?????

???

1. 要求
    1. 需要设计一个健壮的系统，流程化，体系化，理论支撑
    2. 多层级的架构，层级解耦，独立迭代，不断增强
    3. 尽量端到端
2. 本质上需要的能力和处理流程
    1. 语义解析，对 “输入 ”的自然语言的表达的含义 -> LLM
    2. 计算语义的相似度
        1. 查找已有的通用知识和专有知识
    3. 计算语义逻辑，理解问题
        1. 判断表示同一个语义的不同表达
        2. 逻辑推理、选择
    4. 汇总结果进行输出 -> LLM
3. 需要存储和计算那个层级的信息?
    1. RAG只是向量化存储原文，无计算能力
    2. LLM权重存储和计算自然语言的高层级抽象
    3. 明确的语法 +逻辑推理
        1. 需要人工算法处理类似自然语言，非常复杂
    4. **LLM**映射高层级的语义到向量，只要匹配向量就可以（类似于打标签）
        1. 不可靠，向量只是梯度下降后的最优的结果
        2. 黑盒，没有标准的映射关系， **LLM**变更需要重新生成所有的向量
4. 分几个模块，分别负责什么功能?
    1. 语言能力：自然语言的理解和组织能力： LLM
    2. 信息、知识的格式化，检索：
    3. 信息、知识的合理性检查、扩展、举一反三：
5. 存储的信息的数据结构，存储效率，检索效率
    1. LLM

        权重通过向量空间的转换（类似于查找表）来记录所有的信息，相同的语义会高度复用同一个

        向量空间
        1. 不可控，不透明，不可靠
        2. 存储容量有限
    2. RAG只是记录每个 token的向量空间，只有相同的 token才能被索引
        1. 不能支持高级语义的检索， “不聪明 ”
6. 怎么记忆 /学习新的知识
    1. 通过回归 /拟合 /训练 -- LLM 的中间的 Latent Space
        1. LLM不能在推理过程中不断的改变权重，并且保证收敛
        2. 需要用到所有的历史知识

3. 存储的知识量越大，学习越困难
2. 通过 “绝对 ”信息直接分类和记录
1. 需要复杂的人工算法
2. 需要计算和存储语义的绝对信息

7. 分割，建树，识别匹配， 心智 系统 2 意识自洽， 语义计算
8. 存储的知识可以被提前按照信息块通过模型的推理进行向量化，生成 kv cache
9. 向量化的信息快，可以直接拼接到上下文的 KV cache里面，而不用重新计数
1. 插入的信息块和前面的文本没有关系
2. 插入的信息快只和自己内部有关系

AI???????

1. 记忆，为 LLM语言计算器提供海量的数据
2. 离线思考，不断得自由思考，编辑记忆，实现可迭代的，可不断进步的 AI
3. 临时记忆，提供无限的上下文，用于解决复杂问题，意识的存在实体
4. 计算记忆，一次思考的中间过程记忆

?????????????

1. 检索：搜索相关语义的信息
2. 检索相关理念 /想法 /idea的信息
1. 寻找关于，关于大小脑、慢思考 /快思考、第一系统 /第二系统 的概念
2. 寻找之前关于 “结构化表示信息的方法，利用 markdown的层级结构 ”的描述信息
3. 寻找 “transformer缺点 ”
3. 根据相关的理念 /想法进行科研，主动的推理，输出思路和引用，自洽性判断，示例如下
1. 提问：利用 xxx 的概念，应用在 xxx 领域，会有什么效果?
2. 理解信息，结构化信息，根据逻辑推理得到新的结论，产生新的概念
3. 询问当前 AI的问题有哪些? 能检索返回 "
没有反思能力，一个任务，不能越做越顺利，不断改进流程 "
4. 看到文章 “如何科学地给大模型「找茬」? Anthropic联合 Thinking Machines
发布新研究，通过 30万个场景设计和极限压力测试，扒了扒 OpenAI、谷歌、马斯克家 AI
的「人设」。那谁是老好人？谁是效率狂魔? ”能知道，这个是在讲我之前研究过的一个话题《
LLM心理行为学的研究》
5. 请根据当前资料总结出当前 **AI**发展的关键瓶颈
6. 英伟达巧用 **8B**模型秒掉 **GPT-5**，对应到，开发一个专用的小模型提供特殊领域的专家
7. 有哪些文件是在研究我之前提到过的 “LLM心理行为学 ”
8. LLM和 AI可能在大部分场景下会混用，需要能互相检索
9. 能不能帮我总结下，这个礼拜有哪些比较新的研究成果?
10. “maybe know类型的问题是最有用的 finetune数据， 强制给正确的答案做微调，精度最差 ”
能自动看论文，汇总出这样的结论
4. Embedding还是不够理想
1. 句 A表达的意思 >句 B，在检索 B的时候， A的分数不高
2. 主体不清楚：特斯拉前深度学习负责人 Karpathy，公开表示 AGI仍有十年之遥
1. “Karpathy+AGI仍有十年 ” 才是重点
2. “特斯拉前深度学习负责人 Karpathy” 不是重点，不能作为匹配信息依据
3. 卡帕西认为，通用人工智能（ AGI）的实现至少还需要十年时间

4. 同义词语　　还是不理想：后期的知识　　＋中英文　　＋表达方式
    1. 卡帕西认为，通用人工智能的实现至少还需要十年时间
    2. Karpathy公开表示，　　AGI仍有很多年
5. 表达方式的不一致
    1. 反问句
    2. A+B = B+A
5. 一句话可能表达两个或者　　多个主要的意思
6. 支持动态学习，或者说应该支持链式搜索，逻辑处理
    1. 额外增加信息：卡帕西　　==Karpathy

??

1. 一种不断进行推导的知识检索路径规划，利用已有信息进行推导。　　树只是一种高效的简化结构。
2. 扩散 匹配到可能的选项之后，再进行语义精确过滤，　　a>b b>a
   的区分，可以来回多次，类似关键词下钻的原理
3. 记忆的三个层级：逻辑推理，潜意识推理，扩散
4. 语言只是（概念　　/
   思想）的一种表达的方式，一种通过描述，不断得修饰等方法进行表示、表达知识的一种方式
    1. 语言不是知识或者信息本身，可以转换成其他的句式来表达同样的意思
    2. 直接的陈述句，固定格式的陈述句等等
5. 先让一个　　LLM 把原始信息扩展成多个变体在存入，入库时让　　LLM 给每篇文章生成一段抽象摘要
6. 人类记忆的大部分概念，话题，大部分都是跟旧知识有关系的
7. 人类只能记住解析之后的，自己理解的知识，很难记住原话。原话大部分只是暂时放到临时记忆，或
   者就是直接放到无意识区的记忆，这部分属于潜意识，不经常用的信息就会想不起来。

????

1. 快速得在大量的数据里面检索出来需要的信息
2. 大范围：存储大量的信息
3. 支持复杂逻辑、动态的检索
    1. 如果　xx存在，则输出是　　xxx,同时不是　　xxx,可能是　　xx的 xxx
    2. 指代消解
4. 支持复杂的编辑
    1. 快速动态增加一个信息：　　Kapaskey==卡帕斯基

?????

1. 一颗由命题组成的图
    1. 不断进行推理和完善命题的所有逻辑关系和知识点
    2. 非常完善的组织方式，非常方便进行检索和逻辑判断和增加
    3. 输入的信息只是刺激　　/材料做的是转换　　——识别原文里承载的认知信息
    4. 不是「切分」原文。　　原文只是刺激　　/材料。我们做的是转换　　——
       识别原文里承载的认知信息，重新编码成我们事先定义的标准形式。原文的句子
       边界、修辞、语序，对最终入库的形式没有任何约束力。

5. 提一个更高维的视角：认知原语 (Cognitive Primitives)，不是语言原语

2. 第一级原始的信息检索方法
   1. 使用关键词依赖关系的图谱方式
   2. 使用总结的方式进行索引？效果类似于 RAG？
   3. 用于 LLM不断得信息检索
3. 命题
4. 入库的时候整理、建图 --建图很慢、工作量大
   1. 需要对所有已经存在的命题进行交叉判断，如果有相关就建立连接
   2. 如果建立新的命题，就要和所有已经存在的知识进行判断
   1. 大脑做到的自洽，虽然不是实时实现的，可能是在睡觉的时候进行整理
   2. 命题举例： AGI实现的日期是什么时候？

token??????

1. 示例 1原文 :NAND Flash 的读写和擦除操作是基于串进行的。
2. 示例 1输出 :[[NAND Flash] 的 [[读写]和 [擦除]][操作]]是 [[基于][串][进行]的]。]
3. 示例 2原文 :在读取操作时，通过在位线上施加电压，检测各晶体管的导通情况来确定存储的数据。
4. 示例 2输出 :[[[在 [读取][操作]]时]， [通过][[在 [位线][上]][[施加][电压]]]， [[[检测]][[[各][晶体管]][的][[导通][情况]]]][来][[确定][[存储]的 [数据]]]]。]
5. 示例 3原文 :在写入操作时，利用量子隧穿效应等将电子注入浮栅。
6. 示例 3输出 :[[在 [[写入][操作]]时]， [[[[利用]][[[量子][[隧穿][效应]]]等]][[将][电子][[注入][浮栅]]]]。]
7. 示例 4原文 :在擦除操作时，通常是对整个块进行擦除，块是由多个串联的晶体管串组成的。
8. 示例 4输出 :[[在 [擦除][操作]时]， [[通常][是][[对][[整个]块][进行][擦除]]]， [块][是][[由][[[多个][[串联][的][晶体管]]]][[串]组成的]]]]。请按照上面的格式进行格式化 “希望这些步骤能帮助您顺利解决 python3-dev的安装问题！如果尝试后仍有疑问，欢迎随时提出。 ”

1. token之间的关系树 + token的编码 LLM
2. 计算特征值？？ 句子 段落 文章
   1. 是不是不用计算句子的所有细节，而只是计算大的语义和总结
   2. 检索的 “线索 ”不可能包含跨越多层的语义，也有信息量限制
   3. 句子之所以是句子，是人类处理信息的最大单元，一次性能够理解的单元。每个句子就是 **markdown**树的一条 **item**，按照层级，不断总结，形成一个大的句子树。每个句子表示一个独立的语义。
   4. 以句子为节点组成的金字塔结构和 **meaning tree** 的结构，非常像，用这样的结构去训练模型，相当于说对语言的理解强制约束为树状结构或者是金字塔结构，相对于传统的序列编码来说，金字塔结构更符合实际的语言含义，会强制模型去进行高层次的语义表达
   5. 所有的句子作为节点节点是可以用向量进行编码，并且能进行距离计算具有全局唯一性，寓意相近的句子距离也相近。
   6. 每个观点都可以用金字塔的方式来补全所有的方面，但是用户不一定会想到或者写全，**AI**可以进行自动的补充

        1. [illegible] [illegible]
        2. [illegible]
3. [illegible] --[illegible]
    1. [illegible] LLM[illegible]
    2. [illegible] item[illegible]
    3. [illegible]
4. [illegible] LLM[illegible] KVCache[illegible]
    1. [illegible]
5. [illegible]
    1. [illegible]
6. [illegible]
    1. [illegible] /[illegible] /[illegible]

??????

1. [illegible]
    1. LLM[illegible]
2. [illegible]
    1. [illegible] [illegible] **SSM**[illegible]
        1. [illegible] LLM[illegible]
        2. 2[illegible] target
    2. [illegible]
        1. https://www.modelscope.cn/models/jinaai/jina-embeddings-v2-base-zh/summary
        2. [illegible]
        3. [illegible] **Qwen/Qwen3-Embedding-4B**[illegible]
            1. [illegible]
            2. [illegible]
            3. [illegible] [illegible]
            4. [illegible] [illegible] **top**[illegible]
               [illegible]
    3. [illegible] [illegible] [illegible] kv cache[illegible]
        1. kv cache[illegible]
3. [illegible]
    1. [illegible]
    2. [illegible]
    3. [illegible] LLM Agent/MCP[illegible]
4. [illegible]
    1. LLM[illegible]
    2. [illegible]
    3. [illegible]
5. [illegible] token
    1. [illegible]

2. 部署、量化上的提升

??????????

1. 把单个 token/meaning的向量根据固定的语法结构进行直接的拼装后成为一个更大的向量
   1. 使用固定的权重 mapping到空间来映射/表达关系
   2. 这两个向量通过一个固定的模型进行运算，运算后得到一个新的向量
   3. 不同的语法结构使用不同的权重进行计算，比如 A是 B， A和 B， A的 B， A与 B， A或 B或 C
   4. A->vectorA B->vectorB A和 B->vectorA&B
2. 对 **meaning tree**的每个 **node**做 **embedding**
   1. 一句完整的句子，需要按照 meaning的不同层级的拆分分别做 embedding， embedding 做针对性的训练
   2. meaning需要拆分出来具体的逻辑语义、语法规则
   3. 按照 meaning拆封后，分别进行不同组合的结果来训练 embedding， A和 B==B和 A
   4. 所有进行 embedding的 meaning都是一个明确的、最简单的语句
   5. 如果有额外的信息输入，比如 卡帕司 =kaparskey
      1. 需要把所有用到 kaparskey 和卡帕司 的 meaning tree，的所有 embedding进行 embedding的补充
      2. 一个 meaning可能有多个 embedding进行等价表示
   6. 使用翻译的方法，让 **AI**生成语义一致的训练样本
      1. 预先对等价单词进行替换 (小块 token 进行 embed 检索，替换，生成整个句子) 再 emb ，检索
      2. 常识可以持续微调进大模型，比如 人工智能 =AGI ，持续改进 embed 模型，表达真实的语义编码
      3. 当前的 emb还是很注重单词的一致，匹配，不能表示真正的语义接近

??LLM???????

1. 通过 LLM对输入的句子进行结构化: 短语 +关系 **SVO** 语义矢量双子手册
   1. 拆分为 单元之间的作用关系，支持嵌套组合
2. LLM -> AST语法树 -> 高级语义编码（序列编码，不定长的语义编码） -> 距离计算
   1. 编码信息：语义之间的 “与或非 ”操作
   2. 语法树每个节点表示一种操作
   3. LLM有很大的编码空间，多层、不定长、位置编码，各种语义交叉压缩表示
   4. 递归式的编码高级语义
      1. 递归，节省编码量
      2. 编码，为了表示和检索
   5. 链式的不定长的编码
      1. 距离计算，类似图形匹配

??2

1. 基于 LLM的专用 meaning处理大模型：小规模，缓慢更新迭代
    1. 功能
        1. 处理固定的已知的有限的逻辑，生成 meaning，形成 meaning tree
            1. 逻辑关系类
            2. 已知的名词、事物
        2. 支持生成和识别已知的 meaning的组合
        3. 输出和输出特殊的高效的 meaning token： meaning tree
        4. 使用 AI算法，对 meaning tree 进行处理、总结
            1. 分析形成高级抽象 meaning，存储与数据库
                1. 通过低层级的通用语义（单词）的逻辑组合，不断组合成高层级语义
                2. 形成由逻辑关系不断组合形成的 “逻辑树 ”，存储与数据库
                3. 可以先总结底层（短语）级别，逐渐形成更高级的逻辑树
            2. 数据库 +传统算法 +AI处理数据
                1. 由 LLM生成待查询的逻辑需求
                2. 数据库按照固定的逻辑进行查询输出
                3. LLM不断一步步进行 “查询 ”“处理 ”直到不需要查询更多信息
    2. 实现?
        1. 使用更大的 LLM进行训练数据生成：文本 ->meaning token的数据集
2. 结合 meaning tree+原文，推理 meaning，生成 prompt
3. 数据库记录 meaning和原文，以及之间的关系：不断更新数据
4. 公用的 LLM根据 prompt组织生成自然语言：通用的大规模 LLM

??3

??

1. LLM只是一个语言计算器，数据库 +逻辑抽象 +人工调度算法
    1. 当前的 LLM使用大量的权重完全包含和模拟了整个存储和检索，但是容量有限
    2. 人工算法 +数据库不能表达复杂的自然语言推理
2. 类似于把自然语言用传统算法进行表达 “计算 ”
    1. 传统算法模拟大脑的行为，好像不太合理
    2. LLM推理一次等价于大脑思考一次
    3. 每个样本都会存储一个树，经过不断的推理，不断得往高层级的语义合并，引用，减少存储规模
    4. 自动遗忘
3. 怎么复用（链接）相同语义到相同的表示 token？ xx是 xx、因为所以、 A是 B， B是 C， A是 C
4. 人工算法和数据库是不是表示所有的自然语言的关系？？

??

1. 语句 -> LLM生成单词树 -> 单词树查询数据库 -> 单词树匹配 /推理 -> 重新组织语言
2. 语句 -> LLM总结 -> 单词树（ meaning tree）

3. 单词树节点之间的关系的学习:
    1. 不断遍历可能的节点之间的逻辑关系并记录到数据库
        1. xAx == x被 xA
        2. x1 and x2 and x3 == x2 and x3 and x1
        3. 只有抽象高度通用的语义才能穷举
    2. 通过关系树表达，形成一个完全符合自然语言的向量数据库
        1. 明文，非黑盒，可以单点修改
        2. 无需大量计算，可以通过简单的算法进行检索
        3. 和 LLM的区别是，不能进行复杂的逻辑推理和自然语言的组织
4. 正向: 单词树 -> prompt -> LLM -> 单词树的关系 -> 关系数据库
    1. 根据已经有的单词树数据库生成 prompt
    2. LLM进行自然语言推理
    3. 更新单词树节点之间的关系
    4. 不断循环，从低节点开始建立关系，逐步向上总结到高层级
5. 逆向: LLM 生成样本 -> 单词树 -> 关系计算 -> 关系数据库
    1. 使用 LLM生成的同义句
    2. 根据已知的逻辑关系，更新数据库

??4

1. LLM 格式化输入的高级语义到向量空间并存储: seq -> LLM encoder -> vector -> DB
2. 检索距离接近的 vector的 seq，重新组织语言: seq -> LLM encoder -> vector -> DB -> seq -> LLM -> seq
3. 对文本（语义）进行树 tree建立 + 符号分类（向量 V） + 打标签（向量 V）
    1. 模型里面的向量本身就能线性的表示两个语义的相似度，距离
        1. 前提是对已知的信息已经经过大量的训练（分割）
    2. tree
        ，各个层级的语义之间的关系信息，语义由低级别的语义进行的组合，通过组合关系可以确定真实含义
        1. 通过大 LLM生成
    3. 语义空间的多维向量 V（标签）：
        1. 表示两个语义之间的关系和距离
        2. 通过专用模型可以直接计算出两个输入（树 +向量）的关系向量
        3. LLM不能直接对未知的知识进行分类和输出表示向量
            1. 但是可以计算出两个语义的距离
            2. 难以表示绝对值，可以计算相对值
4. 不断得对一整篇文件进行总结，并进行映射输出到单个向量
5. 问题
    1. V表达的空间够不够，需要多大的维度信息才能表达所有的语义
    2. V的分类性能依赖已知数据的训练，不能对新知识进行很好的分类
        1. 通过语言的自然理解，可以进行部分正确的分类
        2. 通过和低级别语义的关系进行推导出语义空间向量

Revision #105
Created 2025-08-30 06:57:13 UTC by Colin
Updated 2026-05-09 15:15:54 UTC by Colin