Introduction to AI & GenAI

Course Code: 4E01

Author: Lloyd Sun

Date: Aug.22.2024

Version: V1.0

1 Overview of AI & GenAI

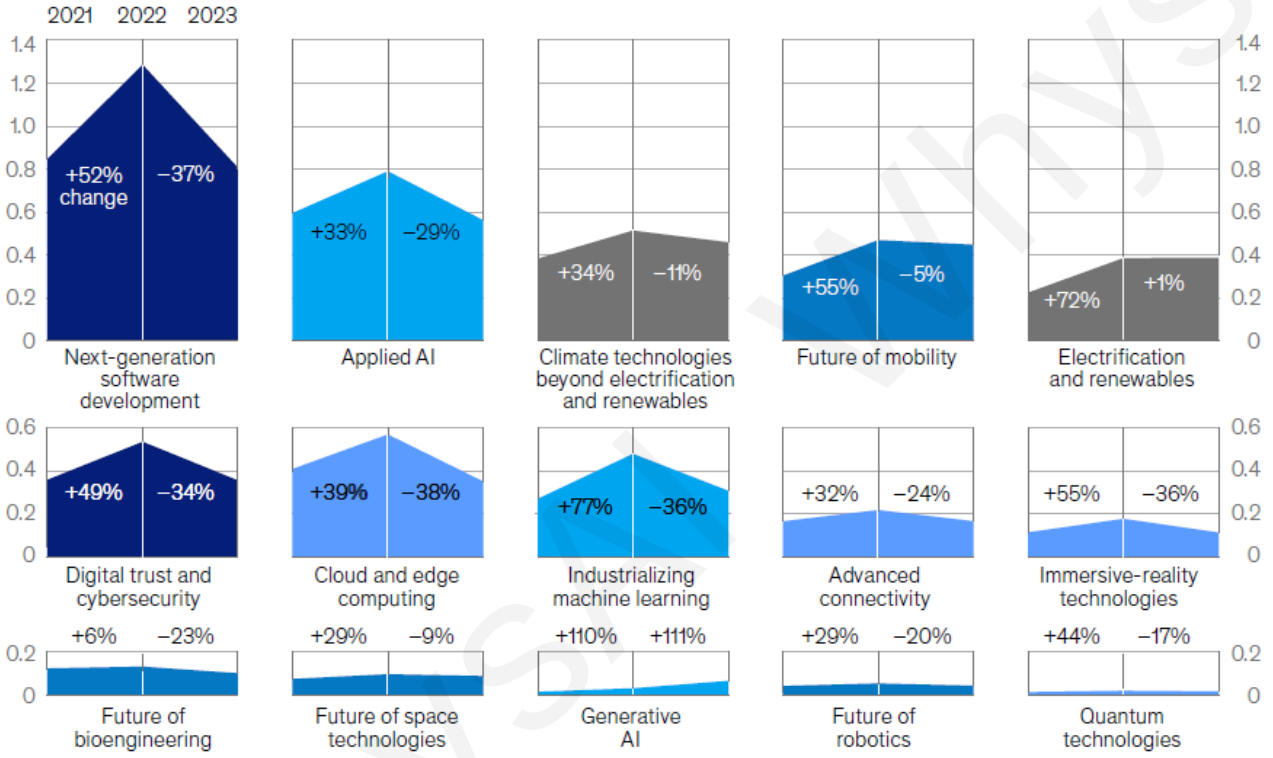
1.1 High-level overview

AI是近几年最被滥用的名词之一。回溯历史，1956年达特茅斯会议让关于“智能”的研究领域有了Artificial Intelligence这个被广泛接受的名称。之后，三大主义相继登上历史舞台，直到AI拥抱统计学、神经网络，才逐渐形成了可以被称作现代人工智能的学科。从当前的视角看AI历史，可以暂且定义以2012年AlexNet为代表的Supervised Learning（有监督学习）为Modern AI 1.0（当然，若追溯其学术渊源，可以追溯到1986年的BP算法、1989年的第一个MLP端到端案例Mnist、2006年的DBN等等）；以2022年ChatGPT为代表的GenAI（生成式人工智能）为Modern AI 2.0（源头是2017年的Transformer，以及更早的注意力机制）。至于Modern AI 3.0，有可能是Cyber Space（网络/虚拟空间）中的AGI（通用人工智能），也可能是融合了Cyber与Physical Space的具身智能（Embodied Intelligence，EI），当然这两者有可能有依赖关系，即AGI是EI的前置条件，或者相反。

现代人工智能有两个基本观点：1）“智能”是从行为主义的角度去定义的，即Rational Behavior（理性行为），具体说是随着数据、训练时长的增加，模型能够产生更好的行为。在这个观点之下，有个著名的比喻，“飞机不是在模仿鸟的飞行方式，而是在拥抱了空气动力学之后才成为了飞机”；2）由强化学习之父Richard Sutton在2019年著名的文章The Bitter lesson（苦涩的教训）中提到，“the only thing that matters in the long run is the leveraging of computation（从长期视角来看，唯一重要的是利用计算的力量）”。

本系列课程的设计目标是帮助（软件）工程师了解、掌握现代人工智能技术。要讨论的一个基本问题是，为什么软件工程师应该学习AI？我们给出三条理由：1）市场需求太大了；2）写代码太重要了；3）学习AI应用技术太简单了。

首先来看麦肯锡在2024年6月发布的关于15大前沿技术领域的相关报告。通过里面关于劳动力市场（New Job Postings）的统计分析，能够看出与AI相关的职位持续保持高需求量，尤其是GenAI领域，在2022、2023年保持了110%的年增速。



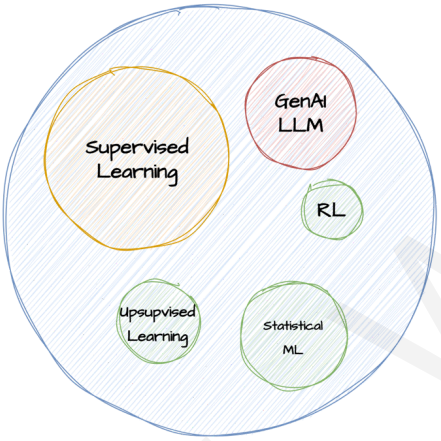
其次，正如著名的AI学者Andrew Ng（吴恩达）所说，“Coding AI is the new literacy：Code is the deepest form of human-to-machine communication. As machines become more central to daily life, that communication becomes ever more important（代码是新的读写能力：代码是人机交流的最深入形式。随着机器在日常生活中扮演越来越重要的角色，这种人机交流能力会越来越重要）”。

最后，软件工程师/程序员入门AI太简单了。著名AI学者李沐曾说：“大模型时代怎么学习AI？动手学”。大量的学习资料、论文、开源项目使现在学习AI的难度跟10年、5年前相比简单很多，但仍然面临着知识范围太广、如何将海量的学习资源系统的整合与整理等等问题。在学习方法论方面，我们认为Andrej Karpathy总结的三条建议最具实用性，也是本系列的设计理念：

1. Iteratively take on concrete projects and accomplish them depth wise, learning on demand(i.e., do not learn bottom up breadth wise): 迭代地、深度地完成扎实的项目，在需求与问题的指引下学习（也就是不要广泛底、自下而上地学习）
2. Teach/Summarize everything you learn in your own words: (用自己的语言总结/教授所学的知识)
3. Only compare yourself to younger you, never to others.（只跟自己比较，永远不要跟他人比较）

1.2 What is AI

关于如何看待AI技术，一个有帮助的比喻是“AI is the new Electricity”，即如同电力一样，AI是一项具有通用性、具有赋能万物潜力的技术。从技术视角看，AI可以看作是工具的集合，其中包括有监督学习、无监督学习、强化学习、生成式人工智能，可以应用在视觉、自然语言、自动驾驶、机器人等等领域。

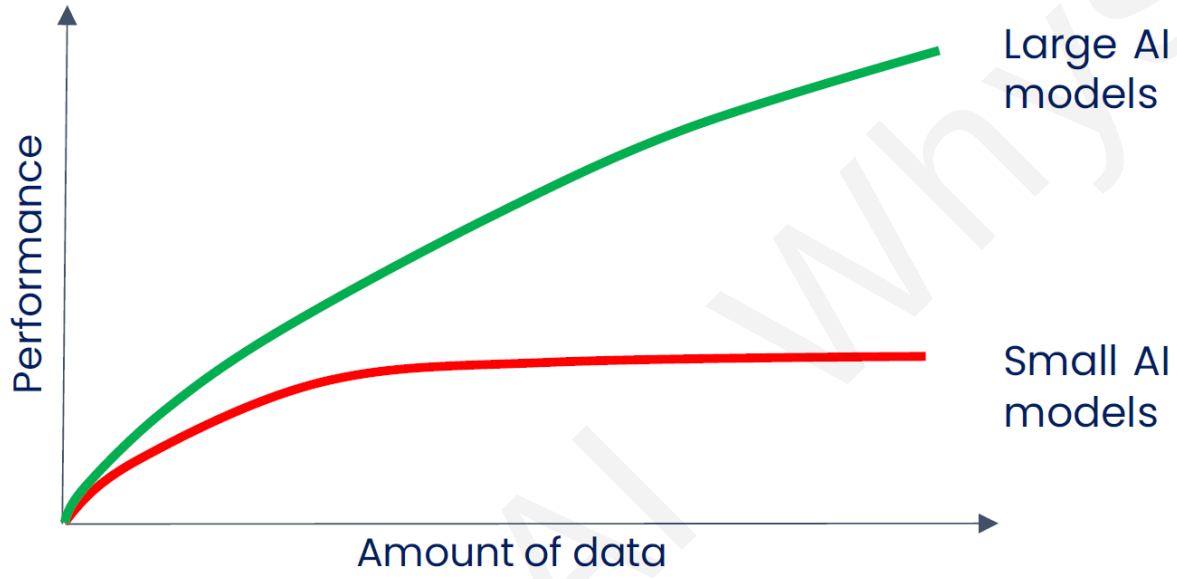


在全部技术领域当中，最成熟与重要的技术是有监督学习。简单说，监督学习就是输入X到输出Y之间的映射关系，与函数f(x)的定义完全一致。整体上理解任何模型的思考方法都是先把模型当作黑箱，确定输入X、输出Y分别是什么，这跟计算机中的抽象/函数是完全一致的，即只需要了解函数的入参、返回值是什么，就可以调用函数了。

至于模型（也就是f（x））内部的结构、流程、原理是什么，一般来说是算法工程师的工作。如果以深度学习/神经网络技术为例的话，模型内部一般是一个神经网络，可以是简单的多层感知机MLP（也就是原始的多层神经网络）、卷积神经网络CNN（一般用来处理图像数据）、循环神经网络RNN（一般用来处理序列数据），或者LSTM、transformer等等。其内部机制简单来说，训练过程是从输入开始，逐层计算输出（每一层都有线性和非线性），进而跟正确答案/标签比较，计算得出损失函数，这就是神经网络的前馈过程；然后（通过链式法则）计算损失函数关于模型参数（weights & biases）的偏导数/梯度，进而通过梯度下降法更新模型权重，这就是反向传播。

上述简要解释当成科普了解就可以了，对具体实现方式、原理有兴趣可以参考CB系列内容。对于本系列来说，更重要的是对模型的输入输出有更直观的理解，例如典型的有监督学习案例是电子邮件分类，即根据电子邮件的内容，判断是否是垃圾邮件。把模型部分当成黑箱，输入是邮件标题、正文、关键字等内容，输出是0（正常邮件）或1（垃圾邮件）；再比如简单的推荐系统，输入是用户的相关信息（年龄、购物历史、浏览历史等）和广告（分类、形式等），输出是0（不点击广告）或1（点击广告）。常见的图片相关算法也是有监督方法，例如缺陷识别、目标识别等Object Dectection算法，输入是图片，模型一般是CNN架构（如SSD、RCNN、Yolo等），输出是五维数组：{x, y, w, h}和label，{x, y, w, h}是定位矩形框的，label是用来分类对象的。

最后再解释一点关于什么是好模型、什么是scaling law（规模法则）。评价一个模型架构好坏（是指CNN /RNN /Transformer这类模型架构，不是指具体算法）的一个很重要思想是模型的智能承载能力。好的模型是具有规模效应的，即随着训练数据、训练时长增加，模型的性能会持续提升，通常的经验是同样模型架构，越大（指参数量）的模型效果越好；而类似Transformer一类的模型架构，具有更好的规模效应。

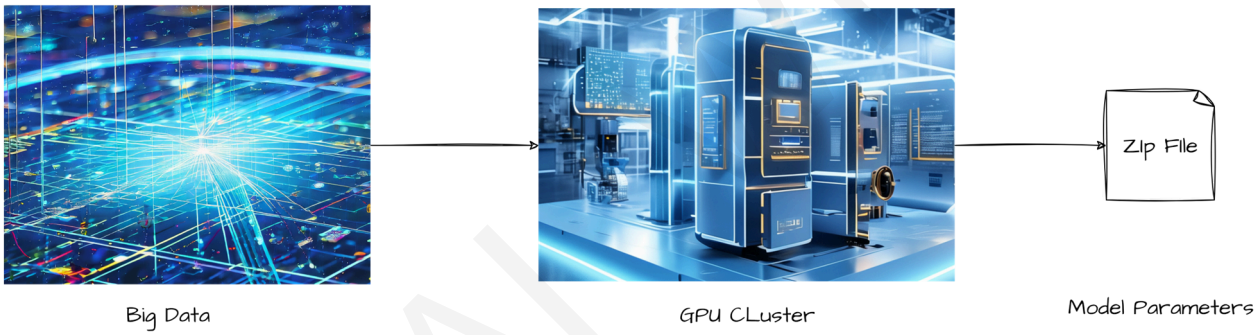


1.3 What is llm

对于大语言模型LLM的理解，需要从多个角度出发。最直观的角度是延续之前的方法，从模型的输入输出开始。LLM任务可以总结成next-word prediction（更准确说是next-token prediction），即基于前文，预测下一个字（实际是词元Token）。重复进行上述预测，就迭代地生成了一段文字。如下例子所示，原始输入是“今天”，LLM预测结果是“天”；第二次的输入是“今天天”，输出是“气”；反复重复上面的过程，直到LLM判断对话结束（即输出一个特殊的标识符，如[end]）。

Step	Input	Output
1	今天	天
2	今天天	气
3	今天天气	很
4	今天天气很	好
5	今天天气很好	[end]

第二个角度理解LLM，可以从数据的角度去理解。LLM在训练的过程中使用大量互联网上的数据（TB级或更高），然后通过GPU算力集群（数千、数万张GPU显卡），调整LLM模型中的权重（几十、几百、几千GB），因此可以看作LLM将大量信息，压缩进了模型（权重）当中。从这个角度看，LLM相当于压缩、汇聚了大量信息。

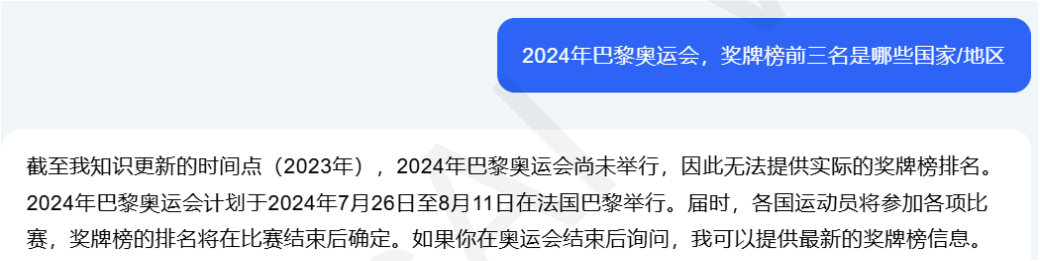


第三个角度理解LLM，可以从计算机或使用的角度理解。模型训练好了之后，使用时可以看到两个文件：第一个是权重文件。这个权重文件的大小取决于：1）参数有多少；2）每个参数占多少空间。比如一个规模是70B的模型，表示其中的参数有700亿个；如果模型的精度是FP/BF16的，也就是用16 bits（2 Bytes）存储一个参数，那么模型文件大约是140GB；如果精度是Int8/Int4的，对应大小就是70GB/35GB。另一个是一段能够运行这个模型程序文件，里

面包含如调用权重文件、神经网络架构等等信息。在使用LLM时，要将模型的权重从硬盘移动到内存/显存（RAM/vRAM）当中，这也是为什么对于推理来说，瓶颈一般是显存而非算力。

1.4 What LLM cannot do

在理解LLM能做什么之前，可以先了解LLM有哪些局限。第一条，也是被很多人熟知的是，LLM的知识有截止时间（Knowledge Cutoffs）。LLM的知识是基于训练数据的，所以并不掌握最新的知识。例如下面示例中(使用Kimi生成)，询问近期的新闻，LLM会表示不了解。当然，目前许多LLM应用会在对话过程中加入网络搜索功能，使得LLM能够获得最新的知识/新闻。



第二条是LLM有幻觉（Hallucination），简单理解就是LLM可能会“编造”一些“事实”，所以对LLM生成的回答（尤其是“严肃”的知识）要进行评估、筛选和判断。例如下面示例中，当让LLM提供一些论文时，会给出一些不存在的文章（如2，3-5未检查）。

第三条是关于LLM的上下文长度的。LLM具有限的上下文长度，比如ChatGPT刚发布时只有4K的上下文长度。但随着技术的进步，越来越多的大模型拥有了百万甚至千万的上下文长度(有部分宣传属性，一般而言，32K的上下文对绝大多数场景都是足够的)。

第四条不擅长处理结构化、表格型数据，比如进行数据分析、数学运算等等。比如近期比较出圈的案例，9.8与9.11哪个大的问题（现在许多大模型都可以给出正确答案了）。但是，并不是说大模型不能应用在数据分析领域，例如可以利用大模型作为推理引擎，以计算器、机器学习模型等作为工具，整体构建成基于大模型的智能体系统进行数据分析。当然，这是后面的话题了。

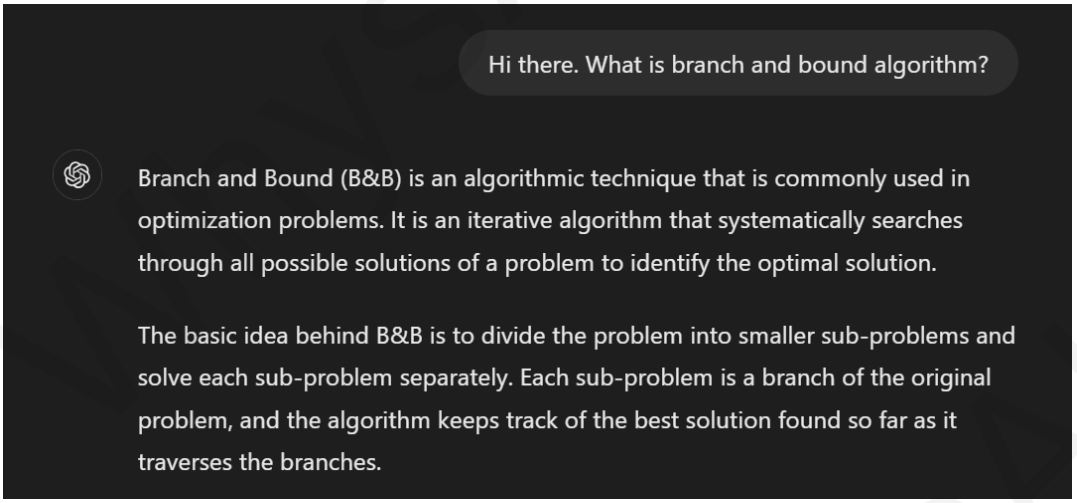
9.8和9.11哪个大

9.8和9.11这两个数字比较起来，9.11更大。

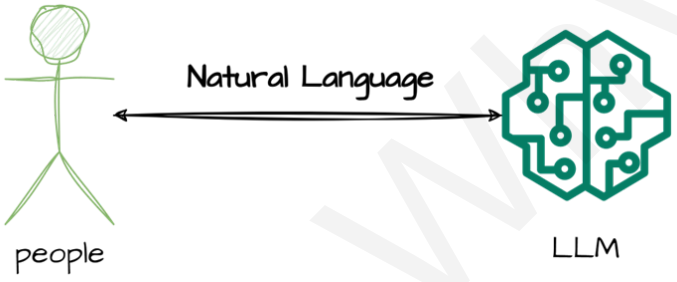
第五条跟安全性、价值观有关，比如大模型有可能会产生偏见、歧视等问题，也可能会输出有害的内容。

1.5 What LLM can do

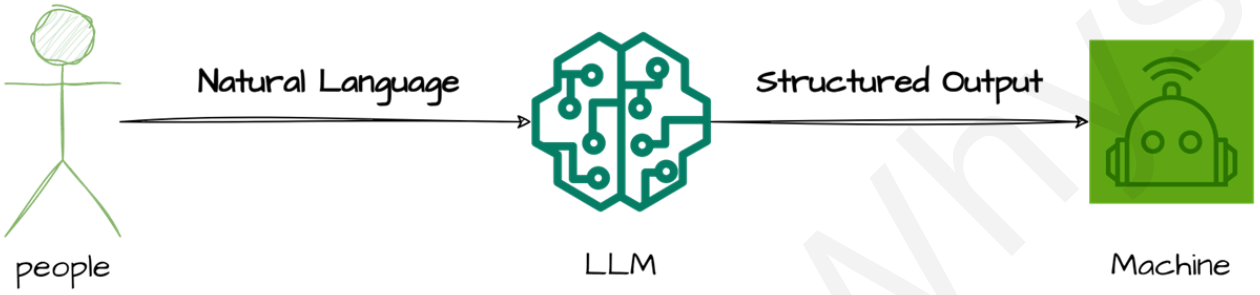
理解LLM能够胜任什么任务，可以从Chat场景说起。大多数人了解LLM是从ChatGPT开始的（下图是我的账号与ChatGPT第一次对话内容），ChatGPT提供了人和LLM通过QA问答进行交互的场景。从Chat场景出发，可以自然而然地延申到三个典型且通用的场景：写作、阅读、搜索。在写作场景下，可以让LLM生成故事、写作思路来帮助头脑风暴，也可以将一段草稿发给LLM，让其润色；在阅读场景下，可以将一段话发给LLM让其总结关键要点，也可以将文档发给LLM，然后进行基于文档内容的对话；在搜索场景下，LLM结合搜索引擎，帮助用户收集、整合相关信息。可见，LLM的基础能力是自然语言，即能够说“人话”。



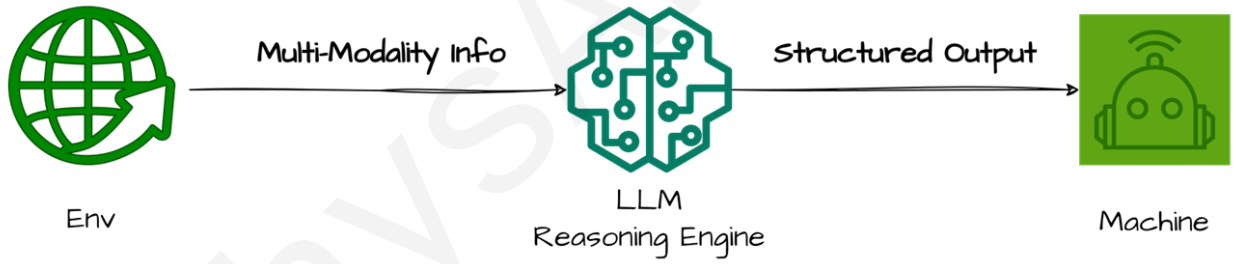
重新思考对话场景，其实质上是人和LLM之间通过Q/A的方式，进行持续交互，即对于LLM来说，输入是人（自然语言）、输出也是人（自然语言）。这是LLM的第一类典型场景。



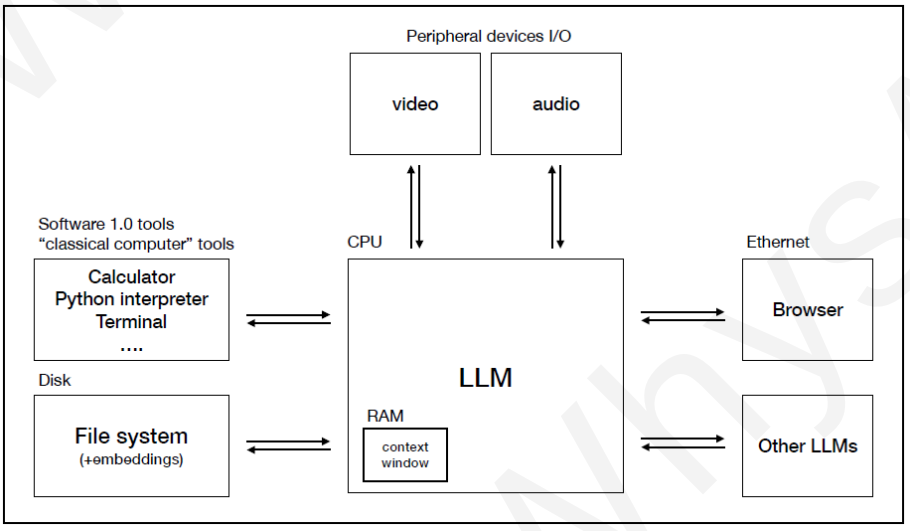
如果LLM的输出对象从人转变为机器，会怎么样？LLM的输出格式就要从自然语言转变为机器语言。机器语言可以是代码，如果是人使用代码就是辅助编程；如果搭配运行/沙盒环境，或执行机构（如机器人），就实现了自然语言执行代码、控制相应机构。机器语言也可以是某种结构化输出，如API接口、JSON字符串、SQL语句等等，再搭配一段执行这些结构化输出的程序，可以实现调用函数、调用工具、数据库查询等等功能。这是LLM的第二类典型场景。



LLM的输入也可以来自于其他对象，比如环境中的传感信息，在这样的范式之下，LLM主要作为信息处理中心或推理引擎。例如当前典型具身智能（简单理解为LLM+机器人）应用范式VLA(Vision-Language-Action)，就是指环境信息以图像等形式输入给LLM，LLM通过自身的推理能力，用机器语言（结构化输出），给出相应的执行指令。这是LLM的第三类典型应用。



总结来说，通过上述典型应用可以看出，LLM核心能力有以下三点：1）自然语言；2）结构化输出；3）逻辑推理。基于以上的核心能力，就能够理解Andrej Karpathy总结的LLM as OS含义：



当然以上只是定性描述了LLM能力范畴，并没有定量描述LLM各项能力强弱。对于不同LLM能力对比，推荐查阅使用ELO机制的LLM排行榜LMSYS ChatBot Arena。对于理解LLM能力强弱（尤其是逻辑推理能力），理念上LLM更类似于快思考的System1系统，而非慢思考的System2系统；直观判断LLM也可以通过这样的方法：刚上大学的年轻人，是否能够通过清晰的指令，完成相应的任务。

1.6 Prompting

上文最后一句话提到了“清晰的指令”，对LLM来说，是指输入给LLM的提示词（Prompt）。关于提示词，目前还没有通用的模板（可能未来也不会有，因为LLM的使用场景太广泛了），但有一些通用的建议与经验。

首先，提示词需要细致且具体，并且包含LLM执行对应任务所需的上下文/背景知识。具体做法比如给出相关的背景知识、给LLM设定角色、给出期望输出的模板/格式、输出限制条件等等。

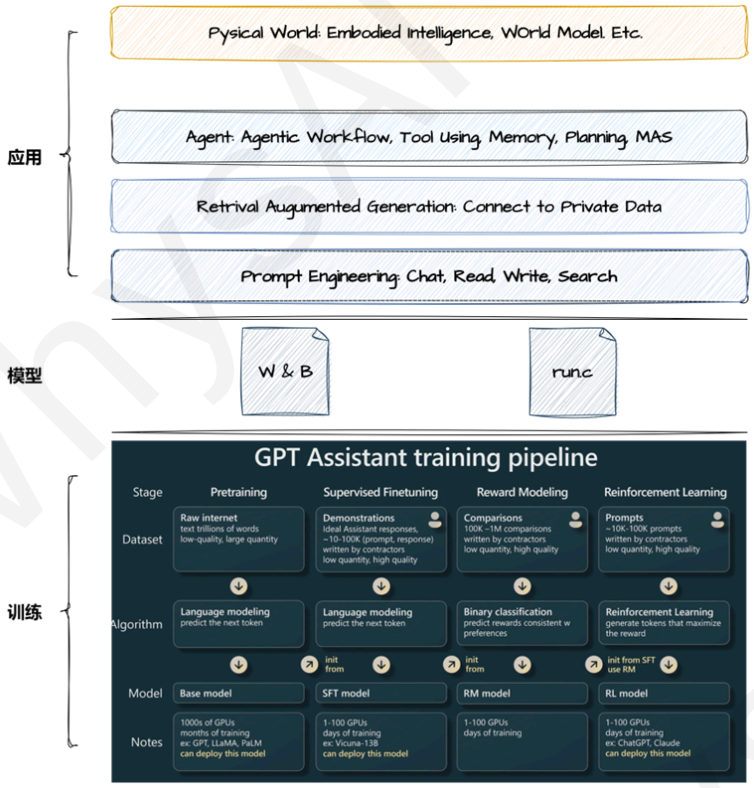
其次，通过提示词指引LLM逐步思考。例如给LLM设定思考流程或步骤，让LLM逐步思考得到最终的答案。

最后也是最重要的，要在具体场景下试验并迭代提示词。最开始的提示词不需要考虑太多，通过试验，再逐步提高即可。

2 High-level Technical Overview

2.1 Whole Picture

宏观上如下图所示，可以以是否改变模型参数为分界，将领域划分成应用技术与算法/训练，大致对应应用工程师与算法工程师的工作。实际上，会有些许不同，比如FT(Fine-tuning, 微调)技术，尤其是LoRA一类的PEFT技术，随着原理、工具的成熟，越来越具有工程属性。



2.2 Training Pipeline: Pre-training & Post-training

对于算法训练相关内容，本系列只是简要介绍（更多内容欢迎关注4S系列）。上图中的训练部分是OpenAI的训练流程，大致分成了四个阶段。当前更多的表述方法是直接分成语预训练（pre-training）和后训练（post-training）两个部分，其中post-training包括了SFT、DPO、RLHF等等。Pre-training的目标是使用非常大量的数据和大规模算力，让模型具备next-word/token prediction的能力，一般称为base model（基础模型）；Post-training的目标是使用（相对PT来说）少量的有标签数据与少量算力，让模型能够遵循指令（也就是说“人话”）、跟人类价值观对齐（也就是不乱说）、增强特定的能力（比如调用函数、编程、垂直领域知识等等）。

以近期发布的Llama3.1为例，在算法上使用“传统”的Dense Transformer架构（这里的传统只是说没有用Mamba等状态模型或MOE等混合专家模型），最大规模的模型是405B：

1. 在预训练阶段：使用了15.6T的多语言数据，数据截至到2023年底，基本上是目前公开数据的上限了；使用了$3.8 * 10^{25}$ FLOPs算力，最大使用了1.6万张H100显卡。
2. 在后训练阶段：选择使用相对传统/简单方法，即SFT（Supervised FT）+DPO（Direct Preference Optimization），使用的数据包括人工标注、人工挑选、针对特定能力合成的数据（代码、多语言、数学、推理、长上下文窗口等）

一般来说，Pre-training只有少数有资源的大公司能承担训练成本，可行的做法是基于开源的Base model，进行post-training，或者在post-training模型的基础上，进一步进行微调。

2.3 Application Techinique Overview: RAG & FT & Agent

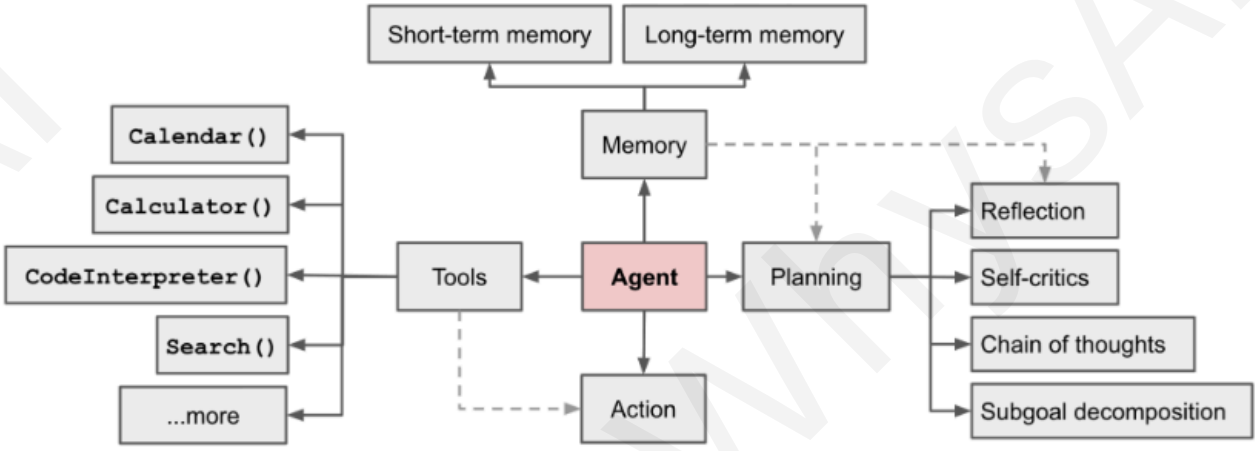
对于LLM应用来说，首先要能通过提示词PE的方式，对LLM问合适的问题。

除了PE之外，还有几项关键的技术/技术领域。第一项是检索增强RAG（Retrival Augmented Generation）技术。RAG的出发点很简单，即大模型的知识是静态的、公开的、有截止时间的，LLM没有经过私有数据的训练，自然也不了解私有的数据与知识。RAG技术本质上，先用私有数据建立知识库（文本等非结构化数据为主），当问题输入后，先到知识库当中找到相关的知识/文字，把“问题”和“相关知识片段”共同作为提示词，输入给大模型进行问答。所以，RAG技术的目标是接入私有数据/知识，本质上甚至可以看作是一种提示词增强手段。

RAG技术中，知识存放在LLM外部，相对的技术就是FT微调，即把知识压缩进模型内部。当然从资源/算力/数据等角度看，FT有更高的成本，所以FT技术一般适合以下几种情况：1）需要完成的任务很难通过提示词的方式描述清楚；2）需要转化LLM风格，例如改变LLM的名字/自我认知/语言风格等；3）需要增强模型某种特定能力，比如编写代码、数学运算等；4）需要通过更小模型完成更大模型能做的任务，提高实时性、降低推理成本，其本质上与前一点一致。

对于PE、RAG、FT三种技术，从时间与资源成本角度来说，优先级是PE > RAG > FT，当前面的技术无法满足时，可以考虑使用后续技术。

最后一种需要单独介绍的技术是Agent智能体技术。下图是23年6月由Lilian Weng在博客中提出的智能体系统，其中Planning模块本质是大模型的推理能力，Memory模块本质是PE和RAG，对于Agent领域来说最重要的是使用工具能力，如同1.5节中讨论的，与工具使用能力直接相关的能力是结构化输出。当前，Autonomous Agent、多智能体MAS（Multi-Agent System）、自进化智能体（Self-improving/healing Agent），甚至具身智能都离不开基础的Agent技术。



3 GenAI in Real-life

3.1 d2d use cases

仅从理论上感受LLM能力是不够的，需要亲自跟LLM进行对话、设计提示词，才能对GenAI/LLM建立直观的理解。以下简要推荐几种适合日常使用的场景与工具：

1. 写作：LLM进行文字翻译、润色、扩写，拓宽写作思路，对于文字工作者很有帮助。推荐工具--万知平台、Kimi。
2. 阅读：上传文档，使用大模型进行文档摘要、总结、问答，适合阅读文章、生成会议纪要等场景。推荐工具--万知平台、Aminer（学术论文阅读）

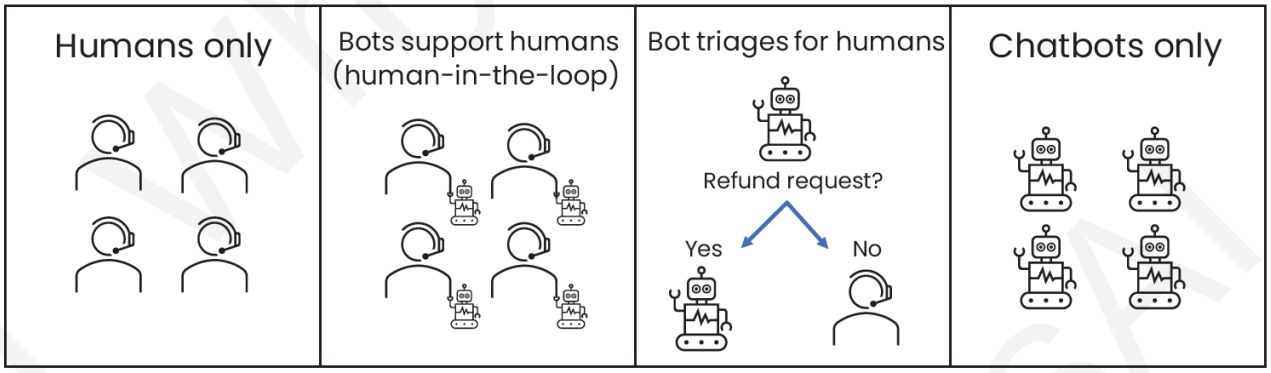
3. 代码：利用代码助手来辅助编程，适合融入开发者日常开发流程，如测试用例、编写文档、辅助代码等。推荐工具--MarsCode、Github Copilot
4. 文生图：给出文字描述，生成图片。推荐工具--通义万相、Flux/SD

3.2 GenAI Project

GenAI项目与其他项目有一致之处，都需要经历“问题->简化->优化->加速->自动化”的流程，需要在持续迭代中前行。只不过，GenAI/AI应用相关的项目，一般形成Demo的时间，要远远短于传统项目，也就意味着更适合“Ready-Fire-Aim”的形式，即有了基本想法就可以直接动手去做，之后再去迭代地调整目标和方向。一个明显的悖论是，完成Demo只需要一天，为什么要花一个礼拜讨论与计划呢。

当然，离需求更贴近的人员与离技术更贴近的人员总会存在信息不对称的情况，比如在数据分析时，一个常见的现象是拿来一些杂乱的数据要进行分析，但数据含义、质量、分析目标、路径等等都没有具体的分析（也是大数据不好产生价值的原因吧）。对于GenAI/LLM相关的场景，分析方法可以有如下两种：1）按照角色/岗位分析：即将某类角色或岗位的任务列出，以任务的维度分析判断，各项任务由AI替代的可能性与可行性；2）按照流程/任务分析：将整体任务拆解为一个个相互关联或独立的子任务，然后还是以任务维度，分析AI替代的可能性与可行性。

另一个有益的思考是，人与GenAI在具体场景下的关系。如下图所示，最开始的状态是没有AI帮助的；下一阶段，仍以人类为主，但AI可以辅助人类的工作，也就是human-in-the-loop，例如Copilot形态的产品；再之后，主要流程都由AI完成，人类只在异常出现时介入，例如当前技术比较先进的自动驾驶；最后，AI全面接管某项具体的任务。



在这样的路径下，人类是否会被替代是值得思考的问题。但我们过去的经验是，无论是蒸汽机还是汽车一类的技术进步，当然会导致传统的职业消失，但同时也会创造更多、更大量的工作需求。只是在这样的技术尚未成熟与大规模普及之前，我们没有那么确切地知道什么会消失、什么会出现而已。

总之，hopefully, AI would change our society in a good way.

4 Ref & Tools:

Highly Recommanded: Karpathy's Lecture on Intro to Large Language Models - YouTube

Our CB-series: LloydS827/WhysAI-LLMFS-CB (github.com)

GenAI Course by Andrew: Generative AI for Everyone - DeepLearning.AI

Sum of all short courses on deeplearnin.ai, cutoff-Aug.15.2024: Resources - DeepLearning.AI

First DL end2end prject introduced by Andrej Karpathy: Deep Neural Nets: 33 years ago and 33 years from now (karpathy.github.io)

The Bitter Lesson by Rich Sutton

智慧的疆界 (豆瓣) (douban.com)

人工智能（第4版）（豆瓣）(douban.com)

mckinsey-technology-trends-outlook-2024.pdf

LMSYS ChatBot Arena

ChatGPT

Llama 3.1 (meta.com)

Llama 3.1论文精读

LLM Powered Autonomous Agents | Lil'Log (lilianweng.github.io)

万知 I 问答、阅读、创作的一站式AI工作平台 (wanzhi.com)

通义万相*AI创意作画*AI绘画_人工智能-阿里云 (aliyun.com)

Kimi.ai - 帮你看更大的世界 (moonshot.cn)

marscode

FLUX.1 AI Model | BlackForestLabs (flux1.io)