

Reverse-o1:OpenAI o1原理逆向工程图解



张俊林 
2023 年度新知答主

已关注

 编辑推荐

段小草、左鹏飞 等 1074 人赞同了该文章

目录

收起

OpenAI o1的重要意义

OpenAI o1的完整训练过程推演

o1应由多个模型构成

OpenAI O1可能采用的训练数据

人工标注数据

合成数据

代码COT数据的反向生成

数学COT的反向生成

Reverse-o1：RL的关键要素及...

O1中RL的状态空间：Token...

O1中RL的可能行为空间：“...

O1中RL模型的奖励模型（Re...

AlphaZero的基本原理

LLM与RL融合后的Reverse-o...

MCTS树搜索下的Reverse-o1

OpenAI o1的推出称为横空出世不为过，尽管关于Q*、草莓等各种传闻很久了，用了强化学习增强逻辑推理能力这个大方向大家猜的也八九不离十，但是融合LLM和RL来生成Hidden COT，估计很少人能想到这点，而且目前看效果确实挺好的。

OpenAI奔向Close的路上越走越远，你要从o1官宣字面来看，除了“强化学习生成Hidden COT”外，基本找不到其它有技术含量的内容。Sora好歹还给出了个粗略的技术框架图，字里行间也透漏不少隐含的技术点，细心点总能发现很多蛛丝马迹，串起来之后整个背后的技术就若隐若现（若对此感兴趣可看下我之前写的分析：技术神秘化的去魅：Sora关键技术逆向工程图解）。而且，尽管目前有不少公开文献在用LLM+RL增强大模型的推理能力，但几乎找不到做Hidden COT生成的工作，所以可供直接参考的内容非常少，这为分析o1进一步增添了难度。

那是否就没办法了呢？倒也不一定，如果多观察细节，再加上一些专业性的推论，我觉得还是有痕迹可循的。本文以相对容易理解的方式来对o1做些技术原理分析，试图回答下列问题：

除了复杂逻辑推理能力获得极大增强，o1还有其它什么重要意义？

o1的完整训练过程大致会是怎样的？

o1是单个模型，还是多个模型？

O1中的RL状态空间如何定义？行为空间如何定义？会用何种Reward Model？可能用何种训练数据？LLM和RM融合后的模型结构可能会是怎样的？.....等等

为行文中指称方便，我将文中推导的模型称为“Reverse-o1”。当然，里面有些部分有判断依据，有些则是根据主流技术作出的推断，只要OpenAI不官宣技术框架，大概一千个从业者能有一千种解法，我主要参考了AlphaZero的做法，试图在此基础上融合LLM和RL，很多看法比较主观，谨慎参考。

OpenAI o1的重要意义

我个人认为OpenAI o1是大模型技术领域的一个巨大突破，除了复杂逻辑推理能力获得极大提升外，这里展开分析下它在其它方面的意义和价值所在。

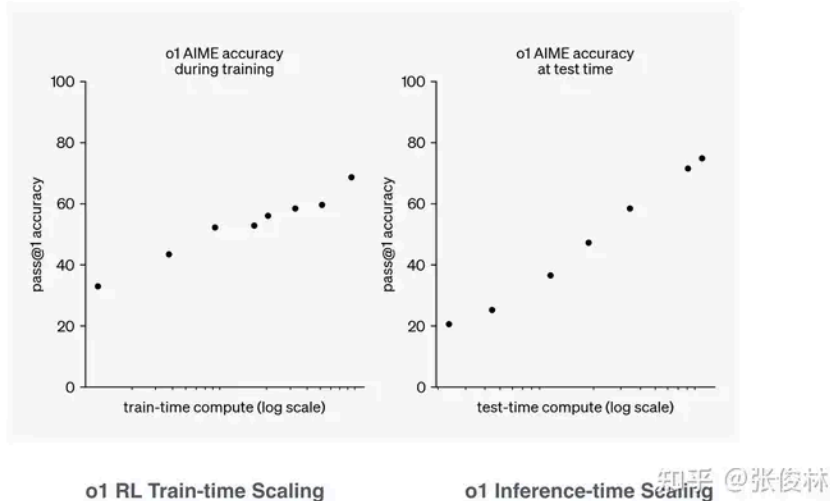
首先，o1给大模型带来了自我反思与错误修正能力，我觉得这点价值特别大。GPT 4这类模型，因为在输出答案的时候是逐个Token输出，当输出长度较长的时候，中间某些Token出错是一定会发生的，但即使LLM后来知道前面输出的Token错了，它也得将错就错往下继续编（这也是大模型幻觉的来源之一，为了看上去逻辑合理，LLM得用100个错误来掩盖前面的第一个错误），因为落Token无悔，没有机制让它去修正前面的错误。

而o1在“思考”也就是生成Hidden COT的过程中，如果你分析过OpenAI官网给出的Hidden COT例子的话，会发现它确实能意识到之前犯错了，并能自动进行修正。这种自我错误识别与修正对于LLM能做长链条思考及解决复杂任务非常重要，相当于越过了一个锁住LLM能力的很高的门槛。

第二，所谓新型的RL的Scaling law。OpenAI自己PR可能更强调这点，各种解读也比较看中这一点。我猜测o1的RL大概率要么用了相对复杂的、类似AlphaGo的MCTS树搜索，要么用了简单树结构拓展，比如生成多个候选，从中选择最好的（Best-of-N Sampling），这种策略如果连续用，其实也是一种简单的树搜索结构。也有可能两者一起用。不论怎样，树搜索结构大概率是用了，COT是线性的不假，但这是产出结果，不代表内部思考过程就一定是线性的，我觉得靠线性思维推导过程很难解决复杂问题，树形结构几乎是不可避免的。

有人问了，你有证据能说明o1大概率用了搜索树结构吗？我没有证据，但是可以推断，我的判断依据来自于o1 mini。之前的研究结论是这样的：尽管小模型语言能力强、世界知识还可以，但逻辑推理能力很难提起来，即使你通过比如蒸馏等措施试图把逻辑能力内化到小模型的参数里，效果

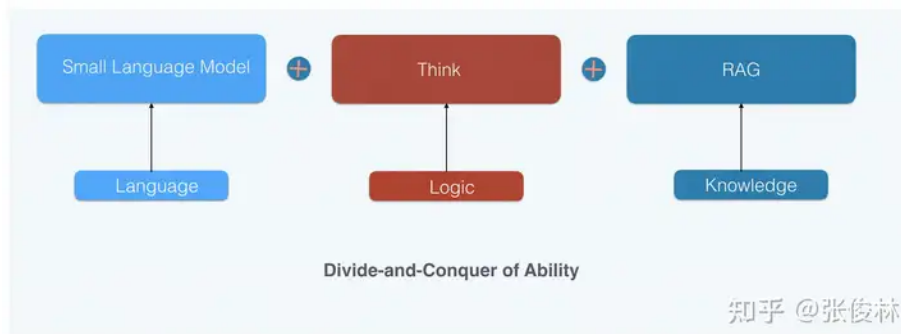
强，而且看样子可通过配置来提升或者降低它的逻辑推理能力（所谓inference-time Scaling law），如果你了解AlphaGo的运作机制的话，会发现这都是比较典型的搜索树的特点，可以通过控制搜索空间大小来提升模型能力。



虽然我个人认为，如果把通过设置参数来控制如何拓展树结构（比如控制搜索的宽度和深度），这种模式如果能被称为Scaling law的话，多少有点勉强，若这样，那我们可以说2006年AlphaGo出来就有Scaling law了。

但不管怎么称呼它，无法忽视的是这种方法的可扩展性极好，无论是在RL训练阶段，还是LLM的Inference阶段，只要改下参数配置来增加树搜索的宽度和深度，就能通过增加算力提升效果，可扩展性好且方式灵活。从这点讲，o1确实具有重要意义，因为这证明了它把怎么融合LLM和树搜索这条路走通了，LLM模型能够达到AGI的上限就被提高了一大截。

第三，在o1之后，小模型大行其道真正成为可能。小模型最近大半年也比较火，但从能力获取角度看，其实还是有上限锁定的，这个锁定小模型上限的就是逻辑推理能力。上面提到了，小模型的能力特点是：语言能力很强不比大模型弱、世界知识不如大模型但是可以通过给更多数据持续提升、受限于模型规模，逻辑推理能力能提升但比较困难。



所以小模型的优化重点是世界知识和逻辑推理能力，而从o1 mini的效果（世界知识弱、逻辑推理强）可推出，之后我们可以采用“能力分治”（DCA, Divide-and-Conquer of Ability）的模式推进小模型的技术发展（参考上图）：把语言、世界知识及逻辑推理三个能力解耦，语言能力靠小模型自身、逻辑推理靠类似o1的通过RL获得的深度思考能力，而世界知识可以靠外挂RAG获得增强。

通过“能力分治”，小模型完全可能具备目前最强大模型的能力，这等于真正为小模型扫清了前进路上的障碍，而SLM做起来成本又比较低，很多人和机构都可以做这事，所以可以预测：之后这种DCA模式将会大行其道，形成一种新的研发小模型的范式。

第四，o1可能会引发“安全对齐”新的范式。O1在做安全对齐方面，大概采用了类似Anthropic的“AI宪法”的思路，就是说给定一些安全守则，指明哪些行为能做，哪些不能做，在o1逻辑推理能力提高之后，它遵循这些法则的能力也获得了极大增强，安全能力比GPT 4o强很多。这可能引发安全对齐新的模式：大家可以先把模型的逻辑推理能力加强，然后在此之上采取类似“AI宪法”的思路，因为OpenAI o1证明这种模式可极大增强大模型的安全能力。

第五，“强化学习+LLM”的领域泛化能力，可能不局限于理科领域。强化学习适合解决Reward比较明确的复杂问题，典型的是数理化、Coding等有标准答案的学科，所以很多人会质疑o1是否

我推测OpenAI可能已经找到了一些非数理学科的Reward定义方法，并将这个方法通过RL拓展到更多领域。既然o1可以读懂并遵循安全规则，以“AI宪法”的思路解决安全问题，我觉得由此可以推导出一种针对模糊标准的Reward赋予方法：就是说针对不好量化的领域，通过写一些文字类的判断标准或规则，让大模型读懂并遵循它，以此来作为是否给予Reward的标准，符合标准则Reward高，否则Reward低。例如，针对写作文，就可以列出好文章的标准（结构清晰、文笔优美等规则），让大模型据此来给Reward。如此就能拓展到很多领域。

当然，想要这么做可能要分步骤，先用好给Reward的数理问题增强模型的复杂推理能力到一定层级，这样它就能看懂规则了，然后再做那些不好量化Reward的领域。（这都是我的猜测，没有依据）

由上述分析可看出，o1这条技术方向不仅增强了模型的复杂逻辑能力，由此可能引发大模型研发很多重要方向的革新，这是为何我说o1重要的主要原因。

OpenAI o1的完整训练过程推演



GPT 4等LLM模型训练一般由“预训练”和“后训练”两个阶段组成（参考上图）。“预训练”通过Next Token Prediction来从海量数据吸收语言、世界知识、逻辑推理、代码等基础能力，模型规模越大、训练数据量越多，则模型能力越强，我们一般说的Scaling Law指的是这一阶段的模型扩展特性，也是LLM训练最消耗算力资源的地方。“后训练”则分为SFT、RM和PPO三个过程，统称为人工反馈的强化学习（RLHF），这一阶段的主要目的有两个，一个是让LLM能遵循指令来做各种任务，另一个是内容安全，不让LLM输出不礼貌的内容。而训练好的模型推理（Inference）过程则是对于用户的问题直接逐个生成Token来形成答案。

OpenAI o1的整个训练和推理过程应与GPT 4这类典型LLM有较大区别。首先，“预训练”阶段应该是重新训练的，不太可能是在GPT 4o上通过继续预训练得到。证据很好找，OpenAI官方一再宣称o1 mini逻辑推理能力极强，但在世界知识方面很弱。如果是在其它模型上魔改的，世界知识不会比GPT 4o mini更弱，所以侧面说明了是重新训练的；另外，这也说明了o1这类侧重逻辑推理的模型，在预训练阶段的数据配比方面，应该极大增加了逻辑类训练数据比如STEM数据、代码、论文等的比例，甚至我都怀疑o1 mini是否引入了通用数据都不好说，否则不需要老强调知识方面能力弱。

在“后训练”阶段，应该有一个环节是用来增强LLM模型的指令遵循能力的，也就是说**RLHF阶段应该是有的**。因为o1在遵循指令方面能力并不弱，而且生成的Hidden COT片段里明显也包含很多指令性的内容，如果遵循指令能力比较弱，估计对于生成Hidden COT也有负面影响。所以，推断起来这个环节大概在“思考”阶段之前。（但是RLHF阶段未必有RM和PPO）。但这里和GPT 4对应的RLHF阶段应有两个重要的不同：首先，o1应该在这个阶段没有做内容安全方面的事情，大概率是挪到后面的阶段了（也有可能这两阶段都做了？）。其次，这个阶段大概率也会极大增强逻辑推理类的指令遵循数据比例，以此进一步加强基座模型的逻辑推理能力，原因我们等会专门说明。

接下来的阶段，就是o1最大的特点，所谓引入了“**系统2**”的慢思考能力。ClosedAI只说用了RL强化学习，其它任何都没提，技术保密工作一流。由此，我们只能推断出o1融合了LLM和RL来实现模型“先想后说”的Think能力。

As part of developing these new models, we have come up with a new safety training approach that harnesses their reasoning capabilities to make them adhere to safety and alignment guidelines. By being able to reason about our safety rules in context, it can apply them more effectively.

From OpenAI o1 Website

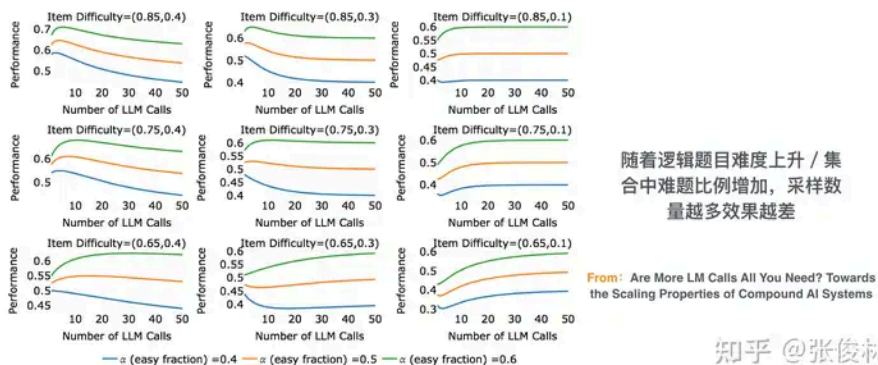
知乎 @张俊林

OpenAI o1应把“内容安全”相关的能力挪到了“Think”阶段之后，而且做法和GPT 4应该也有很大不同。在o1通过Think阶段进一步增强了它的逻辑推理能力之后，并没有用与安全相关的Instruct数据去调整模型参数，而是转为在比如System Prompt里引入人写好的LLM应该遵循的若干安全规则，比如不能输出制造有害产品比如毒品的细节内容等等。O1逻辑能力增强后，在输出时基本能够参考“安全条例说明书”，不输出有害内容，这做法类似于之前Anthropic的“AI宪法”的思路(依据参考上图)。而且，o1在内容安全方面的能力比GPT 4o强很多，这意味着将来大模型安全范式的巨大变化：应该先极大增强模型的逻辑推理能力，继而采取类似“AI宪法”或者叫“AI安全说明书”的模式来做。很明显如果这样，安全这事情做起来就简单多了，因为等于把LLM当人看了，你告诉他哪些能做，哪些不能做，当它逻辑能力强起来，现在就完全看懂了，就这么回事。

以上是o1的训练过程，在模型推理（Inference）阶段，o1体现出了“先思考再发言”的特点，分为三个阶段：首先通过思考，根据用户Prompt的问题生成能体现思考过程的Hidden COT数据，因为很多Hidden COT很长，于是引入了“COT摘要”阶段，从很长的Hidden COT里提取一些关键思考环节展示给用户看看，最后根据COT输出答案。

从上面内容可看出，o1无论在训练还是模型inference阶段，和传统的LLM应该还是有很大区别的。此外，我在这里再展开讲讲两个事情。

第一个，想要仿造模型来达到类似o1的效果，一个很容易想到的取巧的方式是：既不去专门增强基座模型的逻辑推理能力（比如大幅增加预训练中逻辑类数据占比），也不做“慢思考”阶段的RL训练（因为不知道怎么做的），只是侧重在模型inference阶段加入“Think”的过程，比如想办法引入最简单的Best-of-N Sampling这种树拓展策略，再写写Prompt提醒让LLM自己要自我思考、自我反思，两者相结合，也可以让模型自己写Hidden COT。这样做，也能一定程度上提升模型的推理效果。但是，这种做法效果提升的天花板比较低，就是说你模型逻辑推理能力看着提高了一些，然后就会被卡住，即使再增加inference阶段的算力（就是把采样数量N比如从10个拓展到50个，类似这种。Inference-time Scaling law大概其实很可能就是这个意思，您觉得这做法是law还是not law呢？）其实也没用。



这个结论来自于文献“Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters”及“Are More LM Calls All You Need? Towards the Scaling Properties of Compound AI Systems”，它们证明了：对于简单或者中等难度的逻辑推理问题，通过inference-time 增加算力，比如树搜索等方式，比去增强模型的“预训练”阶段的逻辑推理能力来得效果要明显；而对于高难度的逻辑推理问题，则只靠inference-time很难提升，有时还是负面作用，不如去增强模型“预训练”阶段的逻辑能力(参考上图)。

这是为啥呢？您可以想想，其实里面的道理细想一下很好理解。这是因为对于简单或中等难度的问题，模型在inference的时候很可能给出答案中的大部分步骤都是对的（或者多次采样中多数是对的），只有个别步骤错误，导致最终回答错误。通过比如Best-of-N Sampling这种简单树搜索方法来增加输出的多样性，再加上靠谱的Verifier筛一筛，是比较容易把这个小错误修正过来的。但对于高难度的逻辑问题，因为模型输出内容中大部分步骤可能都是错的（或者多次采样中大多数都是错的，这种情况你投个票采取多数人意见看看，结果估计很悲催），你想靠inference-time增加算力无力回天。

力，这是它能在后续inference-time增加算力解决复杂问题的根基。

所以关于这个点的结论应该是这样的：只靠inference-time增加算力，仅对容易和中等难度的逻辑问题有用，想要不断提升模型的复杂推理能力，还需要继续在Pre-Train和Post-Training阶段下功夫。

讲到这就有人就问了：那我也没钱自己训练基座模型啊？这可如何是好？这其实是绝大多数人面临的问题。其实拿来主义也应该可以，但是你得选那些逻辑推理能力强的基座模型，我估计代码类的基座模型相对比较适合，然后想办法在“Think”训练和“Think” inference方面做点工作，感觉应该也是可以的，而且对算力的需求也到不了大多数人做不了的程度。

第二个展开讲讲的事情。其实跟第一个有关，我看现在很多人看了o1后都说，Scaling范式变了，只要去Scale Inference-time的算力，模型推理效果就能一直Scaling。很明显这是进入误区了，原因上面讲了，如果只做Inference-time算力的拓展，模型效果天花板应该不会太高，归根结底还得去拓展Pre-train或者Post-train阶段模型的复杂逻辑推理能力，最起码两者是个相辅相成互相促进的作用，只谈inference-time Scaling大概率是不对的。

o1应由多个模型构成

3.2.2 CoT summarized outputs

We surface CoT summaries to users in ChatGPT. We trained the summarizer model away from producing disallowed content in these summaries. We find the model has strong performance here. We prompted o1-preview with our standard refusal evaluation, and checked for cases where the summary contained disallowed content but the answer didn't contain disallowed content. This would represent a situation in which the summarizer introduced additional harmful content. We found that this happens in only 0.06% of completions. Additionally, we prompted o1-preview with

From OpenAI o1 System Card

知乎 @张俊林

从o1的System Card可以明确看出，o1除了一个主模型外，至少还有一个相对独立的“Hidden COT摘要模型”（参考上图），它的作用是根据用户输入问题及生成的Hidden COT，提供一份简洁且内容安全的COT摘要。所以，**o1至少由两个模型构成**。

那么，问题是：**除了主模型和摘要模型，还有其它模型存在吗？我觉得大概率是有的。**

Model	Pricing	Pricing with Batch API*	Model	Pricing
gpt-4o	\$5.00 / 1M input tokens \$15.00 / 1M output tokens	\$2.50 / 1M input tokens \$7.50 / 1M output tokens	o1-preview	\$15.00 / 1M input tokens \$60.00 / 1M output* tokens
gpt-4o-2024-08-06	\$2.50 / 1M input tokens \$10.00 / 1M output tokens	\$1.25 / 1M input tokens \$5.00 / 1M output tokens	o1-preview-2024-09-12	\$15.00 / 1M input tokens \$60.00 / 1M output* tokens
gpt-4o-2024-06-13	\$5.00 / 1M input tokens \$15.00 / 1M output tokens	\$2.50 / 1M input tokens \$7.50 / 1M output tokens		
GPT 4o价格			o1-Preview价格	
gpt-4o-mini	\$0.150 / 1M input tokens \$0.600 / 1M output tokens	\$0.075 / 1M input tokens \$0.300 / 1M output tokens	o1-mini	\$3.00 / 1M input tokens \$12.00 / 1M output* tokens
gpt-4o-mini-2024-07-18	\$0.150 / 1M input tokens \$0.600 / 1M output tokens	\$0.075 / 1M input tokens \$0.300 / 1M output tokens	o1-mini-2024-09-12	\$3.00 / 1M input tokens \$12.00 / 1M output* tokens
GPT 4o mini价格			o1-Mini价格	

知乎 @张俊林

我们可以从o1的价格入手分析。目前已知：o1 Preview比GPT 4o的输入价格贵3倍，输出价格贵4倍，o1 mini输入和输出价格都是GPT 4o mini的20倍（参考上图）。



知乎 @张俊林

这里插入一段，解释下为何大模型的输入价格和输出价格是不同的，这是因为在大模型推理（inference，相对模型训练来说的，不是指逻辑推理）阶段，分为Prefill和Decoding两个阶段（参考上图）。Prefill阶段首先把用户的输入prompt通过并行计算，产生每个Token 对应Self Attention的Key-Value，并存储在KV Cache中，供Decoding阶段产生每个Token时候计算Self Attention时候用，这个阶段每个Token的Key-Value可并行计算，模型运行一次能输出多个Token的KV，所以GPU利用率高；而Decoding阶段根据用户Prompt生成后续内容，但模型运行一次只能产生一个Token，所以无法有效利用GPU的并行计算特长，资源利用率不足。资源利用率的差异导致了输出阶段成本高，这是为何大模型一般输出价格是输入价格3到4倍的原因。

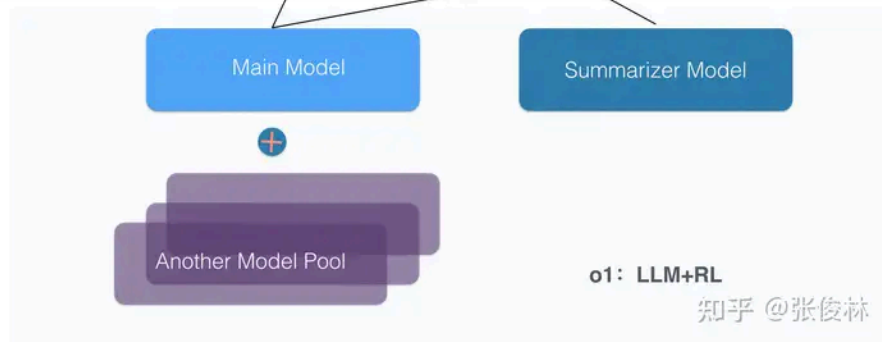
说回来，关于价格的核心问题是：为何比如o1 mini的输入价格（输入部分大模型的处理逻辑简单，只产生Prompt对应的KV Cache，更容易分析）是GPT 4o mini的20倍？这是个很奇怪点，仔细思考的话，这里应包含关于o1内部机制的很多线索。输入价格对应Prefill阶段，Prefill原则上只处理用户输入Prompt和Sys Prompt。Prefill阶段价格贵20倍，只有两个可能：

一种可能是用户Prompt+Sys Prompt的输入长度是GPT 4o mini输入的20倍。用户输入的Prompt对o1和4o来说当然是一样的，如果增加这么多输入，只能是OpenAI往sys Prompt里塞入了很多东西。但是考虑到o1 Preview只比GPT 4o贵3倍，那么sys Prompt特别长的可能性就不大了，否则o1 Preview的输入价格也应该比GPT 4o贵20倍才是。另外，如果只是sys Prompt带来的价格上升，那么o1在Decoding阶段就不应该那么贵了，因为sys Prompt 主要影响Prefill阶段的计算成本（当然，如果KV Cache长了，Decoding在计算Self Attention的时候计算量也会增加，因为要看到更长的上文，但是以目前大模型对长文本的支持能力，成本不会高太多）。

所以，针对这种可能性，结论大概是：OpenAI可能往sys Prompt里塞入东西了（应该是上面提到的“安全规则手册”啥的），但是并没有达到20倍价格的差异。塞入很长的sys Prompt这条原因不能解释20倍价格差距。

我们来考虑第二种可能性。如果假设输入Prompt长度差异没有太大，那么从Prefill计算机制角度来看，只能理解为o1 mini的模型总参数量是GPT 4o mini的20倍左右（除了摘要模型外，还有十多倍的差异）。这里又有两种可能，一种是说o1 mini就一个模型，那么它参数量是GPT 4o的大约20倍，这个很明显不成立，我们可以看到各种测试中o1 mini跑的速度挺快的，所以单个模型不可能太大；

于是，剩下的唯一解释就只能是：o1 mini除了一个主模型，一个摘要模型外，大概还有18个规模相当的其它模型。考虑到o1 Preview输入价格只比GPT 4o贵三倍，可以理解为o1 Preview除了一个主模型，一个摘要模型，还有另外一个规模相当的其它模型。因为o1 Preview的总体工作机制应该和o1 mini是类似的，可知o1 Preview多出来的那一个模型，和o1 mini多出来的那18个模型（那为啥这18个模型不共享KV Cache呢？如果可以共享KV Cache的话，起码在Prefill阶段，18个模型可以缩减为1个模型，那么o1 mini的输入价格只需要比GPT 4o mini贵3倍就够了。既然仍然贵20倍，侧面说明了这18个模型本身模型参数或者配置是有差异的，导致相互之间无法共享KV Cache），大概是干同一类事情的。而这个事情的性质呢，是模型个数可伸缩配置的，就是说你可以设置这个地方是部署1个、5个还是18个这类模型，而且这类模型相互之间还有些差异导致无法共享KV Cache。O1 mini在很多场景的效果要比o1 Preview效果好，一定程度上可能是跟这类模型的部署个数有关的，所以我推断，这类模型大概率跟树搜索有关。



综合起来，也就是说，o1模型大概由三部分构成(参考上图)：一个主模型，一个摘要模型，还有一类可灵活配置个数的跟树搜索相关的模型池子。如果我们自己想要给出技术方案逆向工程o1，你的技术方案可能就需要满足这个约束条件，需要包含这些模型，并能解释清楚这类模型池子的运作机制。

OpenAI O1可能采用的训练数据

人工标注数据

首先，训练o1肯定会人工标注一批COT思考过程，就是说拿到一批<问题，答案>数据，通过人工把解决问题的思考过程和步骤写下来，形成<问题，思考过程（包括思考过程中出现的错误及错误修正过程），答案>。如果没有人工标注过程，那么COT里出现的:Hmm, wait, ...这种，如果是纯靠LLM自己产生的，那估计LLM已经有意识了，这个概率很小，这些大概率最初来自于人工标注数据。可以用这些数据SFT一下o1初始的模型，启动模型的输出模式，让它熟悉这种表达方式，但是仅靠SFT肯定是不够的。

合成数据

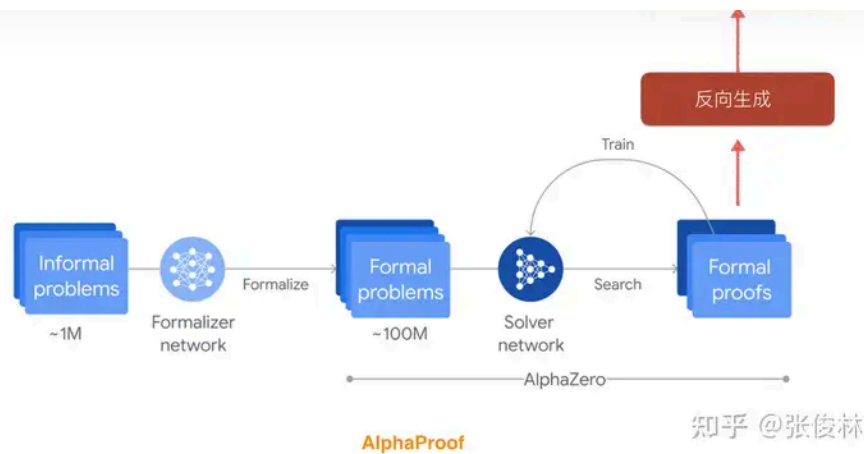
人工标注难度大、成本高，所以人工标注的COT数据数量不会太多，人工标注的问题是可扩展性太差，优点是质量比较高；之后可以采用合成数据的模式，一种最直观的合成数据的方式就类似上面提到制作PRM标注数据的模式：从人工标注的COT里面截取一段人工标注片段，然后使用MCTS树搜索方式去补齐后续推理过程，每个片段跑多次，有的最后答案正确的错误，无论是正确还是错误，都可以作为合成数据来训练o1模型。如果更激进一些，对于有确定标准答案的逻辑问题，可以通过不断试错的模式直接从问题开始搜索正确答案，这里搜索到的正确答案和错误答案都可以用来训练o1模型（但是这貌似就已经是o1了？所以可能性不大）。

代码COT数据的反向生成

有一种极大拓展代码COT数据的办法：我们有大量现成的各种代码，可以教会大模型试着从代码反向生成Hidden COT的推理步骤，这个应该是可行的，并能极大拓展Coding类型的COT数据。

（类似的思路借鉴自：Planning In Natural Language Improves LLM Search For Code Generation）

数学COT的反向生成



AlphaProof这种能力强的数学解题系统，整体思路是首先把自然语言的数学问题通过一个模型转化为形式化语言描述，然后使用lean及类似AlphaZero的模式，通过强化学习和树搜索来不断搜索并验证中间推理步骤。这种方法效果是可以保证的，目前基本可达到IMO银牌选手水准（参考上图蓝色部分）。但是这种需要转化成形式化语言再解题的系统有一个问题，就是通用性差，基本只能用来解决数学题，很难扩展到其它领域。

受到代码反向生成的启发，我觉得也可以反向生成数学COT（参考上图红色部分）。既然AlphaProof可以构造从自然语言问题翻译成形式化数学语言的神经网络，那也可以构造一个反向生成的模型，就是把数学形式语言翻译成自然语言，然后用这个翻译系统把AlphaProof找到的解题推理过程，从形式化语言转换成自然语言思考COT。当然，中间做错的也可以用，因为它有明确的验证系统，每一步即使错了，为啥错也知道，这些也可以翻译成自然语言。这样可以构造出千万量级甚至上亿量级的数学推理COT思维过程数据。我觉得这个思路大体是可行的。

OpenAI o1目前在数学和Coding方面效果最好，可知这方面的训练数据是最多的，我不知道是否会采用类似反向生成的思路来自动构造COT数据，但貌似这种方法看上去可行性还比较高。

Reverse-o1：RL的关键要素及如何将RL与LLM融合

我们从这里开始推导o1可能以何种方式将RL与LLM融合起来，并把推导出的模型称为Reverse-o1。

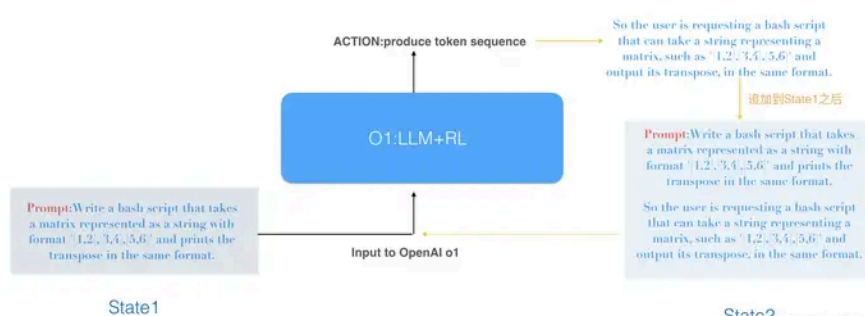
我们会先分析下在Hidden COT场景下RL的关键要素：状态空间（State Space）、行为空间（Action Space）、奖励模型（Reward Model）。至于RL方法，我推测采用类似AlphaGo/AlphaZero的概率较大，有几个原因：

首先，据说OpenAI员工每天要读好几遍萨顿写的“苦涩的教训”，而里面提到“能够发挥算力的通用方法，如搜索和学习，将最终大获成功”，这里的搜索主要指的就是DeepMind AlphaGo的MCST方法，OpenAI员工耳濡目染不把搜索用起来做个实践也说不过去不是？

第二，前几天OpenAI官宣的o1主力成员采访视频里，有员工提到了他们一直以来都在尝试如何将AlphaGo的搜索方法和LLM融合起来，这也是证据之一。

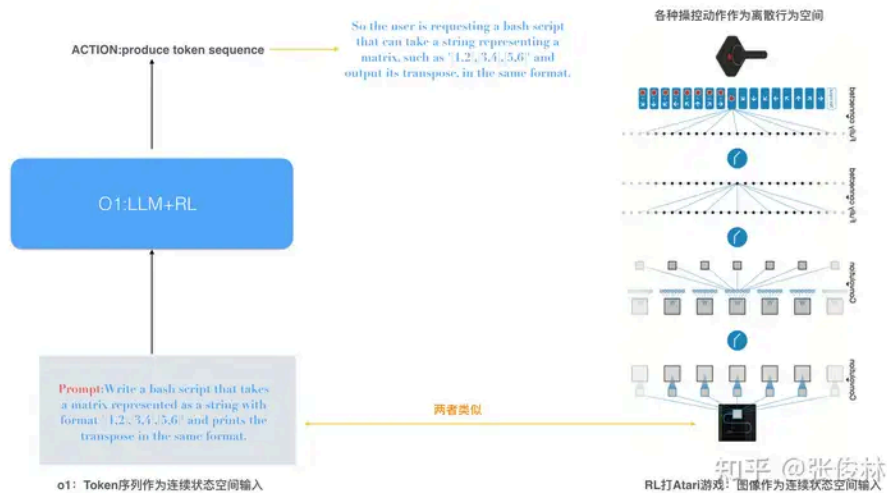
所以，之后会简单介绍下AlphaZero的工作原理，并尝试将其和LLM融合起来构造复杂逻辑推理系统。

O1中RL的状态空间：Token序列组成的连续状态空间



的，这些组成问题的Token序列作为一个整体可以看成第一个状态（State1），State1的Token序列作为o1模型的输入，o1在行为空间里选择某个行为（至于行为空间如何定义后面再谈），先不管这个行为是什么，反正选择这个行为后，o1会输出一个Token序列片段（不会是完整的Hidden COT，应该是其中的某个片段）。之后，o1把新生成的Hidden COT片段内容追加到State1之后，形成State2，再次作为o1的新输入，o1根据新输入选择新的行为，输出新的Token序列片段.....如此往复，直到Hidden COT输出结束。基本是这么个过程。

o1的RL状态空间不太可能由离散状态构成，你很难清晰地划分出若干具体状态。当然，可以说极端情况下，每个Token形成状态空间中的一个离散状态，但是这样基本没有实际的可行性。如果是每个Token代表一个状态S，首先这个状态组合空间太大。假设Token词典大小是10万，2个Token组合空间就是10万的平方，长度为n的Token序列，状态空间就是10万的n次方，基本是天文数字。其次，对于o1的RL来说，每输入一个Token就需要选择某个行为A，并生成下一个Token代表转移到另一个状态S'。如果RL过程带有搜索，意味着每个Token需要做一次搜索，而我们从很多o1的网上例子可以看到，很多时候Hidden COT是非常长的，几十上百K都有可能，这计算量基本是不可接受的。所以把每个Token看成离散状态，是不行，但颗粒度太细，感觉很难在实际中应用。



我觉得把o1的状态空间看成由Token序列组成的连续状态空间比较合适，上面例子尽管提到了State1或者State2，看着好像是离散状态，这只是方便解释过程而已（当然，如果把State1看成在巨大无比的Token组合空间中采样的一个点，这没问题）。就类似RL打游戏或者RL下围棋，RL输入的游戏（或围棋）画面由比如1024*1024个不同像素构成（不同像素可以类比为LLM的不同Token），由于像素组合空间过于巨大，很难清晰定义离散的一个一个State到底是什么，所以一般RL打游戏或者下围棋都是把输入图像当作一个整体，看成连续状态空间，通过一个神经网络来映射到某个具体的行为上。O1的状态空间和图像是类似的（参考上图），可以把一个Token片段类比RL打游戏对应的某个图片输入，看成由Token序列组成的连续状态空间，经过o1的LLM+RL神经网络映射到某个行为空间中的行为。

从上面分析可以看出，打游戏或者下围棋采用的RL技术，大都是以连续状态空间作为网络输入，而输出大都是离散行为空间中的某个行为，所以很明显这些地方采用的RL技术就比较适合用来作为o1的RL部分的解决方案，而采取离散状态空间的RL模型，比如MDP类方法就不太适合。

O1中RL的可能行为空间：“思考因子（Thought-Factor）” 离散行为空间

(10 vs 5, 8 vs 4, 4 vs 2, 8 vs 4)

Idea: Maybe we need to take every other letter or rebuild the plaintext from the ciphertext accordingly.

Let's test this theory.

提出猜测

But 'T' is 20.

Alternatively, perhaps subtract: $25 - 15 = 10$.

No.

否定猜测

Alternatively, perhaps combine the numbers in some way.

Alternatively, think about their positions in the alphabet.

Alternatively, perhaps the letters are encrypted via a code.

提出候选方案

Let's list them properly.

Wait, earlier I missed some letters there.

Let's re-express the sixth word letters:

m y n z n v a a t z a c d f o u l x x z

自我发现&修正错误

So the user is requesting a bash script that can take a string representing a matrix, such as '[1,2],[3,4],[5,6]' and output its transpose, in the same format.

澄清目标

Approach:

- Parse the input string to extract the matrix elements.
- Build the matrix as an array of arrays.
- Transpose the matrix.
- Output the transposed matrix in the same format.

拆解子任务 (知乎 @张俊林)

O1的RL技术方案，其中最关键的环节很有可能是如何定义行为（Action）空间。OpenAI O1的Hidden COT产生过程，本质上是在让机器模仿人在解决复杂问题产生的思考过程，而人在思考复杂问题时，有比较固定且数量并不太多的“思考模式”或者可以叫“思考因子”。比如拿到一个复杂问题，我们一般会首先明确这个问题的目标是什么，然后把复杂问题拆解成几个环节或者步骤，为了得到某一个具体步骤的解法，可能会提出一个假设，然后验证这个假设是否成立，如果不成立，那么继续提出新的假设，直到解决这个子问题.....我们也可能在过程中会进行验算并发现某些中间环节出现错误，并把错误修正过来。

如果仔细分析OpenAI官网放出来的几个Hidden COT，会发现是可以从里面归纳出一些典型的人类思考问题的一些隐含的“思考因子”的（参考上图，我给出了一些具体的例子）。我觉得要是把Hidden COT看成一个一个Token构成的，RL这事情就很难做了（Hidden COT的状态空间已经是连续非离散的，如果行为空间也是非离散的或者组合空间过大，RL很难建模。所以行为空间是离散的，这个极大概率为真，当然怎么定义离散的行为空间应有不同方法），在我的设想中，一个合理的方法是归纳出人类思考复杂问题的隐含的“思考因子”，以此作为候选的行为集合，比如：“拆解问题”、“复述目标”、“检查结果”、“修正错误”、“提出假设”等等，总体数量应该不会太多，即使划分得细致一些的话，估计也就几十到上百种。而针对每个具体的“思考因子”，可以产生符合对应分布概率的Token片段，比如行为若是“提出假设”因子，则生成“Alternatively”这个Token的概率就比较大（通过PPO从训练数据里学到的）。那么，Hidden COT片段很可能其真实面貌是长这样的：

<ACT_Proposer-Start> Alternatively, perhaps combine the numbers in some way.
<ACT_Proposer-End> (提出假设)

<ACT_RephraseTarget-Start> Overall Task: Write a bash script that takes one argument (the string representing the matrix) and outputs its transpose in the same format.
<ACT_RephraseTarget-End> (复述目标)

也就是说，OpenAI的Hidden COT的原始内容或者训练数据，在形式上有可能是这样的二级结构：

<Think_Start> (Hidden COT起始标记)

.....

<ACT-1_Start> token token token.....<ACT-1_End> (思考因子1)

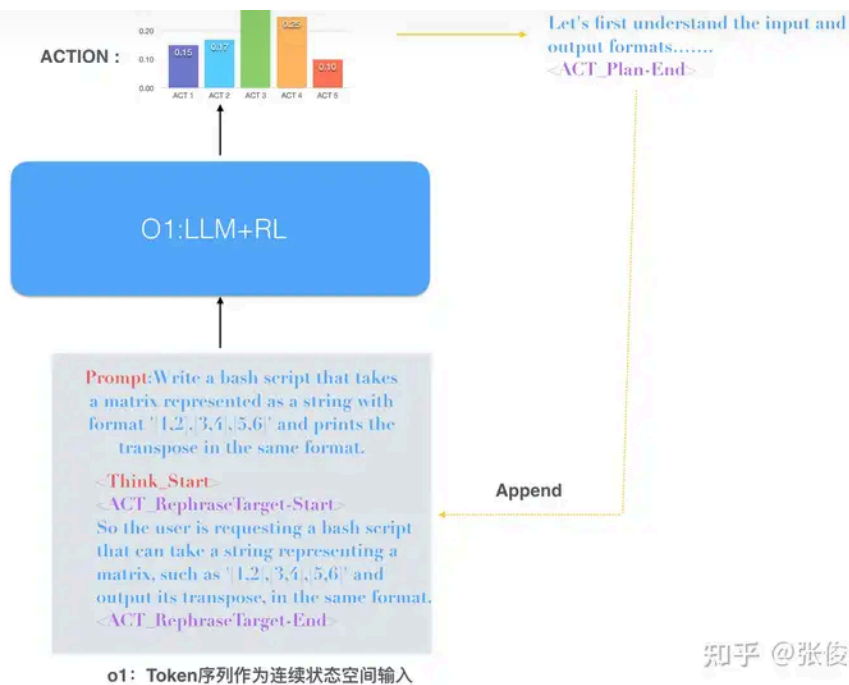
<ACT-2_Start> token token token.....<ACT-2_End> (思考因子2)

<ACT-3_Start> token token token.....<ACT-3_End> (思考因子3)

.....

<ACT-n_Start> token token token.....<ACT-n_End> (思考因子n)

<Think_End> (Hidden COT结束标记)



知乎 @张俊林

这种层级的Hidden COT结构，能体现出RL和LLM的优势结合，离散行为空间比如估算给定状态S采取何种行为，即函数 $Q(S,A)$ 的估算，这是RL擅长做的事情，而思考因子标签中的Token生成则是LLM擅长的事情，LLM可以根据对应“思考因子”的类型，学习调整因子标签内部Token的生成概率。上图展示了如上所述二级“思考因子”离散行为空间后，o1的可能运作形式。在生成Hidden COT的过程中，输入和输出都带有ACT行为Token的起始和结束符号，首先，O1根据当前的问题和已经生成的Hidden COT片段，预测下一个最可能采取的“思考因子”，以决定后面要采取怎样的具体思考模式，然后在这个“思考因子”指导下，LLM生成具体的Token序列，以“思考因子”的结束Token作为这种思维模式的结束标记。并将本步输出的Token序列并入输入，来循环往复地生成下一步思考的对应行为及Token序列。（当然整个过程都是我的设想，没有具体证据）。

那您会问：为啥我在给出的Hidden COT例子里看不到“思考因子”对应的起始和结束Token呢？可能展示给用户的COT是过滤后的版本。你想，Hidden COT的起始和结束Token（<Think_Start>/<Think_End>），这两个Token极大概率是会有，您不也没看到不是？说明输出的是过滤后的COT，那么，原先是有“思考因子”标记，但显示的时候被过滤掉，这也是有可能的。

O1中RL模型的奖励模型（Reward Model）



Pic from: Training Large Language Models for Reasoning through Reverse Curriculum Reinforcement Learning

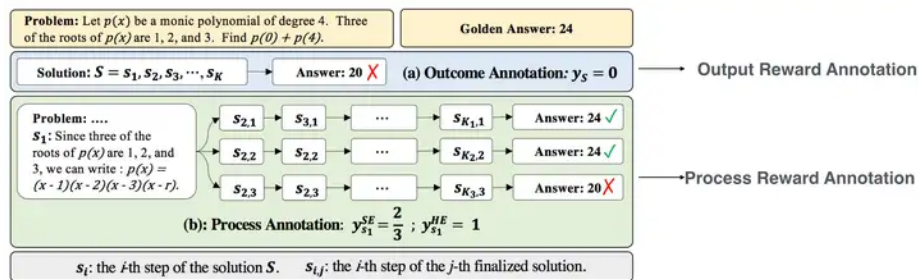
知乎 @张俊林

Reward如何设置对于RL来说至关重要，之前LLM+RL的学术工作其实蛮多的，归纳一下的话，目前常用的Reward模型有两种（参考上图）：结果奖励模型（ORM，Output Reward Model）和过程奖励模型（PRM，Process Reward Model）。

ORM的意思是训练一个模型，不管推导过程有多少步，只对最后结果打分。如果对照Hidden COT看的话，意思是只有o1把Hidden COT完整地写完了，ORM才给出一个奖励信号，模型结果若和标准答案对上了，给奖励1，如果答案错误，给奖励-1，类似这种。很明显，ORM的优点是反馈信号准确，比如对于数学题，模型要么做对了，要么做错了，很明确，所以反馈信号就精准；但ORM的缺点是反馈稀疏，意思就是反馈信号少，这个很直观，哪怕你推导过程写10页纸，反正最后只有一个反馈信号。（OpenAI 训练大模型时RLHF阶段的RM模型，就属于ORM）

PRM的含义是训练一个模型，能对中间每个过程都给予反馈信号，这样在推导过程中错在哪个步骤就很清楚，不用等到最后，所以它的特点是反馈信号丰富不稀疏。但问题来了，要训练PRM就

的数学题推导过程，而且证明了PRM效果要比ORM好。所以，PRM的优点是反馈多效果好，但是训练数据制作成本太高，一般人做不了。



from: Math-Shepherd: A Label-Free Step-by-Step Verifier for LLMs in Mathematical Reasoning

知乎 @张俊林

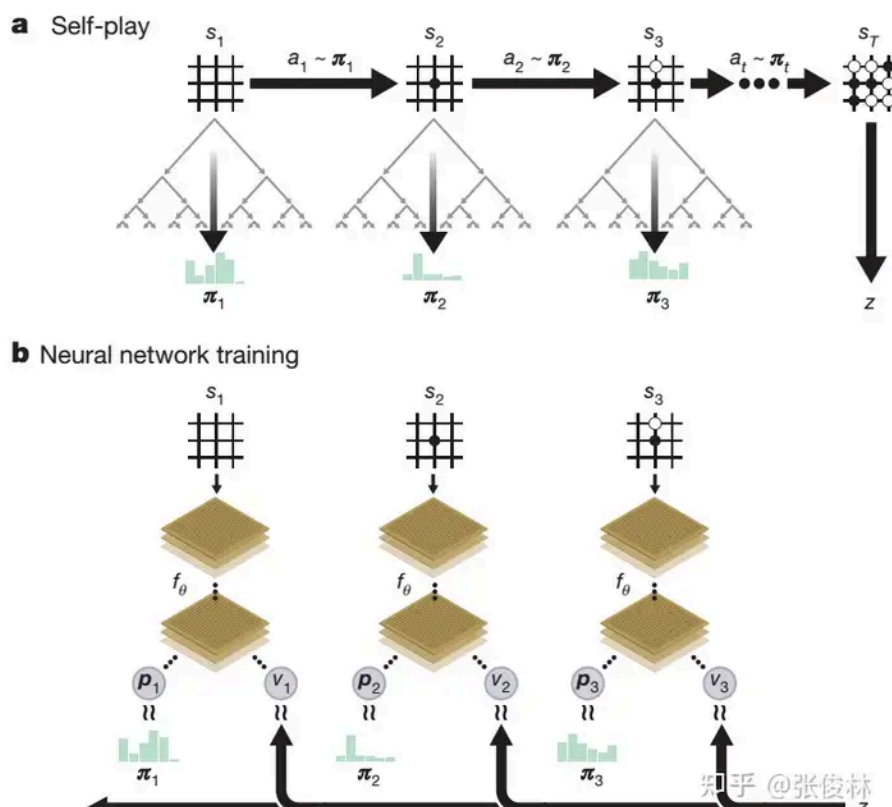
那有没有相对成本低的方法来做给每一步做标注呢？有。我目前看到比较好的做法是这么做的（参考上图）：假设我们手上有一批有完整推导过程的数学题，可以先把第一个解题步骤抄过来，然后用MCTS树搜索的方式去继续往后推导，可以从这个步骤出发做多次推导，其中有些推导过程会得到正确答案，有的结果错误，原则上从这个步骤出发的多次推导中，通向正确答案比例越高，说明抄过来的这步推导过程对于得到正确答案比较重要，则可以标注一个高分，然后可以抄过来第二个解题步骤，依此处理.....这样就能自动给每个推导步骤做质量标注。然后用这种数据去训练PRM模型，PRM就能给每个推理步骤打分。但很明显，通过这种数据训练出来的PRM打分的精准性肯定比不上ORM。

那么OpenAI o1在训练过程会采用ORM还是PRM呢？我估计两者都会用。ORM精准，PRM反馈丰富，两者各有优点，结合起来效果应会更好。另外，o1的官网提到了“*Our large-scale reinforcement learning algorithm teaches the model how to think productively using its chain of thought in a highly data-efficient training process.*”，这里的“*data-efficient*”，应该指的就是PRM。

AlphaZero的基本原理

这里会首先介绍下AlphaZero的基本工作原理，我们后面给出的Reverse-o1方案，核心是如何将RL和LLM融合起来，大框架主要参照AlphaZero的主体思路，所以这里做些说明以方便后续内容的理解。

2017年年底AlphaGo的棋类游戏通用版本Alpha Zero问世，不仅围棋，对于国际象棋、日本将棋等其他棋类游戏，AlphaZero也以压倒性优势战胜包括AlphaGo在内的最强的AI程序。



知乎 @张俊林

两处：一处是将AlphaGo的两个预测网络（策略网络P和价值网络V，策略网络P主要用于预测在当前状态 S 下，执行每个行为 a 也就是可能的落子位置的胜率，即函数 $P(S, a)$ ；而价值网络V主要评估当前状态 S 最终能够赢棋的整体概率，即函数 $V(S)$ ，是一个在比如0到1之间的数值， $V(S)$ 数值越大，从当前局面 S 出发赢棋概率越高。）合并成一个网络，同时产生两类输出 $P(S, a)$ 和 $V(S)$ ；第二处是网络结构从CNN结构升级为ResNet。AlphaZero完全放弃了从人类棋局来进行下棋经验的学习，直接从一张白纸开始通过自我对弈的方式进行学习，并仅仅通过三天的Self Play便获得了远超人类千年积累的围棋经验。

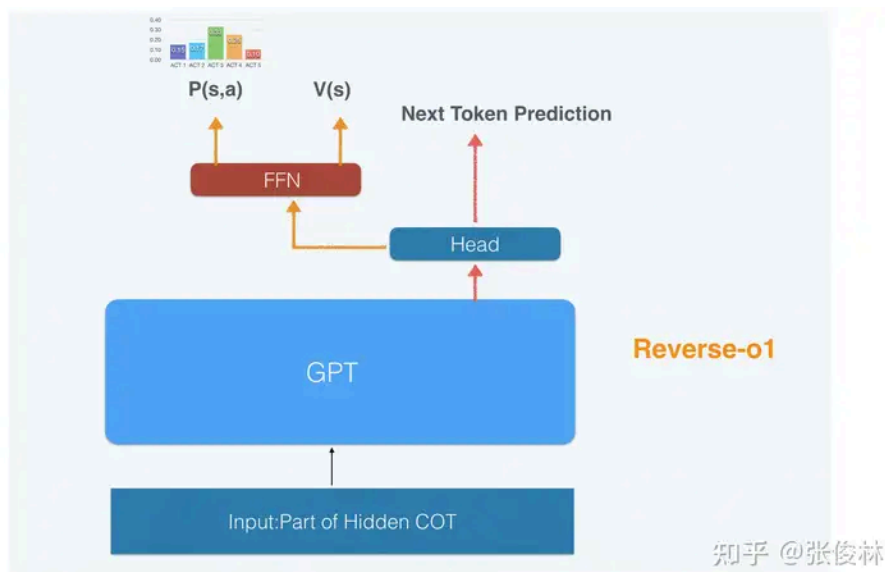
AlphaZero结合了MCTS和RL，MCTS是主体，RL起到了加速搜索速度的作用。在Self Play过程中（参考上图a），对于某个AI棋手，它会用MCTS搜索，对当前状态 S 下各个可能落子（Action）都去搜一下，每个位置经过搜索之后，能获得每个落子位置赢棋的概率分布 π ，从中选择概率最大的位置来落子，之后另一个AI棋手也采用类似的思路去落子.....这么一来一回直到分出胜负（ z 指出谁是胜者，Reward信号）。

到目前为止，貌似我们还没看到神经网络结构的作用，其实它主要是在MCTS搜索某个落子位置的时候发挥作用。因为从某个落子位置出发开始搜索，可搜索空间实在太太大，靠暴力搜索肯定行不通，所以策略网络 P 和价值网络 V （AlphaZero已经融合为一个网络了，分开说主要是为了方便阐述）的作用主要是引导搜索过程，优先搜索赢面大的路径，剪枝掉赢面小的路径，这样来增加搜索效率。

在搜索过程中神经网络参数固定不动，当一盘棋下完最终分出胜负，知道胜者后，可针对下棋路径上经过的每个状态 S 产生对应的训练数据。对于策略网络 P 来说，学习目标是MCST搜索获得的当时状态 S 出发落子概率分布 π ，而对于价值网络 V 来说，学习目标则是“最后的胜者 z 获胜概率大”这一事实。然后根据这些训练数据就可以调整神经网络参数，这样它在下一局对弈过程中能力会更强（可以看出AlphaZero的奖励模型是ORM）。这样通过无限重复的对弈过程，AlphaZero能力就越来越强。

应该意识到：对于AlphaZero来说，其本质其实还是MCTS蒙特卡洛树搜索。围棋之所以看着难度大难以克服，主要是搜索空间实在太太大，单纯靠暴力搜索完全不可行。如果我们假设现在有个机器无限强大，能够快速遍历所有搜索空间，那么其实单纯使用MCST树搜索，不依靠RL，机器也能达到完美的博弈状态。AlphaGo Zero通过自我对弈以及深度增强学习主要达到了能够更好地评估棋盘状态（ V ）和落子质量（ P ），优先选择走那些赢面大的博弈路径，这样能够舍弃大量的劣质路径，从而极大减少了需要搜索的空间，自我进化主要体现在评估棋面状态（ P 和 V ）越来越准，所以能越来越快地找到赢面最大的落子位置。而之所以能够通过自我对弈产生大量训练数据，是因为下棋是个规则定义很清晰的任务，到了一定状态就能够赢或者输，无非这种最终的赢或者输来得晚一些，不是每一步落子就能看到的。

LLM与RL融合后的Reverse-o1模型网络结构



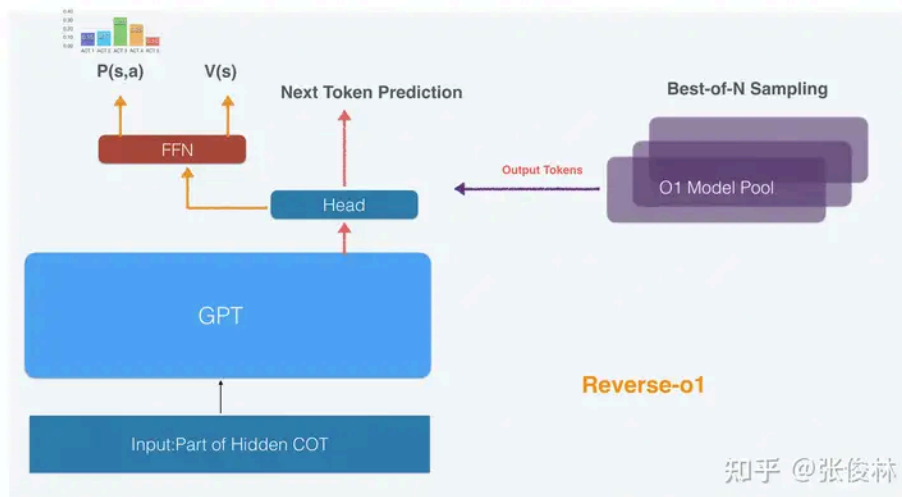
o1和下棋不同的一点是：除了RL，即使是Hidden COT，在背后也是靠一个Token一个Token输出的，LLM一定还是其中的主体结构，但RL肯定也需要一个网络结构去调整模型参数，来逐步学会内部的思考过程。所以，我们首先面临的问题是：如何融合LLM和RL两个模型，来获得一个同时混合LLM和RL两者功能的完整网络结构。

上图给出了一个我设想中的结构：主体仍然是基于Transformer的LLM模型（Dense或MOE都可以，mini版本应是Dense结构），当输入“问题+已经生成的部分Hidden COT”（也就是由连续

上，可以搭建RL模型结构，这里参考了AlphaZero的思路，一个网络两个输出。比如可以用FFN网络结构，一方面输出策略网络 P 结果（ $P(S, a)$ ），代表在当前状态 S 下，下一步Action“思考因子”的分布概率 π ，某个“思考因子”概率越大，则下一步这个Action被选中执行可能性越大；另外一方面会输出价值网络 V 结果（ $V(S)$ ），代表当前状态 S 通向最终正确答案的概率大小，概率越大说明当前状态 S 质量越高，意味着目前已输出的这部分Hidden COT整体质量较高。

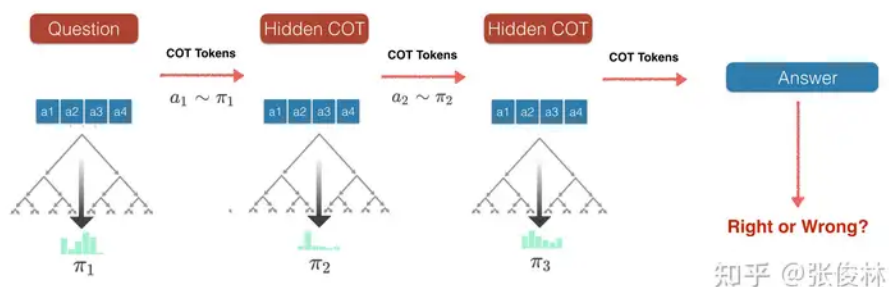
到了这一步，当Hidden COT处于某个状态 S 的时候，经过网络可知下一步应该采取什么动作，也获得了当前状态 S 通向成功答案的概率。但目前仍缺少一部分内容，即在已知下一步“思考因子”行为后，对应的Hidden COT 一系列输出的Tokens。

一种简单的方法是用LLM head之上的LLM部分持续输出后续Tokens(有人工数据训练的时候，可以用PPO来增加对应Token的输出概率)，在输出后续Token的时候并不考虑RL的输出，直到LLM输出到 $\langle \text{Act}_i - \text{End} \rangle$ 之后，再去判断RL的输出选择动作.....持续此过程，结合LLM和RL输出Hidden COT的模型就能运转起来。

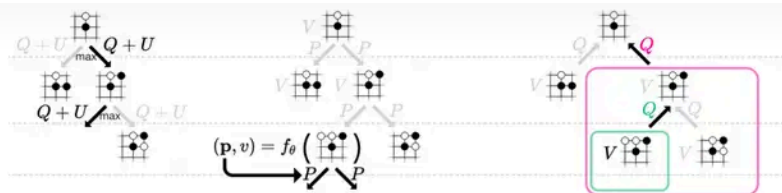


前文我们分析过，o1大概率会使用过程奖励模型PRM，还有，它可能是由多个模型构成的。在这两个约束条件下，可以如此改造上面的模型结构（参考上图）：在已知下一步“思考因子”后，不由主模型来生成后续Tokens，为了增加后续生成COT的质量，可采用Best-of-N Sampling的思路，由多个复制的Reverse-o1模型（不同副本可以设置不同的温度参数，增加输出的多样性）各自给出一个Token序列，然后由离线训练好的PRM作为评委打分，选择得分最高的Token序列，作为本次“思考因子”后续的输出Tokens。选出最佳内容后，可同步给主模型，主模型执行一次类似Prefill的操作，即可同步输出最佳内容，然后开始下一轮的输出.....可如此办理，这么做明显生成的Token序列质量会更高。

MCTS树搜索下的Reverse-o1



我们仿照AlphaZero，引入主体结构MCTS，它的运行流程如下（参考上图）：当用户输入问题后，Reverse-o1使用MCTS树，对于每个可能的“思考因子”进行搜索，在搜索时会用策略网络 P 和价值网络 V 来快速寻找最优搜索路径，这样得到所有“思考因子”的概率分布 π ，概率数值越大则代表采取这类思考通向正确答案概率越高。之后选择概率最大的“思考因子”作为当前状态下的行为，并如上节内容所述，由Reverse-o1生成针对这个行为下的COT Tokens片段。将COT Tokens片段并入用户问题，形成新的状态.....依次往后走，直到产生问题的答案，和标准答案对比后，要么对要么错，由此得到对应的Output Reward。



$$\begin{aligned} Q &\sim f(V(s)) \\ U &\sim f(P(s, a)) \end{aligned} \rightarrow \max(Q + U) \sim \max(f(V(s)) + f(P(s, a)))$$

知乎 @张俊林

仿照AlphaZero，从状态 S 出发，搜索某个“思考因子”指向正确答案的概率时，以 $\max(Q + U)$ 的方式寻找最优下一状态 S' ，而 Q 函数与价值网络 $V(S)$ 正相关， U 函数与策略网络 $P(S, A)$ 正相关，所以 $\max(Q + U)$ 的含义是通过价值网络和策略网络的指引，来寻找高质量的搜索路径；当搜索到叶结点的时候，会进行节点扩展，并用策略网络和价值网络估算初始化相关搜索参数，之后由低向上更新最优路径上所有状态对应的 Q 函数。当每个候选的“思考因子”经过一轮搜索后会得到所有行为的分布概率 π ，完成搜索步骤。O1搜索时与下棋不同的地方在于：如果要往下一状态转移，还需要根据当前选到的行为，产生对应的Hidden COT tokens，这个步骤可由上文讲述的Best-of-N Sampling策略来完成。

当从问题开始，逐步生成Hidden COT片段后，走到答案阶段，会获得Output Reward，完成一次通过中间环节到答案的MCTS搜索过程。如果答案正确，可设置Reward=1，答案错误可设置Reward=-1，在此基础上针对走到答案所经过的所有中间状态 S 构造训练数据，来训练策略网络 P 和价值网络 V ，策略网络的学习目标是对应状态MCTS搜索到的行为概率分布 π ，价值网络的学习目标是Output Reward。

除此外，对于搜索过程每个被选中“思考因子”通过Best-of-N Sampling得到的对应Hidden COT tokens序列（也可以拿到PRM赋予这个tokens序列对应的Process Reward分数），则可以利用PPO（PRM的Reward作为PPO的Reward）来调整LLM模型参数，使得LLM之后在遇到这个“思考因子”后，提高这些Tokens的生成概率。

到目前为止，差不多可以结束o1的整个逆向工程之旅了，前文提到的一些约束条件（o1应该由多个模型构成、应该用了某种或者某几种树搜索、RPM和ORM应该都会用等）在设想中的Reverse-o1中基本都得到了体现。

但是，我个人觉得还有一个问题值得深入思考：“思考因子”是必须存在的吗？毕竟这需要靠人工去归纳人类的潜在思维模式，仍然有比较强的人工痕迹存在，而且会增加人工标注数据的成本。这个问题我确实思考了好几天，结论貌似是：整个框架还可以是这个框架，不用引入“思考因子”或者把“思考因子”改为“隐式思考因子（Hidden Thought Factor）”应该也是可以的，因文章已经太长，解释起来有点复杂，这里先略过。

编辑于 2024-09-28 14:18 · IP 属地中国香港

「真诚赞赏，手留余香」

赞赏

2 人已赞赏



大模型 LLM（大型语言模型） openai-o1

已赞同 1074 58 条评论 分享 喜欢 收藏 申请转载 ...



理性发言，友善互动

58 条评论

默认 最新



习翔宇 我关注的人

“如果没有人工标注过程，那么COT里出现的:Hmm, wait, ...这种，如果是纯靠LLM自己产生的，那估计LLM已经有意识了，这个概率很小，这些大概率最初来自于人工标注数据”

其实PRM800k的数据里面就有这样的数据了，而且是LLM自己产生的。如果看PRM800k的

Conversational capability, 对话能力/对话能力%: 🤖

09-25 · 北京

● 回复 ● 16

**习翔宇** 我关注的人

...

习翔宇: 与Open AI o1有关的一些观察和推测 欢迎讨论

09-27 · 北京

● 回复 ● 1

**字节** ⭐

...

"o1 mini输入和输出价格都是GPT 4o的20倍（参考上图）" 这里应该笔误了，应该是跟GPT-4o-mini比较吧？

09-25 · 浙江

● 回复 ● 2

**司徒青云**

...

刚想说这个问题，一看评论区已经有大佬支出了😄

09-27 · 上海

● 回复 ● 喜欢

**张俊林** 作者 ⭐

...

确实写错了😄

09-25 · 北京

● 回复 ● 喜欢

**挠头的三棱镜**

...

关于最后一个人工的“思考因子”是否必须存在的问题，我认为至少对于o1来说答案是否定的。有两个证据：第一，o1的关键开发人员Hyung Won Chung在最近演讲中强调的不应该教机器如何去思考而是去激发他们学习如何去思考。第二，o1开发人员采访视频中提到的，o1开发过程的“aha”时刻正是发现模型自发涌现诸如“反思”、“重试”这样的“思考因子”。

09-25 · 中国香港

● 回复 ● 12

**习翔宇** 我关注的人

...

我感觉不是必须的，这种数据是自然预训练数据中存在的、也是o1标注数据中去自动



理性发言，友善互动

**wlgqa**

...

深表赞同，我也是第一时间想到了那个演讲

09-25 · 北京

● 回复 ● 喜欢

**宝宝戈**

...

有没有可能它没有用运行时mcts

09-25 · 美国

● 回复 ● 4

**习翔宇** 我关注的人

...

同感，作者这段“有人问了，你有证据能说明o1大概率用了搜索树结构吗？我没有证据，但是可以推断，我的判断依据来自于o1 mini”我感觉就很牵强了，那个inference-time Scaling很有可能就是并行解码+verifier选择，跟o1 blog里面的“On the 2024 AIME exams, GPT-4o only solved on average 12% (1.8/15) of problems. o1 averaged 74% (11.1/15) with a single sample per problem, 83% (12.5/15) with consensus among 64 samples, and 93% (13.9/15) when re-ranking 1000 samples with a learned scoring function”是能对应的

09-25 · 北京

● 回复 ● 7

**zml24** ▶ **习翔宇**

...

要是用 MCTS 就不需要 re-ranking 了，也不需要提交多个答案去算 pass@1k 了，只需要无脑堆 MCTS 的搜索空间就行。下围棋一步最多 361 个选点，复杂局面下 KataGo 单步搜索次数依然可以到几百 k；如果LLM 一步是一句话的话，那搜索空间也可以接近“无限”增长。

09-25 · 上海

● 回复 ● 2

查看全部 6 条回复 >

**scott.cgi**

...

直觉上，一个主模型，一个思维模型——思维模型提供思维链给主模型，主模型的推理过程，会向思维模型提问以获取思维链，然后填充思维链的每一步，以此循环，直到获得最终答案。



类比人脑，分解问题与逐步解题，是两个思维过程，解题遇到问题时，就会继续分解问题，以达到能够解题的程度，然后继续执行解题，以此循环，直到所有问题解决，最后汇总答案。

推理成本，就在于这两个模型的大小，以及两者的交互次数与深度。

09-25 · 安徽

● 回复 ● 2

09-25 · 安徽

 回复  1**习翔宇**  我关注的人 

...

这有点太抽象了

09-27 · 北京

 回复  喜欢**无神小坏**

...

在数据构建上, 我的看法可能不太一样.

在o1的RL阶段, 其实OpenAI给的case都有个特质, 就是验证他的结果是否正确是相对容易的 (或者一些题目已有标准答案), 可能前期会构建少量的专家数据用来加快拟合, 后期更多的应当是使用这些数据自行优化策略和价值网络.

其次是考虑提升基座模型的"小步骤推理"能力, 从g4o 0806开始, 它在代码执行相关的COT任务上呈现出来的结果就与之前很不相似, 变成事无巨细的一行一行一个循环一个循环生成推理结果, 猜测通过获取代码执行结果/简单逻辑推演/数学计算等方式将推理的输入输出一同给到模型构建"小步骤推理"的合成数据. SFT阶段可能包括一些高质量人工标注的COT数据, 到RM PPO阶段可以针对前面产生的小步骤错误进行修正.

09-25 · 北京

 回复  3**十年169**

...

最近alphamath 论文就是通过mcts 和树搜索的方式实现数学推理能力的提高, 其中sbs 就是树搜索方案, 而且从模型功能设计上有思考生成能力和step 级别的打分评判能力

09-25 · 浙江

 回复  3**小仙**

...

看了一下这个论文, 结果很不错。很有参考意义👍

09-27 · 北京

 回复  喜欢**刀刀宁** 

...

我觉得文中提到的“很容易想到的取巧的方式”, 反而有可能有它的可取之处, 因为后边引用的几篇文章证明了 infer 加长不一定能增加整体的逻辑推理能力, 但是这几篇文章实验采用的具体技术, 并不和 OpenAI 一样。所以暂时还不能排除 OpenAI 采用了某些 naive 的想法, 只是不能排除哈。

09-25 · 湖南

 回复  1**刀刀宁** 

...

确切的说, 我比较倾向的方法也不是“很容易想到的取巧的方式”: 确实不去专门增强基座模型的逻辑推理能力, 但是会 RL 训练以期望在 inference 阶段加入 Think 的过程。

09-25 · 湖南

 回复  喜欢**zzzyp**

...

请务必再多讨论一些



09-25 · 陕西

 回复  2**Cppowboy**

...

对基础模型能力要求很高, 首先模型要能理解非常长的内容, 即使已经生成很长的thought也要记得自己现在是在干嘛, 然后每一步的想法都要有些意义, 出错了还得会反思。

09-25 · 北京

 回复  2**习翔宇**  我关注的人 

...

是, 所以基座进行了重新训练, 增强了推理能力和对话能力

09-27 · 北京

 回复  喜欢[点击查看全部评论](#) >

文章被以下专栏收录

**深度学习前沿笔记**

深度学习领域前沿进展科普

**NLP工作站**

NLP论文解读、总结以及个人实践经验汇总

逆向工程（又称逆向技术），是一种产品设计技术再现过程，即对一项目标产品进行逆向分析及研究，从而演绎并得出该产品的处理流程、组织结构、功能特性及技术规格等设计要素，以制作出功能相...

傅初



逆向工程的应用

沪教3d

发表于三维扫描_...



数学规划与运筹学 (4) 线性规划的对偶

金鱼马

发表于优化基本理.