\*Attention

为什么要用Attention： Attention是一种整合有用数据的机制，亦或是一种操作方法，其有较为固定的流程（套路）。对模型而言，需要知道数据，词条的哪个重要（关联度），以及其与上下文的关系，才能更准确的理解语义，生成内容。Attention能表达出与上下文有关联信息的词向量，在训练模型中更准确，意义重大。

Attention的基本原理： Attention 注意力机制本质就是QKV三个向量点乘（求内积），多数情况K约等于V，即二者同源。只是注意力机制先让Q，K（k1，k2，kn）点乘得到相似值s1，s2，sn，再除以一个常数缩小差距，然后做softmax得到（ 对于Q而言，）K的相对概率a1，a2，an。这样，（在模型中就可以）体现出那个词有关联，更需要注意，进而给予差异化的关注。然后我们对生成的a点乘v这样就得到了新的，更准确的V值，这样K，V中的关键点就被Q标注出来了。

Self-Attention： 而self-Attention自注意力机制与Attention的区别就在于它的Q也是跟KV同源。假设QKV都源于X（一个语句）。这样自注意力作用抓取对X语句里的每一个单词取空间变换的Q而言，X语句中的关键点。换言之，在I have a dream这个语句中，我have这个单词与哪几个单词关联度高，这个信息就被我包含在更新后的词向量中了。

Attention的分类【本质是根据功能分类】： 另外，Attention的定义是一种包含了QKV三个变量的，让它们以规定的套路点乘的一大类操作机制。而self-Attention是它旗下的子集，它额外添加了对QKV的来源的定义，即X分别乘以不同的参数矩阵。同属于a这一大类下的子集还有Cross-Attention，它强调Q不与KV同源。

Attention相对于RNN和LSTM的优劣势： Attention的计算量是非常庞大的通常，因为他要计算每个词与句子里的所有词的点积。虽然相对于RNN（和LSTM）他有处理长序列的先天优势，但庞大的计算量和成本让他在处理大于50个词的语句效率不一定如LSTM。还有一个值得一提的优点就是他的计算可以做并行的，(每个词与其他词的点积的过程可以同时进行)，有效利用电脑算力，可以节省时间。

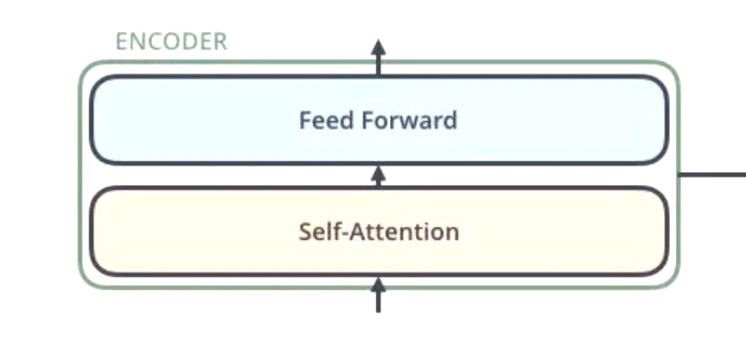
Multi head self-Attention（多头自注意力机制）和Masked self-Attention（掩码自注意力机制）：

Multi head self-Attention（多头自注意力机制）其效果是让self-Attention对词向量的改进基础上，进一步生成更合理准确的词向量。其做法为把多头数（h）设置为8，把原始词向量X分为8块，得到8个新的词向量拼接起来，再做线性变换，让维度与X相同。其原理是相对于1个空间向量X通过非线性变换得到准确词向量表达，把这个X分为空间里的8个点再合并拼接要更准确。

Masked self-Attention（掩码自注意力机制）是对self-Attention的进一步改造，让其适应生成模型的特定场景（比如完成特定生成任务）。我们要对生成模型生成的单词做注意力机制，这样可以动态捕捉输出和输入的关联性以便调整，让生成单词互相做Attention还可以避免信息均等化。在对生成单词实时做注意力机制时，我们的情况较self-Attention有所变动，因为单词是一个一个生成的，无法一次完整的给一整个句子的单词，所以我们用掩码把未生成的给masked掉，用已生成的和之前的做Attention——一个单词一个单词生成的顺序来，最后一个单词生成后再给全完整的矩阵。

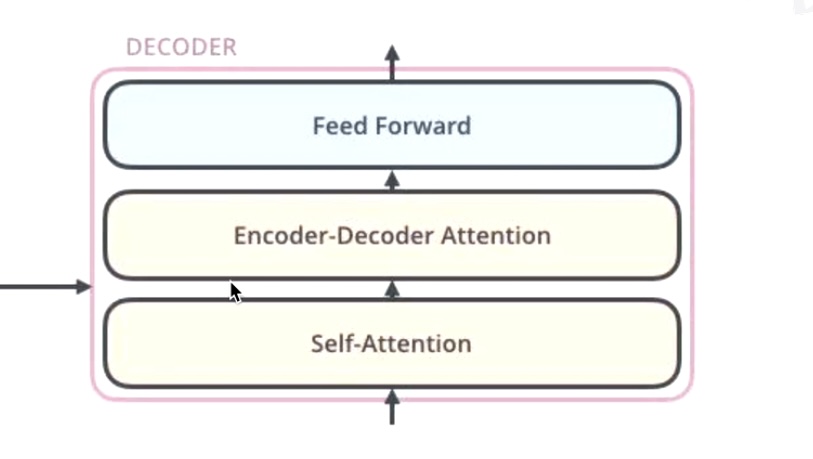
\*transformer（decoder是生成（翻译）任务）： 预训练模型的进化过程中，Attention的出现为transformer奠定了基础，因为transformer本质就是attention的堆叠：让模型学习，调整，更新参数（反向更新）。

注意：transformer每层信息传输过程中箭头都代表经过一次残差网络和归一化（LayNorm）。前者防止梯度消失，后者防止梯度爆炸。



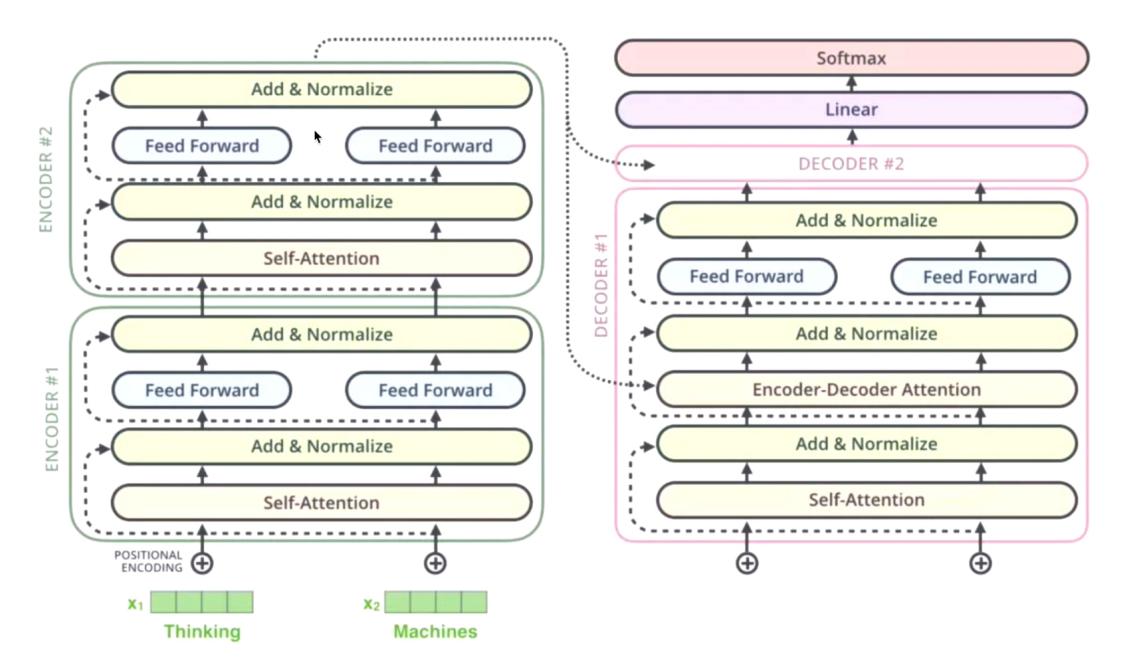
编码器（encoder） to 解码器（decoder）： 堆叠6个encoder就是堆叠self-Attention，以此一步一步强化词向量的准确性。Feed Forward就是一层激活函数，这样做了很多层线性变换后，做一层非线性变换相当于激活前面的变换工作：这样可以保证一定的无规则空间变换，就可以无限拟合空间里的任何一种状态。源语句（要翻译的）输入进来之前先要经过词间位置编码，这样self-Attention输出的就是涵盖X1在语句中的位置属性和句法特征，语义特征（self-attention自带的效果）。

Encoder里的“multi head self-Attention”： 把一句话拆成不同的部分（单词），每个单词与一句话做Attention，得到Z1。再把Z1，Z2拼接起来，得到语句的向量。



Decoder中的Masked self-Attention和cross-Attention： 在训练过程中，decoder下面的输入是masked后的目标语句，这里的self-Attention作用是编码，更新已生成的目标语句的词（相对应翻译过去的那个词）相较于源语句的词向量。

Masked self-Attention就是保证训练阶段和测试阶段的一致性，不让训练阶段所有信息都给decoder的第一个self-Attention，让这两个阶段decoder的过程一模一样。注意cross-Attention，encoder提供K，V；decoder提供Q，这样在翻译过程中，模型就可以学习对目标（结果）语句，源语句（需要翻译的）中除了自己，哪个词对我翻译出这个结果更有意义（即更有关联度，这样调整参数）。这两个Attention可以提升训练效果，训练效率。



图中Add即+X防止残差消失，Normalize即归一化，控制范围，防止梯度爆炸，便于做非线性变换的Relu（）函数（或softmax）操作。

最后linear是变换维度，列出所有词向量；然后再由softmax做归一化概率分布，生成相应概率最大的词。

Transformer强大的原因归根到底是self-Attention和multi head self-Attention。重复的叠加，只要有算力的支撑，它就会很准确，高效。

简述对DeepSeek，Gpt，LLama使用体验：

Llama应该是在AI at Meta官网里美国账户可以使用，而chat GPT已商业化，除美洲以外，推广到香港，中南亚，等地。由国内开发团队研发的deepseek是全球都能使用，现今，服务器繁忙的问题也修复并优化了。Deepseek我最常用的就是深度思考功能，它可以随多个照片或文件的传入，识别读取并做深度思考。同对于普通用户，deepseek思考数学，推理问题相对于gpt 4o和o3-mini的准确率明显更高。而GPT要更擅长生成性任务，但体验下它生成的内容较同质化，实质性自己思考结果不多，而且内容不长，无法生成长文段。给的特定场景的任务（如生成实验报告，创新性地生成历史剧剧本）要多次分点描述要求后，通过引导性指令，反复修改才能初步达到预期的效果。包括一些创新性的数学题，逻辑题，deepseek和gpt都无法准确一次性抓取题目要求，精准回答问题。

总之，这两个模型还无法通过简短的命令和要求短时间达到人类的要求，或完全替代人类思考计算。目前AI也不是说能有多创新地帮助科研，它优势是①可以搜集全网资料信息，背后有大量数据支撑，它可以总结调用你所需的知识，让人省去繁忙的搜索链，让人更容易产生自己的理解并深度思考。②根据已有知识，不同层面的知识推理。通过逻辑链推理思考，然后得出正确结果。