Transformer详解

在Transformer之前，多数基于神经网络的机器翻译方法依赖于循环神经网络(RNN)，后者利用循环（即每一步的输出馈入下一步）进行顺序操作。尽管RNN在建模序列方面非常强大，但其序列性意味着该网络在训练时非常缓慢，因为长句需要更多的训练步骤，其循环结构也加大了训练难度。与基于RNN的方法相比，Transformer不需要循环，而是并行处理序列中的所有单词，同时利用自注意力机制将上下文与较远的单词结合起来。通过并行处理所有单词，并让每个单词在多个处理步骤中注意到句子中的其他单词，Transformer的训练速度比RNN快很多，而且其翻译结果也比RNN好的多。

一、总体流程

模型结构如下：

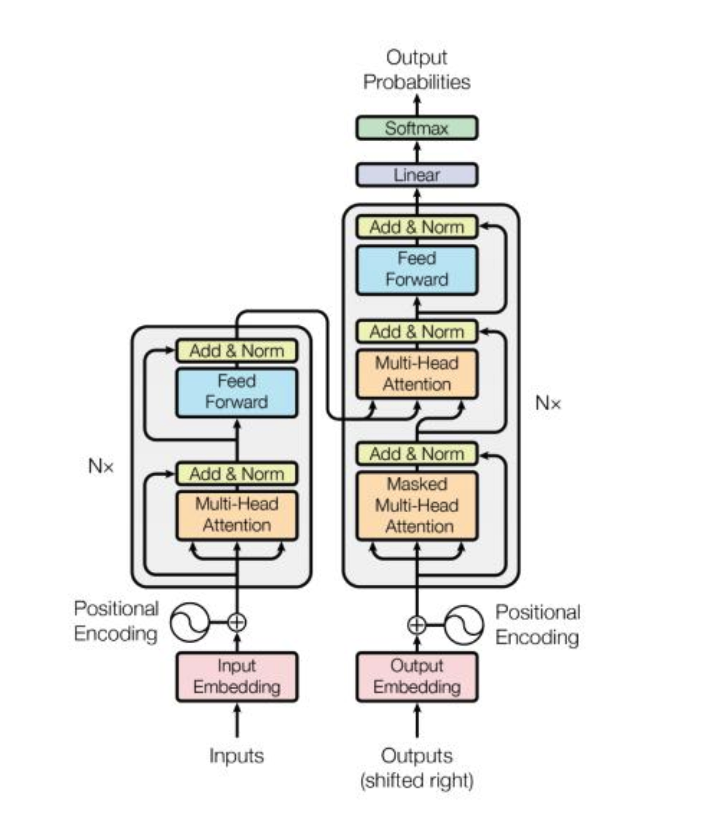


图1： Transformer结构

和大多数seq2seq模型一样，transformer的结构也是由encoder和decoder组成。

* Encoder：由N=6个相同的层组成，每一层包含两个sub-layers，第一个sub-layer是多头注意力层（multi-head attention layer），。然后是一个简单的全连接层。其中每个sub-layer都加了residual connection（残差链接）和normalization（归一化）。
* Decoder：由N=6个相同的层组成，但这里的每一层包含三个sub-layer，其中一个是self-attention layer，还有一个encoder-decoder attention layer，最后一个是全连接层。前两个层都是基于multi-head attention layer。这里有个特点就是masking，masking的作用就是防止在训练的时候使用未来的输出单词。比如训练时，第一个单词是不能参考第二个单词的生成结果的。Masking会把这个信息变为0（第二个单词的输出结果），用来保证预测位置i的信息只能基于比i小的输出。

如上所示，Decoder和Encoder的结构差不多，但是多了一个encoder-decoder attention的sub-layer，这里明确一下decoder的输入输出和解码过程：

* 输出：对应i位置的输出词的概率分布
* 输入：encoder的输出、对应i-1位置的decoder的输出，所以图1中间位置的attention（第二个的attention）不是self-attention，它的K、V来自encoder，Q来自上一位置的decoder的输出。
* 解码：**这里要特别注意一下，编码可以并行计算，一次性全部encoding出来，但是解码不是一次把所有序列解出来的，而是像rnn一样一个一个解出来的，因为要用到上一个位置的输入当做attention的query。**

**二、Attention**

2.1 Scaled dot-product attention

Scaled dot-product attention的具体过程如下图2所示，其输入由维度为d的查询(Q)和键(K)以及维度为d的值(V)组成，所有键计算查询的点积，并应用 softmax 函数获得值的权重。

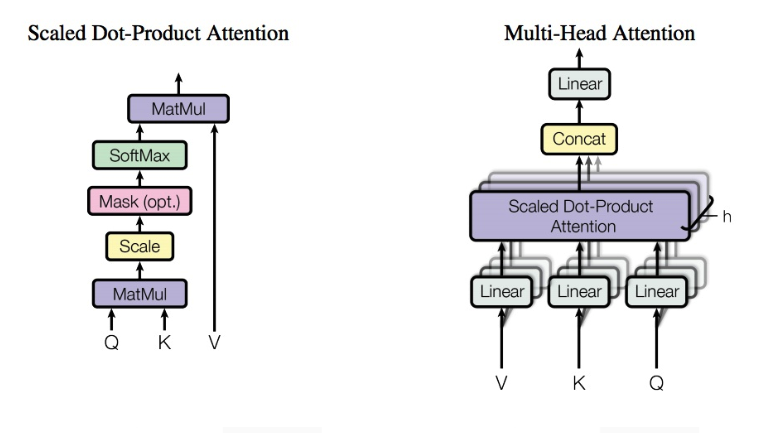


图2：两种Attention的实现框图

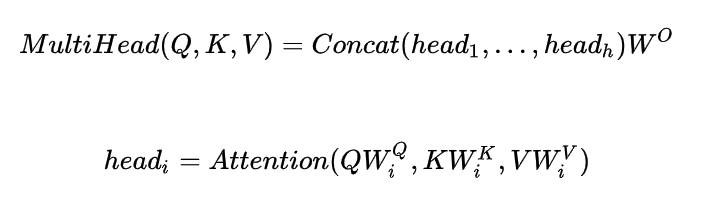
Scaled dot-product attention具体的操作有三个步骤：

* 每个query-key 会做出一个点乘的运算过程，同时为了防止值过大除以维度的常数。
* 最后使用softmax将他们归一化
* 再到最后会乘以V用来做attention vector。

数学公式如下：

2.2 Multi-head attention

上面介绍的scaled dot-product attention，看起来还有点简单，网络的表达能力还有一些简单所以提出了多头注意力机制（multi-head attention）。Multi-head attention则是通过h个不同的线性变换对Q,K,V进行投影，最后将不同的attention结果拼接起来，公式如下：



Self-attention则是取Q,K,V相同。论文中使用了8个平行的注意力层或者头部。

2.3 Position-wise feed-forward networks

第二个sub-layer是个全连接层，之所以是position-wise是因为处理的attention输出是某一个位置i的attention输出。全连接层公式如下所示:

2.4 Positional Encoding

除了主要的Encoder和Decoder，还有数据预处理的部分。Transformer抛弃了RNN，而RNN最大的优点就是在时间序列上对数据的抽象，所以文章中作者提出了两种Positional Encoding的方法，将encoding后的数据与embedding的数据求和，加入了相应位置信息。

三、优点

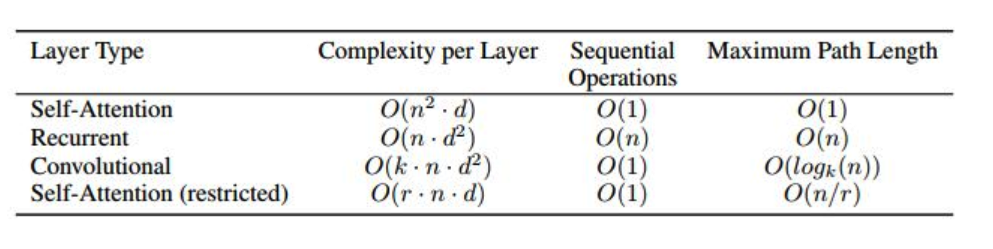


图3：模型对比

1. 每层计算复杂度
2. 作者用最小的序列化运算来测量可以被并行化的计算。也就是说对于某个序列, self-attention可以直接计算的点乘结果，而rnn就必须按照顺序从计算到。
3. Path Length：这里的path length值的是要计算一个序列长度为n的信息要经过的路径长度。Cnn需要增加卷积层数来扩大视野，rnn需要从1到n逐个进行计算，而self-attention只需要一步矩阵计算就可以了。所以也可以看出来，self0-attention可以比rnn更好的解决长时依赖问题。当然，如果计算量过大，比如序列长度n>序列维度d这种情况，也可以用窗口限制self-attention的计算数量。