**你说一下为什么Transformer推理需要做KV的缓存呢？**

这一块的话主要是减少这个生成token的时候它的矩阵的运算。因为在Transformer当中里面，这个文本它是逐个token生成的，每一次新的预测它是基于之前生成的所有的token的上下文。这种的话它的顺序、数据的依赖呢会减少生成的过程。因为每次预测下一个token都需要重新处理序列当中，比如说我们要去预测第一百个token的时候，那么模型必须使用前九十九个token的信息，这个时候需要对这些token做矩阵的运算。而这个矩阵运算它是非常耗时的，所以说KV缓存就是为了减少这种耗时的矩阵运算。在推理的过程当中里面把这个键和值放在缓存当中里面，这样的话模型就可以在后续生成token的时候直接访问缓存，而不是重新去进行计算。

**那你既然提到了KV缓存可以减少矩阵运算量，那你具体说明一下它怎么做的？**

这里的话我画个图吧，因为通过画图来讲的话可能比较清晰一点，这张图的话分别是有缓存和没有缓存的情况下面。

对于第一个token，每次Q和K做矩阵相乘的时候，通过Softmax计算注意力得分，再和V相乘，然后进行最后的输出。因为第一个token，所以有没有缓存的计算量其实是一样的。不过下面这张图呢，就是把这一步计算过的K和V，然后通过第二第二个token，我们可以看到只是标注了这个就是没有缓存下面的这个K值和Y值。那没有缓存下面的这个KV需要重新计算的，如果是做了缓存的话，就需要把历史的KV拿出来，同时只需要计算新的token的KV，再拼接成一个大的矩阵就可以了。那对比下来，有没有缓存计算量是明显减少了一半。

然后的话，后面的token，后面的token的话也是一样的，每次历史计算过的键和值就不需要重新计算了。这样就极大的减少了自注意力机制的计算量，从序列长度的二次方直接变成了线性。

**那你说明一下我们在解码的时候，一般典型的KV Cache大模型的推理过程中，一般包含哪几个阶段呢？**

一共的话两个阶段，一个是Pre fill和Decoding两个阶段。

**那这两个阶段分别干什么的呢？**

Pre fill阶段的话会输出一个prompt序列，为每一个Transformer层申请一个PP缓存，同时的话输出第一个token。

而Decoding阶段呢，发生计算第二个token的时候到最后一个token的过程当中里面，这个时候Cache是有值的，所以每一轮推理只需要读取Cache，同时的话将当前计算出的新的K和Value追加写入到Cache当中就OK了。

这样的话Float降低矩阵，矩阵层操作变成了相连层操作，推理速度相比第一阶段的话它会变得更快。

**那你觉得我们一般如果说要去做KV Cache的缓存，会用什么样的数据类型去做这个储存张量呢？**

这一块的话通常是Float16或者BFloat16，十六位的精度。

**那你觉得如果我们要去算这个KV缓存消耗了多少显存，你会怎么去算呢？**

这个的话对于每个token，K V缓存的话为每一层和每个注意力图存储一张K V张量，K V张量的消耗的总显存的话可以通过这个公式来进行计算。这个公式的话我来简单的解释一下，最后的二是因为有两组张量，也就是键和值，然后的话位宽通常十六位等于八位一个字节嘛，所以说除以八这样的缓存中，每十六位参数会占用两个字节。