传统的卷积神经网络需要大的感受野不断地卷积，才能获得更大的感受野，但是在 transformer 模式下，由于 attention 计算时会看到所有的信息，即感受野为整张图片，所以不需要堆叠层来增大感受野，此时可以直接获得全局信息。

两个 Transformer 在图像中的经典应用：

1. IGPT
2. VIT 《An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale》：先将图像拉平（3D->2D）切片，然后使用可训练的线性投影映射输出 patch embedding（类似 NLP 中的 word embedding，其目的是使计算机能识别）。再将 patch embedding 接入 transformer 的 encoder 块（包含 MHA，MLP：MSA 得到用于分类的 token（可以视为 feature map）接入 MLP），最后输出概率向量，其中概率最大，即相似度最强，完成分类。

总的来说，在计算机视觉（CV）中，卷积体系结构仍然是 SOTA（目前最先进的）。对于少量数据集，如中等规模数据集（imagenet），基于 transformer 的模型对于 CV 得到的精度比同等规模的 Resnet 低几个百分点，这是因为 transformer 缺乏 CNN 固有的一些感应偏差，如平移不变性和局部性，因此在训练不足的数据量时不能很好地概括。而在大数据集上，基于 transformer（如 VIT）在图像识别中达到了基准或超越了最新水平。

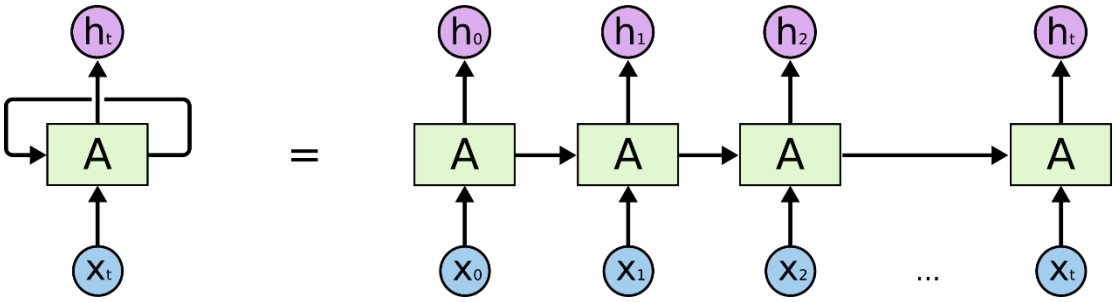
NLP 下游任务主要分为：`NLU` 自然语言理解,`NLG` 自然语言生成

发展历程（在 NLP 中）：
RNN、LSTM-->attention、self-attention-->multi-head self attention-->transformer-->GPT、BERT

1. RNN、LSTM

1.1

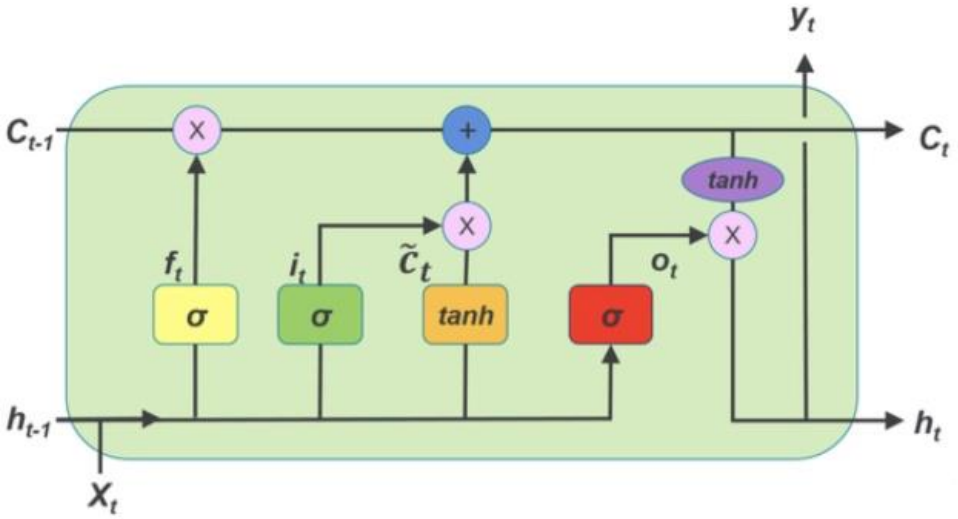
传统的神经网络（CNN）无法获取时序信息，所以 RNN 被设计用来处理时序信息，其基本单元如下：



由基本单元可知，每一个模块都会把当前的信息传递给下一个模块，因此解决了短距离时序信息依赖的问题，但是在长距离中，RNN 容易出现梯度消失（反向梯度传播到底层时是很多个梯度的链式相乘，若梯度小于 1，那么到底层的梯度就会接近 0，从而发生梯度消失），因此导致难以获取长距离信息。

1.2

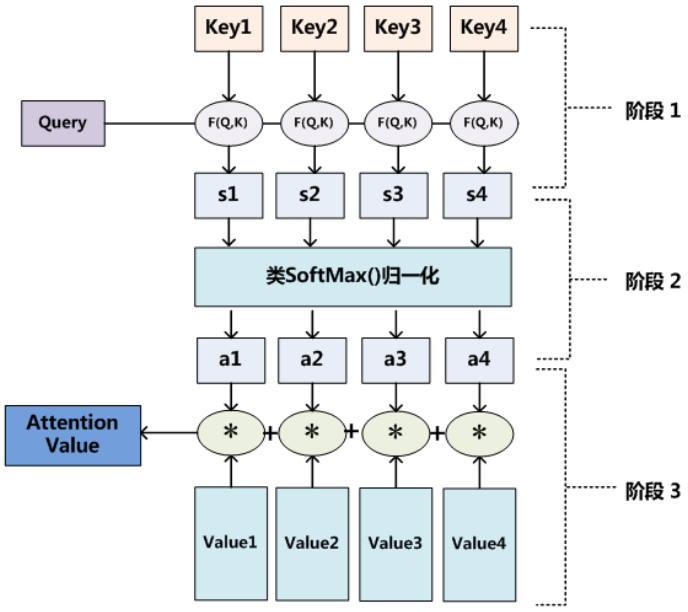
LSTM 解决了 RNN 中梯度消失的问题，它使反向传播的每一步梯度始终大于 1。LSTM 模型参数量约为 RNN 的 4 倍，因此它可以更加精细地进行时序处理，但当文本在 200 左右，LSTM 也捉襟见肘了。LSTM 结构如下：



2. Attention & Self-attention

2.1

Attention 机制分为 3 个向量，Query（查询表）、Key（键）、Value（值），Key-Value 是数据 X 的一个 key-pairs，attention 就是算 Q 与 V 的重要（相似）程度，看对于 Q 来说，哪些信息是值得注意的。其计算流程如下：



为了防止 softmax()出现 overflow 的情况，所以对 softmax 的输入都除以$\sqrt{d_k}$(d_k表示维度)，

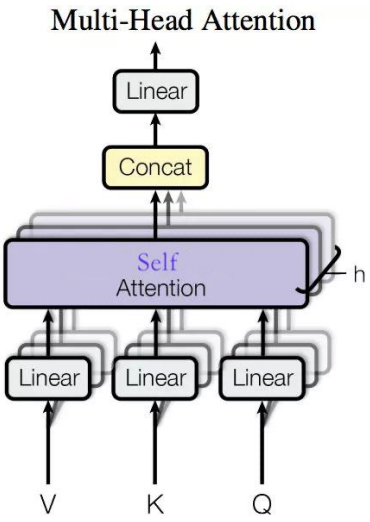
attention 经过 softmax 后输出的是每个 token（NLP 中的最小词元，可以是一个单词、标点等）相对于此时 Q 的重要程度分数，而 attention value 则是分数和 value 的加权和。Attention 机制可以更加好的解决序列长距离依赖问题，并且具有并行计算能力。

2.2

Self-attention 就是将 Attention 中的 Query、Key、Value 都是同源的，即都来自数据 X。但是 Q、K、V 是不等的，因为它们会由三个可学习的权重矩阵线性变化（即如 $Q=Q*W_Q$）。简单的说，Self-attention 就是算 source 中每个 token 相对于其它所有 token 的重要程度。

3. Multi-head self attention

多头自注意力机制是自注意力机制的进阶版，它就是为了捕捉到不同子空间的信息，将输入数据 X 分成了多份，然后分别用多个不同的 self-attention 去处理，再将得到的不同加权和值 merge 起来就行了。注意：不同 self-attention 块中的三个线性变化权重矩阵通常是不一样的。多头注意力结构如下：



4.Transformer

Transformer 其实就是 self-attention 的叠加。Transformer 是一个 encoder-decoder 结构。

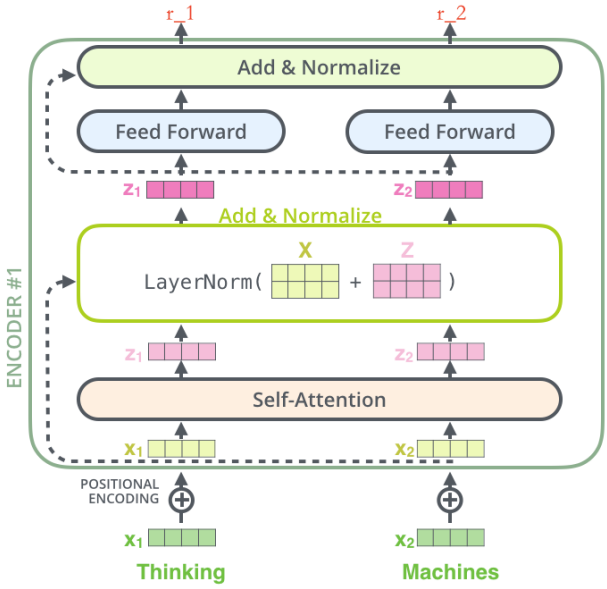
4.1 Encoder

通常每层 encoder 包括两个 sub-layer:

（1） Multi-head self-attention

（2） 前馈神经网络（感知机）

Encoder 流图如下:



4.2 Decoder

通常每层 decoder 包括三个 sub-layer:

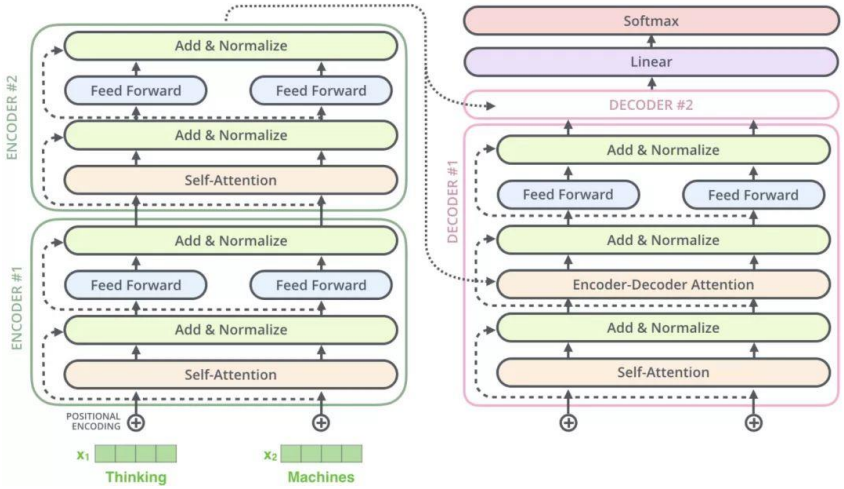
（1） Masked Multi-head self-attention（为了训练过程和测试过程相匹配，所以要 mask）

（2） Multi-head self-attention

（3） 前馈神经网络（感知机）

注意：对于 decoder，训练阶段是能并行的（即一个 time step），而测试阶段只能是串行的。

Encoder 和 decoder 的交互流图如下：



5. GPT

两阶段过程：

（1） 无监督预训练语言模型

（2） Fine-tuning 解决下游任务

GPT 采用的是单向语言模型，即它只会用上文信息来进行预测，它采用了无监督的预训练方法，不需要人工标注的数据，GPT-3 是在 webtext 数据集训练的。GPT 使用 transformer 的 decoder 作为特征提取器、具有良好的文本生成能力。

6. BERT

BERT 使用了 transformer 的 encoder 作为特征提取器，并使用了掩码训练的方法：

掩码方法指的是随机掩盖一些 token，然后被遮挡的 token 就作为标签进行训练。这样就是通过预料信息自动产生标签，所以训练数据就可以想要多少就有多少，从而训练出一个很大的模型。BERT 是从大量无标记数据集中训练得到的深度模型，可以显著提高各项 NLP 任务的准确率。需要注意的是 BERT 采用的策略是：80%用 MASK 替代；10%不发生变化，该做法是为了缓解训练文本和预测文本的偏差带来的性能损失；10%将选中的词用任意词来代替，该做法的为了让 BERT 学会根据上下文信息自动纠错。

BERT 是双向语言模型，虽然这样使它文本生成（这里的文本生成指的是只看上文信息得到下文的一种 NLP 下游任务）能力较差，但是它提取语义信息（特征提取）的能力就变强了。