

# github上langchain的介绍

---

先看看github上面的介绍：

大型语言模型（LLM）正在成为一种变革性的技术，使开发者能够建立他们以前无法建立的应用程序。然而，孤立地使用这些LLM往往不足以创建一个真正强大的应用程序--当你能将它们与其他计算或知识来源相结合时，真正的力量才会出现。

这个库的目的是协助开发这些类型的应用。这些应用程序的常见例子包括：

- 对特定文件的问题回答
- 聊天机器人
- 代理

## 文档

---

- 入门（安装，设置环境，简单的例子）。
- 操作实例（演示、集成、辅助功能）
- 参考（完整的API文档）
- 资源（核心概念的高级解释）

## 这能帮助什么？

---

LangChain旨在帮助六个主要领域。这些领域按复杂程度递增：

### LLMs和提示：

这包括提示管理、提示优化、所有LLM的通用接口，以及使用LLM的通用工具。

### 链

链超越了单一的LLM调用，涉及到一系列的调用（无论是对LLM还是不同的工具）。LangChain为链提供了一个标准接口，与其他工具进行了大量的集成，并为常见的应用提供了端到端的链。

### 数据增强的生成

数据增强生成涉及特定类型的链，它首先与外部数据源交互以获取数据用于生成步骤。这方面的例子包括对长篇文字的总结和对特定数据源的提问/回答。

### 代理

代理涉及到LLM决定采取哪些行动，采取该行动，看到一个观察结果，并重复该行动直到完成。LangChain提供了一个标准的代理接口，提供了一些可供选择的代理，以及端到端代理的例子。

# 内存

内存指的是在链/代理的调用之间持续保持状态。LangChain提供了内存的标准接口、内存实现的集合，以及使用内存的链/代理的例子。

# 评估

**[BETA]** 生成式模型是出了名的难以用传统的指标来评估。评估它们的一个新方法是使用语言模型本身来进行评估。LangChain提供了一些提示/链来协助这个工作。

关于这些概念的更多信息，请看我们的完整文档。

# 贡献

作为一个快速发展领域的开源项目，我们对贡献极为开放，无论是新功能、改进的基础设施还是更好的文档。

## github上介绍自我总结

---

看完github上面的介绍，我们有一个初步的认识了，假设现在我问你langchain是干嘛用的，你能怎么回答呢？

- 首先，langchain是一个开源框架，旨在方便使用LLM进行计算和开发应用，比如基于知识的问答、聊天机器人、使用代理进行各种任务。
- 其次，它主要应用在六大领域里面：
  - 管理和优化prompt。不同的任务使用不同prompt，如何去管理和优化这些prompt是langchain的主要功能之一。
  - 链，初步理解为一个具体任务中不同子任务之间的一个调用。
  - 数据增强的生成，如名。
  - 代理，代理的话，初步认识就是根据不同的指令采取不同的行动，直到整个流程完成为止。
  - 评估，如名。
  - 内存：在整个流程中帮我们管理一些中间状态。
- 最后，可以理解为：在一个流程的整个生命周期中，管理和优化prompt，根据prompt使用不同的代理进行不同的动作，在这期间使用内存管理中间的一些状态，然后使用链将不同代理之间进行连接起来，最终形成一个闭环。

这就是初步看完github上面的一些介绍所得到的总结，当然可能会有所错误，我们继续慢慢看下去。

接下来我们去看看它的官方文档，看看有什么说法没有。

## 官方文档介绍

---

官方文档里面也有一小部分介绍：

LangChain是一个框架，用于开发由语言模型驱动的应用程序。它使应用程序具有以下特点：

- 数据感知：将语言模型与其他数据源连接起来。
- 代理性：允许语言模型与环境互动。

多了一个数据感知，也就是可以处理不同的数据源。这里的代理性是允许语言模型和环境交互，我们之前的理解是使用代理能够通过指令执行不同的任务，意思应该相差不大。

LangChain的主要价值道具是：

- 组件：用于处理语言模型的抽象概念，以及每个抽象概念的实现集合。无论你是否使用LangChain框架的其他部分，组件都是模块化的，易于使用。
- 现成的链：用于完成特定高级任务的组件的结构化组合。现成的链使人容易上手。对于更复杂的应用和细微的用例，组件使得定制现有链或建立新链变得容易。

链，我们之前理解的差不多，也就是将一个应用的整个流程连接起来。额外的，这里多了一个组件概念。可以这么理解，链是有模块化的组件构成的，而一个应用程序则是由许多这种链组成的。langchain给我们提供了许多现有的链组成的高级应用程序。而我们也可以使用自定义的链来进行相关应用程序的开发。

## 组件

---

既然说到了组件，那我们来看看都有哪些组件。LangChain为以下模块提供标准的、可扩展的接口和外部集成，这些模块从最不复杂到最复杂排列：

- model I/O：语言模型接口
- data connection：与特定任务的数据接口
- chains：构建调用序列
- agents：给定高级指令，让链选择使用哪些工具
- memory：在一个链的运行之间保持应用状态
- callbacks：记录并流式传输任何链的中间步骤

## 最终总结

---

到这里为止，我们应该对langchain有了一个初步的认识，所以langchain是什么呢，我们简要进行一个回答：

- langchain是一个开源框架，它旨在系统化LLM进行计算和构建应用程序的整个流程。（整体介绍）
- 它允许我们使用不同的数据源。它可以帮助我们管理和优化相关的prompt。（从输入的角度）
- 一个应用程序可以表示为多个链的组成。通过代理，可以让链选择不同的工具、LLM。而链由不同的模块化组件构成。我们可以自定义这些组件来实现特定的需求。（从具体结构出发）
- 总而言之，langchain可以帮助我们快速开发出应用程序，比如知识库的问答，聊天机器人，处理传统自然语言处理中的各种任务等。（从最终的应用场景出发）

由于这只是一个初步的了解，可能有一些不完备的地方，欢迎指出。接下来可能先去了解一下langchain中的六大组件，然后在结合不同的案例加深理解。