Indirect Object Identification（IOI）

Background

这是大模型可解释性经典论文Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small中的一个实验。

对于一个经典的transformer模型（如GPT, Deepseek），它除了一个处理输入的"embedding"模块和一个处理输出的"unembedding"模块以外，中间会有很多长得完全一样的层，每一层里面又有两个重要的模块MLP和Attention。其中Attention一般使用“多头注意力”，也就是一个Attention它有多个"head"，研究表明每个head都有专属的功能，比如有些head的功能是识别颜色，有些head是识别形状等等。

“间接宾语识别”是一个有趣的任务，给定一句这样的话“When John and Mary went to the shops, Mary gave the bag to"，按照语言习惯，下一个单词应该是"John"，事实证明模型也会这样认为（你把这句话输入一个简单的模型，这个模型预测出的下一个词大概率是John）。一个合理的推断是“模型先识别出下一个词应该与第一个出现的名字John一样，然后再将John这个词的信息移动到输出的位置，然后输出”。无论模型的机理如何，在这个任务执行中attention在扮演了很重要的角色，我们想要做的就是找出哪些attention head在这个任务中被激活了，以及它们都是做什么的。

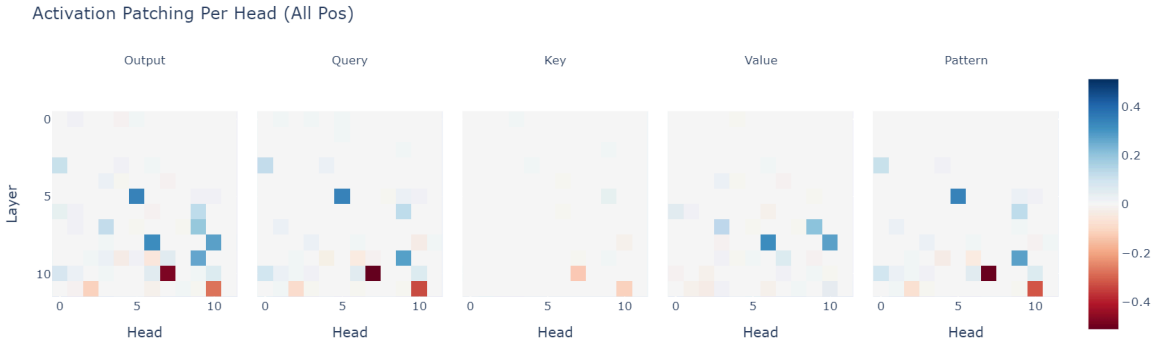
Methology and Results

我们使用的方法是可解释性领域一个经典的方法“激活值修补”（Activation patching），它的原理是制造一个正常的输入和损坏的输入（这里损坏的输入是指有意修改正常的输入，引导模型生成错误或相反的答案），然后将正确的输入放入模型，并缓存模型运行过程中的中间值（也就是正常的激活值）。然后我们再给模型一个损坏的输入，在模型运行的过程中使用缓存下来的某个部件的正常激活值替换掉正在运行的部件，然后继续运行，最后查看模型的输出从“损坏”向“正常”变化了多少。按理来讲，如果被替换的部件对该任务其正作用，那么替换以后模型的输出应该“向正常移动”；反之则会不移动甚至“向更损坏的方向移动”。

对于正常和损坏的输入，这里举一个例子。例如对于正常的输入When John and Mary went to the shops, Mary gave the bag to，对应损坏的输入应该是When John and Mary went to the shops, John gave the bag to，那么正常的输出应该是John，损坏的输出应该是Mary。如果我们用某个正常的attention head替换掉损坏的，然后发现损坏的运行得到了正常的结果（或者结果向正常的方向偏离了），那就说明这个attention head起到了某些作用。

对于衡量“输出向正常移动了多少”，我们使用一个线性指标logits diff，具体原理在这里不陈述了，但通过这个方法，我们可以得到一个介于-1到1的值。如果值为0则表明替换前和替换后没有变化，靠近1说明向正常靠近，靠近-1说明更加损坏了。

最终我们应该会得到这一堆图的第一个：



我们可以看出有一些head很蓝，这代表这些head对识别John，移动John起到了关键的作用，而还有一些Head很红，这代表这些“正常的head”反而会使答案更加损坏，一些论文推测这些Head是为了防止模型变得“过于自信”

Pipeline

我们使用经典的小型语言模型`gpt2-small`，这个模型层数（12层）和head数（每层12个）都不多，性能也完全可以执行我们的任务，部署占据的空间和推理占显存量也很小。

目前我想到的一个实验的pipeline如下所示：

1. 数据生成

   我们需要生成如When John and Mary went to the shops, Mary gave the bag to和When John and Mary went to the shops, John gave the bag to这样的输入对，显然可以通过先找一些人名，然后通过字符串操作进行插入；同时我们也可以使用不同的模型（API）来完成这个任务

2. 数据格式筛选

   对于模型生成的数据，我们需要通过正则表达式匹配来检查模型是否按照我们规定的格式生成了数据，筛除模型胡言乱语的case

3. 数据正确性筛选

   对于原始数据，我们必须要保证正确的输入会在GPT2-small上得到的是正确的输出，而损坏的输入得到的是损坏的输出。少数不匹配的输入输出（假如一个特定的case：When John

> and Mary went to the shops, Mary gave the bag to，模型输出了一个Mike）会干扰我们的探究，需要被剔除

4. 缓存激活值与修补激活值

> 这里有许多可操作的地方，不管是模型本地部署快还是autodl部署快，到底是4060pi快还是4090d快，一个一个case执行快还是一整个batch送进去一起做快等等

5. 画图分析

> 我们最终会得到一个[batch, layer, head]形状的三维矩阵，里面每个数代表着特定一个case，修补模型某一层的某个head后得到的logits diff。我们只需要在batch方向取平均，消除第零维，然后画个热力图就可以了