# Transformer论文笔记

## 摘要

占主导地位的序列转导模型是基于包含编码器和解码器的复杂循环或卷积神经网络。表现最好的模型还通过注意力机制连接编码器和解码器。作者提出了一种新的简单的网络架构Transformer，它完全基于注意力机制，不需要循环和卷积。在两个机器翻译任务上的实验表明，Transformer在质量上更胜一筹，同时具有更好的并行性，训练时间也显著减少。作者的模型在WMT2014英德翻译任务上取得了28.4 BLEU，比现有的最好结果（包括集成神经网络的模型）提高了2 BLEU。在WMT 2014英法翻译任务上，作者的模型在8个GPU上 训练3.5 天后，建立了一个新的单模型，其BLEU评分为41.8，训练成本是最好模型的一小部分。作者通过将Transformer成功地应用于具有大量和有限训练数据的英语短语结构分析，表明Transformer能很好地推广到其他任务。

像循环神经网络的输出由上一个输出和当前输入决定，不具有很好的并行性，只能串行处理。

## 一、导论

循环神经网络，特别是LSTM和GRU神经网络，已经成为时序模型和像语言建模、机器翻译等转导问题的主流方法。它们主要使用语言模型和编码器-译码器架构来构建。

循环神经网络模型通常沿着输入和输出序列的符号位置进行计算，每一步生成一个隐藏状态序列ht，作为前一个隐藏状态ht - 1和位置t输入的函数。这种固有的顺序性限制了训练实例内部的并行化，而并行处理在更长的序列长度上十分重要，因为内存约束限制了跨样本的批处理。通过因子分解技巧和条件计算在计算效率方面取得了显著的改进，同时也提高了模型性能。然而，顺序计算的基本约束依然存在。

注意力机制已经成为各种任务中引人注目的序列建模和转导模型的组成部分，允许在输入或输出序列中不考虑它们之间的距离的依赖关系建模。然而大多数情况下，这类注意力机制都与递归网络结合使用。

像卷积神经网络，由于卷积核的机制只能处理相邻的信息，不能像Transformer一样不用考率输入序列之间的距离。

在本文中，作者提出了Transformer，一种不使用循环的模型架构，完全依靠注意力机制来绘制输入和输出之间的全局依赖关系。Transformer允许更好的并行化。

## 二、背景

减少序列计算量的目标也形成了Extended Neural GPU、ByteNet和ConvS2S等网络，它们都使用卷积神经网络作为基本构建模块，对所有输入和输出位置并行计算隐藏表示。在这些模型中，将来自两个任意输入或输出位置的信号关联起来所需的操作数量随着位置之间的距离而增加，对于ConvS2S是线性的，而对于Byte Net是对数增长的。这使得学习远距离位置之间的依赖关系变得更加困难。在Transformer中，这被减少到恒定的操作次数，尽管代价是平均注意力加权位置导致有效分辨率降低，作者使用多头注意力来抵消这种影响。

自注意力，有时也称为帧内注意力，是一种将单个序列的不同位置联系起来以计算序列的表示的注意力机制。自注意力已成功应用于多种任务，包括阅读理解、抽象概括、文本蕴含和学习任务无关的句子表征。

简单理解自注意力就是以往的输出作为输入的一部分和当前输入来产生新的输出。

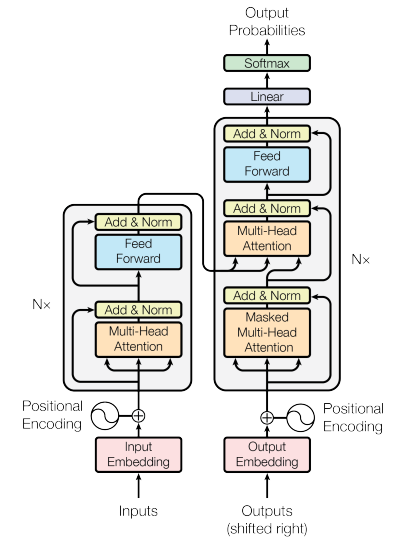
端到端记忆网络基于循环注意力机制而非序列对齐的循环，在简单语言问答和语言建模任务中表现良好。

Transformer是第一个完全依靠自注意力计算其输入和输出表示的转导模型，而不使用序列对齐的RNN或卷积。

## 三、网络架构

大多数有竞争性的神经序列转导模型都具有编码器-解码器结构。这里，编码器将输入的符号表示序列 (x1, …, xn) 映射为连续表示序列z = (z1, …, zn)。解码器根据z一次产生一个符号元素的输出序列( y1, … , ym)。在每一步中，模型都是自回归的，在生成下一步的符号时，将先前生成的符号作为额外的输入。Transformer遵循这种整体架构，编码器和解码器都使用堆叠的自注意力和全连接层，分别下图的左半部分和右半部分所示。

简单理解自回归就是以往的输出作为输入的一部分和当前输入来产生新的输出。



### 编码器和解码器

编码器由N = 6个相同层堆叠而成。每层有两个子层。第一个子层是多头自注意力机制，第二个子层是简单的全连接前馈网络。作者在两个子层上分别使用了残差连接，然后进行层归一化。即每个子层的输出为LayerNorm( x +子层( x) )，其中子层( x )为子层自身实现的函数。为了便于这些残差连接，模型中的所有子层以及嵌入层都产生维度为dmodel = 512的输出。

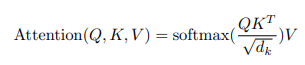
解码器也是由N = 6个相同层堆叠而成。除了每个编码器层中的两个子层外，解码器还插入了第三个子层，对编码器的输出进行多头注意力。与编码器类似，作者在每个子层上使用了残差连接，然后进行层归一化。还修改了解码器中的自注意力子层，以防止该位置出现在后续位置。这种屏蔽和输出嵌入被一个位置偏移的事实，确保了对位置i的预测只能依赖于小于i位置的已知输出。

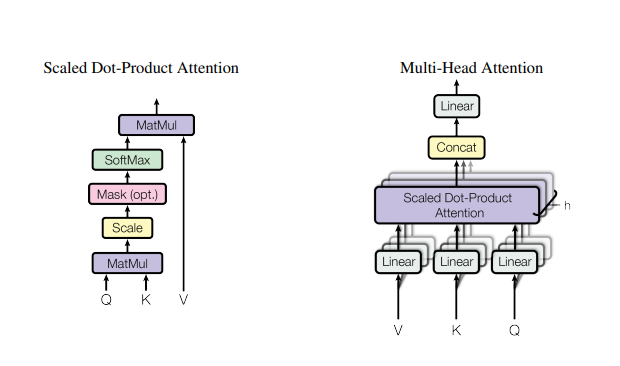
### 注意力机制

注意力函数可以描述为将一个查询和一组键值对映射到一个输出，其中查询、键、值和输出都是向量。输出被计算为值的加权和，其中分配给每个值的权重通过查询与相应键的配对函数计算。

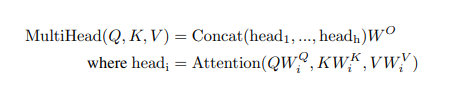
### 点积注意力

输入由维度为dk的查询和键与维度dv的值组成。计算查询的所有键的点积后，除以，并应用softmax函数来获得值上的权重。

在实际中，可以同时计算一组查询上的注意力函数，并打包成矩阵Q，将键和值也打包成矩阵K和V，输出矩阵如下：

最常用的两种注意力函数是加法和点积。点积注意力与上面的算法完全相同，只是比例因子为。加法注意力使用带有单隐藏层的前馈网络计算相容性函数。虽然两者在理论复杂度上相似，但点积注意力在实际应用中速度更快，空间效率更高，因为它可以使用高度优化的矩阵乘法代码实现。而对于较小的dk值，两种机制的表现类似，对于较大的dk值，相加注意力优于无尺度的点积注意力。作者猜想，对于dk的大值，点积增长的幅度很大，将softmax函数推进到具有极小梯度的区域。为了抵消这种影响，作者对点积进行缩放。

### 多头注意力

与使用dmodel维度的键、值和查询执行单个的注意力函数不同，作者发现将查询、键和值的h次线性投影到dk、dk和dv维度是有益的。在每个投影版本的查询、键和值上，并行地执行注意力函数，产生dv维输出值。将这些进行拼接并再次投影，得到最终值。多头注意力允许模型在不同位置共同关注来自不同表征子空间的信息。在单个注意头的情况下，平均抑制了这一点。

在本文中，作者采用了h = 8个平行的注意力层。其中，k = dv = dmodel / h = 64。由于每个头的维度降低，总的计算成本与全维度的单头注意力相似。

### 注意力机制在Transformer模型中的应用

Transformer以三种不同的方式使用多头注意力：

* 在"编码器-解码器注意力"层中，查询来自前一个解码器层，键和值来自编码器的输出。这使得解码器中的每个位置都可以参与输入序列中的所有位置。
* 编码器包含自注意力层。在一个自注意力层中，所有的键、值和查询来自同一个地方，在这种情况下，编码器中前一层的输出。编码器中的每个位置都可以参与到编码器前一层的所有位置。
* 类似地，解码器中的自注意力层允许解码器中的每个位置关注解码器中的所有位置，包括该位置。由于需要在解码器中防止向左的信息流动以保持自回归特性。通过掩盖softmax输入中所有与非法连接对应的(设置为-∞)值来实现点积缩放注意力。

### 前馈网络

除了注意力子层之外，编码器和解码器中的每一层都包含一个全连接的前馈网络， 分别且相同地应用于每个位置。这包含两个线性变换，ReLU激活函数介于两者之间。

虽然线性变换在不同的位置是相同的，但它们使用不同的参数。描述这一点的另一种方式是两个核大小为1的卷积。输入输出维数dmodel = 512，内层维数dff = 2048。

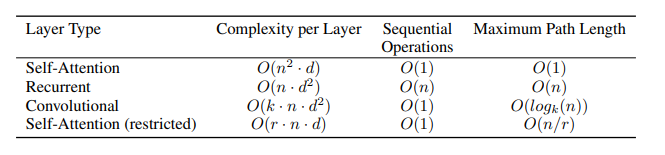
### 嵌入层和Softmax

嵌入层将输入和输出的标记转换为d维向量。还使用线性变换和softmax函数将解码器输出转换为预测的下一个token概率。在Transformer模型中，两个嵌入层和预softmax线性变换之间共享相同的权重矩阵。在嵌入层，这些权重乘以dmodel。

# 四、使用自注意力的原因

本节将自注意力层的三个方面与通常用于将一个可变长度的符号表示序列映射到另一个序列的循环层和卷积层进行比较，如典型的序列转导编码器或解码器中的隐藏层。一是每层的总计算复杂度。另一个是可以并行化的计算量，用所需的最少顺序操作数来衡量。第三是网络中长程依赖之间的路径长度。

在许多序列转导任务中，学习长距离依赖是一个关键的挑战。影响这种依赖关系学习能力的一个关键因素是网络中前向和后向信号的路径长度都是横贯的。输入和输出序列中任意位置组合之间的路径越短，就越容易学习到长程依赖关系。因此，我们还比较了不同层类型构成的网络中任意两个输入和输出位置之间的最大路径长度。



# 五、训练

### 训练数据与批处理

句子对通过近似的序列长度被批处理在一起。每个训练批次包含一组包含大约25000个源token和25000个目标token的句子对。

### 优化器

使用Adam优化器，β1 = 0.9，β2 = 0.98，根据公式在训练过程中改变学习率：

对应于第一个warmup\_steps训练步中线性地增加学习率，而在此之后按与步数的倒数平方根成比例地减少学习率，这里warmup\_steps= 4000。

### 正则化

在训练过程中使用了三种类型的正则化：

* 残差Dropou：将dropout应用于每个子层的输出，然后将其添加到子层输入并进行归一化。
* 此外，对编码器和解码器堆栈中的嵌入和位置编码之和使用dropout。对于基准模型，取Pdrop = 0.1。
* 标签平滑：在训练过程中，我们使用了∈ls = 0.1的标签平滑。这增加了噪声，模型学会了不确定，但提高了准确性和BLEU评分。

# 六、结论

Transformer是第一个完全基于注意力机制的序列转导模型，用multi-header注意力机制替换编码器-解码器架构中最常用的循环层。对于翻译任务，Transformer的训练速度明显快于基于循环或卷积层的架构。在WMT2014英译德和WMT2014英译法的翻译任务上，Transformer都达到了新的水平。在前一项任务中，作者的最佳模型甚至优于所有以前报道的模型。

作者对注意力模型的未来感到兴奋，并计划将其应用于其他任务。作者计划将Transformer扩展到涉及除文本以外的输入和输出问题，并研究局部的、受限的注意力机制来有效地处理大的输入和输出，如图像、音频和视频。使生成的序列时序化不是那么高是作者的另一个研究目标。

* 论文试图解决什么问题？

**1、传统的循环神经网络（RNN）在处理长序列时存在的梯度消失和梯度爆炸问题。这个问题会导致长序列的预测精度下降，因为当梯度在反向传播过程中被反复相乘时，小的梯度会逐渐消失，而大的梯度则会增长，这会干扰模型的训练和性能。**

**2、传统的卷积神经网络（CNN）虽然可以有效地处理图像等局部特征，但在处理自然语言处理（NLP）等序列数据时，无法捕获序列数据中的长期依赖关系。这是因为卷积操作只对局部区域进行操作，无法像RNN那样对整个序列进行全局信息交互。**

**3、RNN的并行性问题**

* 这是否是一个新的问题？

**在Transformer之前，传统的循环神经网络（RNN）和卷积神经网络（CNN）在处理序列数据时存在一些问题，如梯度消失和梯度爆炸、无法捕捉序列数据中的长期依赖关系等。虽然在这之前已经有一些工作试图解决这些问题，例如使用LSTM和GRU等门控机制，但是Transformer提出了一种全新的解决方案，通过自注意力机制实现对序列数据的全局信息交互，从而更好地捕捉序列数据中的长期依赖关系。**

* 这篇文章要验证一个什么科学假设？

**通过使用自注意力机制和残差连接等技术，可以有效地处理序列数据，并提高模型对数据的理解能力和生成能力**

* 有哪些相关研究？如何归类？谁是这一课题在领域内值得关注的研究员？

**注意力机制在该论文之前大多与RNN结合使用，而Transformer则避免了递归网络，所以依据是否使用递归来进行分类；Ashish Vaswani：Vaswani是Transformer模型的主要贡献者之一**

* 论文中提到的解决方案之关键是什么？

**自注意力机制、多头注意力机制等**

* 论文中的实验是如何设计的？

**在翻译质量和训练成本与论文中的其他模型架构进行了比较**

* 用于定量评估的数据集是什么？代码有没有开源？

**WMT英德数据集；代码开源**

* 论文中的实验及结果有没有很好地支持需要验证的科学假设？

**实验结果表明，Transformer在机器翻译方面远超其他模型，而且模型的泛化性能很好**

* 这篇论文到底有什么贡献？

**提出了自注意力机制、多头注意力机制和Transformer等新方法和模型。**

* 下一步呢？有什么工作可以继续深入？

**模型压缩、可解释性、模型优化、跨模态数据转换（如图文和音频之间的转换）。**