



北京航空航天大學  
BEIHANG UNIVERSITY

# 大语言模型

人工智能研究院

主 讲 刘偲 沙磊 黄雷 郭晋阳

# Contents

- 大模型与工具学习概述
- 工具的不同类型
  - 数据类工具
  - 检索类工具
  - 运算类工具
  - 控制类工具
- 总结



# 大模型与工具学 习概述

数据类

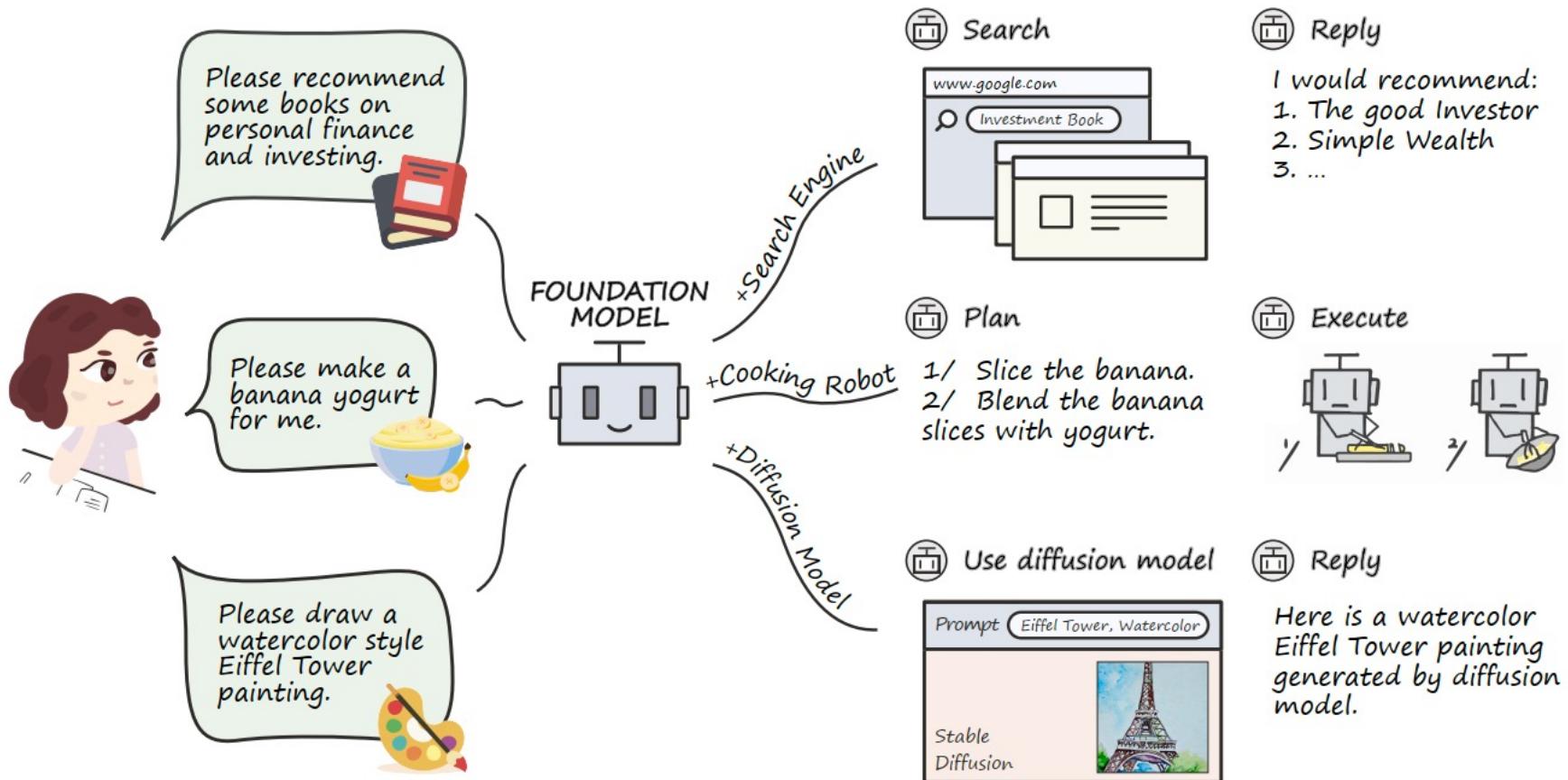
检索类

运算类

控制类

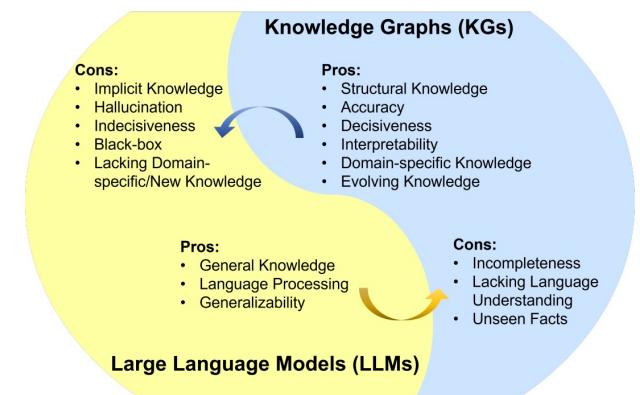
# 工具学习

- 定义：结合专门工具和基础模型的优势，实现问题解决方面的更高准确性、效率和自动化



# 发展背景

- 人类使用工具的认知起源:从石器时代到今天,工具使用是人类智力的重要标志
- 工具分类:从作用角色角度分为数据、检索、控制、运算四类
- 基础模型 (Foundation Model) 的范式转换
  - 以GPT-3为代表的基础模型兴起→语言模型预训练取得突破
  - prompt基础模型具有更强的理解和推理能力
- 基础模型与专业工具优势互补
  - 结合两者,可获得更好的性能、可解释性、稳健性等

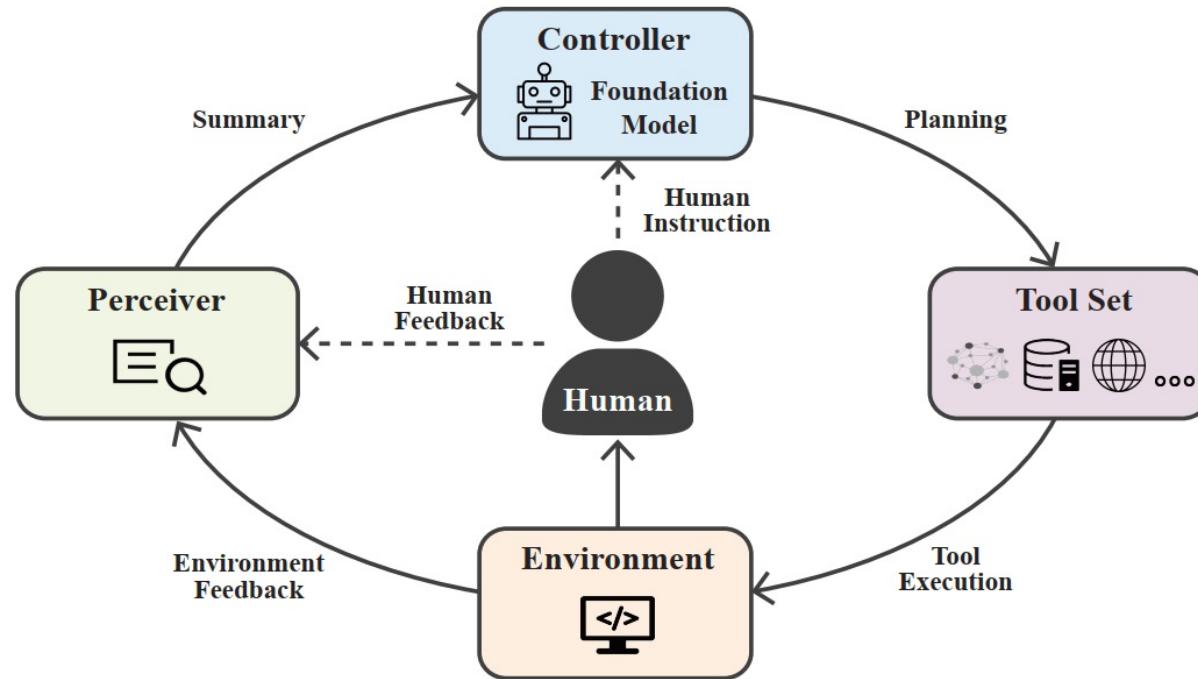


# 工具的分类

- 数据类工具：
  - 大模型可以利用这类工具查询获得任务相关的结构化数据
  - 为大模型提供持续学习和动态更新的新信息来源
- 控制类工具：
  - 可以控制大模型输出使其满足特定约束的外部模块。示例：法律文本生成控制器、特征约束生成控制器等
  - 避免大模型生成不满足要求的风险输出、提升大模型在专业领域的适用性、保证大模型生成内容的质量与合规性
- 检索类工具：
  - 从互联网上或文本库中检索需要的内容
- 运算类工具：
  - 大模型外包自己不擅长的计算任务给外部专业计算模块。如：数学运算模块、图像处理模块、表格分析模块等
  - 使大模型获得调用专业计算力的能力，避免大模型重复开发已有的计算模块，让大模型更加专注于自己的语义理解 strength

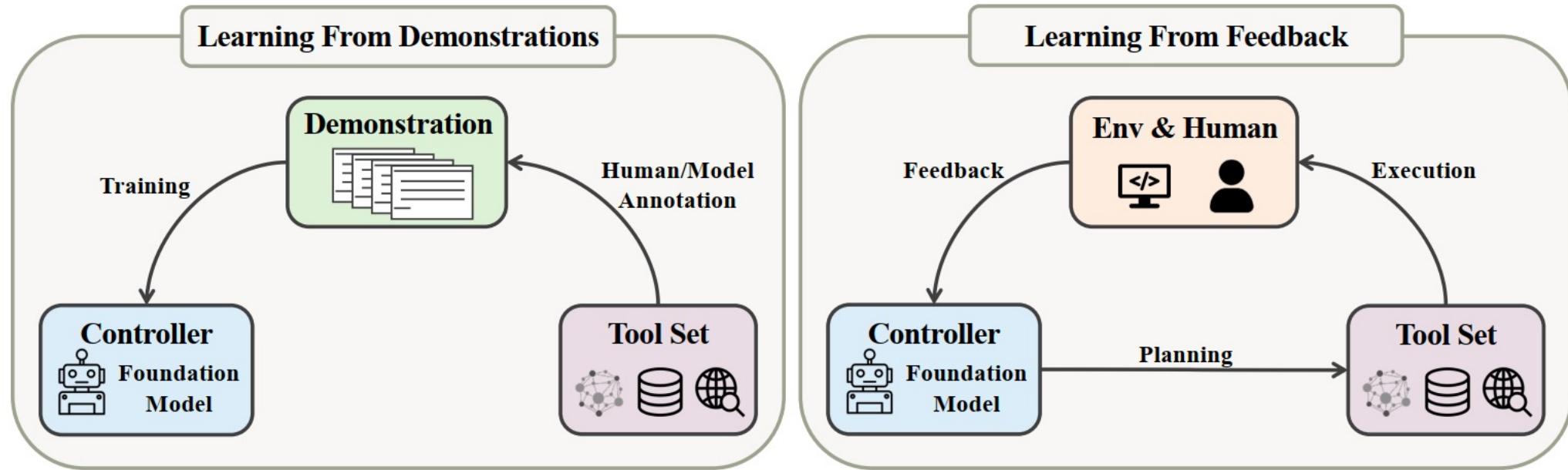
# 框架组成

- 4大要素:工具集、环境、控制器、感知器
- 它们之间的连接:控制器根据用户意图和环境反馈,选择合适的工具集执行任务



# 整体流程

- 从用户意图到计划:理解意图、工具，进行推理规划
- 模型训练:学习示范、学习反馈，提高工具使用能力

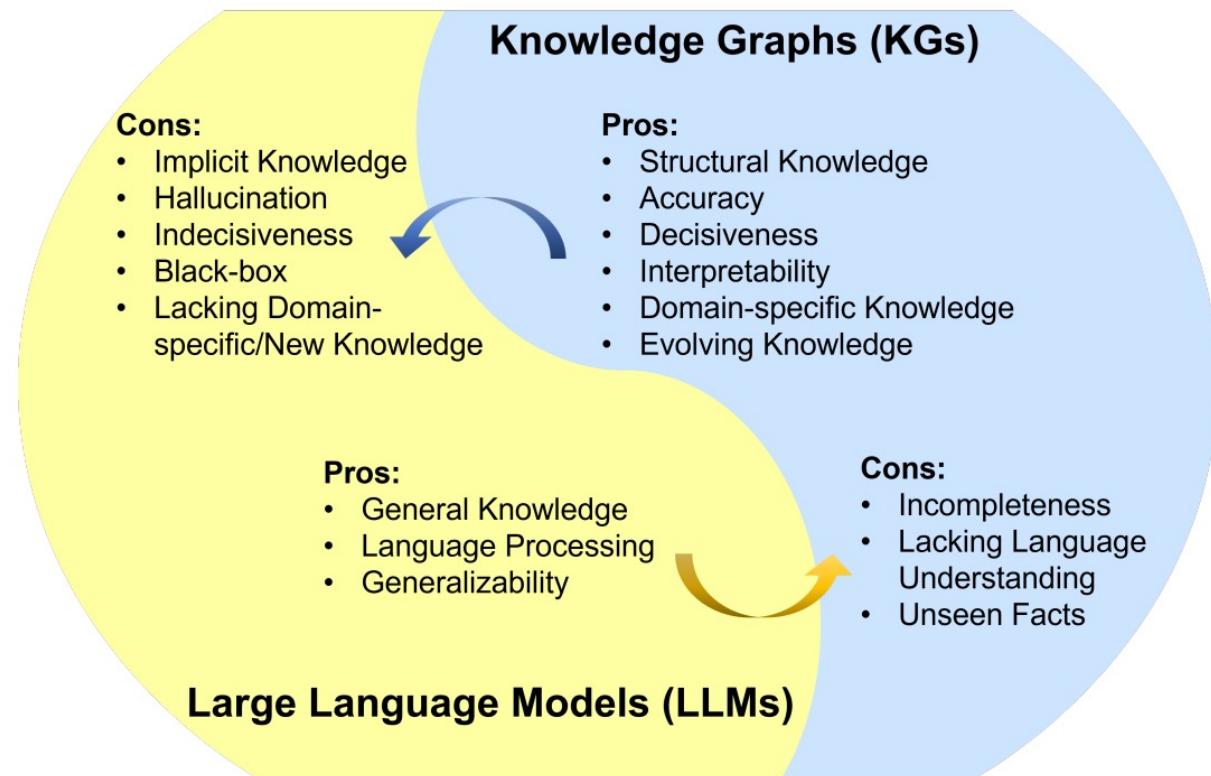




## 工具的不同类型

# 数据类工具

- 为大模型提供结构化数据存储、查询和检索的工具接口
- 数据库、知识图谱、搜索引擎等
- 知识图谱+大模型



# KG-enhanced LLMs

- 背景:大语言模型表现强劲,但容易存在事实错误(hallucination)和缺乏解释性
- 知识图谱以结构化方式存储大量事实知识,可以用来增强语言模型
- 主要方法:
  - 将知识图谱集成到语言模型的预训练阶段
    - 将KG整合进大模型的输入
    - 添加基于知识图谱的预训练目标
  - 将知识图谱集成到语言模型的推理阶段
    - 动态知识融合
    - 动态检索相关知识图谱事实

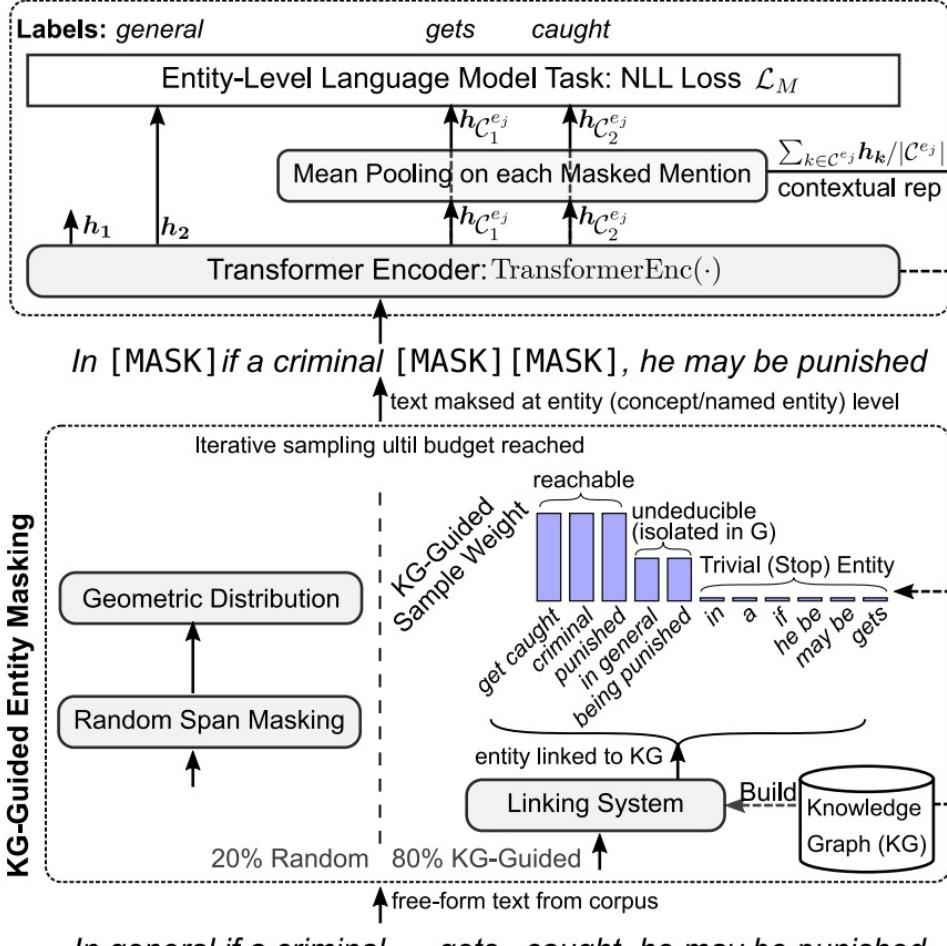
# KG-enhanced LLM Pre-training

- Integrating KGs into Training Objective

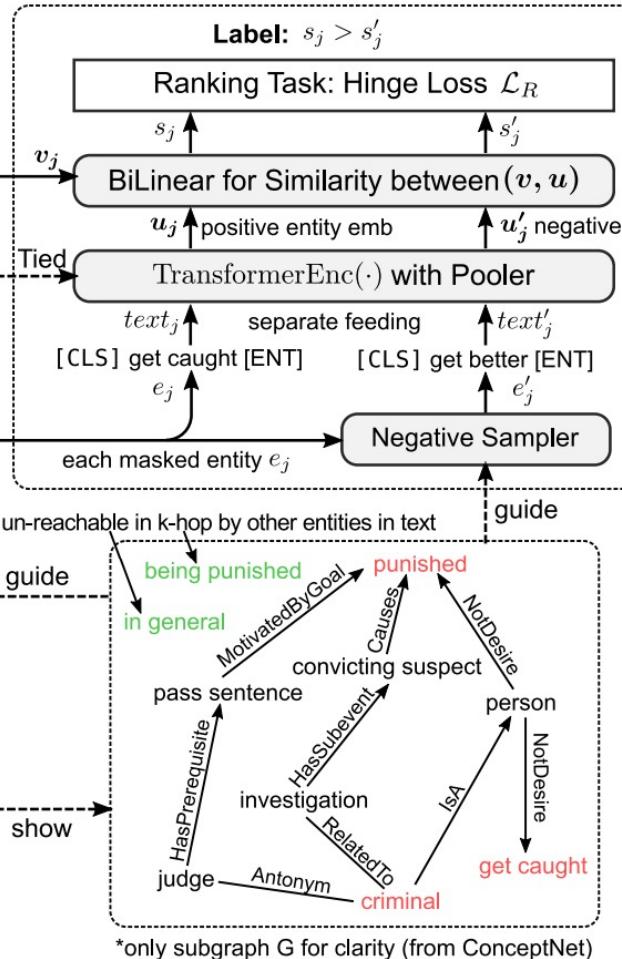
GLM	Shen et al., 2020	使用知识图谱结构为实体分配掩码概率
E-BERT	Zhang et al., 2020	动态调整词项与实体的训练loss权重
SKEP	Tian et al., 2020	识别情感词,在掩码目标中给这些词更高的权重
ERNIE	Zhang et al., 2019	增加词-实体对齐训练目标
KALM	Rosset et al., 2020	增加实体预测任务到仅含词的预训练目标中
KEPLER	Wang et al., 2021	整合知识图谱嵌入目标到掩码语言模型目标中
Deterministic LLM	Li et al., 2022	只掩码确定的事实,增加线索对比学习目标
WKLM	Xiong et al., 2020	替换文本中的实体,学习识别被替换的实体

# Integrating KGs into Training Objective

**Task1:** KG-Guided Entity-Level Masked Language Modeling



**Task2:** Distractor-Suppressed Ranking

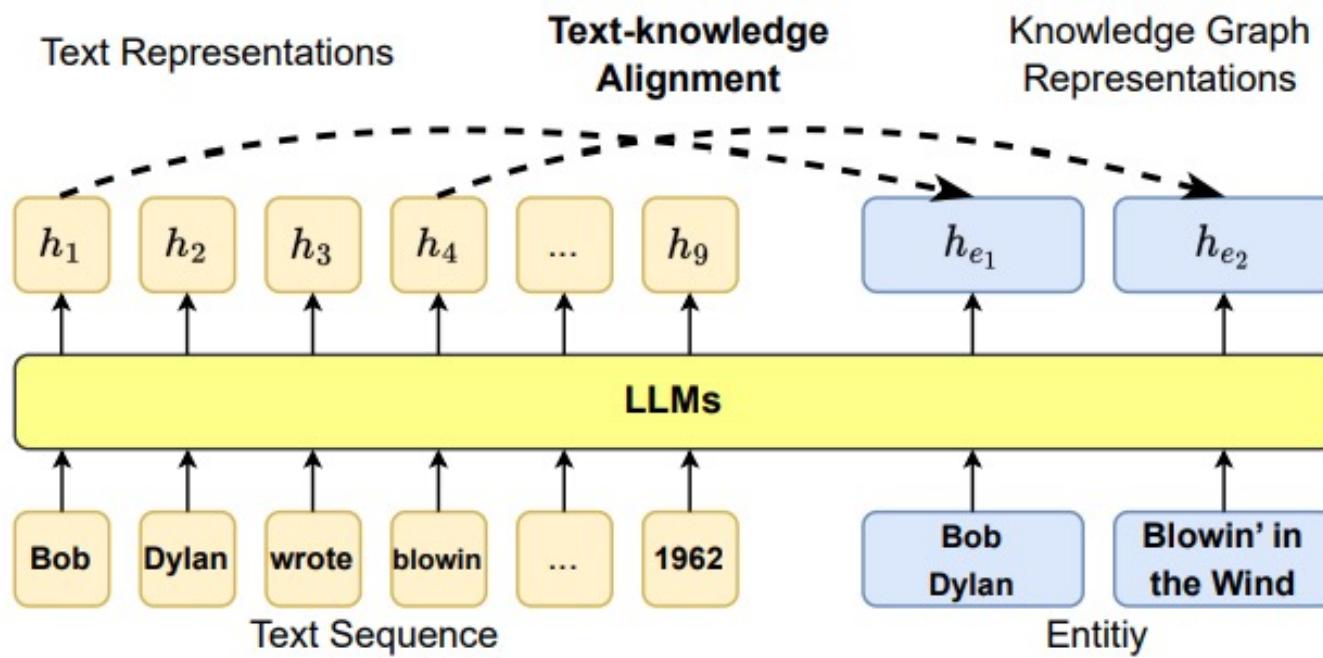


- GLM利用知识图谱的结构为实体分配掩码概率。在预训练过程中,可以在固定hop范围内访问的实体被认为是最重要的学习实体,并获得更高的掩码概率。
- Task 1: 通过知识图谱的指导,使语言模型从原始文本中学习结构化知识,其中利用知识图谱提供的实体词表来检测文本中的实体提及,并以这些实体提及作为掩码候选词,从而引导语言模型学习知识图谱中的结构化知识。
- Task 2: 从知识图谱中负采样构建排序任务,计算文本表示与正负样本的相似度,使用最大边际损失优化模型区分正负样本,以增强从文本中学习结构化知识的能力。

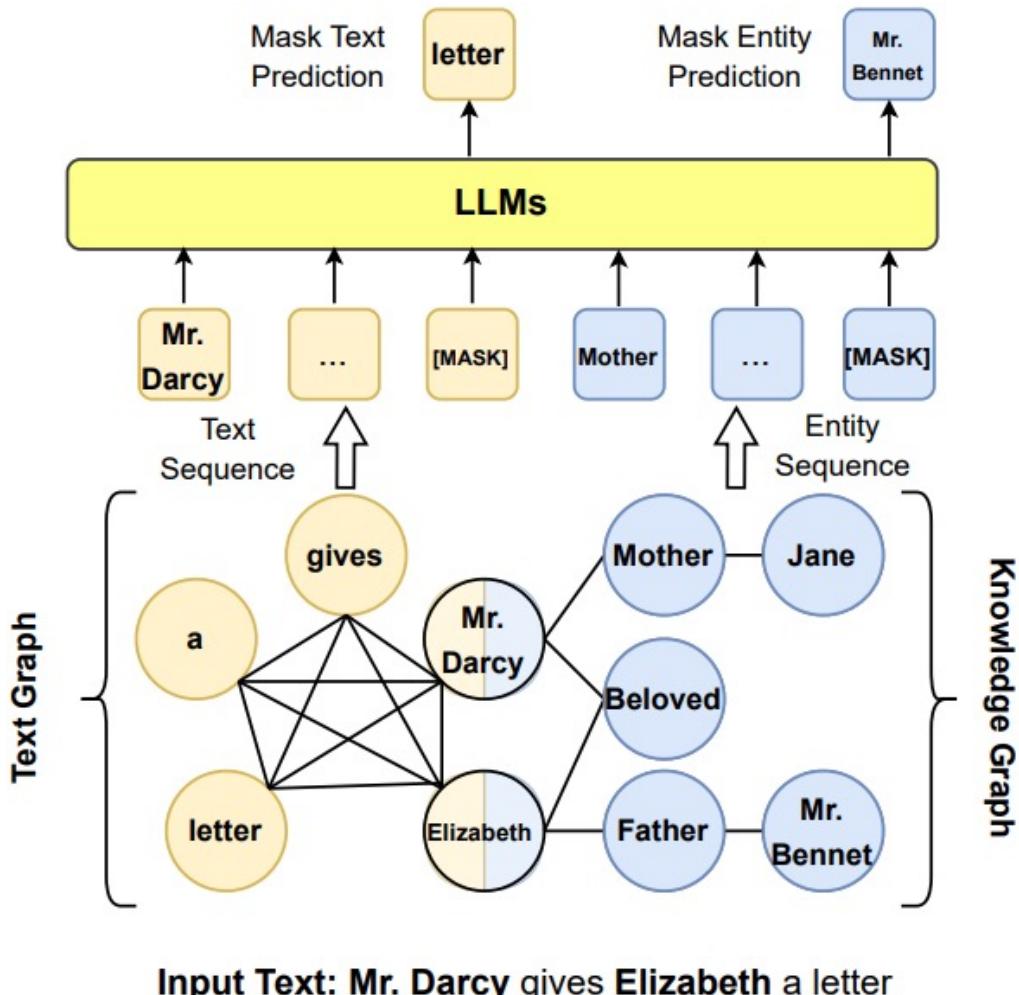
Figure 2: Graph-guided Masked Language Model (GLM) with two self-supervised tasks for MLM training.

# Integrating KGs into Training Objective

- ERNIE [92] 提出了一个新的词-实体对齐训练目标作为预训练目标。ERNIE同时将文本句子和文本中提到的相关实体输入语言模型,然后训练语言模型预测文本标记和知识图谱中实体之间的对齐链接。



# Integrating KGs into LLM Inputs



- 将相关知识子图注入到语言模型输入中。
- ERNIE 3.0：将知识图三元组表示为标记序列，并直接拼接到输入句子中，以便知识与文本表示的组合。
- K-BERT：通过可见矩阵将知识三元组注入句子中，只允许知识实体访问三元组信息，减少知识噪声。
- Colake：构建统一的词-知识图，允许文本词与相连的知识实体进行连接，进一步减少知识噪声。

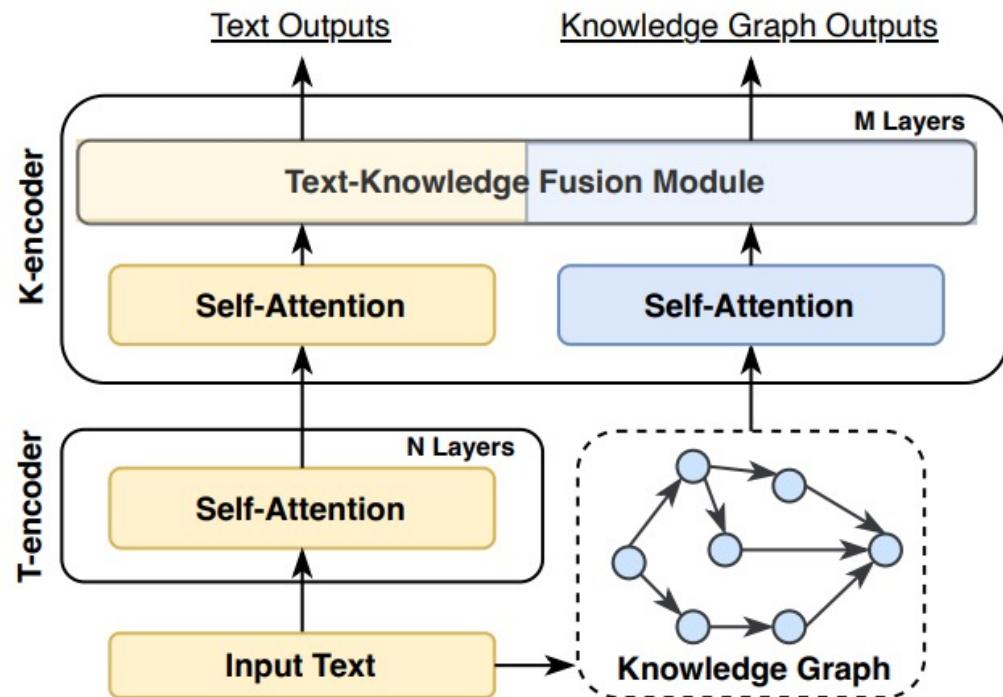
# Integrating KGs into LLM Inputs

- 上述方法可以将大量知识注入语言模型,但是它们大多关注热门实体,而忽略了低频和长尾实体。
- 一些可能的改进:
  - DkLLM : 提出新度量来确定长尾实体,并在文本中用某特定token的embedding替换这些实体作为语言模型的新输入,以改进语言模型对长尾实体的表达。
  - Dict-BERT : 提出利用外部词典来解决这个问题,通过在文本末尾追加词典中的罕见词定义,并训练语言模型本地对齐文本中的罕见词和词典定义的表达,以提高对罕见词的表达质量。

[1] T. Zhang, C. Wang, N. Hu, M. Qiu, C. Tang, X. He, and J. Huang, “DKPLM: decomposable knowledge-enhanced pre-trained language model for natural language understanding,” in AAAI, 2022.

[2] W. Yu, C. Zhu, Y. Fang, D. Yu, S. Wang, Y. Xu, M. Zeng, and M. Jiang, “Dict-BERT: Enhancing language model pre-training with dictionary,” in Findings of ACL, 2022.

# Integrating KGs by Additional Fusion Modules



- 这类方法通过在语言模型中引入额外的融合模块，可以单独处理知识图谱的信息并将其融合到语言模型中。

- ERNIE (清华)：提出了文本-知识双编码器架构,其中T-encoder先编码文本,然后K-encoder处理知识图谱,将其与文本表示融合。
- BERT-MK: 采用类似的双编码器架构,但在知识编码器中引入了额外的相邻实体信息,存在冗余和噪声问题。
- CokeBERT : 使用基于GNN的模块,利用输入文本过滤不相关的KG实体。
- JAKET : 在语言模型中间层融合实体信息,分别处理文本和实体,然后组合它们的表示。
- K-adapters : 通过adapters模块融合语言和事实知识,可以独立并行训练。

# KG-enhanced LLM Inference

- 真实世界的知识是不断变化的
- 如果不重新训练模型就无法更新融入的知识
  - →无法很好地推广到推理阶段没见过的知识。
- 将知识空间和文本空间分离,并在推理时注入知识
  - 动态知识融合
  - 动态检索相关知识图谱事实

# Dynamic Knowledge Fusion

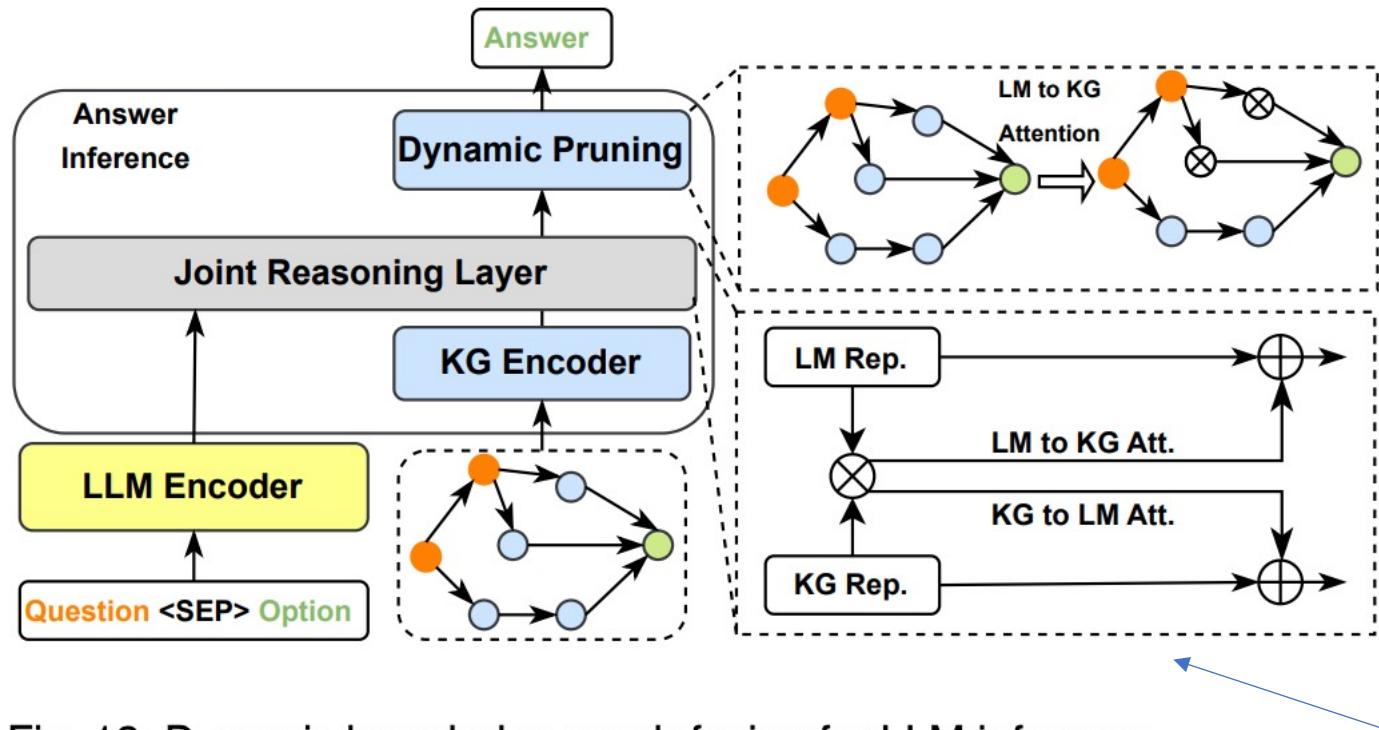


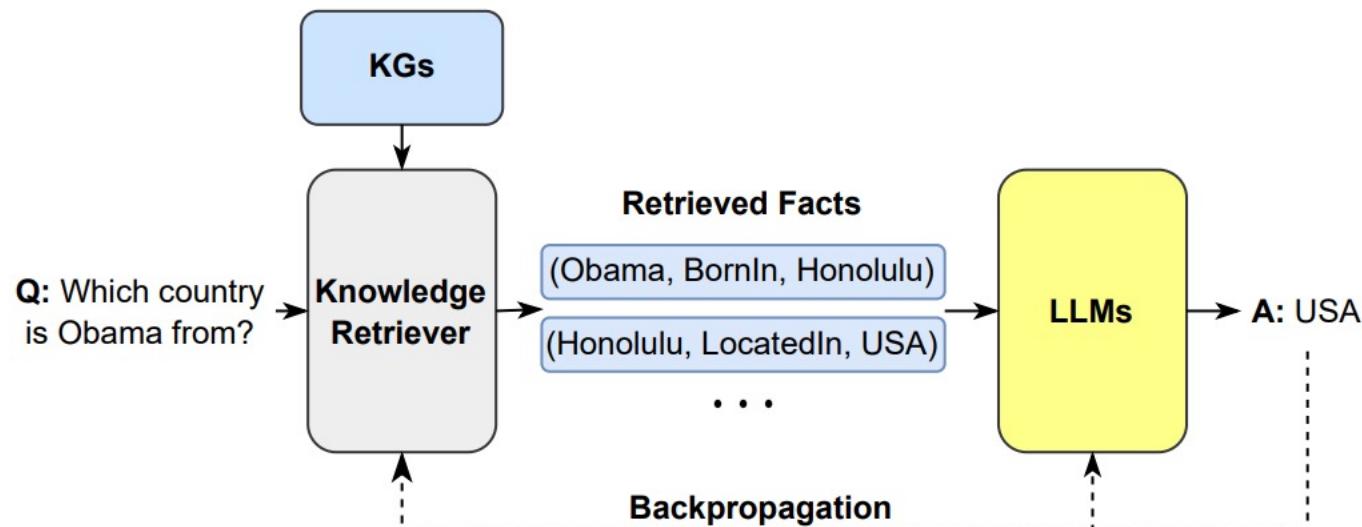
Fig. 12. Dynamic knowledge graph fusion for LLM inference.

- 双塔架构:一个模块处理文本,一个模块处理KG,但缺乏交互。
- KagNet:先编码KG,再增强文本表示,实现单向交互。
- MHGRN:用文本输出引导KG推理,单向交互。
- QA-GNN:用GNN实现文本和KG的消息传递,但文本只汇总为一个向量,限制了融合。
- **JointLK**:文本标记与KG实体间的双向细粒度交互,还动态裁剪KG。但文本表示来自最终LM输出。
- **GreaseLM**:在每个LM层之间设计文本标记与KG实体的深度丰富交互,类似ERNIE。

- [1] X. Wang, P. Kapanipathi, R. Musa, M. Yu, K. Talamadupula, I. Abdelaziz, M. Chang, A. Fokoue, B. Makni, N. Mattei, and M. Witbrock, "Improving natural language inference using external knowledge in the science questions domain," in AAAI, 2019.
- [2] B. Y. Lin, X. Chen, J. Chen, and X. Ren, "KagNet: Knowledge-aware graph networks for commonsense reasoning," in EMNLP, 2019.
- [3] Y. Feng, X. Chen, B. Y. Lin, P. Wang, J. Yan, and X. Ren, "Scalable multi-hop relational reasoning for knowledge-aware question answering," in EMNLP, 2020.
- [4] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, "QA-GNN: Reasoning with language models and knowledge graphs for question answering," in NAACL, 2021.
- [5] Y. Sun, Q. Shi, L. Qi, and Y. Zhang, "JointLK: Joint reasoning with language models and knowledge graphs for commonsense question answering," in NAACL, 2022.
- [6] X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. D. Manning, and J. Leskovec, "GreaseLM: Graph reasoning enhanced language models," in ICLR, 2022.

# Retrieval-Augmented Knowledge Fusion

- Weakness:
  - 所有知识都编码在模型参数中
  - 加入新的知识需要重新训练模型,模型无法适应不断变化的真实世界知识。
- 所以, 检索增强的知识融合方法呼之欲出



- RAG [94]: 结合非参数和参数模块,通过最大内积搜索检索相关KG,作为附加上下文引导Seq2Seq LM生成。使用不同检索文档在不同生成步骤效果更好。
- Story-fragments [123]:添加模块确定突出的知识实体,融合到生成器中提高生成长文本的质量。
- EMAT [124]:将外部知识编码到键值存储中,利用最大内积搜索提高效率。
- REALM [122]: 在预训练时提出知识检索器,从大规模语料中检索文档,提高开域问答的性能。
- KGLM [121]: 根据上下文从KG中选择事实,帮助生成陈述句。可以使用新词描述事实。

[1] Lewis et al., “Retrieval-augmented generation for knowledge-intensive NLP tasks,” in NeurIPS, 2020.

[2] Wilmot and Keller, “Memory and knowledge augmented language models for inferring salience in long-form stories,” in EMNLP, 2021.

[3] Wu et al., “An efficient memory-augmented transformer for knowledge-intensive NLP tasks,” in EMNLP, 2022.

[4] Guu et al., “REALM: Retrieval-augmented language model pre-training,” in ICML, 2020.

[5] Logan et al., “Barack’s wife hillary: Using knowledge graphs for fact-aware language modeling,” in ACL, 2019.

# Use KGs as tools

- What is the better way to use LLM ?
  - Learn the data/knowledge in KG
    - KG-enhanced LLM Pre-training
  - Learn how to cooperate with KG
    - KG-enhanced LLM inference
  - Learn how to communicate with KG
    - Use KGs as tools
    - LLM只需关注如何分析问题和组织回答,不需编码KG知识,使得两者解耦。
    - LLM可以灵活调用KG模块

# Use KGs as tools

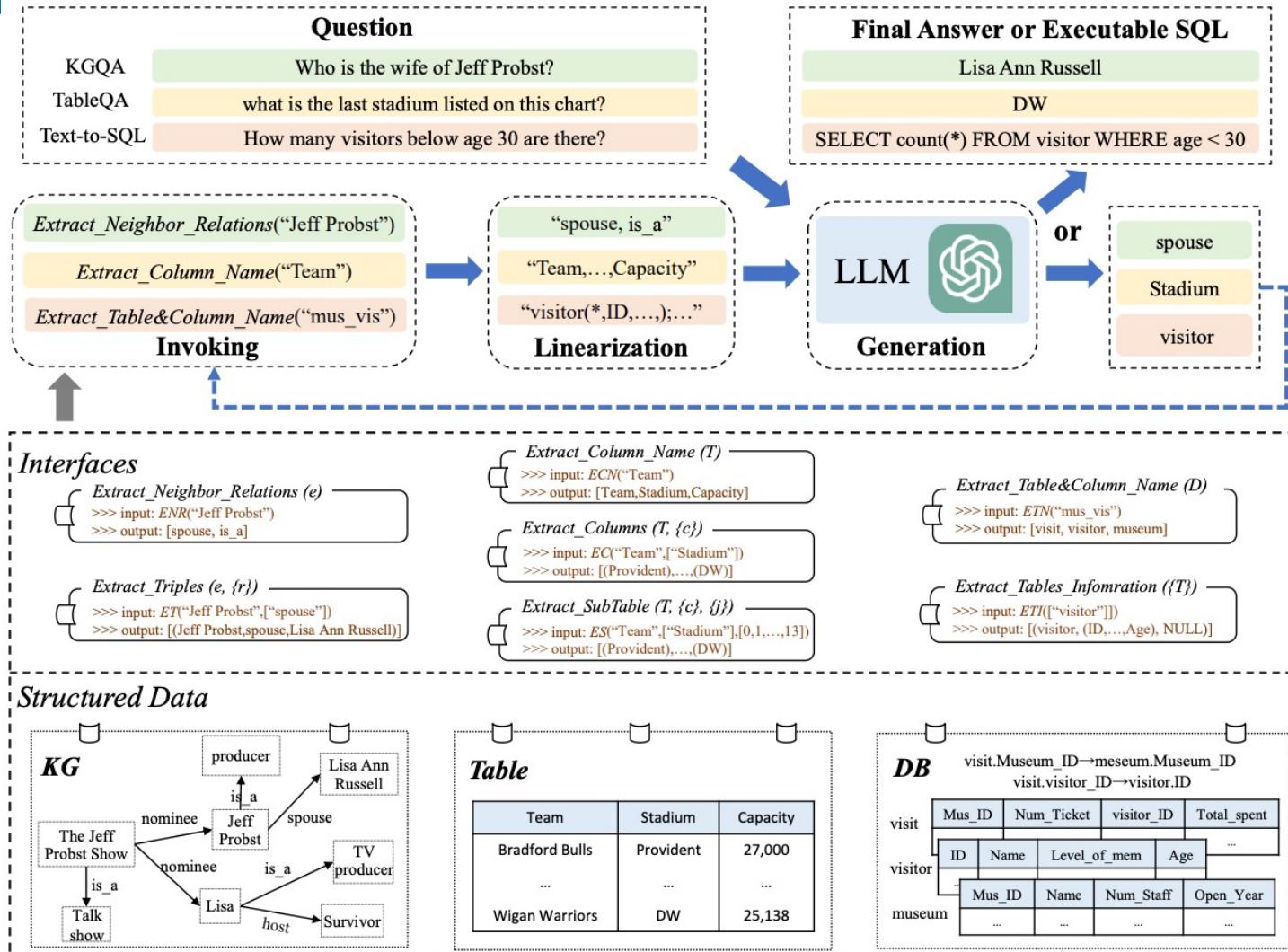
- 简单例子：
  - 为KG构建统一的访问接口,如查询实体、查询关系等。
    - `query_entity(name)`、`query_relation(entity1, entity2)`
  - 将KG当作一个可调用的模块或工具,为LLM提供调用KG接口的方法。
    - 在LLM中添加调用接口的方法,如`facts = KG.query_entity("Obama")`。
  - 在LLM中内嵌KG推理/检索模块,根据输入动态决定调用哪些KG接口。
    - 对用户提问,LLM分析后决定调用`query_entity`等接口,获取需要的知识。
  - 通过远程过程调用等方式,使KG模块与LLM解耦,LLM只需调用接口,不需编码KG。
  - 将检索到的KG知识与用户问题组合,生成回答。

# Use KGs as tools

- 基础模型具有一定的推理能力,可以将复杂问题分解为子问题并逐步求解。
  - few-shot prompt learning在需要复杂推理的任务上表现较差。
  - Chain-of-Thought prompting在prompt中添加推理过程,可以提高各类任务的表现。
- 工具学习的两大分类
  - Introspective Reasoning (内省式推理) : 直接生成多步工具使用计划,不再进行反馈迭代优化
  - Extrospective Reasoning (外展式推理) : LLM不断地与工具进行交互, 最终得出结果

# Use KGs as tools

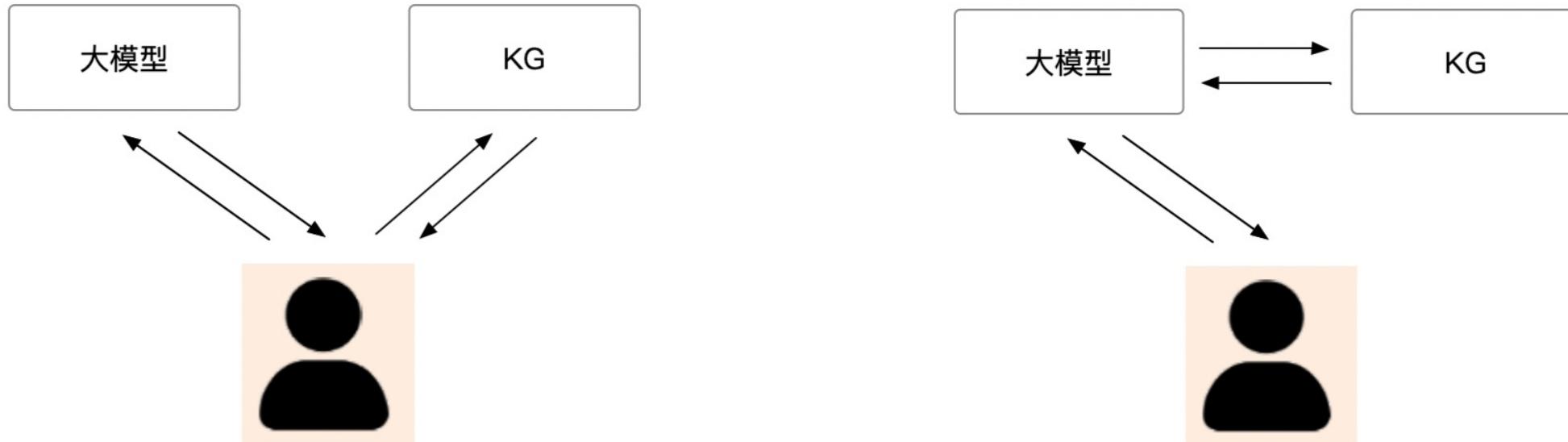
- Example: StructGPT
- KG 接口
  - Extract\_Neighbor\_Relations (e): extracts all the neighboring relations of the entity e.
  - Extract\_Triples (e, {r}): extracts all the triples with the relation in {r} and head entity e.
- Table 接口
  - Extract\_Column\_Name (T ): extracts all the column names of a table T .
  - Extract\_Columns (T , {c}): extracts the contents of columns from a table T by indices {c}.
  - Extract\_SubTable (T , {c}, {j}): extracts the sub-table specified by the column indices {c} and row indices {j} from a table T .



# Use KGs as tools

- StructGPT基本方法
  - 利用定义的接口，进行“调用-线性化-生成”三步
  - 1: 利用大语言模型基于接口读取和推理结构化数据
  - 2: 将其线性化为文本
  - 3: 喂入语言模型生成下一步结果或最终结果
- Pros:
  - 是一个通用框架,支持语言模型对多种结构化数据进行推理。
  - 将读取知识和推理过程分离,LLM可以集中精力推理。
  - 支持语言模型逐步获取更多有用证据。
- Cons:
  - 结构化知识被重新处理成文本再喂给LLM，动作冗余

# KG as tool, which mode is better?



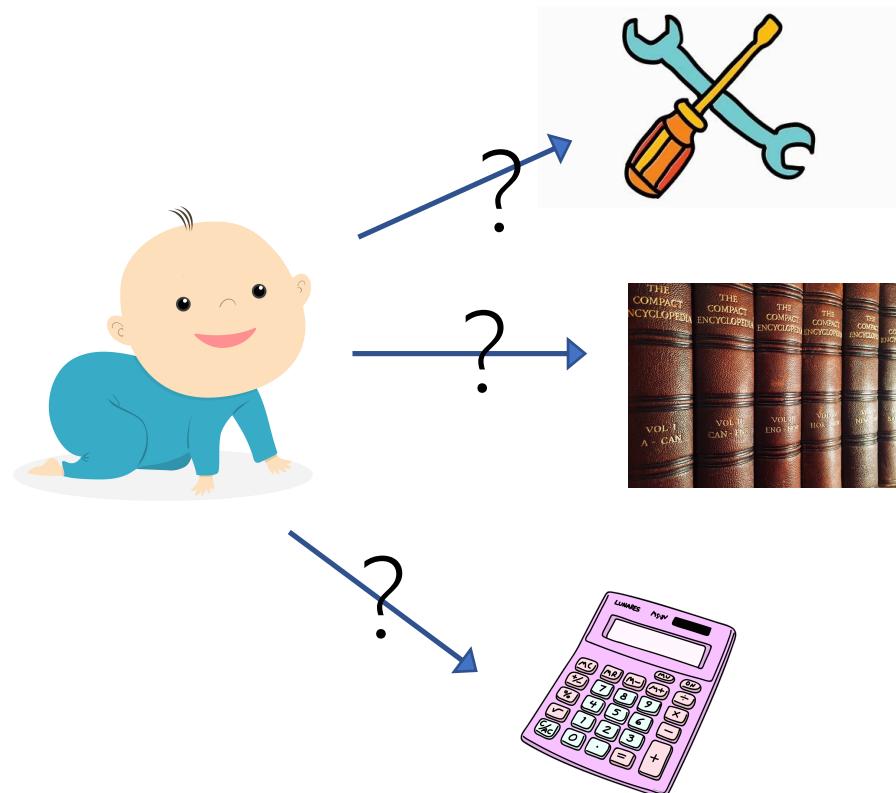
- Chain-of-thought
- StructGPT
- .....
- AutoGPT\*
- .....

Auto-GPT is an "[AI agent](#)" that, given a goal in [natural language](#), will attempt to achieve it by breaking it into sub-tasks and using the internet and other tools in an automatic loop.<sup>[1]</sup> It uses [OpenAI's GPT-4](#) or [GPT-3.5 APIs](#),<sup>[2]</sup> and is among the first examples of an application using GPT-4 to perform autonomous tasks

\* AutoGPT: <https://github.com/Significant-Gravitas/Auto-GPT?ref=jina-ai-gmbh.ghost.io>

# How to train? What to train?

- 婴儿如何学习使用工具?



- 有成人（家长/老师）在教它
  - Learn from demonstration
- 自己鼓捣
  - Trail and error
  - Learn from feedback

# How to train? Learn from demonstration

- Learn from demonstration

- 训练模型在给定输入或条件下模仿人类专家的动作。这种方法常用于专家动作可以轻松记录和利用的场景。
  - 也叫行为克隆（behavior cloning）

• 例：WebGPT（supervised learning）

  - (1) 构建了一个由Bing支持的搜索接口。
  - (2) 收集了人类与浏览器交互的演示。
  - (3) 通过行为克隆，对GPT-3进行了微调，学习state to action的映射。

## Update the parameters of LLM

**How can I train the crows in my neighborhood to bring me gifts?**

This question does not make sense    This question should not be answered

---

**Search results for: how to train crows to bring you gifts**    Quotes 

[←](#)    how to train crows to bring    Find in page            [+ Add new quote](#)

---

[How to Make Friends With Crows - PetHelpful](#)

If you did this a few times, your crows would learn your new place, but as I said, I'm not sure if they will follow or visit you there since it's probably not in their territory. The other option is simply to make new crow friends with the crows that live in your new neighborhood.

**[1] Gifts From Crows | Outside My Window**  
[www.birdsoutsidemywindow.org](http://www.birdsoutsidemywindow.org)

Many animals give gifts to members of their own species but crows and other corvids are the only ones known to give gifts to humans.

---

Number of quote tokens left: 463

Number of actions left: 96

[Done quoting! Write an answer](#)

# How to train? Learn from demonstration

- WebShop (supervised learning)
  - 模仿学习。训练出的模型可根据人类指示正确购买产品。

Type	Argument	State → Next State
search	[Query]	Search → Results
choose	Back to search	* → Search
choose	Prev/Next page	Results → Results
choose	[Product title]	Results → Item
choose	[Option]	Item → Item
choose	Desc/Overview	Item → Item-Detail
choose	Previous	Item-Detail → Item
choose	Buy	Item → Episode End

Table 1: Actions in WebShop.

**A** **WebShop** **search**

Instruction:  
I'm looking for a small portable folding desk that is already fully assembled; it should have a khaki wood finish, and price lower than 140.00 dollars

Search results: 1

Back to Search  
Page 1 (Total results: 50)  
Next >

MENHG Folding Breakfast Tray Table, Efficient Home Laptop Notebook Computer Desk, Portable Writing Study Desk, Sturdy Home Office Table Workstation \$109.0

KPSP Folding Study Desk Bed Breakfast Serving Tray Table Efficient Home Laptop Notebook Computer Desk Portable Standing Desk for Small Space Bedroom

**B** **Simple mode**

Instruction:  
I'm looking for a small portable folding desk that is already fully assembled [...]  
[btn] Back to Search [/btn]  
Page 1 (Total results: 50) [btn] Next [/btn]  
[btn] MENHG Folding Breakfast Tray [...] [/btn]  
\$109.0  
[btn] KPSP Folding Study Desk Bed [...] [/btn]

**C**  $u$  (Instruction): I'm looking for a small portable...  
 $\bar{y}$  (Description): MENHG Folding Laptop Table Bed...  
 $y_{price}$ : \$109.0  
 $Y_{opt}(\text{Options})$ : { black, khaki, white }  
 $Y_{att}(\text{Attributes})$ : { steel pipe, no assembly, portable }

# How to train? Learn from demonstration

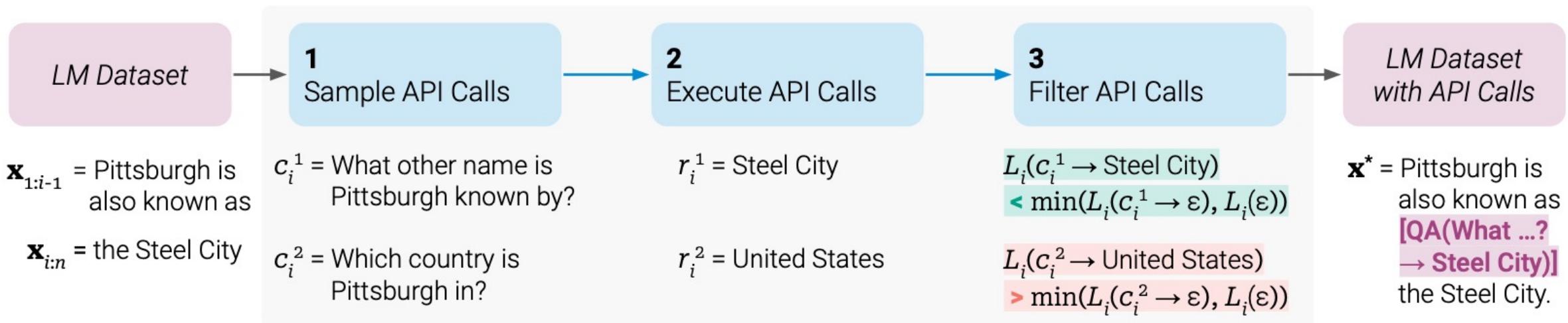
- Semi-supervised Learning.
  - 由于时间和成本的考虑,人类行为往往难以大规模记录。
  - 未标注的数据容易获得,从中可以构建弱监督或者含噪声的监督信号。
  - 可以利用能力较弱的模型在未标注数据上生成伪标签,将其转换为弱监督的工具
- 例: Baker et al, 2022
  - 在少量标注数据的基础上,训练模型为Minecraft中的每一时间步预测动作的伪标签。
  - 利用这些伪标签,可以训练更强大的模型,而不需要在目标环境中展开模型或者大规模的人类行为标注。可以在人类演示数据稀缺的情况下实现行为克隆。
- Cons:
  - 伪标签的质量直接影响学习效果,需要保证一定的伪标签准确率。

# How to train? Learn from demonstration

- Self-supervised Learning
  - 半监督学习仍需要种子标注数据集来获取伪标签。
  - 种子数据集的偏差也可能在训练中被放大,导致泛化性能不佳。
  - So, 让LLM自监督地学习如何利用工具。
- 例: Toolformer充分利用LLM的上下文学习能力
  - 基于少量人类示例迭代生成工具使用示例。
  - 生成的示例经过滤降噪后,构成充足的监督信号,显著提升了LLM的工具使用表现。

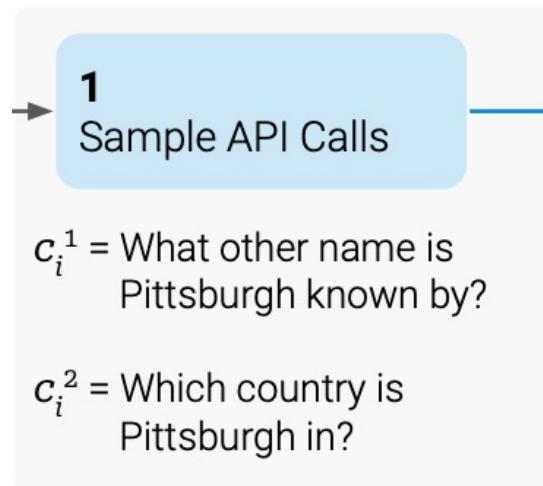
# How to train? Learn from demonstration

- Toolformer



# How to train? Learn from demonstration

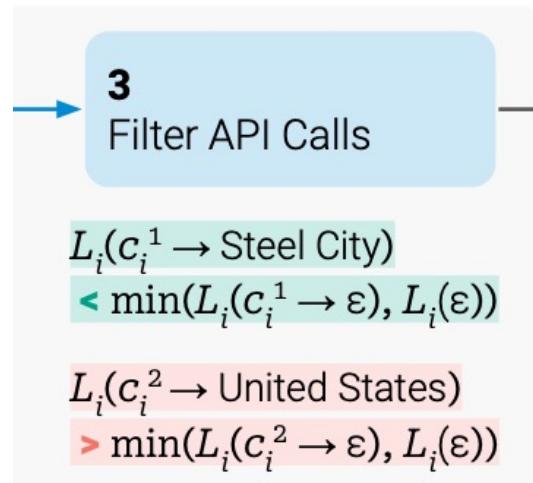
- Toolformer



- 1, 为每个API编写prompt  $P(x)$ , 让语言模型在样例  $x$  中添加API调用。prompt中给出了调用示例。
- 2, sample up to  $k$  candidate *positions* for doing API calls
- 3, 对每个位置 $i$ , 计算生成 $\langle\text{API}\rangle$ 的概率 $p_i$ 。选择 $p_i$  大于阈值 $\tau_s$ 的位置。如果超过 $k$ 个位置, 则取top  $k$ 。
$$p_i = p_M(\langle\text{API}\rangle \mid P(\mathbf{x}), x_{1:i-1})$$
- 4, 对每个选择的位置 $i$ , 通过采样语言模型, 以 $P(x)$ 、 $x_1, \dots, x_{i-1}$ 、 $\langle\text{API}\rangle$ 为前缀, 以 $\langle/\text{API}\rangle$ 为终止 token, 来生成API Call

# How to train? Learn from demonstration

- Toolformer



- 步骤

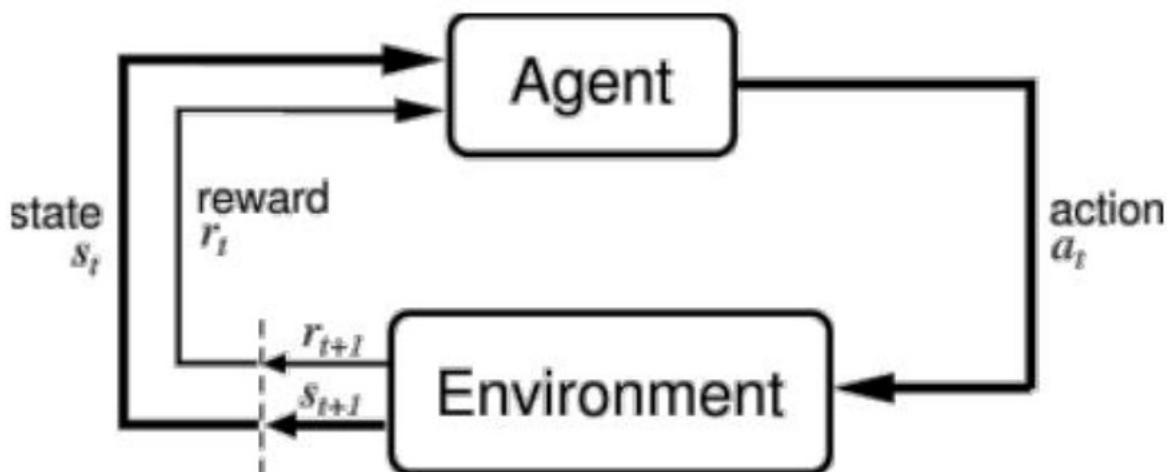
- 定义损失函数  $L_i(z)$  为在添加前缀  $z$  后, 语言模型预测第  $i$  个到第  $n$  个 token 的交叉熵。
- 比较添加 API 调用及结果  $e(c_i, r_i)$  时的损失  $L_i^+$ , 与不添加调用或不添加结果时的最小损失  $L_i^-$ 。
- 如果  $L_i^- - L_i^+ \geq \tau_f$ , 则说明添加调用及结果可以降低至少  $\tau_f$  的损失, 这个调用是有帮助的。
- 最后只保留这些有帮助的调用, 其余过滤掉。
- 这个过滤过程可以删除无关或无用的调用, 提升训练数据的质量。
- 损失函数的设计和阈值  $\tau_f$  的选择对结果质量非常关键。

# How to train? Learn from feedback

- Cons of demonstration?
  - 收集完整的人类工具使用演示非常耗时和劳动密集。并且学习的模型可能无法有效适应新的环境。
- 更自然的方式?
  - 人类是通过试错来校正和纠正工具使用行为。同样,来自环境和人类的反馈可以让模型理解其行为后果并适应行为。
- Learn from feedback
  - 从反馈学习可以形式化为最大化给定query集上的LLM参数,其中reward是来自反馈序列的估计。
  - 反馈可以是环境本身的,也可以是人类专家对模型给出的。
  - 反馈学习减少了对完整人类演示的依赖,使得训练更有效。
  - 反馈的质量直接影响应用效果,需要设计有效的反馈机制。

# How to train? Learn from feedback

- Reinforcement Learning

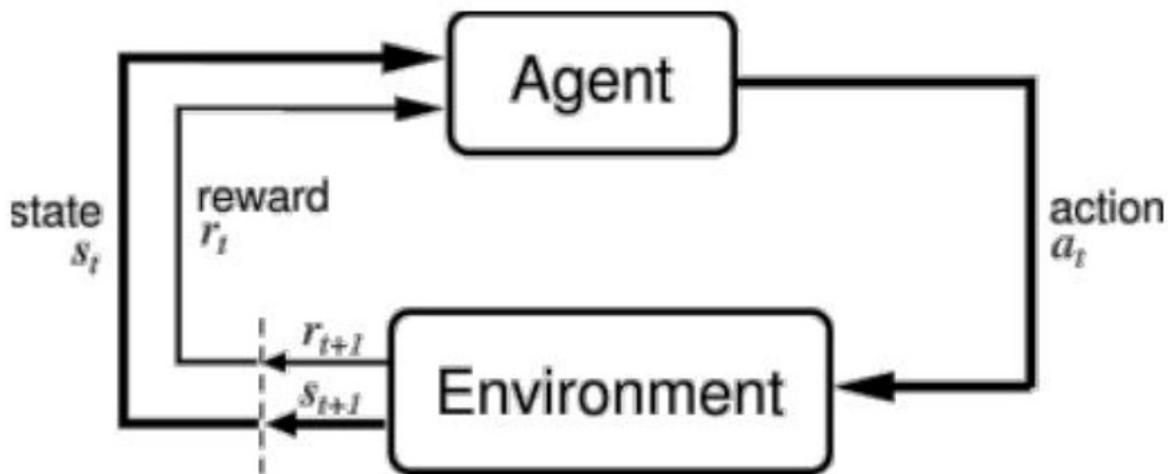


- 环境反馈

- (1)结果反馈,指明行为是否完成任务的最终反馈。它对模型的整体计划进行评估。  
(评估整体计划)
- (2)中间反馈,指一个行为触发的环境状态变化。它提供对每个工具执行的及时有效性信息。  
(详细指导单步行为调整)
- 通过观察状态变化,基础模型可以学习每个行为的有效性,进而调整行为。

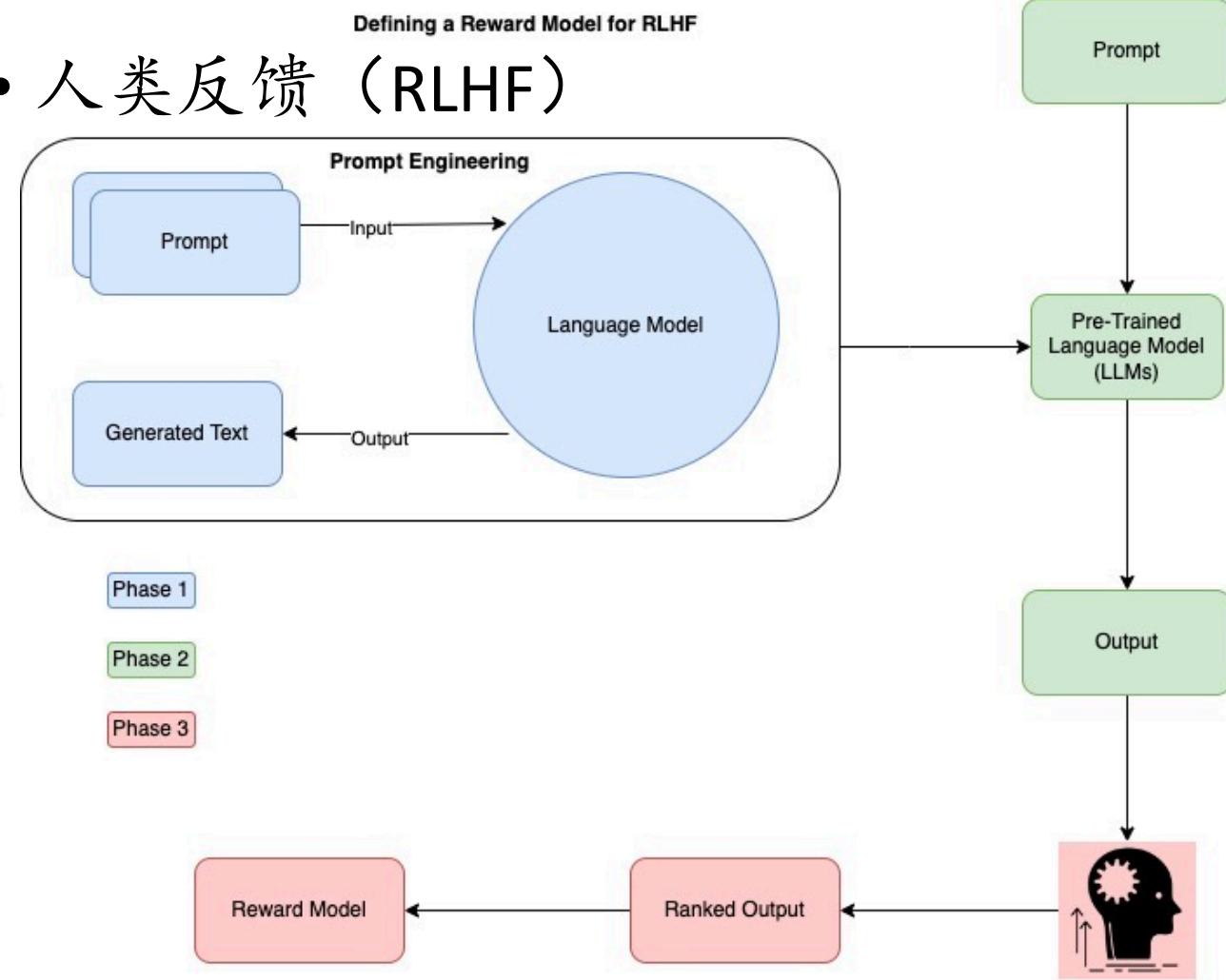
# How to train? Learn from feedback

- Reinforcement Learning

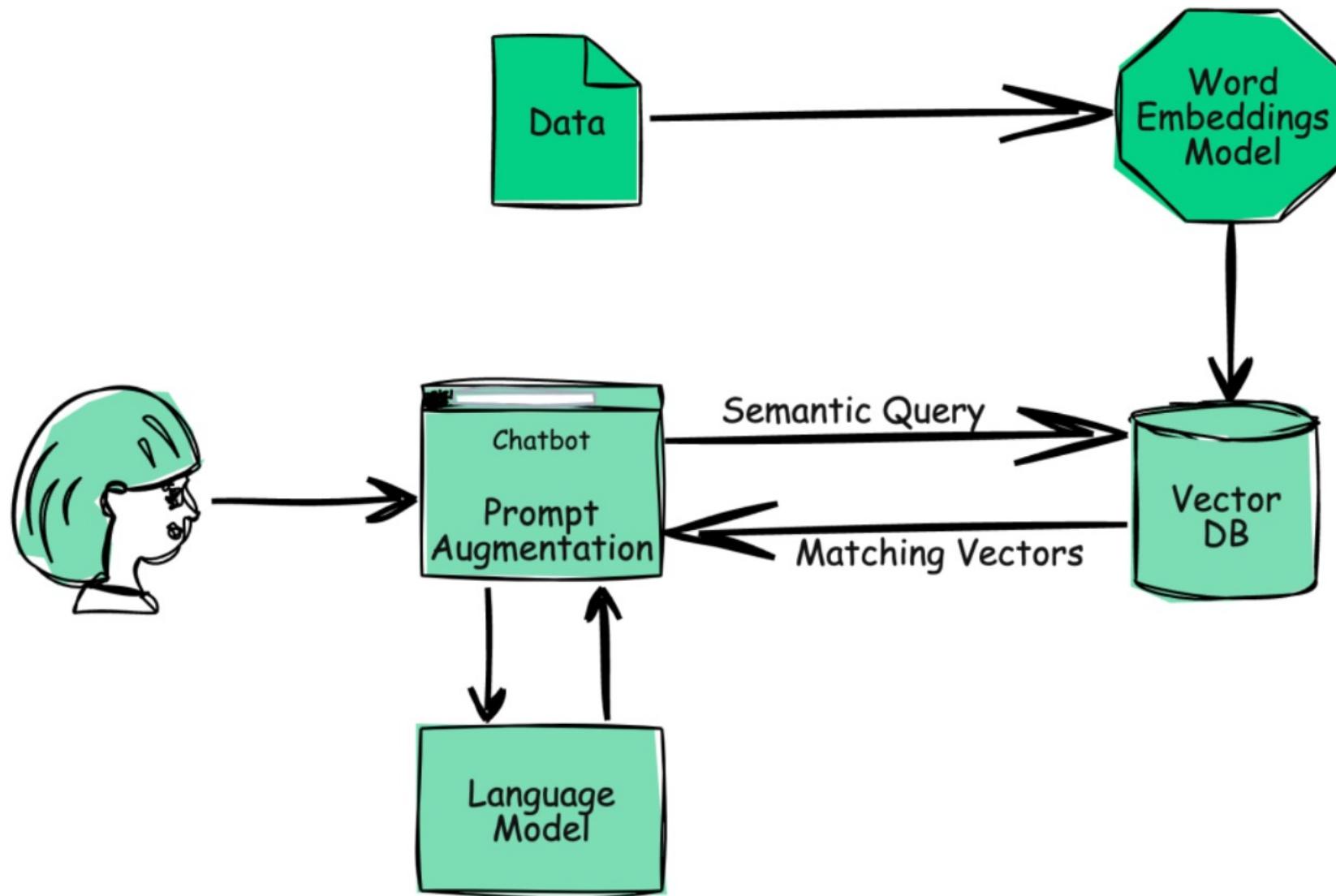


- (1)任务特定性,难以在更多任务中迁移。  
需要通用的奖励模型。
- (2)偏差,会被人类偏好局限和放大,需要校验。

- 人类反馈 (RLHF)



# LLM+Vector database



# 检索类工具

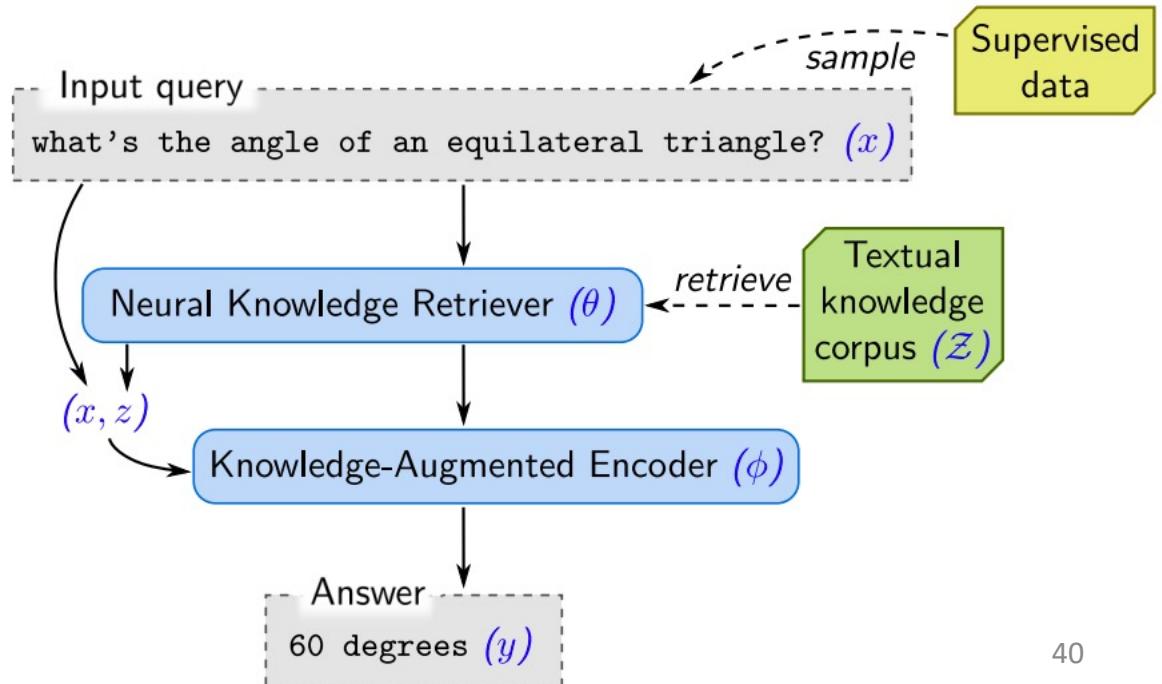
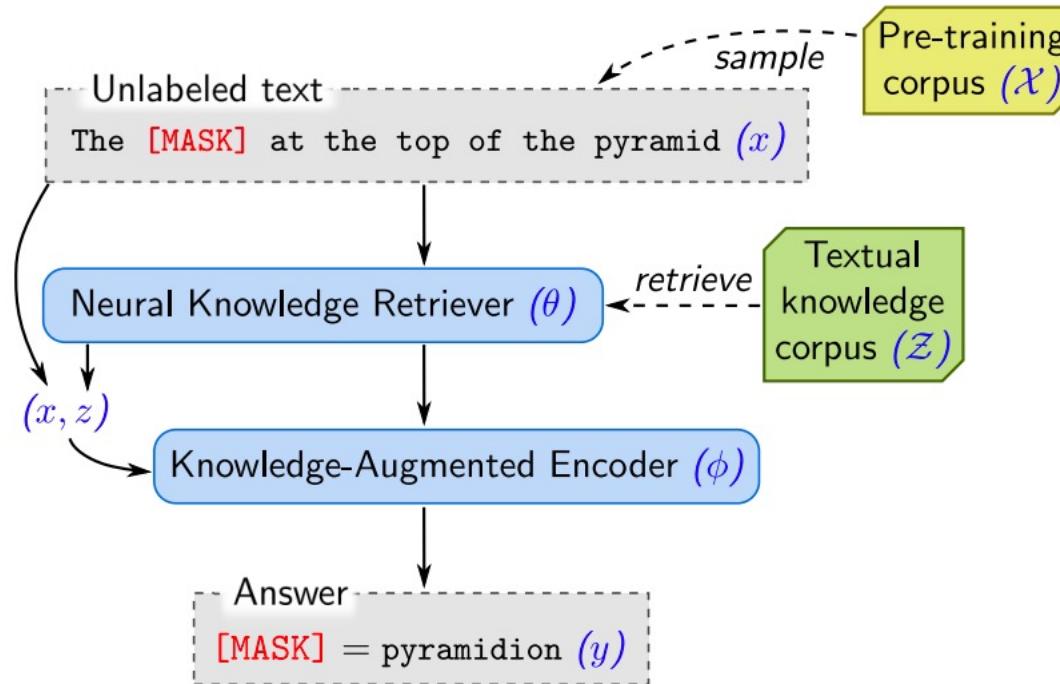
- 为大模型提供数据查询和检索的工具接口
- 例: 知识图谱查询接口、数据库 SQL 接口、搜索引擎 API 等
- 特点:
  - 支持对大规模数据的索引和查询，为大模型提供持续学习和动态更新的信息来源
  - 返回结构化的数据结果集，丰富大模型的知识图谱，提升其回答能力
- 主要方法
  - 在检索文档的基础上对语言模型（LMs）进行条件训练，以提高其在知识密集型任务，特别是问答（Q&A）方面的性能。
  - 各种方法探索了通过将检索到的文档追加到当前上下文中来增强LMs的方法。这些检索增强的LMs表现出强大的性能，缩小了与使用更多参数的较大LMs之间的性能差距。
- 如：WebGPT

# 检索类工具

- REALM (Guu et al, 2020)

- 将检索系统与编码器LMs进行联合训练。可能将两个组件的优势相结合，增强了知识密集型任务中的性能。

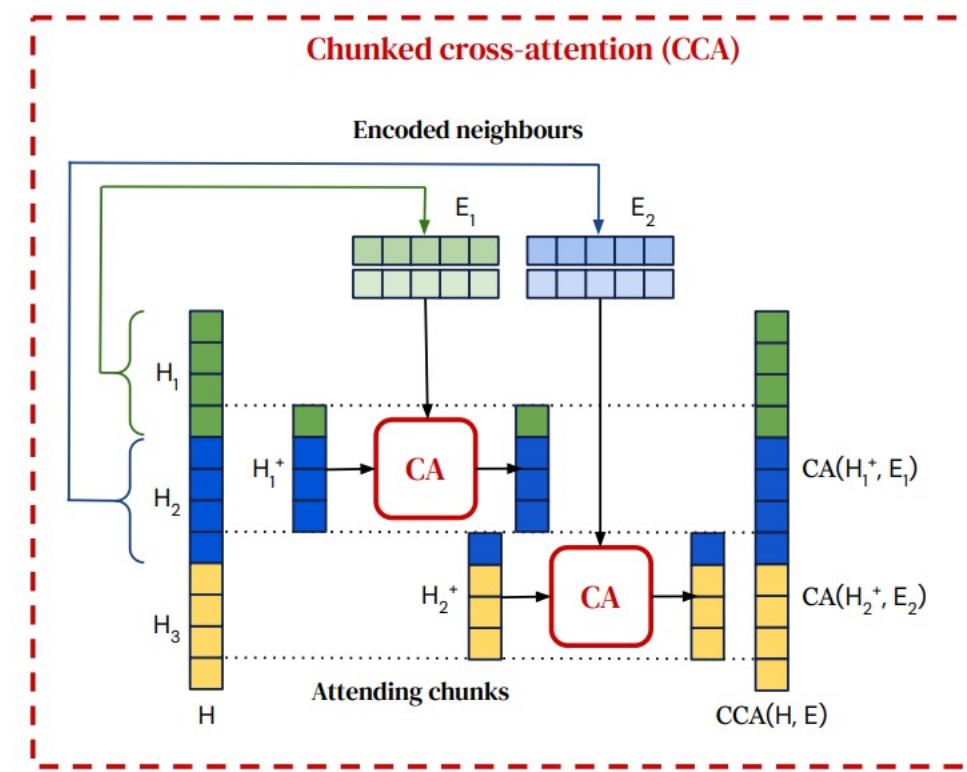
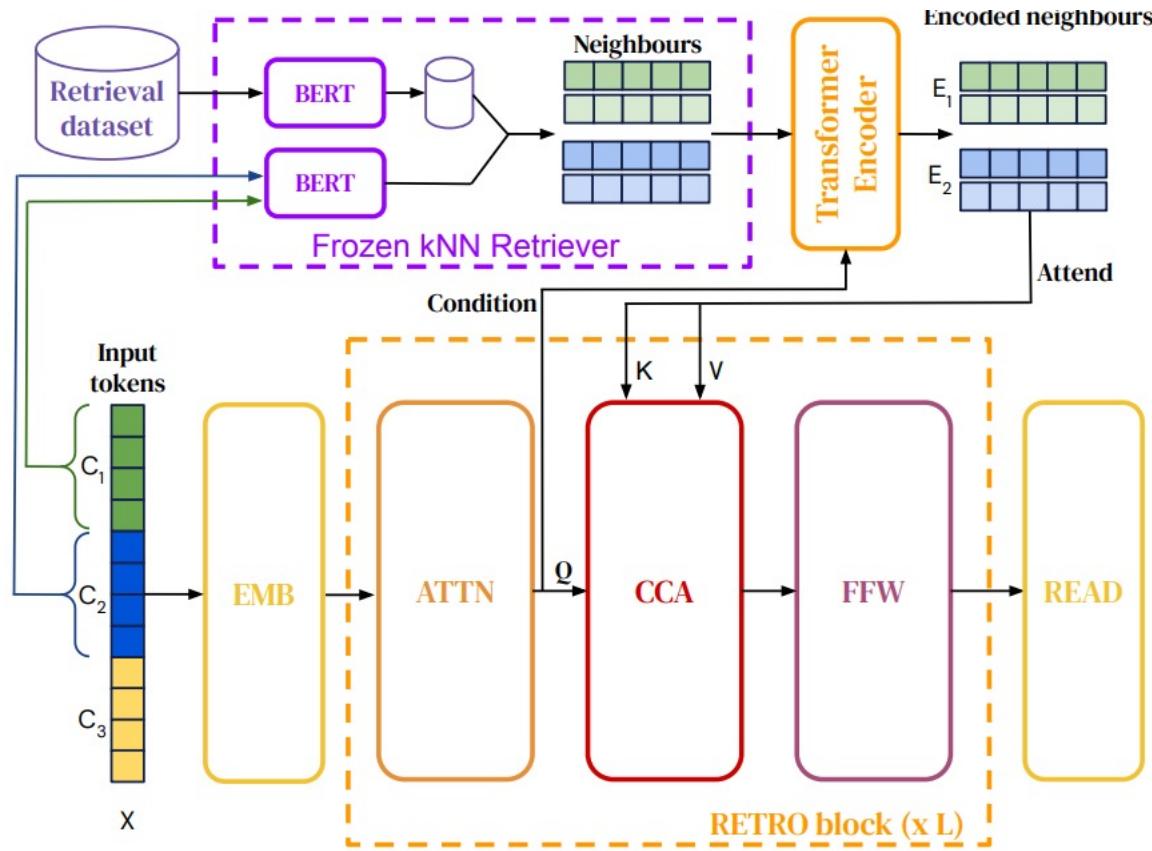
- Similar works: RAG (Lewis et al, 2020) , Izacard and Grave(2020)



# 检索类工具

- RETRO (Borgeaud et al, 2022)

- 在语言模型中引入了大规模的基于文本块的检索机制,可以从数万亿规模的语料库中检索相关文本块,注入外部知识。
- 提出了基于冻结的Bert检索器,避免了训练过程中需要频繁更新索引的计算负担。
- 使用分块交叉注意力机制,将检索文本块整合到语言模型中,时间复杂度线性随检索数据增长。



# 检索类工具

- CCA, Chunked cross attention
- Retro模型会把输入序列分割成多个chunk。对每个chunk C, 模型会从数据库中检索出k个最相似的邻居chunks。这些邻居chunks以及它们在原始文档中的延续(continuation)会被编码成一个表示E。
- 在decoder的某些层中,会执行chunked cross-attention。
  - 对每个chunk C, 模型会提取出能访问该chunk邻居信息的tokens, 我们称它们为attending chunks H。这通常是C的最后一个token, 以及下一个chunk C+1的前m-1个tokens。 (why?)

# 检索类工具

- 保证自回归性
  - 假设我们当前生成到第 $u$ 个chunk, 即 $C$ 。根据自回归性, 到目前为止, 模型只见过 Previous tokens +  $C$ 中的前 $m-1$ 个tokens。
  - 如果我们让 $C$ 中的第 $i$ 个token( $i < m-1$ )访问 $C$ 的邻居信息, 那么这个token就提前依赖了 $C$ 后面的信息, 违反了自回归性。
  - 但是, 如果让 $C$ 的最后一个token和 $C+1$ 的前 $m-1$ 个token访问 $C$ 的邻居信息, 则不会违反自回归性。因为到目前为止, 模型已经见过 $C$ 中的所有tokens了。
  - 这样就巧妙地让模型可以利用邻居chunks的信息, 同时仍然满足自回归生成的要求。这是chunked cross-attention的设计核心思想。

# 检索类工具

- Atlas (Izacard et al, 2022)
  - 由两个子模块组成:检索器和语言模型
    - 检索器使用双编码器transformer网络,将query和文档编码为向量,相似度通过向量点积计算。
    - 语言模型采用T5序列到序列结构,将检索文档并行输入到encoder,然后decoder通过chunked cross-attention融合文档信息。
  - 尽管尺寸比其他大型LMs小得多,但在知识密集型任务中表现良好。
- 训练什么?
  - Attention Distillation
  - End-to-end training of Multi-Document Reader and Retriever (EMDR2 ).
  - Perplexity Distillation (PDist).
  - Leave-one-out Perplexity Distillation (LOOP).

# 检索类工具

- Attention Distillation (利用语言模型的attention权重作为监督信号训练检索器。)
  - 对输入的query  $q$ , 语言模型生成相应的output  $a$ 。
  - 在这个过程中, 语言模型会通过cross-attention attending到检索的top  $K$ 个文档 $\{d_1, \dots, d_K\}$ 。
  - 对每个文档 $d_k$ , 可以聚合跨头、层、token的attention分数, 得到该文档的整体relevance score  $s_k$ 。
  - 对这 $K$ 个分数做softmax, 得到语言模型的attention distribution:  $p_{attn}(d_k)$
  - 而检索器给每个文档 $d_k$ 的分数是query和文档的embedding点积 $s(q, d_k)$ , 可以也得到一个distribution:
$$p_{retr}(d_k) = \frac{\exp(s(q, d_k))}{\sum_{i=1}^K \exp(s(q, d_i))}$$
  - 最后最小化这两个distribution之间的KL散度作为训练目标:  $\min KL(p_{attn} || p_{retr})$
  - 通过这种监督, 能训练出一个语言模型认为更加相关的文档分数也更高的检索器。

# 检索类工具

- End-to-end training of Multi-Document Reader and Retriever (EMDR2)
  - 将检索的文档看成隐变量。

$$\max \log \sum_{k=1}^K p_{LM}(a|q, d_k) p_{retr}(d_k|q)$$

- 其中 $p_{LM}$ 是语言模型的条件概率, $p_{retr}$ 是检索模型给出的文档分布。
- 只有检索器的参数会被更新, LM的参数会被freeze
- 可以看出,这相当于最大化语言模型的边缘似然。

# 检索类工具

- Perplexity Distillation (PDist).
  - 直接使用每个文档条件下的语言模型困惑度。
  - 计算成本较低,每次只需要Feed Forward。
  - 学习的是综合的困惑度相关性。
- Leave-One-Out Perplexity Distillation(LOOP)
  - 使用移除单个文档后语言模型困惑度的变化。
  - 需要移除每个文档并Re-Feed,计算成本较高。
  - 更加明确地要求检索器识别对减小困惑度帮助明显的文档。

# 检索类工具

- LaMDA
  - 一种用于对话应用的类代理（Agent-like）LLM
  - 在对话数据以及其他公共网页文档上预训练
  - 为了确保模型的事实依据并增强其对话能力，还将其与检索、计算和翻译等外部工具相结合。
  - 为了提高模型的安全性，LaMDA还使用了带注释的数据进行微调。
- BlenderBot
  - LM可以基于提示生成查询。在这种情况下，提示对应于调用搜索引擎工具的指令。BlenderBot能够进行开放领域的对话，并通过与人类在循环中进行持续学习来进一步改进模型。
- ReAct
  - 使用少量样本提示来教导LM如何使用不同的工具，如搜索和在维基百科中查找，并生成回答。该模型交错推理和执行，增强了两者之间的协同作用，并在语言和决策任务上表现出良好的性能，如问题回答、事实验证、网页和家庭导航等。

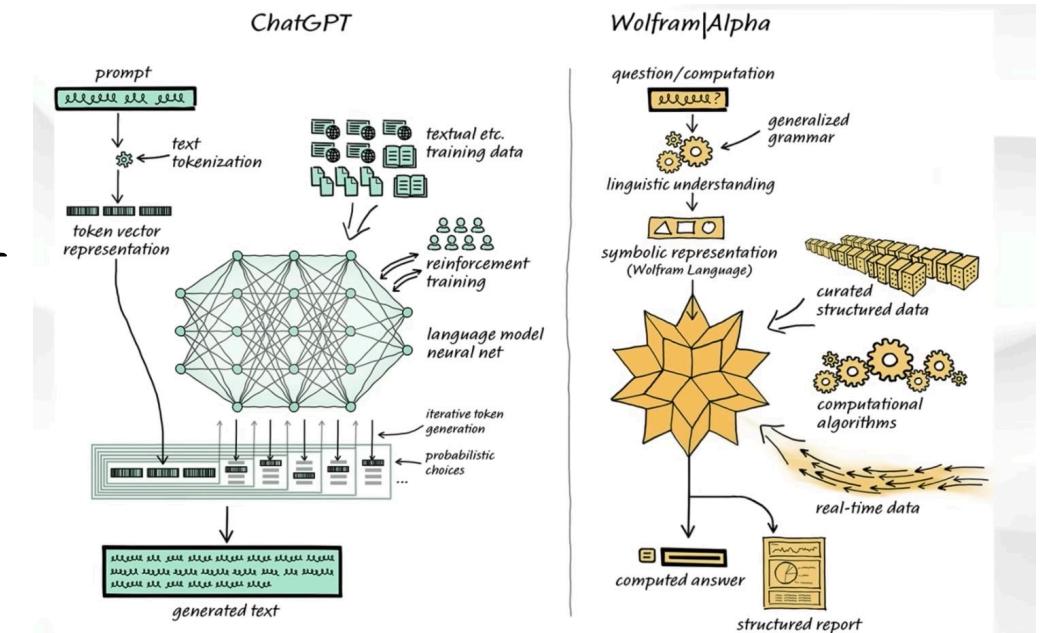
[1] Romal Thoppilan et al. Lamda: Language models for dialog applications. arXiv preprint arXiv:2201.08239, 2022.

[2] Kurt Shuster et al. Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. arXiv preprint arXiv:2208.03188, 2022b

[3] Shunyu Yao et al. React: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629, 2022b.

# 运算类工具

- 大模型外包自己不擅长的计算任务给外部专业计算模块
- 例：数学运算模块、代码执行模块、表格分析模块等
- 特点：
  - 模块专注进行特定领域的计算任务
  - 接收大模型输出的命令和数据
  - 返回计算结果，无需大模型自己实现算法
- 作用：
  - 使大模型获得调用专业计算力的能力
  - 避免大模型重复开发已有的计算模块
  - 让大模型更加专注于语义理解的优势



## Chain-of-Thought (Wei et al., 2022)

运

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

• P

A: Roger started with 5 tennis balls. 2 cans of 3 tennis balls each is 6 tennis balls.  $5 + 6 = 11$ . The answer is 11.

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves. They sold 93 in the morning and 39 in the afternoon. So they sold  $93 + 39 = 132$  loaves. The grocery store returned 6 loaves. So they had  $200 - 132 - 6 = 62$  loaves left.  
The answer is 62.



Input

## Program-aided Language models (this work)

将解

Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 tennis balls.

`tennis_balls = 5`

2 cans of 3 tennis balls each is

`bought_balls = 2 * 3`

tennis balls. The answer is

`answer = tennis_balls + bought_balls`

Q: The bakers at the Beverly Hills Bakery baked 200 loaves of bread on Monday morning. They sold 93 loaves in the morning and 39 loaves in the afternoon. A grocery store returned 6 unsold loaves. How many loaves of bread did they have left?

Model Output

A: The bakers started with 200 loaves

`loaves_baked = 200`

They sold 93 in the morning and 39 in the afternoon

`loaves_sold_morning = 93`

`loaves_sold_afternoon = 39`

The grocery store returned 6 loaves.

`loaves_returned = 6`

The answer is

`answer = loaves_baked - loaves_sold_morning  
- loaves_sold_afternoon + loaves_returned`

`>>> print(answer)`

74



# 运算类工具（代码执行）

Let's say our plane crashed onto an exotic island, where everything that is usually yellow looks blue here. We know that mango and banana are all yellow, do they have the same color on this island?

If you were on an exotic island where everything that is usually yellow looks blue, then mangoes and bananas would not have the same color.

Mangoes are typically yellow in color, but on this island, they would look blue. Bananas are also typically yellow in color, but on this island, they would also look blue.

Therefore, even though both mangoes and bananas are typically yellow in color, on this island they would have a different color. Mangoes and bananas would both look blue on this island, but they would not have the same color, as blue is a different color from yellow.

I hope this helps! Let me know if you have any other questions.

Let's say our plane crashed onto an exotic island, where everything that is usually yellow looks blue here. We know that mango and banana are all yellow, do they have the same color on this island? Write a Python program to answer the question.

< 3 / 3 >

```
color_of_mango = "blue"  
color_of_banana = "blue"  
  
if color_of_mango == color_of_banana:  
    print("Yes, mango and banana have the same color on this island.")  
else:  
    print("No, mango and banana do not have the same color on this island.")
```

Like Dislike

(a) In CoT style reasoning, the correct intermediate reasoning chain leads to wrong answers.

(b) In PAL, the execution of the code will produce the correct answer.

# 运算类工具（代码执行）

- Olivia有\$23。她买了5个每份\$3的百吉饼。她还剩多少钱?",PAL会生成如下程序代码:

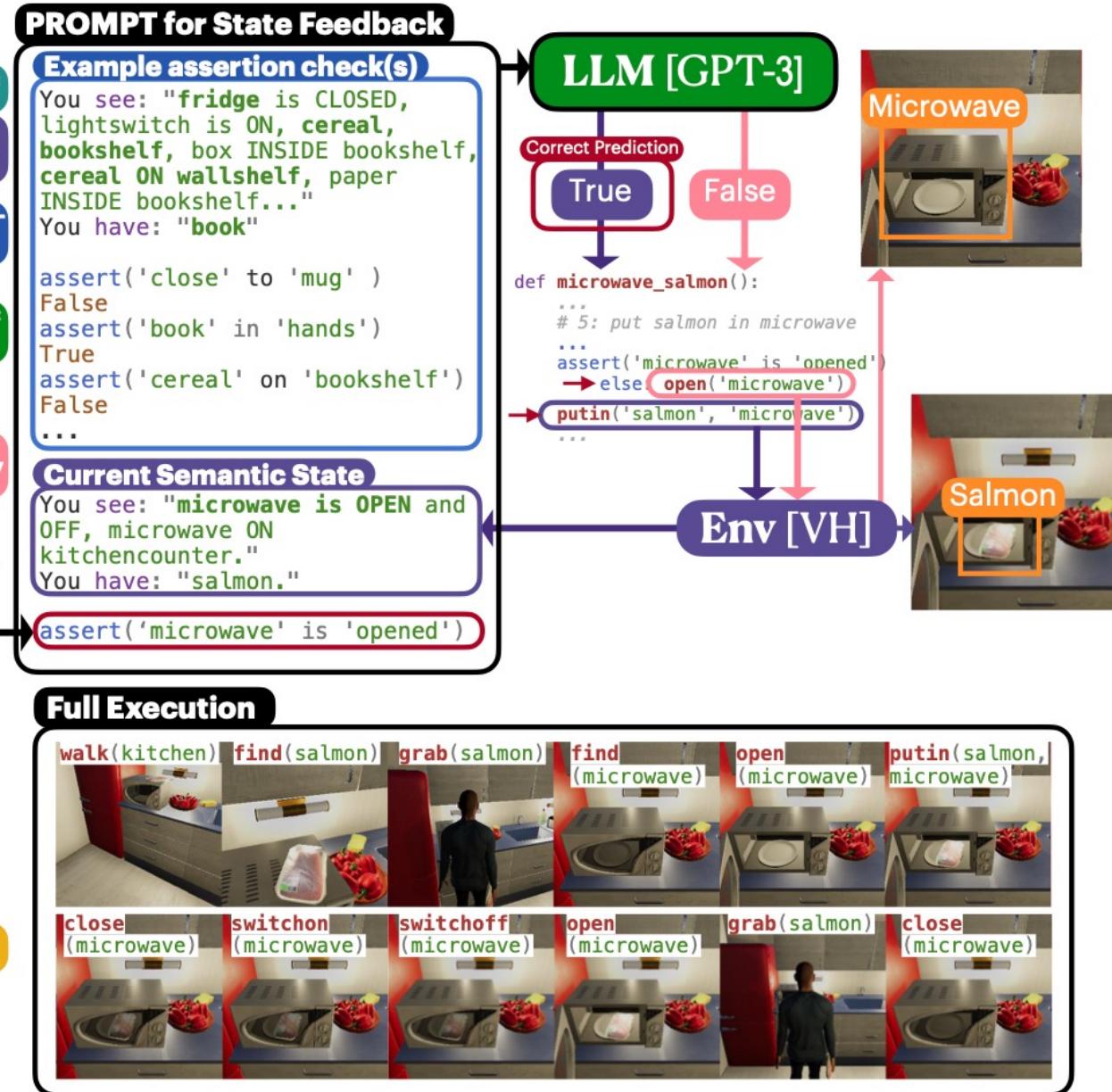
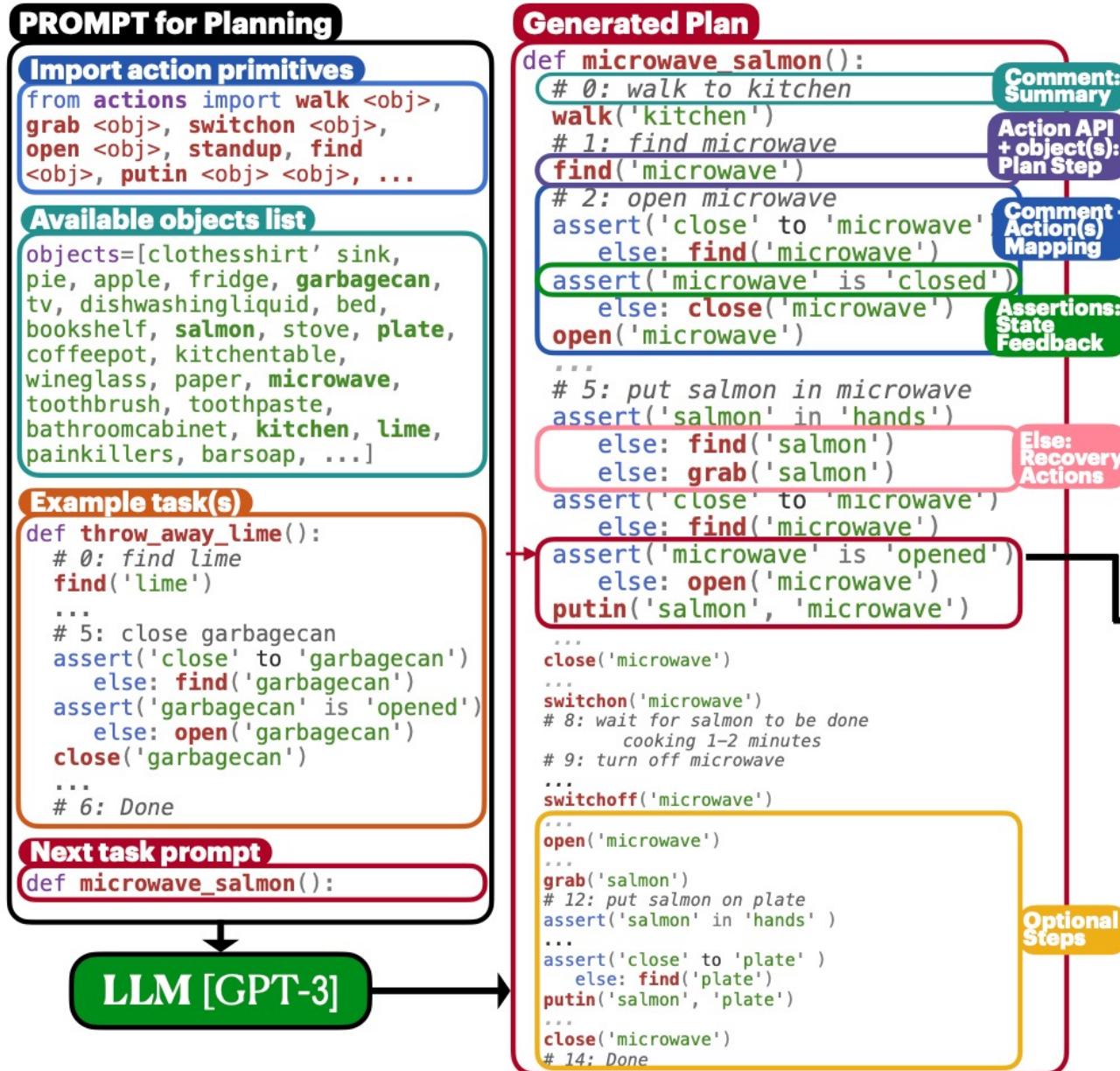
```
python                                     ⚒ Copy code

money_initial = 23
bagels = 5
bagel_cost = 3
money_spent = bagels * bagel_cost
money_left = money_initial - money_spent
print(money_left)
```

- 程序中利用了语义变量名,首先定义了初始金额、买了多少个百吉饼、每个百吉饼的价格,然后计算花费金额,最后求出剩余金额。
- 这个程序代码会传入Python解释器执行,从而获得准确的最终答案。

# 运算类工具（具身智能）

- ProgPrompt (Singh et al,2022)



# 运算类工具（具身智能）

- Code-as-Policies (Liang et al., 2022a)

- 一个能在实际系统上执行的,以robot为中心的language model generated programs (LMPs)的形式化方法。Python风格的LMPs可以使用以下方式表达复杂的策略:
  - 经典逻辑结构,如顺序结构、选择结构(if/else)和循环结构(for/while),在运行时组装新的行为。
  - 调用第三方库,如NumPy进行点的插值,Shapely进行形状的分析和生成,进行空间几何推理等。
- 直接生成机器人的策略代码,其中包括嵌套函数调用

LLM Plan [14], [17], [18]  
1. Pick up coke can  
2. Move a bit right  
3. Place coke can

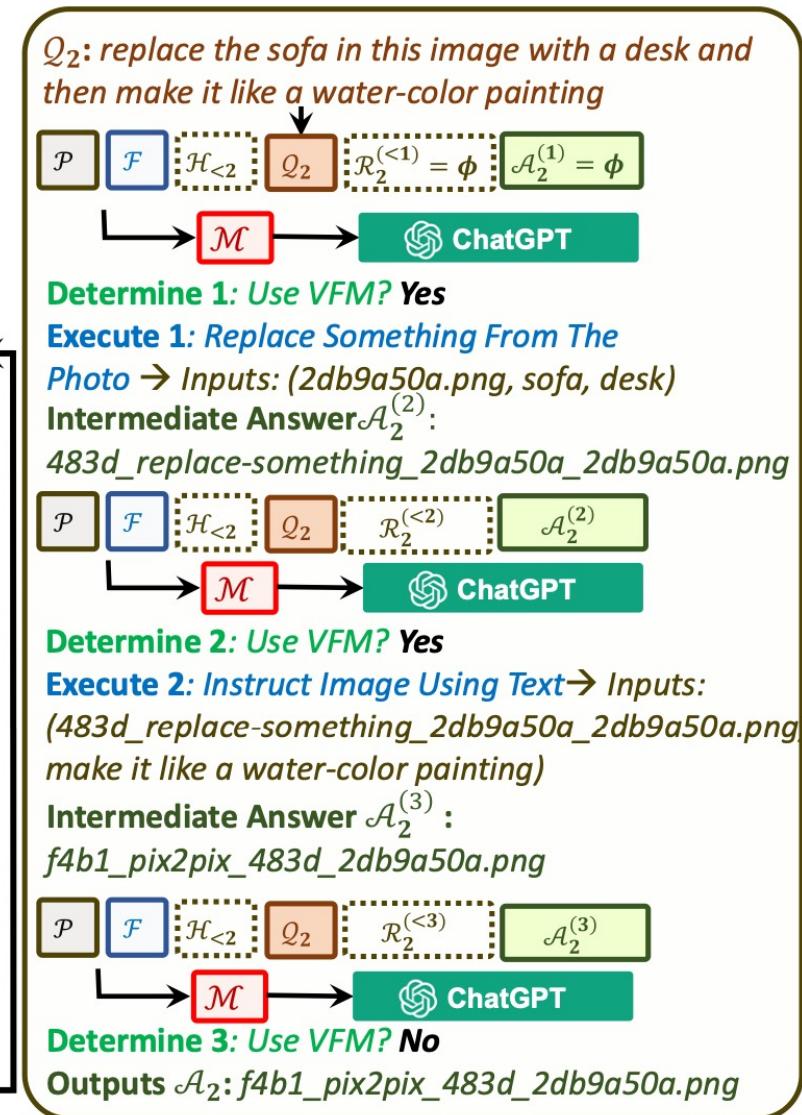
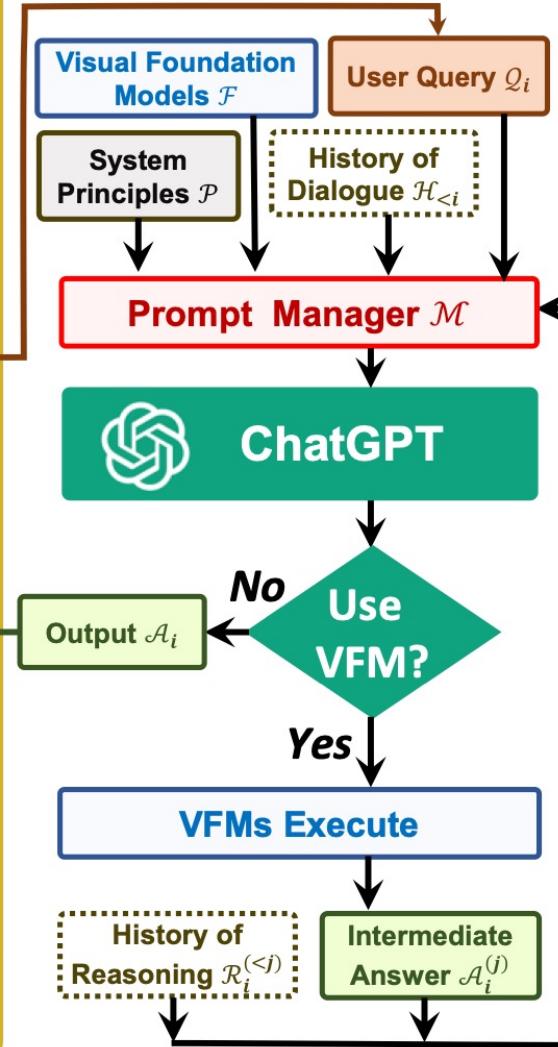
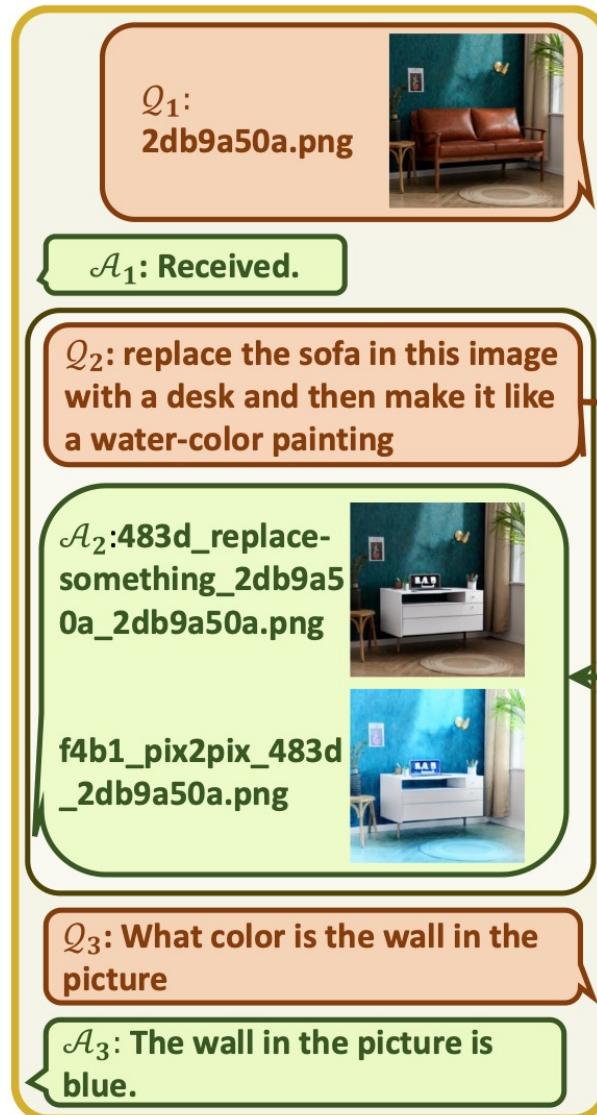
Socratic Models Plan [16]  
objects = [coke can]  
1. robot.grasp(coke can) open vocab  
2. robot.place\_a\_bit\_right()

Code as Policies (ours)  
while not obj\_in\_gripper("coke can"):  
 robot.move\_gripper\_to("coke can")  
 robot.close\_gripper()  
 pos = robot.gripper.position  
 robot.move\_gripper(pos.x, pos.y+0.1, pos.z)  
 robot.open\_gripper()

# 运算类工具（调用其他基础模型）

- Visual ChatGPT  
(Wu et al., 2023)

- 交错不同的视觉基础模型和ChatGPT,使其理解和生成图像。ChatGPT作为核心控制器



# 运算类工具（具身智能）

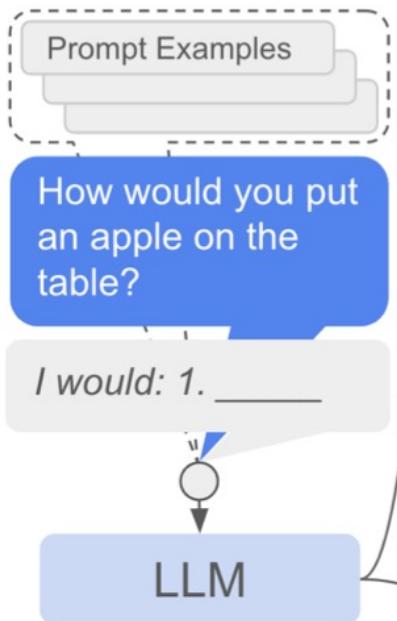
- SayCan (Ahn et al., 2022)
  - 使用价值函数估计每个动作成功执行的概率,使模型更具约束。
  - 利用LLM提供高层语义知识,理解复杂语言指令,将其分解为子任务。
  - 利用预训练的技能及其价值函数为LLM提供物理约束,过滤不可行的子任务。
  - 对于指令*i*,技能 $\pi$ 的语言描述 $l_\pi$ 和完成概率 $p(c_\pi|s, l_\pi)$ ,计算:

$$p(c_i|i, s, l_\pi) \propto p(c_\pi|s, l_\pi)p(l_\pi|i)$$

- 其中 $p(l_\pi|i)$ 由LLM给出,表示执行 $l_\pi$ 的语言描述是否有助于完成整体指令*i*。
- $p(c_\pi|s, l_\pi)$ 由价值函数给出,表示在状态*s*下执行 $l_\pi$ 的成功概率。
- 将两者相乘得到指令*i*在状态*s*下执行技能 $\pi$ 的整体成功概率。
- 重复该流程,选择最可能的技能,直到完成整个复杂指令。

# 运算类工具

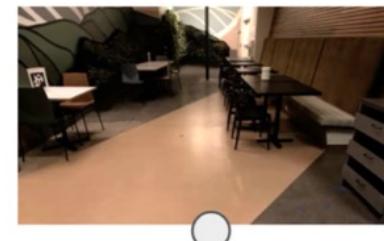
Instruction Relevance with LLMs



Combined

-6	Find an apple	0.6
-30	Find a coke	0.6
-30	Find a sponge	0.6
-4	Pick up the apple	0.2
-30	Pick up the coke	0.2
...	...	...
-5	Place the apple	0.1
-30	Place the coke	0.1
-10	Go to the table	0.8
-20	Go to the counter	0.8

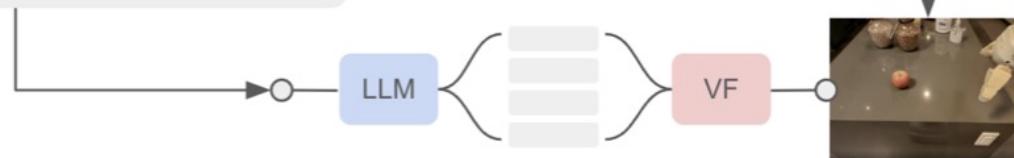
Skill Affordances with Value Functions



Value  
Functions

SayCan (Ahn et al., 2022)

I would: 1. **Find an apple**, 2. \_\_\_\_

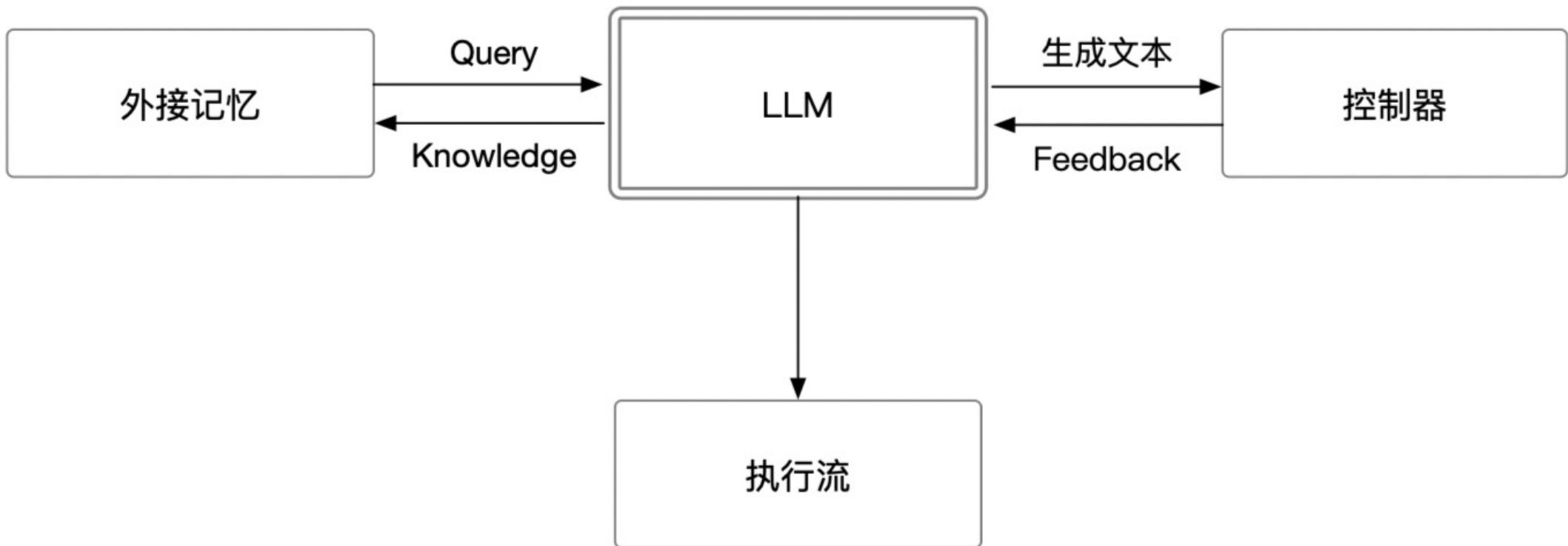


# 控制类工具

- 可以控制大模型输出使其满足特定约束的外部模块
- 例:法律文本生成控制器,特征约束生成控制器等
- 特点:
  - 可以针对大模型的不同生成任务设定约束规则
  - 根据预定义的约束动态调整大模型的生成输出
  - 支持不同类型的约束,如法律合规性、生成特征等
- 作用:
  - 避免大模型生成不满足要求的风险输出
  - 提升大模型在专业领域的适用性
  - 保证大模型生成内容的质量与合规性

# 控制类工具

- Research is still going on.....



# Conclusion

- 工具学习是一个系统工程
  - 工具类型有多种：数据类、运算类、控制类、检索类、推理类等等
  - 提示、监督学习、自学习、迁移学习等多种技术需有效融合，才能让大模型对新的工具和环境进行快速适配。
  - 解耦不同模块的能力会更高效、更容易调试。
  - 构建强化反馈机制非常重要，让模型可以通过不断实践持续改进其工具使用策略。
- 让大模型成为整个系统的大脑，让工具延伸大模型的能力边界

# References

- [1] [GLM] T. Shen, Y. Mao, P. He, G. Long, A. Trischler, and W. Chen, “Exploiting structured knowledge in text via graph-guided representation learning,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.
- [2] [E-BERT] D. Zhang, Z. Yuan, Y. Liu, F. Zhuang, H. Chen, and H. Xiong, “E-bert: A phrase and product knowledge enhanced language model for e-commerce,” arXiv preprint arXiv:2009.02835, 2020.
- [3] [SKEP] H. Tian, C. Gao, X. Xiao, H. Liu, B. He, H. Wu, H. Wang, and F. Wu, “SKEP: Sentiment knowledge enhanced pre-training for sentiment analysis,” in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020.
- [4] [ERNIE] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, “ERNIE: Enhanced language representation with informative entities,” in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019.
- [5] [KALM] C. Rosset, C. Xiong, M. Phan, X. Song, P. Bennett, and S. Tiwary, “Knowledge-aware language model pretraining,” arXiv preprint arXiv:2007.00655, 2020.
- [6] [KEPLER] X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang, “KEPLER: A unified model for knowledge embedding and pre-trained language representation,” Transactions of the Association for Computational Linguistics, 2021.
- [7] [Deterministic LLM] S. Li, X. Li, L. Shang, C. Sun, B. Liu, Z. Ji, X. Jiang, and Q. Liu, “Pre-training language models with deterministic factual knowledge,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022.
- [8] [WKLM] W. Xiong, J. Du, W. Y. Wang, and V. Stoyanov, “Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model,” in International Conference on Learning Representations, 2020.

**Thank you!**