总结（1）战术应用

下一篇
【LLM学习】Applied LLMs：LLMs构建应用程序的实践经验总结（3）战略层面