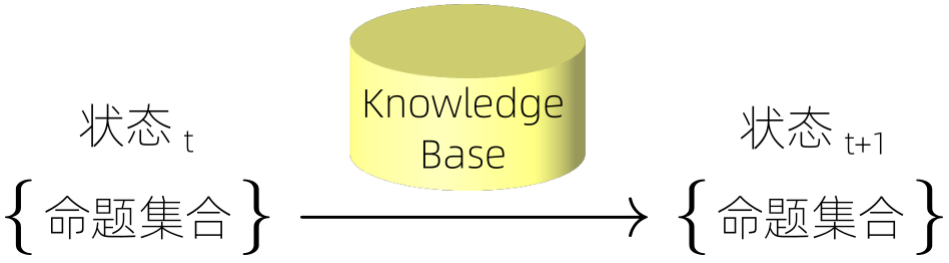
①

# 逻辑与深度学习的关系

这是经典逻辑 AI 的最基本运作模式：




(1)

它其实包含了两个算法：

- matching (unification):
  逻辑 rules 是包含变量的条件命题，
  例如　　$\forall x.\text{是人}(x) \Rightarrow \text{会死}(x).$
  Unification 判定一条 rule 是否可以 apply 到某逻辑命题上，
  例如：$\text{是人}(\text{苏格拉底})$ 可以跟上式的左边 unify.
  Matching 的结果是得到一推 instantiated（特例化，即不包含变量）的命题。
- forward- or backward-chaining (resolution):
  由已知事实 推导出新结论，或反过来，判断某给定的新结论是否成立。
  例如：$\text{是人}(\text{苏格拉底}) \Rightarrow \text{会死}(\text{苏格拉底}) \ \wedge \text{是人}(\text{苏格拉底})$
  可以推出：$\text{会死}(\text{苏格拉底})$。

深度学习的特点，就是将

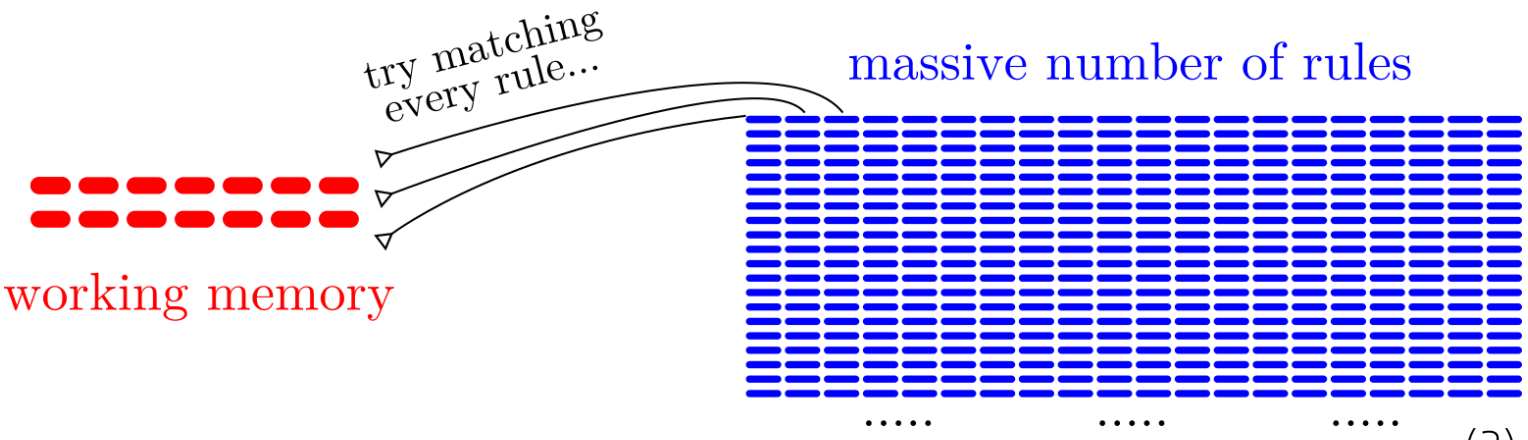
$$\text{状态}_t \vdash \text{状态}_{t+1} \tag{2}$$

的逻辑推导过程，通通纳入进去一个非常复杂的非线性函数（= 深度神经网络）里面。这样做以后，上述的逻辑结构被 “mingled” 在一起，以至于很难分辨了。但也正是由于这种「大杂烩」，深度神经网络 将一套复杂的组合算法压缩成数量不算太多的一层层的参数。它同时可以做 learning 和 inference 这两个动作。这种简单粗暴的方法，其实非常有效率，要超越它的速度并不容易！

我们知道（或推测）一个智能系统 应该具有 符号逻辑的结构。这点知识可不可以用来 约束/加速 深度神经网络？答案似乎是有可能的。现时 state-of-the-art 处理 视觉的 CNN 和 处理文字的 BERT，它们都有内部结构，**而不是 fully-connected**，而且 这内部结构 对应于 被处理的资料的结构。因此我们有理由相信，逻辑结构 可以用来约束 深度神经网络的结构，达到加速。

接下来我们详细一点看逻辑系统的结构：

Knowledge Base 里面有很多 rules，系统要将这些 rules 逐一 match with 系统状态 (= working memory) 里面的命题：



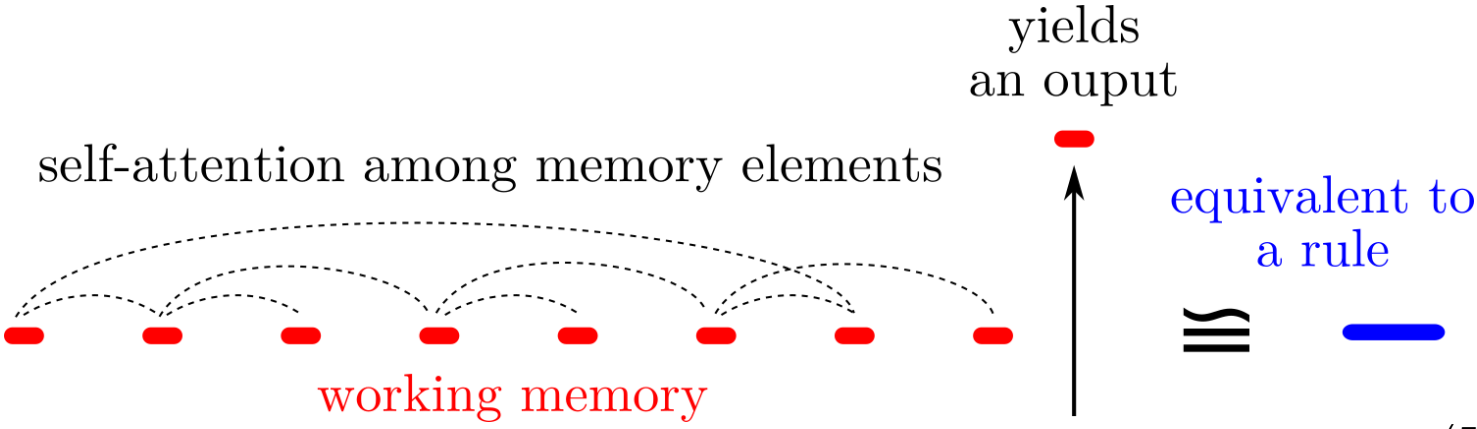

(3)

成功 matched 的 rules 可以导出新的结论，加进 working memory 的状态 里面。

这个复杂的操作，完全被一个神经网络取代。或者可以更抽象地说：

$$\text{状态}_t \longrightarrow \boxed{\text{？某种记忆？}} \longrightarrow \text{状态}_{t+1} \tag{4}$$

而以 Transformer 来说，它是一种 输入元 **之间** 的记忆体（这记忆就储存在 Q, K, V 矩阵里），而它 implicitly 做到了 rules 的作用：




(5)

换句话说，Transformer 内部有某种（扭曲了的）逻辑 rules 的结构。那么很自然的问题就是：能否发掘更多 逻辑 / 逻辑系统 的结构？也就是说，公式 (4) 可以有怎样的代数结构约束？这个问题 可以参考 范畴逻辑 的理论，还有 经典 logic-based AI 系统的理论。

我们希望 勾画出公式 (4) 需要具备的代数约束，但暂时先用文字描述比较容易：

- 状态是 **颗粒化** 的，它是某集合的元素，元素之间可交换，也就是 Transformer 的 equivariance. （注意：Transformer 有 equivariance，但 equivariance 未必一定要用 Transformer 实现）
- **深度结构**：例如多层网络，每层是函数的复合 (composition). Transformer 也用了深度结构。
- 逻辑 包括了 **命题**层次 和 **命题内部**层次 的 颗粒化。后者是**谓词** (predicate) 逻辑的结构，例如：*loves( John, Mary )*，也可以简单地将它视为**代数元**之间的**乘积**，例如：*John • loves • Mary*, 后者也叫做 "word". （不同类别的代数元之间不一定容许乘积，因此有 groupoid 的概念，但暂时来说这细节不重要。）现时重点是如何将 这两层的 颗粒化 结构 施加到深度神经网络上。
- 逻辑推导 每步只产生**一个**新的结论（或其概率分布），然后这个新的结论，再加入到旧的状态中，作为一个命题集合的元素，而旧状态也要 **遗忘** 一些命题，否则需要无限记忆。这跟 Transformer 每次输出**一列**的 tokens 有点不同（虽然我们不太肯定 Transformer tokens 究竟对应于命题 还是 谓词 / 原子概念）。
- 逻辑 rule 通常只跟某几个前提有关，其它前提是**无关**的，例如：*眼睛好看 ∧ 鼻子好看 ∧ 嘴巴好看 ⇒ 帅*，跟 *有钱* 或 *穷* 无关。Transformer 的 **softmax** 结构似乎也可以排除一些无关的 tokens 的影响。
- (可能还有其他的结构特征.....)


4