KIMIK1.5: SCALING REINFORCEMENT LEARNING WITH LLMS

注意到KIMI和DEEPSEEK 在训练最新的大模型中提出了一个新的思路，不依赖于更复杂的技术，如蒙特卡洛树搜索、价值函数和过程奖励模型，但依然在多个基准和模式上实现了最先进的推理性能。

注意到与deepseek R1不同的是KIMI K1.5是基于文字和图片一起训练的，它们都简化了中间奖励模型，但是KIMI有一个最终奖励模型。

通过改进Tree of thoughts的反馈方式，使训练的模型可以自动回归预测来提升模型的思维能力，这里着重学习该论文提出的Policy Optimization部分

策略梯度定理可以简单的概括为以下公式

公式关联图示

$$\underbrace{\nabla_\theta J(\theta)}_{\text{梯度}} = \mathbb{E}\left[\underbrace{\nabla_\theta \log \pi(a|s;\theta)}_{\text{偏导数}} \cdot \underbrace{A(s,a)}_{\text{系数}}\right] \quad \text{受} \quad \underbrace{\mathrm{KL}(\pi_{\text{old}}\|\pi)}_{\text{离散度}} \leq \delta \quad \text{约束}$$

这里策略更新需限制新旧策略的差异，避免崩溃，所以需要**KL 散度**衡量策略分布的“离散度”

论文里提出的公式没有采用KL散度，这里是这篇论文的创新点之一。

论文中通过relative entropy regularized policy optimization problem，为了达到最佳的输入x，输出y,z,在每一次循环通过一个类似于softmax的函数去选择最大化的系数。得到在surrogate loss中最佳状态是里面括号这一部分相减等于0.

$$L(\theta) = \mathbb{E}_{(x,y^*)\sim\mathcal{D}}\left[\mathbb{E}_{(y,z)\sim\pi_{\theta_i}}\left[\left(r(x,y,y^*) - \tau\log Z - \tau\log\frac{\pi_\theta(y,z\mid x)}{\pi_{\theta_i}(y,z\mid x)}\right)^2\right]\right].$$

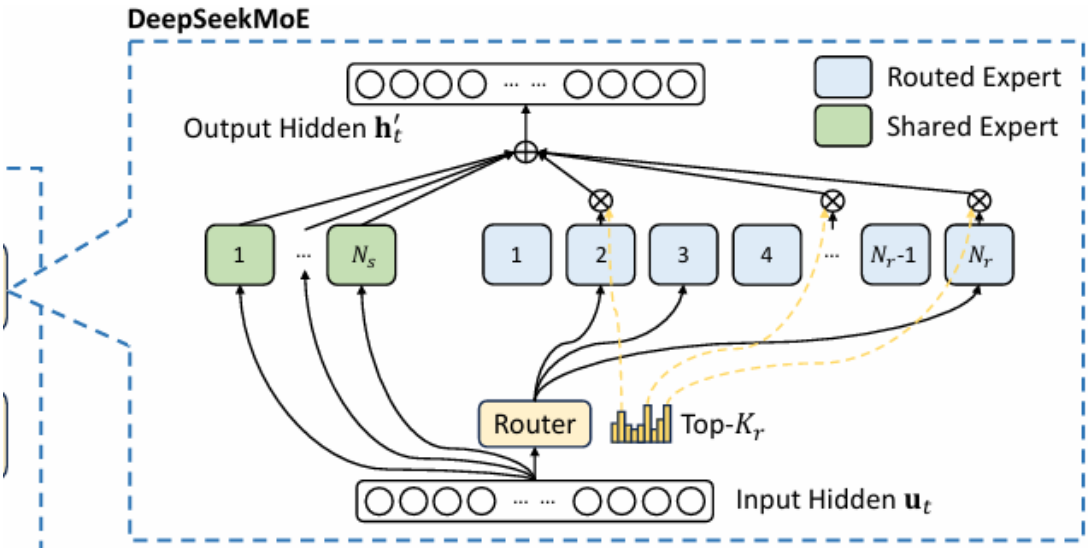
其中logZ是每一次多个步骤采样的平均值，这里计算量非常大，所以优化为去取K步的平均值来作为基准，这里将所得的系数取平方论文中未做解释，推测应该是为了避免负数的出现影响优化的效果。

在最终得到的策略梯度公式中，τ/2没有相应的说明，推测可能是整体的更新过程中方差依然比较大，为了使梯度更新更加平滑一些引入该步骤去控制。

$$\frac{1}{k}\sum_{j=1}^{k}\left(\nabla_\theta \log \pi_\theta(y_j,z_j|x)(r(x,y_j,y^*)-\bar{r}) - \frac{\tau}{2}\nabla_\theta\left(\log\frac{\pi_\theta(y_j,z_j|x)}{\pi_{\theta_i}(y_j,z_j|x)}\right)^2\right). \tag{3}$$

DeepSeek-V3 Technical Report

这篇论文我主要关注的是**ShareExpert SparseMOE**技术，注意到最早使用MOE技术来训练LLM的是Mistral AI ,deepseek引入**Share 专家** 和 **Router 专家**，**Share 专家** 是一直激活的，即输入的 token 都会被 Share 专家头计**Router 专家头** 会先和上图中的 ut 计算亲和度（代码中直接用一个 Linear 层进行投影），选择 top-k 各专家进行推理。（代码中推理计算 top-k 时，会先将 N 各专家进行分组为 `n_groups`，将每个组中 top-2 各专家的亲和力加起来，算出亲和力最高的 `top_k_group` 个组，然后在这些组里选 top-k 个专家）。最终将所有的 Share 输出和 Router 专家进行亲和度加权相加，得到 MoE 层的输出。



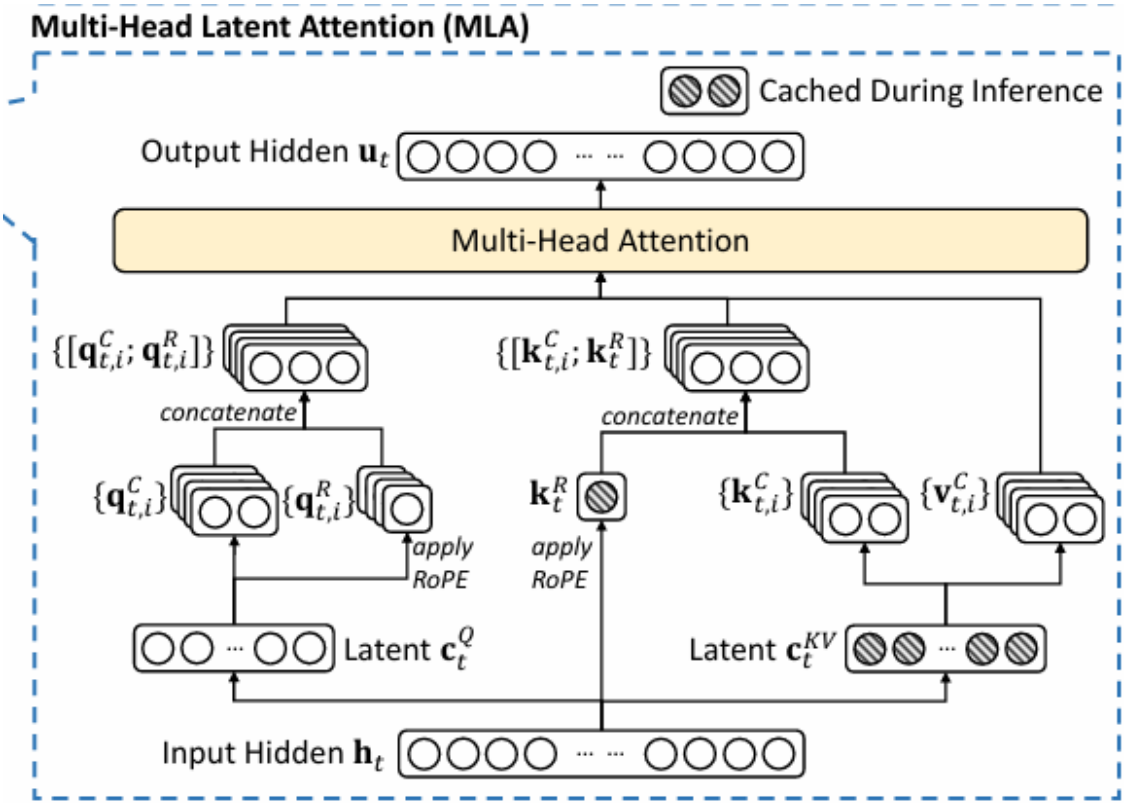
选择MOE的好处是可以选择部分专家（K个）进行计算，计算量和通信量都下降为MLP的K/N,

在模型训练阶段deepseek也采取了数据并行和流水线并行，它们是正交的，可以同时使用，这将在所有GPU上形成子组，并在子组中使用集合通信。**水平方向是完整的一个模型，垂直方向是相同层的不同副本**

Multi-Head Latent Attention

通过学习得知MLA的核心是对KV进行压缩后，再送入标准的MHA算法中，用一个更短的k，v向量来进行计算，进而减少KV Cache的大小。

计算代表 KV Cache 的潜在向量C_t^{KV},这种方式使得KV Cache变小，相对与原来的隐藏状态h_t要小很多。注意到为了同时使用潜向量计算和旋转位置编码RoPE两个技术多创建一个新的向量k_t^R来编码位置信息，将来通过向量合并将位置信息带入键向量。特殊的是对于查询向量q，也进行了潜向量的计算，与减少KV Cache无关，个人认为在注意力机制中，q、k、v是平等的输入，对它们进行相同的潜在向量计算可以保持模型的对称性和一致性。



学习了以上两种技术之后我正在尝试复现这两种技术。

To CoT or not to CoT? Chain-of-thought helps mainly on math and symbolic reasoning

这篇论文主要提出的一个结论是CoT主要在涉及数学或逻辑的任务上提供了强大的性能优势，在其他类型的任务上的收益要小得多。在MMLU上，直接生成没有CoT的答案会导致与CoT几乎相同的精度，除非问题或模型的响应包含等号，表示符号操作和推理。

为什么CoT在这两种问题上面有优势：CoT主要帮助执行计算和符号操作的执行步骤，生成可执行的正式解决方案计划。

关键是问题是否具有符号推理。如果一个问题可以建立在一个自然的、公认的正式系统中，我们就认为它是象征性的。

同时，在可能的情况下，在求解符号任务时，llm应该在推理时与符号求解器配对，以获得比直接回答和CoT始终更好的性能。

Deep Dive into LLMs like ChatGPT

这是Andrej Karpathy的youtobe视频，我认为这个视频的内容很好，于是就拿来简要概括一下LLM的训练阶段。

1、预训练和分词：通过互联网提取公开文本（URL-文本提取-语言过滤-删除过滤等），再通过编码压缩为token存储 再通过神经网络训练，输入序列化token，模型预测随机输出，不断训练模型参数直至输出与训练集一致。模型参数固定下来。此阶段需要大量数据+GPU，需要调节数十亿参数，所以需要较长时间（几个月）。

2、推理阶段：根据既定模型，通过既定模式（如助手）进行预测输出。此阶段可能会产生幻觉，由于输出是预测，修正方法为通过让模型知道自己不知道不要瞎随机回答；大预言模型不擅长计数、字符拼写，在输出过程中可调用网页搜索、程序代码等方式让问题得到更准确的解决（而非通过推理），以及通过一些硬编码/PE固定化一些回答（如你是谁）。

4、有监督学习（SFT）和强化学习（RL）：有监督学习像是教课书上给的例题，有题目有答案；强化学习像是教课书的课后题。从效果上看SFT提升有一定上限（趋近专家水平），但RL可以涌现一些更好的方式突破。目前deepseek和openai o3都是RL 奖励模型（RLHF）：一种强化学习用来训练模型打分的模型。也是一种神经网络。但有局限性在于1）人类判断有损不完全准确，2）多次训练后会将一些无意义对抗样本赋予高分（如1000次以后），所以需要及时剪枝。

我从该视频中得到一点结论，从chatgpt 4o开始，模型训练的参数大小尚未取得“gpt4”级别的突破，不难从openai 发布的o1模型上看，其模型相比 4o只是在原基础上引入CoT等技术，未来模型的突破主要聚焦于强化学习，推理速度，超长上下文，agent代理等方面。不同的模型厂家正在对相关技术进行突破和测试。

下周我将继续阅读deepseek v3的技术报告中有关流水线优化的部分和deepseek R1的技术报告，并初步计划完成deepseek V3的训练部分，阅读Tree of Thoughts（arXiv:2305.10601），InternVL: Scaling up Vision Foundation Models and Aligningfor Generic Visual-Linguistic Tasks，这两篇论文，目前还在学习github开源项目browser use。