最近读了一些LLM-based agent的工作和一些相关理论，做一个学习笔记。一直也想做一种个人级别的agent挂在后台解闷，强调agent的生动性和生活气息，同时辅助一些通用 or 细分工作，因而争取做一些技术实践。最近意难平《边缘行者》里的Lucy，AI agent的能力似乎也到了一种边缘，所以给想象中模糊的目标agent取名为LucyAgent。Lucy并非一定要是一个具体的人，而是指代这种生动的个体形象。

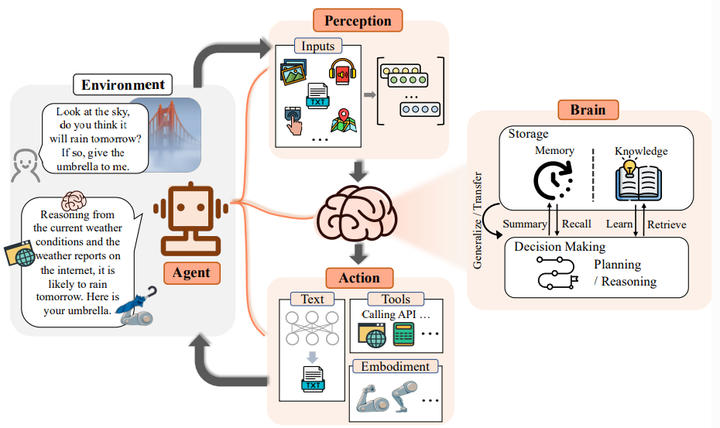
第一章介绍LLM-based agent的总体框架，第二章介绍一些agent构件的细节框架。

**1 理论：宏观框架**

最近涌现了一大批LLM-based agent的理论框架，如人大和复旦的两篇相关survey。理论框架的指导是很重要的，因此在本章做一些介绍，主要是基于米哈游和复旦的这篇综述。

**1.1 Agent的整体架构**

arxiv.org/pdf/2309.07864v1.pdf​arxiv.org/pdf/2309.07864v1.pdf



Conceptual framework of LLM-based agent with three components: brain, perception andaction.

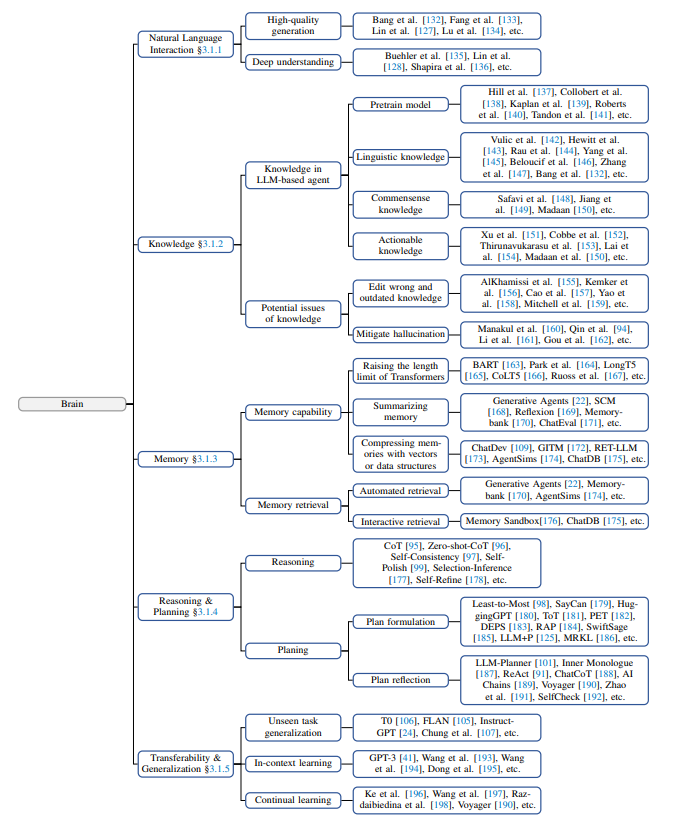
米哈游和复旦的这篇综述给定了一个LLM-based agent的概念框架图，较为清晰，细细读图其实就能理解这个框架，加上一些认识和细化后，大概意思是说：

* Agent首先是“运动”在”环境“（Enviroment）中的，环境是一种状态空间，而运动是一个抽象概念，可以将其理解为Agent的根本属性和存在形式（疑似有点马哲）。可以说agent的一切都是和这个环境相关联的，包括感知（输入），大脑（各种类型的内部处理），行动（输出）。**Agent也是环境的一部分，因而agen的行动改变环境时也可以改变agent自身。**
* Agent的“运动”可以被大致分类为感知，内部处理，行动三个部分。分别对应图中**Perception，Brain，Action**三个模块。Brain模块承担记忆、思考和决策等内在的任务；感知模块负责感知和处理来自外部环境的多模态信息；行动模块负责使用工具执行任务并影响周围环境。三个模块描述了一个agent自身状态的全部组成。其机理分别于类似于人类用眼睛，耳朵等获得信息，大脑用于处理信息和驱动四肢，四肢用于改变环境和改变人类自身。
* **Agent的内部信息通路是Perception->Brain->Action，而信息通路的设计本身也应该是Agent的一部分。**类似于人类的神经信号传输，决定了一些Agent可表征的信息、信息损失和处理效率问题。宏观来看，agent应该通过一轮或多轮的”输入->处理->输出“来完成一个任务（类似Re-Act框架），是否结束任务应由外界中断和反馈信息来确定。

为什么是多轮呢？作为人类，很多时候我们做了一件事才能看到结果，通过对上一件事的结果再次”输入->处理“，我们会推理出所做的上一件事的效果如何，而正是通过这种评价，我们决定要不要继续做这件事和下一步怎么进行。这其实是多轮的”输入->处理->输出“。从这个角度看，**如果一个agent评估了自身完成任务的情况，并且能够对没有完成好的情况制定进一步方案，或是向人类寻求帮助，是感知环境和接受反馈的智能表现。**这种方法在LLM内部已经有所体现：如向LLM提问时给出不完全的信息，LLM可能会向你索要进一步的信息。但这不是一种可控的机制性能力，它不是agent的结构所带来的，而是LLM内部的一种不太可解释的能力。

显然，每个模块都可以做进一步的细分，每个模块也有独立的运动机制，当然也有跨模块的运动机制。

**1.2 Brain模块的构件框架**



Typology of the brain module.

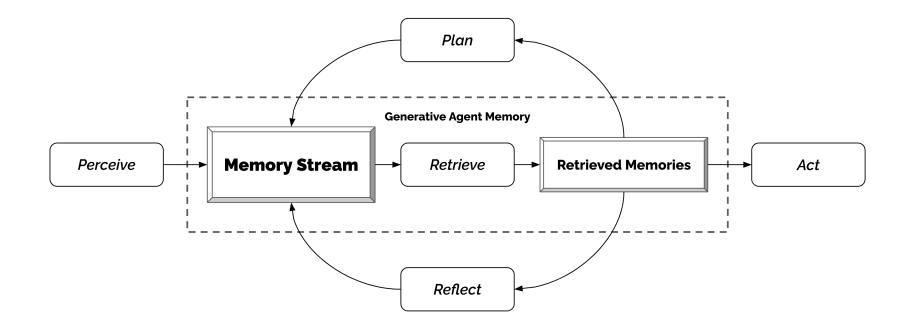
Brain理所当然是最核心的模块。Brain的运动机制大概为：**从内部信息通路接受感知模块传来的输入->将输入作为参数，从模块内的知识和记忆构件检索（似乎可统称为 ”内存“）->综合输入和检索出的数据信息进行推理和计划->从内部信息通路输出行动序列到行动模块，并且进行模块内部知识和记忆的更新。**

Brain模块有5个关键构件：自然语言交互，知识，记忆，推理和规划，可迁移性和通用性。

* **自然语言交互：**提到了多轮交互对话，高质量的自然语言生成，意图和含义理解三个部分。大概就是LLM的基础能力部分。
* **知识：**提到了三种类型的知识，即语言知识，常识知识，专业领域知识，以及潜在问题。潜在问题主要是过时、错误和幻觉。
* **记忆：**提到了agent需要特定的记忆机制来处理连续任务，记忆可以分为短期记忆和长期记忆，短期记忆即输入的上下文，长期记忆则有多种实现方式。长期记忆是一种”外骨骼“式的增强方法，主要需要获得记忆的方法，存储记忆的方法和检索记忆的方法三个内容。《Generative Agents》提供了一种可行的记忆流框架。记忆模块是提高LLM能力的重要方式之一，具体可以提高Transformer架构处理长序列的能力（短期），改善总结记忆的方法，检索记忆的方法等（长期）。

知识和记忆好像没有明显的界限，这和人类类似。我们似乎可以认为后天学习的知识就是一种记忆。因而，本文对“知识”的界定在于agent被初始化并投放到环境之前，agent所固有的知识。

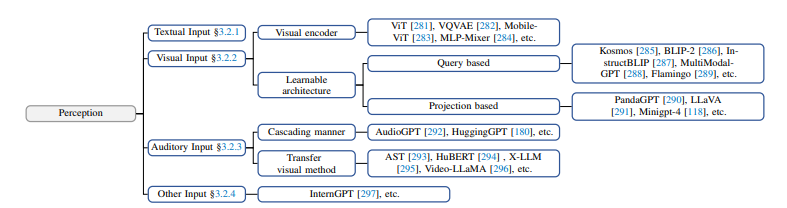
* **推理和规划：**推理更多是LLM内部的能力，但是可以通过外部方法提高，如思维链（CoT）技术及其变种。规划的核心是推理能力，但是也需要人类”要求“来激发。人类要求LLM进行规划后，LLM可以将复杂任务分解为子任务逐步解决。规划又可细分为计划制定和计划反思，制定的方法如《Generative Agents》的plan模块，反思则可以从外部评价反馈获得。



generative agent architecture.

* **可迁移性和通用性：**提到了任务泛化，情景学习，持续学习三个部分。任务泛化和情景学习大致是涉及到zero-shot learning，few-shot learning两个概念。持续学习的核心挑战是灾难性遗忘。

**1.3 Perception模块的构件框架**

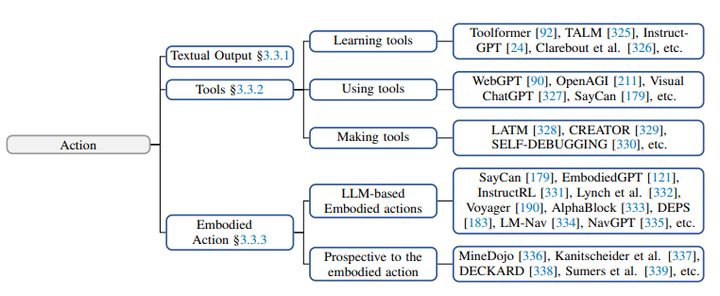


Typology of the perception module.

Perception模块的4个关键构件是:文本输入，视觉输入，听觉输入和其他输入。核心目的是将agent的感知空间从扩展到包括文字、听觉和视觉模式在内的多模态领域。如当前的GPT-4V，已经能够理解一些视觉提示（Visual Referring Prompting），并且能够排列一些打乱的序列帧（Temporal ordering）。

* **文本输入：**基础能力。重点在于理解人类的隐含含义和未知任务的文本指示。
* **视觉输入：**一种方式是为图像输入生成相应的文本描述然后链接，但是会丢失信息。一种方式是将图像编码器和 LLM 结合起来，并将图像编码转换为 LLM 可以理解的嵌入。agent用于感知图像的方法可能适用于视频领域。
* **听觉输入：**将 LLM 用作控制中心，以级联方式调用现有工具集或模型库来感知音频信息。
* **其他输入：**大致是和真实物理世界相关的一些传感器输入，同样要做预处理。

**1.4 Action模块的构件框架**



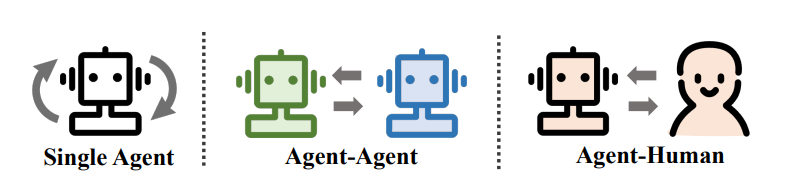
Typology of the action module.

Action模块的3个关键构件是:文本输出，工具，具身行动（embodied actions）。行动模块接收大脑模块发送的行动序列，并执行与环境互动的行动。

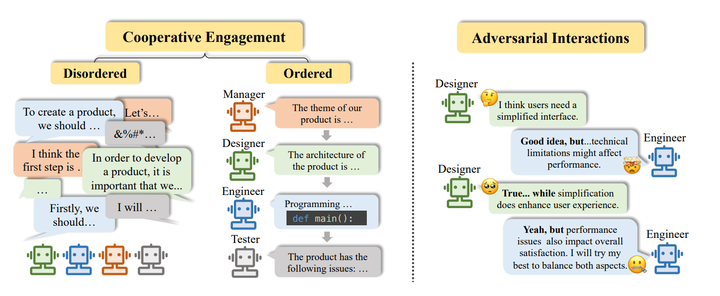
* **文本输出：**基础能力。包含内部指令的log和对外部环境的输出。
* **工具：**主要是三个方面，弥补LLM缺陷，增强agent能力，拓展LLM-based agent的行动空间。LLM有幻觉的问题，并且对于特定领域的能力不够强，所以需要一些专有工具，如科学计算，地图导航等。agent需要理解工具，使用工具，甚至制作工具。agent需要实现工具和知识库的内在协作机制。
* **具身行动：**高级话题，使agent能够以近似人类行为的方式与世界互动并理解世界。

**1.5 Agent与环境**

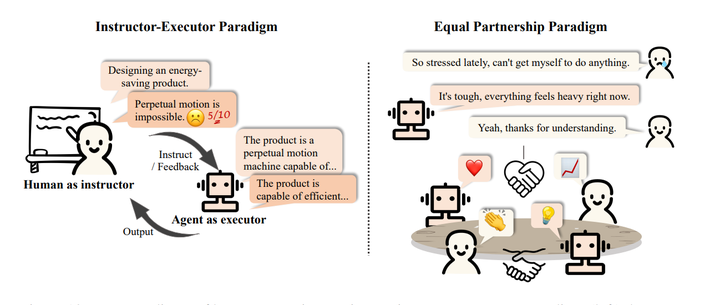
大致是一些框架，包括agent的交互形式，multiple agents的交互形式，agent human的交互形式，以及agent society的框架。都比较直观。



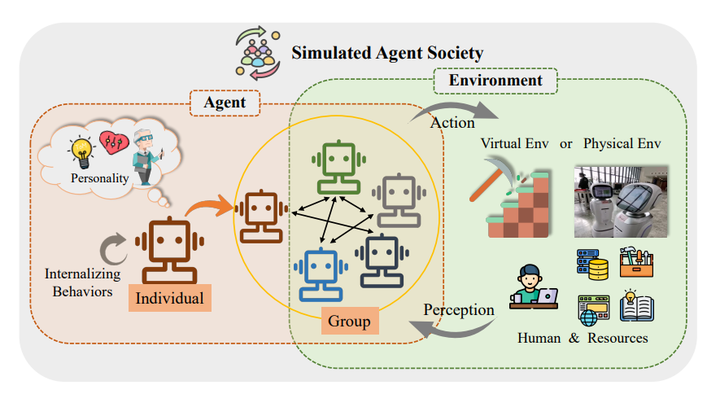
Scenarios of LLM-based agent applications.



Interaction scenarios for multiple LLM-based agents.



Two paradigms of human-agent interaction.



Overview of Simulated Agent Society.

**2 骨骼：构件的细节技术**

按照第1章的框架拆分agent有一些好处：可以像拆卸一个单元一样更改agent的某个构件。在设想里，可以像《Cyberpunk 2077》的义体系统一样来组成一个agent的构件系统，在组成模块时，应当保持构件的独立性和通信能力，同时保留模块可扩展性（当然，有些端到端的模型不可不用，这种模型的通用能力有很大优势，不应该有极端的“分解”主义），这也符合程序设计中“低耦合高内聚”的概念。当一个构件有更好的实现的时候，可以更替掉原来的构件，也方便给模块加上一些新的构件。

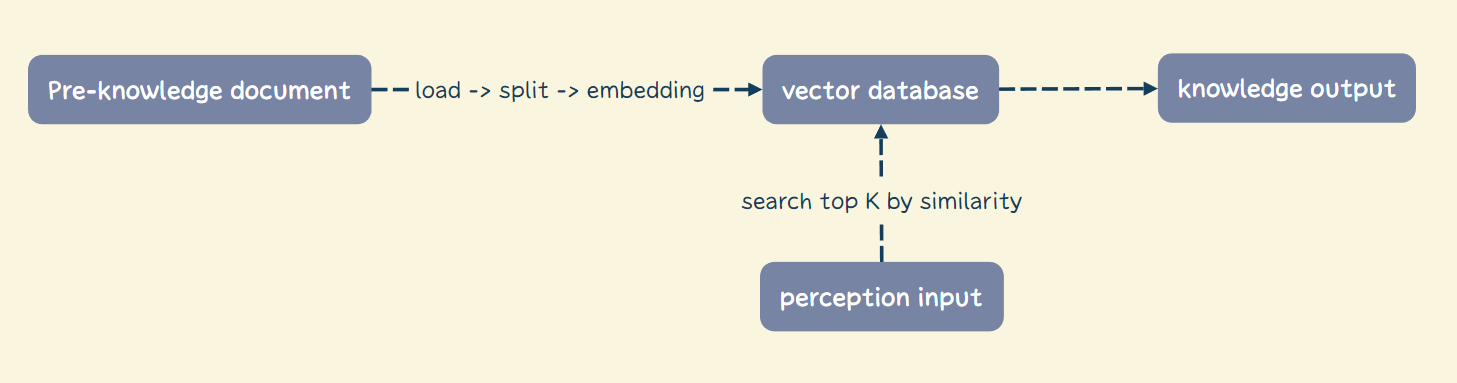
**2.1 可用的Brain构件**

**2.1.1 核心：生成式大语言模型**

核心技术点：深度学习基础，Transfomer，Bert，大模型预训练，模型微调，推理部署等。

基础的模型如Transfomer，Bert。LLM即ChatGPT，ChatGLM等，可见一些LLM的概述。对于不同的任务可以选用不同的语言模型，比如角色扮演用Chat模型，Brain内部逻辑用Instrcut模型。后续会跟进对基础模型和ChatGLM（x）-6B的学习和实践，因为开源可商用 & 轻量，可本地部署微调，吸引力很大。

**2.1.2 外置知识库**



已有的知识库构建方案：核心流程

核心技术点：数据预（后）处理，文本切分，向量数据库，Embedding方法，相似性匹配等。

实践来看，将知识存到向量数据库中然后按需检索是一种外置知识库的方法，已经有不少的这样的应用，如LangChain + ChatGLM定制私有知识库问答。将这种知识库也视为agent模块内部的一部分，会有下述优点：

* 拓宽agent知识范围，解决私有垂直领域问题。打开想象力，垂直领域事实上是一种广义的角色扮演任务。如果是要做角色扮演任务，将世界观，人设背景，人物基本社交关系，人设中该角色应该知道的知识等都放到静态知识库中，伴随agnet进行初始化，便能够进行更好的，没有上下文限制的角色扮演。知识库可以采用现有的一些解决方案。

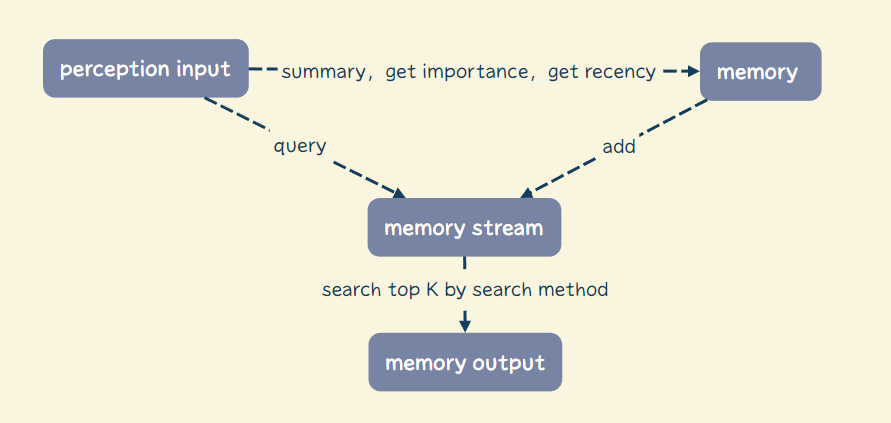
不过值得注意的是，现在的知识库都是表征自然语言信息的，是否可以拓宽到图像等？先不考虑直接嵌入，可以考虑给图像生成一些自然语言描述，然后把描述作为chunk嵌入进向量数据库，再将该chunk和图像建立一个映射表即可，这样可以以检索普通text chunk相同的方式检索出图像。 本文经常强调角色扮演，是因为agent要扮演一种特定的角色。本文认为对于agent来讲，拥有一致的自我认识和外在表现是极重要的，即使驱动agent的核心（LLM）没有这种能力。这种能力也体现为情感，性格等。

* 能够较为直观地实现（半）自主学习。一种可能的逻辑是，人类输入指令让agent进行学习某领域的知识，然后经过一些LLM自我询问，网络检索，人类审核反馈等，将学习到的知识嵌入到向量数据库即可。当然，也可以设置一种agent的生物钟，让agent自主在某个时间点进行学习，可以是人类预设也可以是agent自发的。agent的生物钟似乎是一个还没被提出，但值得讨论的话题。

agent能否自主地感知环境，自主地和环境（包括人类）交互，有自主的任务日程，是很重要的一个智能表现。

* 缓解了灾难性遗忘的问题：没有对LLM进行微调，仅仅是外置的方法。不过这也导致了，知识不属于LLM，而是属于整体的agent，对于一些跨度长的知识检索（语义大于一个chunk的语义），会导致查不全的问题。

**2.1.3 长期记忆流**



长期记忆的构建方案：核心流程

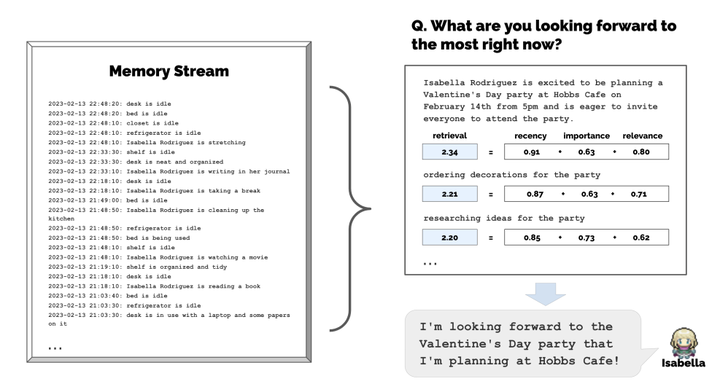
核心技术点：记忆结构设计，检索函数设计，记忆存储设计等。

长期记忆使得agent能够记录以往的环境信息，并根据记录的环境信息处理新的状况。LLM是无状态的，但长期记忆的存在使得agent是有状态的。《Generative Agents》提出了一种记忆流框架，实现了记忆生成，检索和反思，下面的研究问题也以该文为例子。长期记忆的主要研究问题包括：

* 记忆单元如何组织：描述一个记忆对象是如何组织的。例如，一个对象可以包含事件的[自然语言描述，自然语言描述的embedding，记忆生成的时间点，最近访问的时间点，事件的重要性（由LLM生成）]组成。
* 记忆系统如何组织：描述记忆系统是怎么组成的，即如何组织上述的记忆对象。例如，可以简单的把记忆对象组织成列表。但agent较复杂时，显然会有更高效的方式。

人类似乎会隐式地将一些记忆总结为一个主题，在接收到相关主题的信息时，能够飞速地做出反应。是否可以认为，在agent学习到关于一种主题的记忆多次后，应该会像人类的条件反射一样，接收到一些外部刺激后（perception模块），不经过思考（brain模块）就直接进行反应（action模块）？即使这种反应是不利的，和人类无法控制的一些条件反射一样，即“agent不一定要深度思考”。

* 记忆生成：即如何把环境信息转化为agent可组织的记忆。注意到，记忆可以是从perception的环境输入而来，也可以从agent内部模块而来（如更早期的记忆，关于工具的用法等）。例如，agent感知环境某个物件的情况作为记忆，或者是对用户文本输入和agent自身文本输出做一些text summary得到。agent也可以对自身的记忆做一些反思，从而生成更高语义，更概括性的信息。
* 记忆检索：描述如何检索并使用记忆。例如，可用利用[当前输入和记忆对象的相似度，记忆事件的重要性，记忆对象的时效性]三个参数进行检索，取出最高分的几个。

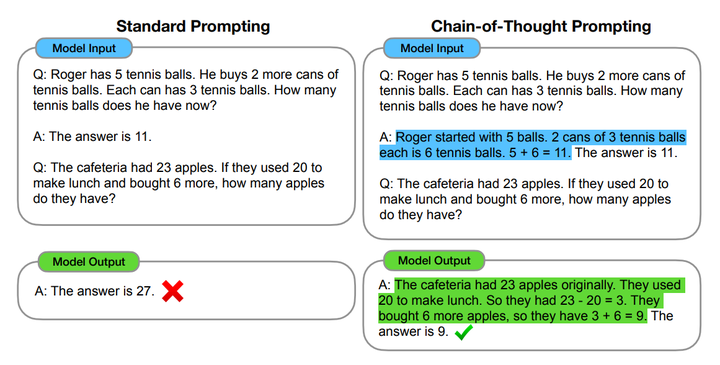


《Generative Agents》设计的记忆流框架

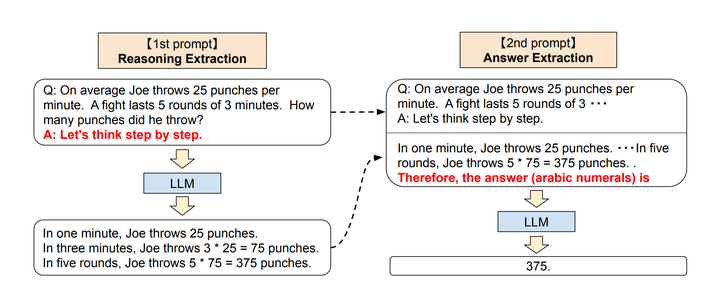
**2.1.4 推理与规划**

核心技术点：CoT，prompt engineering，推理框架设计，agent生物钟等。

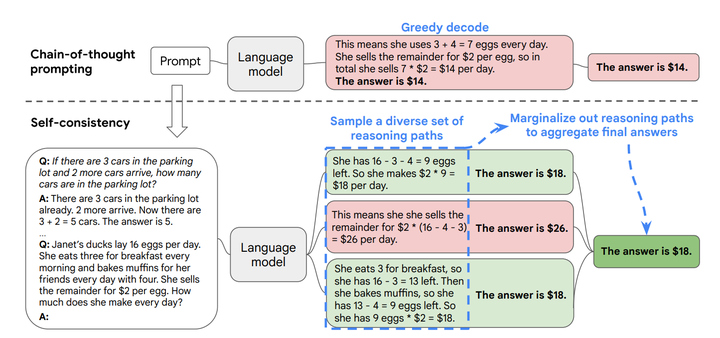
* CoT及其变种技术的目的都是激发LLM的推理能力。基础的Few-shot-CoT是在给出解决任务的例子时，附带上推理和逐步解决问题的方法，让LLM进行下上文学习（模仿）。Zero-shot-CoT是通过两次LLM的级联，在第一次进行分析，第二次提取任务输出的方法。Self-consistency的思路是通过多轮的LLM问答，再取多数投票选出最终答案。其他变种大致也是引入一些树，图等结构。



Few-shot-CoT



Zero-shot-CoT



Self-consistency

* 提问的技巧：高质量prompt的撰写。可以参考这篇文章。
* agent生物钟：agent应该有一套自主的行动逻辑，能够感受时间流逝，应对日常的情况，这很像人类的生物钟。有一些工作的做法是提供让agent做计划的模板，或者告诉agent的长期目标，让agent根据LLM的理解自己做计划。这种生物钟是比任务规划更高级的层面，强调agent的自主性。

我曾也有过怀疑prompt engineering的可靠性，总感觉力气没用用到正确的位置。但是再想想，作为人类，无论是向老师，专家教授，搜索引擎提问，还是向朋友，家人，伴侣提问，一些任务是为了获取知识，一些任务是为了获取他人的信息或感受等，其实都极其需要提问的技巧。我们时常看一些采访，访谈，主持，也能够评鉴出：“这人根本不会提问题，这人怎么老说错话。”所以，如果承认agent不会比人类的理解能力强，那么对agent提问的方式只会比人类重要。想通这一点，承认prompt engineering研究的重要性就是理所当然，prompt和模型调参都是同样有价值的工作。

**2.1.5 一致的角色扮演**

核心技术点：长期记忆设计，prompt要素构成，prompt injection等。

agent应该能够识别自身的身份，并保持一贯性。agent应该能够处理其所在环境（沙盒 or 物理）的信息，并以自身的身份做出反应和自然语言描述。agent应当有一贯的社交关系，对某个agent or human的看法等等。似乎可以用好的角色扮演要素和记忆流方案去解决。暂时挂上之前写的一篇学习记录文章。

**2.2 可用的Perception构件**

**2.2.1 文档输入**

例子：类ChatPDF的应用。

流程：load->split->embedding->search。和外置知识库的方案类似，只是一个常驻外存，一个运行时输入，可以理解为计算机系统的外存（硬盘）和内存。LangChain封装了这些接口。

**2.2.2 从沙盒环境感知**

agent内部有一些和沙盒有关的属性，例如生命值，可以转为自然语言描述->查询agent的外部属性接口（位置等）->转化为自然语言描述->agent内部处理。

**2.2.3 从真实环境中感知**

获取时间，位置，天气等物理概念的方法。大致还是自然语言描述进去，再用brain模块做处理。

**2.3 可用的Action构件**

**2.3.1 agent的外在形式**

核心技术点：客户端开发，live 2d，实时TTS，数字人等。

agent需要以某种形式存在，被人类感知。可能的形式有：对话机器人（文本框+前端界面），虚拟人（live2d，unity，ue等），执行特定任务的物理实体（如商汤围棋机器人），具身智能，仿人设计（高级话题）。

对于本文的目标（挂后台解闷，处理通用 or 细分问题），在这里给出一种简单的类桌宠应用方案。需要的一些构件为：

* 一个agent形象概念
* 一组形象对应的动画（待机，不同情绪等）
* 一个独特的文本转语音（TTS）模型
* 一个有限状态机控制对话，不同动画的触发
* 一个桌面客户端用于进行human-agent交互
* 一个角色特色的微调对话模型（可选，专用于对话生成，内部逻辑不用微调）
* 一个live2d模型+脸部，口型驱动方案等（可选，生动性问题）

**2.3.2 网络检索工具**

网络检索工具可以让agent接触的环境扩展到新的维度。难点：如何让agent真正学会搜索，学会判断搜索的结果好坏，如何设计一套好的行动机制，如何高效地执行搜素-反馈-再搜索流程。

**2.3.3 日志**

记录原始输入，环境信息，内部模块信息的构件。“人都要检查，agent当然也要。”

**2.3.4 细分专有工具**

科学计算器，地图，数据分析工具等。大致类似ChatGPT的插件形式实现？

**2.3.5 沙盒环境中的行为驱动**

关键点大致是把接口结构化，固定化，保证内部逻辑格式正确

**2.4 关于LangChain**

赞同LangChain有些过度封装，简单问题复杂化的观点，很多功能实际上可以自己实现。

**3 实例**

构建一个完美的agent在现在看来是不可能的，所以能实践的方法将针对于具体的对agent的要求。LucyAgent在理想中要实现可成长的知识库，长期记忆，角色扮演（包含个性化的对话生成和动画驱动），环境感知等功能。当然，能加上（半）自主学习，（模拟的）自我认知，物理世界接口等更好。

*\*后续跟进开发过程中agent用得到的所有相关技术点，包括基础模型和新的想法，功能。同时也会边实践边学习后，补齐前面一些简单带过的内容。*

*\*后续跟进开发过程和实现记录，拟写一篇《LucyAgent（二）：实现》。拟将一些agent的构件（如各类api调用，检索增强生成接口，物理世界感知接口，网络检索接口）封装成函数库 ，agent作为一个类实现。然后用角色相关知识去实例化一个LucyAgent。*