基于langchain+LLM大模型，从0到1教你搭建AI agent 应用

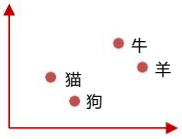
基于langchain+LLM大模型，从0到1教你搭建AI agent 应用

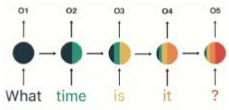
- 课程目标、受众、形式
- 课程规划大约11章大课
- 理论+实践相结合
- 课程特色

多模型强应用：AI2.0时代应用开发者机会

LLM(大语言模型)发展一览

猫[1,0,0,0,0]
狗[0,1,0,0,0]
牛[0,0,1,0,0]
羊[0,0,0,1,0]





OpenAI

词向量 → **词嵌入 embedding** → **句向量 全文向量** → **理解上下文** → **大模型**

1.无法表达词语之间的关系
2.效率不高

1.低维向量表达
2.语义相似向量空间相近
3.迁移学习

1.有效处理序列数据
2.短期记忆与选择性遗忘
3.文本生成、语音识别、图像描述等

1.支持并行训练
2.替代RNN与CNN
3.完形填空

1.BERT一个任务一个模型
2.GPT基于prompt范式训练模型

国内外主要LLM及特点介绍

Google T5 → GPT-3 → GLM130B → LLaMa → GPT-4 → Falcon → GPT-4v

名称	参数	特点	发布时间
GPT-2	15亿	英文底模，开源	2019年
Google T5	110亿	多任务微调,开源	2019年
GPT-3.5	1750亿	人工反馈微调	2022年
Meta OPT	1750亿	英文底模，开源	2022年
LLaMA	70亿～650亿	最受欢迎的开源模型之一	2023年
GPT-4	1.8万亿	史上最强大模型	2023年
Vicuna-13B	130亿	开源聊天机器人	2023年
Falcon	400亿	阿联酋先进技术研究委员会	2023年

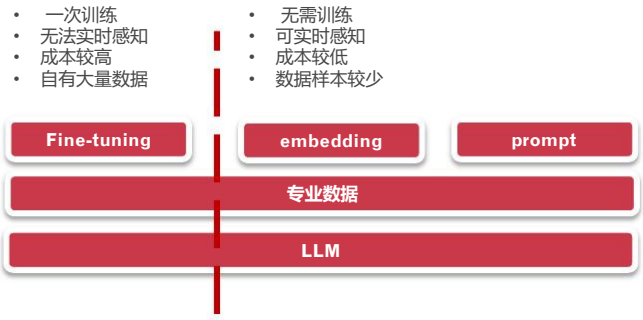
ChatGLM/6B/1T/可商用
ChatGLM2/6B/1T/可商用
LLaMA/7B/13B/33B/65B/1T/不可商用
LLaMA2/ 7B/13B/33B/65B /2T/可商用
BLOOM / 1B7/7B1/176B-MT /1.5T/可商用

名称	参数	特点	发布时间
百川智能	70亿	王小川，开源	2023年
文心一言	2600亿	中文语料85%	2023年
通义千问	70亿～700亿	总体相当GPT-3	2023年
ChatGLM6B	60亿	10B以下最强中文开源	2023年
腾讯混元	超千亿	腾讯出品多模态	2023年
MOSS	160亿	多插件，开源	2023年
Aquila	70亿~330亿	首个中文数据合规	2023年
PolyLM	130亿	对亚洲语种友好	2023年

Baichuan/7B/13B/1.2T/1.4T/可商用
Falcon/7B/40B/1.5T/可商用
Qwen/7B/7B-Chat/2.2T/可商用
Aquila/ 7B/7B-Chat /可商用

大模型的不足以及解决方案

- 强语言弱认知：大模型的幻觉
- 实时信息更新慢，新旧知识难以区分
- 逻辑推理不靠谱：本质上是预测下一个词语
- 无法为领域问题，提供专业靠谱的答案



MaaS

Model as a Serivce 模型即服务
通过“微调”技术，在LLM基础上灌入行业数据，实现行业模型

prompt engineering

提示词工程，通过上下文提示词设计引导LLM输出精确答案

AIGC产业拆解以及常见名词

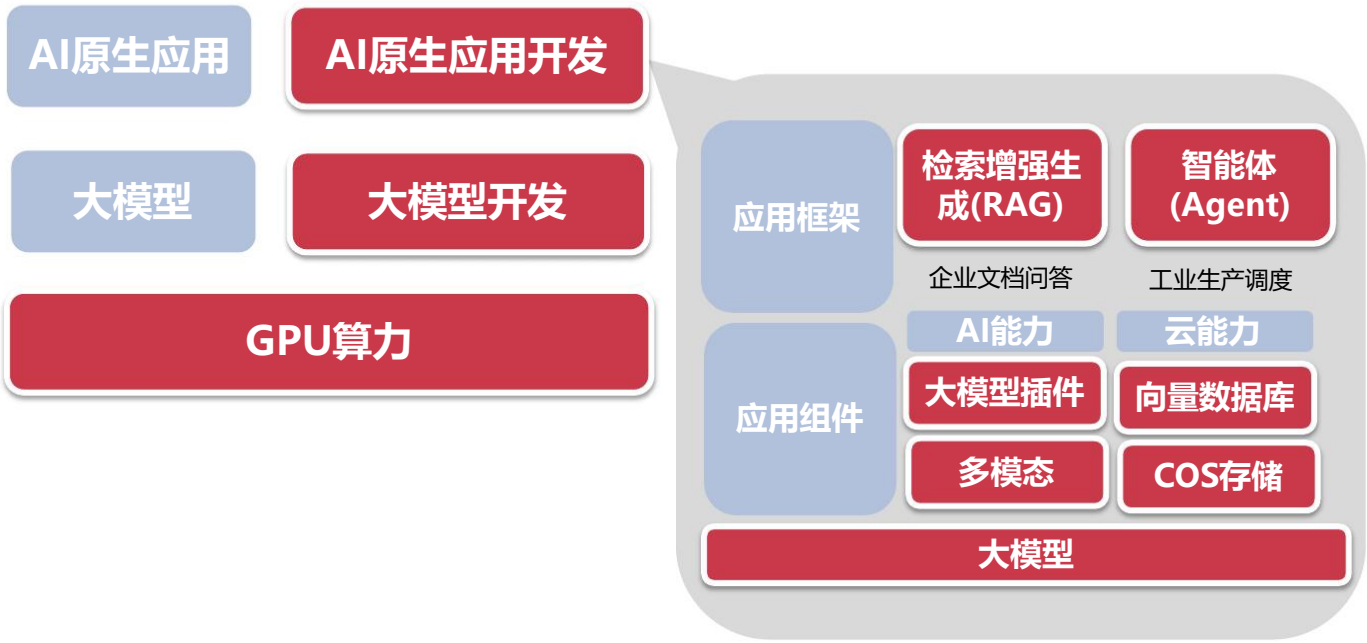
AIGC产业拆解以及常见名词

- 多模态
- token
- RPM
- 扩散模型
- CV
- LLM
- HuggingFace
- 行业模型
- Few-shot
- Azure
- CNN
- Fine-tunes
- stable diffusion
- chatGPT
- One-shot
- AI-Agents
- RNN
- RAG
- AIGC
- midjourney
- Zero-shot
- 模型训练
- 知识图谱
- DALL·E
- openAI
- RWKV
- Temperature
- RLHF
- 过拟合
- D-ID
- 具身智能
- 并行训练
- 分析式AI
- 情感识别
- 咒语
- Heygan
- AGI
- 知识幻觉
- 上下文
- 向量搜索
- 吟唱
- 炼丹
- 自监督学习
- PGC&UGC
- 生成对抗网络
- 向量数据库
- AI推理
- 炸炉
- 预训练
- TPM
- 元学习
- NLP
- CDN
- Copilot

应用级开发者该如何拥抱AIGC

模型层面竞争日趋激烈，而AI应用则市场广阔，可以说所有的应用都值得用AI重新做一遍

应用级开发者该如何拥抱AIGC



应用级开发者该如何拥抱AIGC

Preparation work

- 学习机器学习和深度学习基础知识
- 掌握AI开发工具和框架（python）
- 实践AI项目
- 深入了解特定领域的AI应用（场景知识）

Career opportunities

- AI软件工程师
- 数据科学家
- 自然语言处理工程师
- 计算机视觉工程师
- 机器学习工程师

- AI+金融
- AI+医疗
- AI+教育
- AI+制造业
- AI+零售
- AI+地产
- AI+家居
- AI+农林牧渔
- AI+文旅
- AI+汽车
- AI+能源
- AI+科研
- AI+公共管理
- AI+物流快递
- AI+对话私域
- AI+人才招聘

AI淘汰的不是人，而是不会使用AI的人

本课程虚拟项目案例

项目目标

开发一个人性化的具备专业知识的AI Agent

项目分解

- 自然语言交互（文本、语音）
- 具备鲜明个性，而非千人一面
- 具备知识学习能力
- 具备实时感知能力
- 具备记忆能力
- Api化可以嵌入IM中
- 可扩展

项目技术拆解

- API层开发 (python/fastApi)
- LLM+BOT开发(openai+**langchain**+telegram)
- 语音能力(barker/TTS/whisper)
- 记忆与学习(Vector database)

开发一个有手有耳有嘴有脑有性感的智能体，做专业的任务

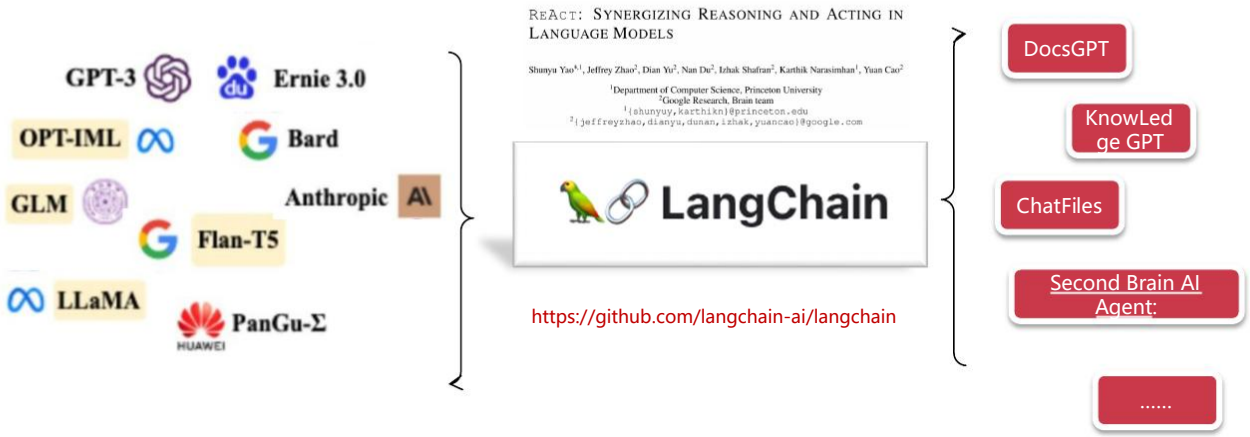
初识langchain：LLM大模型与AI应用的粘合剂

本章介绍

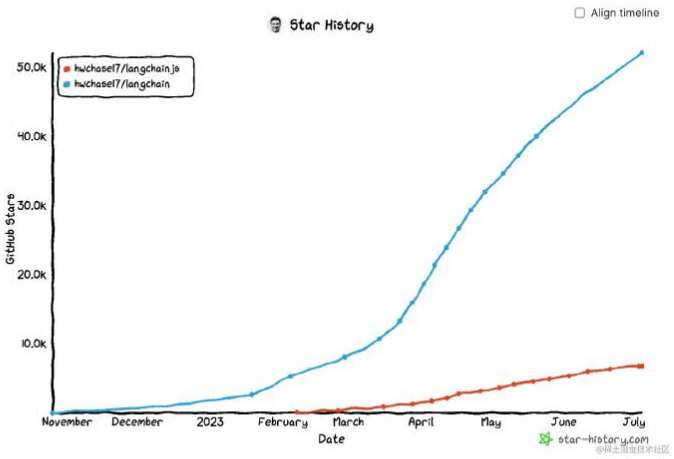
- langchain是什么以及发展过程
- langchain能做什么和能力一览
- langchain的优势与劣势分析
- langchain使用环境的搭建
- 第一个实例，了解langchain的基本模块

langchain是什么以及发展过程

- LangChain是一个开源框架，旨在简化使用大型语言模型构建端到端应用程序的过程，它也是ReAct（reason+act）论文的落地实现。

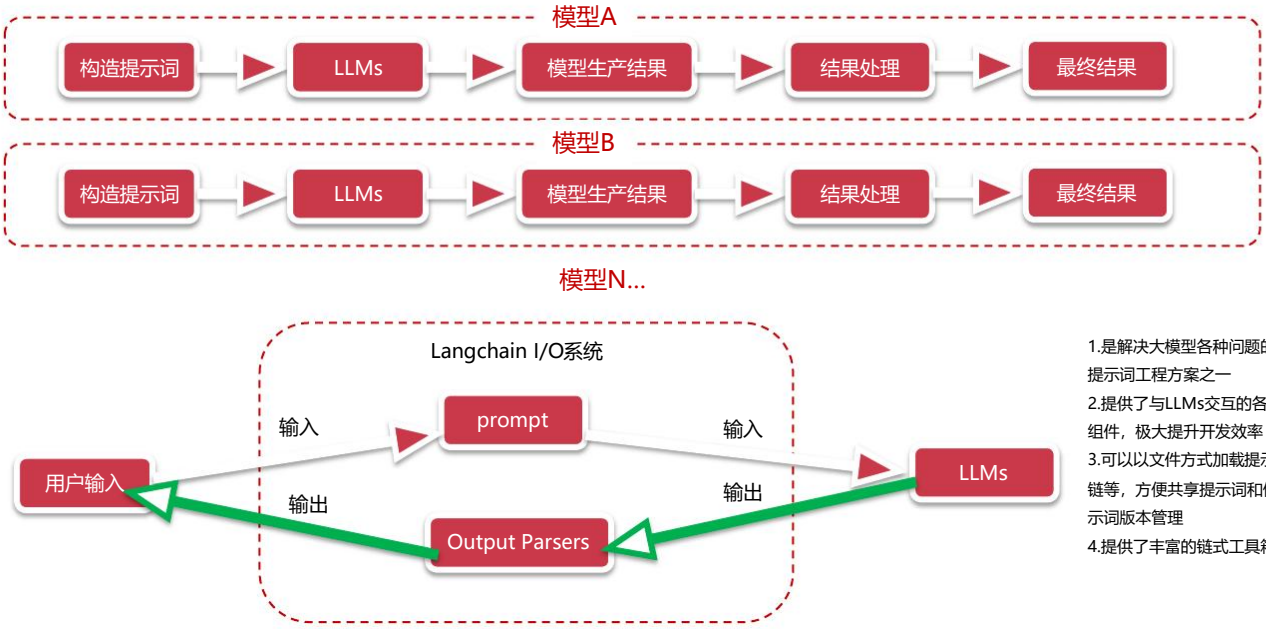


langchain是什么以及发展过程



- 2022年10月25日开源
- 54K+ star
- 种子轮一周1000万美金，A轮2500万美金
- 11个月里累计发布200多次，提交4000多次代码

Langchain能做什么和能力一览



1.是解决大模型各种问题的提示词工程方案之一
2.提供了与LLMs交互的各种组件，极大提升开发效率
3.可以以文件方式加载提示词、链等，方便共享提示词和做提示词版本管理
4.提供了丰富的链式工具箱

Langchain能做什么和能力一览

LLMs & Prompt

提供了目前市面上几乎所有 LLM 的通用接口，同时还提供了 **提示词** 的管理和优化能力，同时也提供了非常多的相关适用工具，以方便开发人员利用 LangChain 与 LLMs 进行交互。

Chains

LangChain 把 **提示词**、**大语言模型**、**结果解析**封装成 **Chain**，并提供标准的接口，以便允许不同的**Chain** 形成交互序列，为 AI 原生应用提供了端到端的 **Chain**

Retrieval Augemented Generation

检索增强生成式 是一种解决预训练语料数据无法及时更新而带来的回答内容陈旧的方式。LangChain 提供了支持 **检索增强生成式** 的 **Chain**，在使用时，这些 **Chain** 会首先与外部数据源进行交互以获得对应数据，然后再利用获得的数据与 **LLMs** 进行交互。典型的应用场景如：基于特定数据源的问答机器人。

Agent

对于一个任务，**代理**主要涉及让 **LLMs** 来对任务进行拆分、执行该行动、并观察执行结果，**代理** 会重复执行这个过程，直到该任务完成为止。LangChain 为 **代理** 提供了标准接口，可供选择的代理，以及一些端到端的**代理**的示例。

Memory

指的是 chain 或 agent 调用之间的状态持久化。LangChain 为 **内存** 提供了标准接口，并提供了一系列的 **内存** 实现。

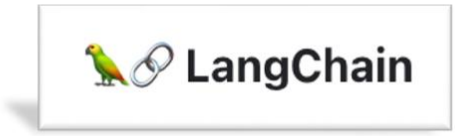
Evaluation

LangChain 还提供了非常多的评估能力以允许我们可以更方便的对 **LLMs** 进行评估。

Langchain的优势和劣势

优势

- 大语言模型调用能力，支持多平台多模型调用，为用户提供灵活选择
- 轻量级SDK(python、javascript)，可以将LLMs与传统编程语言集成在一起
- 多模态支持,提供多模态数据支持，如图像、音频等



劣势

- 学习曲线相对较高
- 文档相对不完善，官方文档不是很完善
- 缺乏大型工业化应用实践

环境搭建

- Python环境
- Jupyter Notebook的安装与使用
- LangChain安装

Python简介

- 高级的接近人类语言的编程语言，易于学习
- 动态语言
- 直译式语言，可以跳过编译逐行执行代码
- 广泛应用于web应用、软件、数据科学和机器学习
- AI方向的主流语言
- 活跃的python社区
- 数据巨大且丰富的库

Python环境

Python版本要求　>=3.8.1 推荐 3.10.12

https://www.python.org/downloads/

Active Python Releases

For more information visit the Python Developer's Guide.

Python version	Maintenance status	First released	End of support	Release schedule
3.13	prerelease	2024-10-01 (planned)	2029-10	PEP 719
3.12	bugfix	2023-10-02	2028-10	PEP 693
3.11	bugfix	2022-10-24	2027-10	PEP 664
3.10	security	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569

Jupyter Notebook简介

- 一个交互式开源笔记工具，可以用于编写、运行和共享代码、文本、图形等内容
- 多语言支持，python、R等,可在任意浏览器运行
- 可以交互式运行代码，用户可以执行某个片段而不用从头开始
- 非常适合数据分析、机器学习、可视化等任务
- 在AI方向使用非常广泛

Jupyter Notebook简介

文本框

代码框

运行结果

An example: visualizing data in the notebook ✨

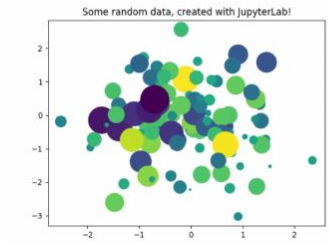
Below is an example of a code cell. We'll visualize some simple data using two popular packages in Python. We'll use NumPy to create some random data, and Matplotlib to visualize it.

Note how the code and the results of running the code are bundled together.

[1]:
```
from matplotlib import pyplot as plt
import numpy as np

# Generate 100 random data points along 3 dimensions
x, y, scale = np.random.randn(3, 100)
fig, ax = plt.subplots()

# Map each onto a scatterplot we'll create with Matplotlib
ax.scatter(x=x, y=y, c=scale, s=np.abs(scale)*500)
ax.set(title="Some random data, created with JupyterLab!")
plt.show()
```



Jupyter Notebook安装

官网 https://jupyter.org/install

命令安装 `$ pip install jupyterlab`

启动 `$ jupyter-lab`

也可以使用VS code的jupyter插件启动

Langchain的安装

官网 https://python.langchain.com

命令安装 `$ pip install langchain`

`$ conda install langchain -c conda-forge`

也可以使用VS code的jupyter插件启动

Langchain的第一个实例

创建一个LLM

- 自有算力平台+开源大模型（需要有庞大的GPU资源）
- 第三方大模型API（openai/百度文心/阿里通义千问...）
- 让LLM给孩子起具有中国特色的名字

自定义提示词模版

- 将提问的上下文模版化
- 支持参数传入
- 让LLM给孩子起具有美国特色的名字

输出解释器

- 将LLM输出的结果各种格式化
- 支持类似json等结构化数据输出
- 让LLM给孩子起4个有中国特色的名字，并以数组格式输出而不是文本

第一个实例

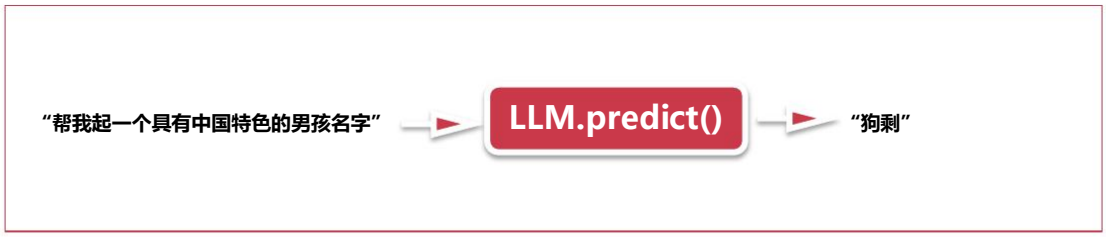
- 让LLM以人机对话的形式输出4个名字
- 名字和性别可以根据用户输出来相应输出
- 输出格式定义为数组

Langchain的第一个实例

创建一个LLM

在langchain中最基本的功能就是根据文本提示来生成新的文本

使用方法: **predict**

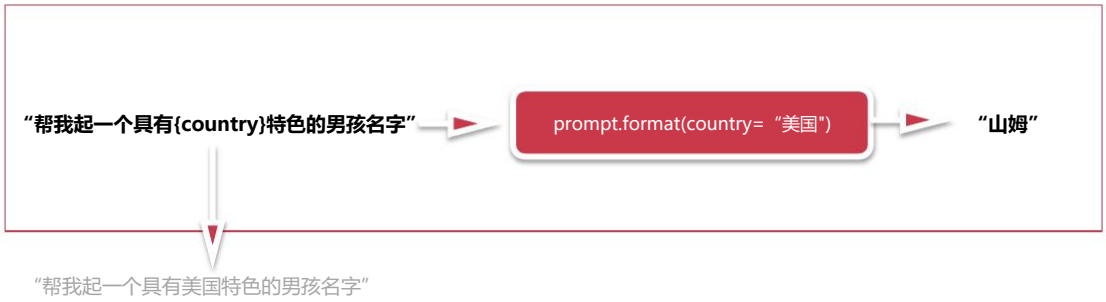


生成结果根据你调用的模型不同而会产生非常不同的结果差距，并且你的模型的tempurature参数也会直接最终结果

Langchain的第一个实例

自定义提示词模版

将提示词模版化后会产生很多灵活多变的应用，尤其当它支持参数定义时
使用方法: **langchain.prompts**



Langchain的第一个实例

输出解释器

与chatGPT只能输出文本不同，langchain允许用户自定义输出解释器，将生成文本转化为序列数据
使用方法: **langchain.schema**