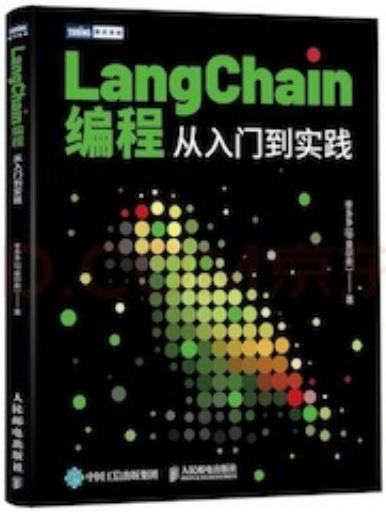
LangChain 就是一个 LLM 编程框架，你想开发一个基于 LLM 应用，需要什么组件它都有，直接使用就行；甚至针对常规的应用流程，它利用链(LangChain中Chain的由来)这个概念已经内置标准化方案了。下面我们从新兴的大语言模型（LLM）技术栈的角度来看看为何它的理念这么受欢迎。

我的新书《LangChain编程从入门到实践》已经开售！推荐正在学习AI应用开发的朋友购买阅读！



我们先看看官方的定义

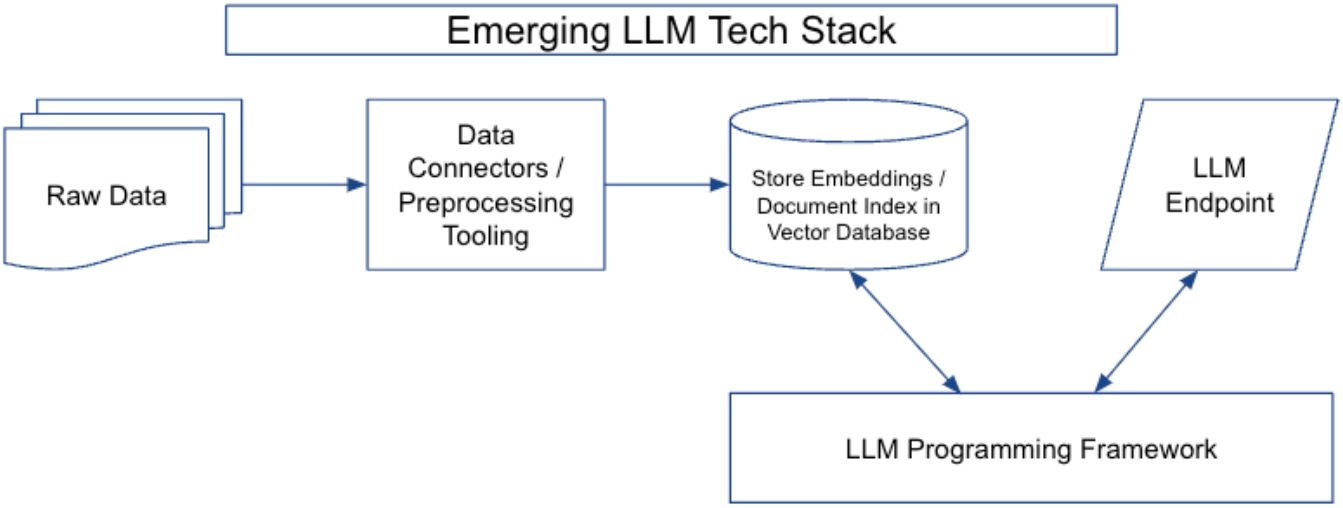
> LangChain是一个基于语言模型开发应用程序的框架。它可以实现以下应用程序：
>
> - 数据感知：将语言模型连接到其他数据源
> - 自主性：允许语言模型与其环境进行交互
>
> LangChain的主要价值在于：
>
> - 组件化：为使用语言模型提供抽象层，以及每个抽象层的一组实现。组件是模块化且易于使用的，无论您是否使用LangChain框架的其余部分。
> - 现成的链：结构化的组件集合，用于完成特定的高级任务
>
> 现成的链使得入门变得容易。对于更复杂的应用程序和微妙的用例，组件化使得定制现有链或构建新链变得更容易。

LangChain 就是一个 LLM 编程框架，你想开发一个基于 LLM 应用，需要什么组件它都有，直接使用就行；甚至针对常规的应用流程，它利用链(LangChain中Chain的由来)这个概念已经内置标准化方案了。下面我们从新兴的大语言模型（LLM）技术栈的角度来看看为何它的理念这么受欢迎。

新兴 LLM 技术栈

大语言模型技术栈由四个主要部分组成：

- 数据预处理流程（data preprocessing pipeline）
- 嵌入端点（embeddings endpoint ）+向量存储（vector store）
- LLM 终端（LLM endpoints）
- LLM 编程框架（LLM programming framework）



数据预处理流程

该步骤包括与数据源连接的连接器（例如S3存储桶或CRM）、数据转换层以及下游连接器（例如向矢量数据库）。通常，输入到LLM中的最有价值的信息也是最难处理的（如PDF、PPTX、HTML等），但同时，易于访问文本的文档（例如.DOCX）中也包含用户不希望发送到推理终端的信息（例如广告、法律条款等）。

因为涉及的数据源繁杂（数千个PDF、PPTX、聊天记录、抓取的HTML等），这步也存在大量的 dirty work，使用OCR模型、Python脚本和正则表达式等方式来自动提取、清理和转换关键文档元素（例如标题、正文、页眉/页脚、列表等），最终向外部以API的方式提供JSON数据，以便嵌入终端和存储在向量数据库中。

嵌入端点和向量存储

使用嵌入端点（用于生成和返回诸如词向量、文档向量等嵌入向量的 API 端点）和向量存储（用于存储和检索向量的数据库或数据存储系统）代表了数据存储和访问方式的重大演变。以前，嵌入主要用于诸如文档聚类之类的特定任务，在新的架构中，将文档及其嵌入存储在向量数据库中，可以通过LLM端点实现关键的交互模式。直接存储原始嵌入，意味着数据可以以其自然格式存储，从而实现更快的处理时间和更高效的数据检索。此外，这种方法可以更容易地处理大型数据集，因为它可以减少训练和推理过程中需要处理的数据量。

LLM终端

LLM终端是接收输入数据并生成LLM输出的终端。LLM终端负责管理模型的资源，包括内存和计算资源，并提供可扩展和容错的接口，用于向下游应用程序提供LLM输出。

LLM编程框架

LLM编程框架提供了一套工具和抽象，用于使用语言模型构建应用程序。在现代技术栈中出现了各种类型的组件，包括：LLM提供商、嵌入模型、向量存储、文档加载器、其他外部工具（谷歌搜索等），这些框架的一个重要功能是协调各种组件。

关键组件解释

Prompts

Prompts用来管理 LLM 输入的工具，在从 LLM 获得所需的输出之前需要对提示进行相当多的调整，最终的Promps可以是单个句子或多个句子的组合，它们可以包含变量和条件语句。

Chains

是一种将LLM和其他多个组件连接在一起的工具，以实现复杂的任务。

Agents

是一种使用LLM做出决策的工具，它们可以执行特定的任务并生成文本输出。Agents通常由三个部分组成：Action、Observation和Decision。Action是代理执行的操作，Observation是代理接收到的信息，Decision是代理基于Action和Observation做出的决策。

Memory

是一种用于存储数据的工具，由于LLM 没有任何长期记忆，它有助于在多次调用之间保持状态。

典型应用场景

- 特定文档的问答：从Notion数据库中提取信息并回答用户的问题。
- 聊天机器人：使用Chat-LangChain模块创建一个与用户交流的机器人。
- 代理：使用GPT和WolframAlpha结合，创建一个能够执行数学计算和其他任务的代理。
- 文本摘要：使用外部数据源来生成特定文档的摘要。

Langchain 竞品

（个人认为）在商业化上，基于大模型业务分为三个层次：

- 基础设施层：通用的大模型底座
- 垂直领域层：基于大模型底座+领域场景数据微调形成更强垂直能力
- 应用层：基于前两者，瘦前端的方式提供多样化应用体验

类似 LangChain 这种工具框架可以做到整合各个层能力，具备加速应用开发和落地验证的优势，因此也出现了很多竞争者。

名称	语言	特点点
LangChain	Python/JS	优点：提供了标准的内存接口和内存实现，支持自定义大模型的封装。 缺点：评估生成模型的性能比较困难。
Dust.tt	Rust/TS	优点：提供了简单易用的API，可以让开发者快速构建自己的LLM应用程序。 缺点：文档不够完善。
Semantic-kernel	TypeScript	优点：轻量级SDK，可将AI大型语言模型（LLMs）与传统编程语言集成在一起。 缺点：文档不够完善。
Fixie.ai	Python	优点：开放、免费、简单，多模态（images, audio, video...） 缺点：PaaS平台，需要在平台部署

不可错过 👉：之前创建的 LLM 应用开发交流群已经运行 1 年了，里面会实时分享 AI 应用开发生态最新进展，欢迎感兴趣的朋友加入交流，请备注**【入群交流】**

如果觉得内容不错，欢迎**订阅邮件和RSS，转发文章**～

参考链接

1. LangChain官方文档
2. LLMs和新兴的机器学习技术栈