注意力是你所需要的

Ashish Vaswani*谷歌大脑
avaswani@google.com

Noam Shazeer*谷歌大脑
noam@google.com

Niki Parmar*谷歌研究部
nikip@google.com

Jakob Uszkoreit*谷歌研究
usz@google.com

Llion Jones*谷歌研究
llion@google.com

Aidan N. Gomez* t多伦多大学aidan@cs . toronto. edu

Łukasz Kaiser*谷歌大脑
lukaszkaiser@google.com

伊利亚·波洛苏金*?
illia.polosukhin@gmail.com

摘要

主流的序列转导模型基于复杂的递归或卷积神经网络，包括编码器和解码器。表现最佳的模型还通过注意力机制连接编码器和解码器。我们提出了一种新的简单网络架构——Transformer，完全基于注意力机制，摒弃了递归和卷积。在两个机器翻译任务上的实验表明，这些模型不仅质量更优，而且更加并行化，训练时间显著减少。我们的模型在WMT 2014英德翻译任务中达到了28.4的BLEU值，比现有的最佳结果，包括集成方法，提高了超过2个BLEU值。在WMT 2014英法翻译任务中，我们的模型经过3.5天的训练，在八个GPU上达到了41.8的新单模型最先进BLEU分数，这仅占文献中最佳模型训练成本的一小部分。我们展示了Transformer在其他任务上的良好泛化能力，成功应用于英语成分句法分析，无论是在大量还是有限的训练数据下。

*等额贡献。列表顺序随机。雅各布提议用自注意力机制替代循环神经网络，并开始评估这一想法。阿什什与伊利亚设计并实现了首个Transformer模型，在这项工作的各个方面都发挥了关键作用。诺姆提出了缩放点积注意力、多头注意力和无参数位置表示，几乎参与了每一个细节。妮基设计、实现、调优并评估了我们原始代码库和tensors2tensors中的无数模型变体。利昂也尝试了新的模型变体，负责我们的初始代码库以及高效的推理和可视化。Lukasz和Aidan花了无数漫长的日子设计和实现tensor2tensor的各个部分，取代了我们早期的代码库，极大地提高了结果，并大大加速了我们的研究。

t在谷歌Brain工作期间完成的工作。

在谷歌研究期间完成的工作。

第31届神经信息处理系统会议（NIPS 2017），美国加利福尼亚州长滩。

1 介绍

循环神经网络，特别是长短期记忆[13]和门控循环[7]神经网络，在序列建模和转换问题如语言模型和机器翻译[35,2,5]中，已被确立为最先进方法。此后，许多努力继续推动循环语言模型和编码器-解码器架构的边界[38,24,15]。

循环模型通常沿输入和输出序列的符号位置进行计算。将这些位置与计算时间的步骤对齐，它们生成一系列隐藏状态ht，作为前一个隐藏状态ht- 1和位置t的输入的函数。这种固有的顺序性质使得训练示例之间无法并行化，这在较长的序列长度下变得尤为关键，因为内存限制了跨示例的批量处理。最近的研究通过因子分解技巧[21]和条件计算[32]显著提高了计算效率，同时在后者的情况下还提升了模型性能。然而，顺序计算的基本约束依然存在。

注意力机制已成为各种任务中引人注目的序列建模和转换模型的重要组成部分，使得可以建模依赖关系而不考虑它们在输入或输出序列中的距离[2,19]。然而，在大多数情况下[27]，这种注意力机制是与循环网络结合使用的。

在这项工作中，我们提出了Transformer模型架构，该架构摒弃了递归方法，完全依赖于注意力机制来捕捉输入和输出之间的全局依赖关系。Transformer允许显著更多的并行化处理，并且在仅使用八块P100 GPU训练十二小时后，就能达到翻译质量的新高度。

2 背景

减少顺序计算的目标也构成了扩展神经GPU [16]、ByteNet [18]和ConvS2S [9]的基础，所有这些都使用卷积神经网络作为基本构建块，对所有输入和输出位置的隐藏表示进行并行计算。在这些模型中，连接两个任意输入或输出位置的信号所需的运算次数随着位置之间的距离线性增长，对于ConvS2S而言是如此，而对于ByteNet则是呈对数增长。这使得学习远距离位置之间的依赖关系变得更加困难[12]。在Transformer中，这一问题被减少到常数级别的运算次数，尽管由于平均注意力加权位置导致有效分辨率降低，我们通过第3.2节所述的多头注意力机制来抵消这一影响。

自注意力机制，有时也称为内部注意力，是一种关注机制，它关联单个序列的不同位置，以计算该序列的表示。自注意力机制已在多种任务中成功应用，包括阅读理解、抽象摘要、文本蕴含以及学习任务无关的句子表示[4,27,28,22]。

端到端记忆网络基于循环注意力机制，而不是序列对齐的循环，并且已经在简单语言问题回答和语言建模任务上表现良好[34]。

据我们所知，Transformer是首个完全依赖自注意力机制来计算输入和输出表示的转换模型，无需使用序列对齐的RNN或卷积。在接下来的部分中，我们将介绍Transformer，阐述自注意力机制，并讨论其相对于[17,18]和[9]等模型的优势。

3 模型架构

大多数竞争性的神经序列转导模型都采用编码器-解码器结构[5,2,35]。在这里，编码器将输入的符号表示序列（x1，...，xn）映射到连续表示序列z =（z1，...，zn）。给定z，解码器则逐个生成输出符号序列（y1，...，ym）。在每一步中，模型都是自回归的[10]，在生成下一个符号时，会使用之前生成的符号作为额外输入。

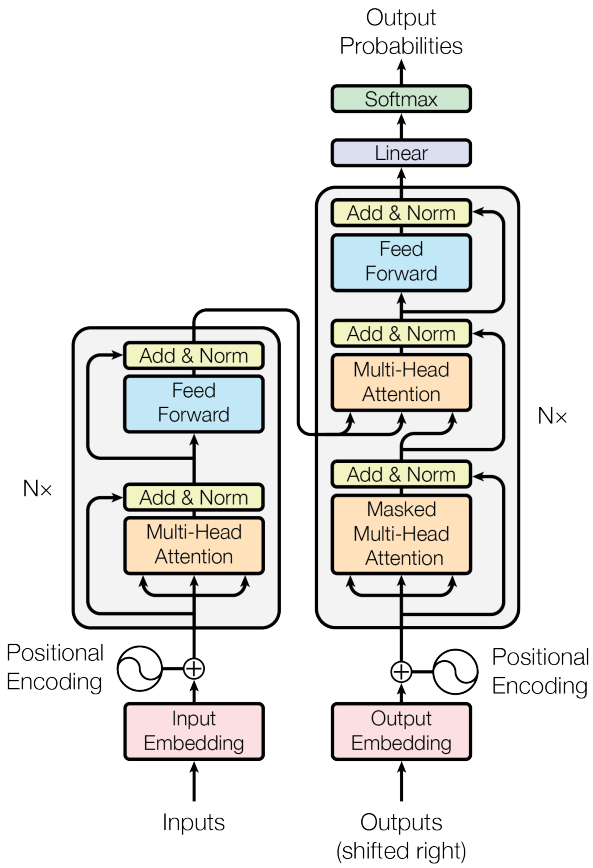

图1：Transformer -模型架构。

Transformer遵循这种整体架构，使用堆叠的自注意力和逐点全连接层作为编码器和解码器，分别如图1的左半部分和右半部分所示。

3.1 编码器和解码器堆栈

编码器：编码器由N个=6的相同层堆叠而成。每一层包含两个子层。第一个是多头自注意力机制，第二个是简单的、位置感知的全连接前馈网络。我们在每个子层周围使用残差连接[11]，随后进行层归一化[1]。也就是说，每个子层的输出经过层归一化（x+子层(x)），其中子层(x)是该子层本身实现的功能。为了便于这些残差连接，模型中的所有子层以及嵌入层都生成维度为dmodel = 512的输出。

解码器：解码器也由N个=6的相同层组成。除了每个编码器层中的两个子层外，解码器插入第三个子层，该子层对编码器堆栈的输出执行多头注意力。类似于编码器，我们在每个子层周围使用残差连接，并随后进行层归一化。我们还修改了解码器堆栈中的自注意力子层，以防止位置关注后续位置。这种掩码结合输出嵌入偏移一个位置的事实，确保位置i的预测仅依赖于小于i的位置的已知输出。

3.2 注意

注意力函数可以描述为将查询和一组键值对映射到输出，其中查询、键、值和输出都是向量。输出是作为加权和计算的

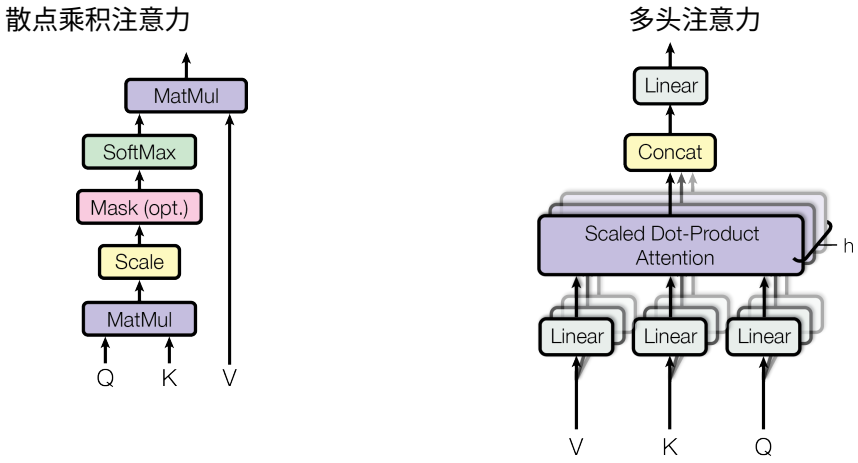

图2：（左）缩放点积注意力。（右）多头注意力由几个并行运行的注意力层组成。

其中，每个值的权重是通过查询与相应键的兼容性函数计算得出的。

3.2.1 散点乘积注意力

我们称这种注意力机制为“缩放点积注意力”（图2）。输入由维度为dk的查询和键以及维度为dv的值组成。我们计算查询与所有键的点积，然后将每个点积除以$\sqrt{}$dk，并应用softmax函数以获得值的权重。

在实践中，我们同时计算一组查询的注意力函数，将其打包成一个矩阵Q。键和值也打包成矩阵K和V。我们计算输出矩阵如下：

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

两种最常用的注意力函数是加法注意力[2]和点积（乘法）注意力。点积注意力与我们的算法相同，只是缩放因子不同。of $\frac{1}{\sqrt{d_k}}$。通过使用具有单个隐藏层的前馈网络，加性注意力计算兼容性函数。虽然两者在理论复杂性上是相似的，但点积注意力在实践中要快得多，也更节省空间，因为它可以使用高度优化的矩阵乘法代码来实现。

虽然对于dk的较小值，这两种机制表现相似，但当dk [3]的值较大时，加性注意力优于不缩放的点积注意力。我们怀疑对于较大的dk值，点积的大小会变得很大，将softmax函数推入其具有优势的区域。

极小的梯度4.为了抵消这种效应，我们通过缩放点积来实现$\frac{1}{\sqrt{d_k}}$.

3.2.2 多头注意力

我们发现，与其使用d模型维度的键、值和查询执行单一注意力函数，不如将这些查询、键和值分别通过不同的、学习得到的线性投影线性地投影h次，分别映射到dk、dk和dv维度。然后，在这些投影后的查询、键和值上并行执行注意力函数，从而得到dv维的结果。

4为了说明为什么点积会变大，假设q和k的分量是均值为0、方差为1的独立随机变量。然后它们的点积q · k = Σdi^{k}1 qi，均值为0，方差为dk。

输出值。这些值被连接起来，然后再次投影，得到最终值，如图2所示。
多头注意力使得模型能够联合关注不同位置的不同表示子空间的信息，而单个注意力头的平均抑制了这一点。

MultiHead (Q, K, V) = Concat (head1, ..., headh) WO其中headi = Attention (QW-iQ, KWiK, VWiV)

其中，投影为参数矩阵WiQ∈Rdmodel × dk、WiK∈Rdmodel × dk、WiV∈Rdmodel × dv和WO∈Rhdv × dmodel。
在这项工作中，我们采用了h = 8个并行注意力层，或称为头。对于每个头，我们使用dk = dv = dmodel /h = 64。由于每个头的维度降低，总计算成本与全维单头注意力相似。

3.2.3 注意力在我们模型中的应用

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，而记忆键和值来自编码器的输出。这使得解码器中的每个位置都能关注输入序列中的所有位置。这模拟了序列到序列模型中典型的编码器-解码器注意力机制，如[38,2,9]。
- ? 编码器包含自注意力层。在自注意力层中，所有的键、值和查询都来自同一个地方，在这种情况下，就是编码器前一层的输出。编码器中的每个位置都可以关注编码器前一层的所有位置。
- ? 同样地，解码器中的自注意力层允许解码器中的每个位置关注到该位置及其之前的全部位置。我们需要防止解码器中向左的信息流动，以保持自回归特性。我们通过在缩放点积注意力机制中屏蔽（设置为?∞）所有softmax输入中对应于非法连接的值来实现这一点。见图2。

3.3 位置上的前馈网络

除了注意力子层之外，我们的编码器和解码器中的每一层都包含一个全连接前馈网络，该网络分别且相同地应用于每个位置。它由两个线性变换组成，中间有一个ReLU激活函数。

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$

虽然线性变换在不同位置上是相同的，但它们在每一层使用不同的参数。另一种描述方式是两个卷积核大小为1的卷积。输入和输出的维度为dmodel = 512，内层的维度为dff = 2048。

3.4 嵌入和Softmax

类似于其他序列转导模型，我们使用学习到的嵌入将输入标记和输出标记转换为维度为dmodel的向量。我们还使用常规的学习线性变换和softmax函数将解码器输出转换为预测的下一个标记概率。在我们的模型中，我们在两个嵌入层和预softmax线性变换之间共享相同的权重矩阵，类似于[30]的做法。在嵌入层中，我们将这些权重乘以$\sqrt{\text{dmodel}}$。

表1：不同层类型的路径长度最大值、各层复杂度和最小顺序操作数。n是序列长度，d是表示维度，k是卷积的内核大小，r是受限自注意力中邻域的大小。

层类型	每层的复杂性	顺序操作	最大路径长度
自注意力	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
循环	$O(n \cdot d^2)$	$O(n)$	$O(n)$
卷积	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
自我关注（受限）	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

3.5 位置编码

由于我们的模型不包含循环和卷积，为了使模型能够利用序列的顺序，我们必须注入一些关于标记在序列中相对或绝对位置的信息。为此，我们在编码器和解码器堆栈底部的输入嵌入中添加了“ 位置编码 ”。位置编码与嵌入具有相同的维度dmodel，因此两者可以相加。位置编码有许多选择，包括学习的和固定的[9]。

在本工作中，我们使用不同频率的正弦和余弦函数：

PE（pos，2i）=sin（pos/100002i/dmodel）PE（pos，2i+1）=cos（pos/100002i/dmodel）

其中pos是位置，i是维度。也就是说，每个位置编码的维度对应一个正弦波。波长从2π到10000 · 2π形成几何级数。我们选择这个函数是因为假设它能让模型轻松学习通过相对位置来关注，因为对于任何固定的偏移k，PEpos+k可以表示为PEpos的线性函数。

我们还尝试使用学习到的位置嵌入[9]，发现两个版本的结果几乎相同（见表3第(E)行）。我们选择了正弦波版本，因为它可能允许模型外推到比训练过程中遇到的更长的序列长度。

4 为什么自我关注

在本节中，我们将自注意力层与常用的循环层和卷积层进行比较，这些层通常用于将一个变长的符号表示序列（x1，...，xn）映射到另一个等长的序列（z1，...，zn），其中xi，zi∈Rd，例如典型序列转导编码器或解码器中的隐藏层。为了说明我们使用自注意力层的原因，我们考虑了三个理想特性。

一是每层的总计算复杂度。二是可以并行化的计算量，以所需最少顺序操作的数量来衡量。三是网络中长程依赖之间的路径长度。学习长程依赖是许多序列转换任务中的关键挑战。影响学习此类依赖能力的一个重要因素是信号在网络中前后传播所需路径的长度。输入和输出序列中任意位置组合之间的这些路径越短，学习长程依赖就越容易[12]。因此，我们还比较了由不同层类型组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。

如表1所示，自注意力层通过固定数量的顺序执行操作连接所有位置，而循环层需要O(n)次顺序操作。就计算复杂度而言，当序列

长度n小于表示维度d，这在机器翻译中使用最先进的模型时尤为常见，例如词块[38]和字节对[31]表示。为了提高处理非常长序列任务的计算性能，可以将自注意力机制限制为仅考虑输入序列中围绕相应输出位置的r个邻域。这样会将最大路径长度增加到O（n/r）。我们计划在未来的工作中进一步研究这种方法。

单个卷积层的核宽度k < n并不能连接所有输入和输出位置。这样做需要在连续核的情况下使用O（n/k）的卷积层堆栈，或在膨胀卷积[18]的情况下使用O（logk（n））的卷积层堆栈，这会增加网络中任意两个位置之间最长路径的长度。卷积层通常比递归层昂贵k倍。然而，可分离卷积[6]使复杂度显著降低至O（k·n·d + n·d²）。然而，即使k = n，可分离卷积的复杂性也等于自注意力层和逐点前馈层的组合，这是我们模型中采用的方法。

作为附带的好处，自注意力机制可以产生更易解释的模型。我们检查了模型中的注意力分布，并在附录中展示了示例进行讨论。不仅各个注意力头明显地学会了执行不同的任务，许多注意力头还表现出与句子的句法和语义结构相关的行为。

5 训练

本节描述了我们模型的训练方案。

5.1 训练数据和批次

我们在标准的WMT 2014英德语数据集上进行了训练，该数据集包含约450万句对。句子使用字节对编码[3]进行编码，该编码共享约37000个词的源目标词汇表。对于英法语，我们使用了规模显著更大的WMT 2014英法语数据集，该数据集包含3600万句，并将词切分为32000个词块词汇表[38]。句子对根据近似序列长度批量处理。每个训练批次包含一组句子对，包含大约25000个源标记和25000个目标标记。

5.2 硬件和时间表

我们在一台配备8个NVIDIA P100 GPU的机器上训练了模型。对于使用文中所述超参数的基本模型，每个训练步骤大约需要0.4秒。我们总共训练了100,000步或12小时。对于我们的大型模型（见表3底部），每步时间是1.0秒。大型模型训练了300,000步(3.5天）。

5.3 优化器

我们使用了Adam优化器[20]，β1 = 0 .9，β2 = 0 .98和?= 10 -9。我们在训练过程中根据公式调整学习率：

$$lrate = d_{\text{mode}}^{-0.5} \cdot \min(\text{step\_num}^{-0.5}, \text{step\_num} \cdot \text{warmup\_steps}^{-1.5}) \quad (3)$$

这相当于在前warmup_steps个训练步骤中线性增加学习率，之后按步数的平方根倒数比例降低学习率。我们使用了warmup_steps = 4000。

5.4 正则化

我们在训练过程中使用了三种正则化方法：

表2：在英语到德语和英语到法语的news test2014测试中，Transformer在训练成本的一小部分下实现了比以前最先进的模型更好的BLEU分数。

模型	有蓝色霉菌花纹的		培训成本（FLOPs）	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL集成[38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S集合[9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
变压器（基础型号）	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
变压器（大）	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

残差丢弃我们在每个子层的输出上应用丢弃[33]，然后再将其加入到子层输入并进行归一化。此外，我们还对编码器和解码器堆栈中的嵌入和位置编码的总和应用丢弃。对于基础模型，我们使用Pdrop = 0 .1的丢弃率。

训练过程中，我们采用了?ls=0.1[36]的标签平滑方法。这会损害困惑度，因为模型会学会更加不确定，但会提高准确率和BLEU分数。

6 结果

6.1 机器翻译

在WMT 2014英语到德语翻译任务中，大变压器模型（表2中的Transformer（big））比之前报道的最佳模型（包括集成）高出2倍以上。0 BLEU，建立了新的最先进BLEU分数28.4。该模型的配置列于表3底部。训练耗时3.5天，在8个P100 GPU上完成。即使我们的基础模型也超越了所有先前发布的模型和集成，而训练成本仅为任何竞争模型的一小部分。

在WMT 2014英法翻译任务中，我们的大型模型达到了41.0的BLEU分数，超越了所有先前发布的单模型，且训练成本不到之前最先进模型的四分之一。Transformer（大型）模型用于英法翻译时，使用了Pdrop = 0. 1的丢弃率，而不是0.3。

对于基础模型，我们使用了通过平均最后5个检查点得到的单一模型，这些检查点以10分钟为间隔记录。对于大型模型，我们平均了最后20个检查点。我们采用了束搜索，束大小为4，长度惩罚 α = 0 .6 [38]。这些超参数是在开发集上实验后选定的。我们在推理过程中将最大输出长度设置为输入长度+ 50，但尽可能提前终止[38]。

表2总结了我们的结果，并将我们的翻译质量和训练成本与其他文献中的模型架构进行了比较。我们通过将训练时间、使用的GPU数量以及每个GPU的持续单精度浮点运算能力的估计值相乘，来估算训练模型所需的浮点运算次数。

6.2 型号变更

为了评估Transformer的不同组件的重要性，我们以不同的方式改变了我们的基础模型，测量了在英语到德语翻译上的性能变化

5我们分别使用了K80、K40、M40和P100的2.8、3.7、6.0和9.5 TFLOPS值。

表3：变压器架构的变化。未列出的值与基础模型相同。所有指标均基于英语到德语的翻译开发集newstest2013。列出的困惑度是按字块计算的，根据我们的字节对编码，不应与按单词计算的困惑度进行比较。

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	火车步骤	人民党 (dev)	有蓝色霉菌花纹的 (dev)	参数 $\times 10^6$
基础	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)			1	512	512					5.29	24.9	
			4	128	128					5.00	25.5	
			16	32	32					4.91	25.8	
			32	16	16					5.01	25.4	
(B)				16						5.16	25.1	58
				32						5.01	25.4	60
(C)	2									6.11	23.7	36
	4	256			32	32				5.19	25.3	50
	8	1024			128	128				4.88	25.5	80
			1024							5.75	24.5	28
			4096							4.66	26.0	168
										5.12	25.4	53
										4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	位置嵌入而不是						正弦波			4.92	25.7	
大的	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

开发集，newstest2013。我们使用了前一节中描述的束搜索方法，但没有使用检查点平均法。我们在表3中给出了这些结果。

在表3的第(A)行中，我们改变了注意力头的数量以及注意力键和值的维度，同时保持计算量不变，如第3.2.2节所述。虽然单头注意力比最佳设置低0.9 BLEU，但过多的注意力头也会导致质量下降。在表3的第(B)行中，我们观察到减少注意力键大小dk会损害模型质量。这表明确定兼容性并不容易，可能需要一个比点积更复杂的兼容性函数。我们在第(C)和(D)行中进一步观察到，正如预期的那样，更大的模型更好，而丢弃在避免过拟合方面非常有帮助。在第(E)行中，我们将正弦位置编码替换为学习的位置嵌入[9]，并观察到与基础模型几乎相同的结果。

6.3 英语语料库解析

为了评估Transformer是否能够泛化到其他任务，我们在英语成分句法分析上进行了实验。这项任务存在特定的挑战：输出受到强烈的结构约束，并且显著长于输入。此外，RNN序列到序列模型在小数据集情况下未能取得最佳结果[37]。

我们在宾夕法尼亚树库[25]的《华尔街日报》（WSJ）部分训练了一个4层变压器模型，dmodel = 1024，大约40,000个训练句子。我们还在半监督设置下进行了训练，使用了更大的高置信度和伯克利解析器语料库，包含约1700万句子[37]。对于仅WSJ部分的设置，我们使用了16,000个词汇量，而在半监督设置下则使用了32,000个词汇量。

我们只在第22节开发集上进行了少量实验来选择dropout、注意力和残差（第5.4节）、学习率和束大小，所有其他参数都与英语到德语基础翻译模型保持不变。在推理过程中，我们

表4：Transformer在英语成分分析中表现良好（结果见《华尔街日报》第23节）

字幕	训练	WSJ 23 F1
Vinyals & Kaiser等人（2014）[37] Petrov等人（2006）[29] Zhu等人（2013）[40] Dyer等人（2016）[8]	《华尔街日报》独家，歧视性 《华尔街日报》独家，歧视性 《华尔街日报》独家，歧视性 《华尔街日报》独家，歧视性	88.3 90.4 90.4 91.7
变压器（4层）	《华尔街日报》独家，歧视性	91.3
Zhu等人（2013）[40] Huang和Harper（2009）[14]McClusky等人（2006）[26] Vinyals和Kaiser等人（2014）[37]	半监督 半监督 半监督 半监督	91.3 91.3 92.1 92.1
变压器（4层）	半监督	92.7
Luong等人（2015）[23] Dyer等人（2016）[8]	多任务 能生产的	93.0 93.3

将最大输出长度增加到输入长度+ 300。对于WSJ仅和半监督设置，我们使用了21的光束尺寸和α = 0 .3。

表4中的结果表明，尽管缺乏特定任务的调整，我们的模型表现出了惊人的效果，比所有先前报道的模型都好，除了循环神经网络语法[8]。

与RNN序列到序列模型[37]相比，即使只在40K个句子的WSJ训练集中进行训练，Transformer也优于BerkeleyParser [29]。

7 结论

在这项工作中，我们提出了Transformer，这是第一个完全基于注意力的序列转换模型，用多头自注意力取代了编码器-解码器架构中最常用的循环层。

对于翻译任务，Transformer的训练速度远超基于循环或卷积层的架构。在WMT 2014英德和英法翻译任务中，我们取得了新的世界领先成果。在前者任务中，我们的最佳模型甚至超越了所有先前报道的集成方法。

我们对基于注意力机制模型的未来充满期待，并计划将其应用于其他任务。我们打算将Transformer扩展到涉及文本以外的输入和输出模态的问题，并研究局部受限注意力机制，以高效处理如图像、音频和视频等大型输入和输出。减少生成过程的顺序性也是我们的另一个研究目标。用于训练和评估我们模型的代码可在https://github.com/ tensorflow/tensor2tensor获取。

我们感谢Nal Kalchbrenner和Stephan Gouws的有益评论、纠正和启发。

参考文献

[1] Jimmy Lei Ba、Jamie Ryan Kiros和Geoffrey E Hinton。层归一化。arXiv预印本arXiv：1607.06450,2016。

[2] Dzmitry Bahdanau、Kyunghyun Cho和Yoshua Bengio。通过联合学习对齐和翻译实现神经机器翻译。CoRR，abs/1409.0473,2014。

[3] Denny Britz、Anna Goldie、Minh-Thang Luong和Quoc V. Le。大规模探索神经机器翻译架构。CoRR，abs/1703.03906,2017。

[4] Jianpeng Cheng、Li Dong和Mirella Lapata。用于机器阅读的长短期记忆网络。arXiv预印本arXiv：1601.06733,2016。

[5] Kyunghyun Cho，Bart van Merrienboer，Caglar Gulcehre，Fethi Bougares，Holger Schwenk和Yoshua Bengio。使用rnn编码器-解码器学习短语表示以进行统计机器翻译。CoRR，abs/1406.1078,2014。

[6] Francois Chollet。Xception：使用深度可分离卷积进行深度学习。arXiv预印本arXiv：1610.02357,2016。

[7] Junyoung Chung，Caglar Gulcehre，Kyunghyun Cho和Yoshua Bengio。门控循环神经网络在序列建模中的经验评估。CoRR，abs/1412.3555,2014。

[8] Chris Dyer，Adhiguna Kuncoro，Miguel Ballesteros和Noah A. Smith。循环神经网络语法。在NAACL会议论文集，2016年。

[9] Jonas Gehring，Michael Auli，David Grangier，Denis Yarats和Yann N. Dauphin。卷积序列到序列学习。arXiv预印本arXiv：1705.03122v2,2017。

[10] Alex Graves.用循环神经网络生成序列。arXiv预印本arXiv：1308.0850,2013。

[11] Kaiming He，Xiangyu Zhang，Shaoqing Ren和Jian Sun.用于图像识别的深度残差学习。在IEEE计算机视觉与模式识别会议论文集，第770-778页，2016年。

[12] Sepp Hochreiter，Yoshua Bengio，Paolo Frasconi和Jurgen Schmidhuber。递归网络中的梯度流：学习长期依赖的困难，2001年。

[13] Sepp Hochreiter和Jurgen Schmidhuber。长短期记忆。神经计算，9(8)：1735-1780,1997。

[14]黄中强和玛丽·哈珀。跨语言潜在注释的自我训练PCFG语法。收录于《2009年自然语言处理经验方法会议论文集》，第832-841页。ACL，2009年8月。

[15] Rafal Jozefowicz，Oriol Vinyals，Mike Schuster，Noam Shazeer和Wu Yonghui。探索语言建模的极限。arXiv预印本arXiv：1602.02410,2016。

[16]Łukasz Kaiser和Samy Bengio.主动记忆能否取代注意力？在神经信息处理系统进展（NIPS），2016年。

[17]Łukasz Kaiser和Ilya Sutskever。神经GPU学习算法。在国际学习表示会议（ICLR），2016年。

[18] Nal Kalchbrenner，Lasse Espeholt，Karen Simonyan，Aaron van den Oord，Alex Graves和Koray Kavukcuoglu。线性时间下的神经机器翻译。arXiv预印本arXiv：1610.10099v2,2017。

[19] Yoon Kim，Carl Denton，Luong Hoang和Alexander M. Rush。结构化注意力网络。*在2017年国际学习表征会议上。*

[20] Diederik Kingma和Jimmy Ba. Adam：一种随机优化方法。见ICLR，2015年。

[21] Oleksii Kuchaiev和Boris Ginsburg。LSTM网络的因式分解技巧。arXiv预印本arXiv：1703.10722,2017。

[22] Zhouhan Lin，Minwei Feng，Cicero Nogueira dos Santos，Mo Yu，Bing Xiang，Bowen Zhou，和Yoshua Bengio.结构化的自我关注句子嵌入。arXiv预印本arXiv：1703.03130,2017。

[23] Minh-Thang Luong，Quoc V.Le，Ilya Sutskever，Oriol Vinyals和Lukasz Kaiser。多任务序列到序列学习。arXiv预印本arXiv：1511.06114,2015。

[24] Minh-Thang Luong、Hieu Pham和Christopher D Manning。基于注意力的神经机器翻译的有效方法。arXiv预印本arXiv：1508.04025,2015年。

[25] Mitchell P Marcus，Mary Ann Marcinkiewicz和Beatrice Santorini。构建一个大型的英语注释语料库：Penn树库。计算语言学，19(2)：313-330,1993。

[26] David McClosky、Eugene Charniak和Mark Johnson。有效的自训练用于解析。在NAACL人类语言技术会议的主会议上，第152-159页。ACL，2006年6月。

[27] Ankur Parikh、Oscar Tackstrom、Dipanjan Das和Jakob Uszkoreit。可分解注意力模型。在自然语言处理中的经验方法，2016年。

[28] Romain Paulus，Caiming Xiong和Richard Socher。抽象摘要的深度强化模型。arXiv预印本arXiv：1705.04304,2017。

[29] Slav Petrov，Leon Barrett，Romain Thibaux和Dan Klein.学习准确、紧凑且可解释的树标注。在第21届国际计算语言学会议和第44届年度ACL大会上，第433-440页。ACL，2006年7月。

[30] Ofir Press和Lior Wolf。利用输出嵌入来改进语言模型。arXiv预印本arXiv：1608.05859,2016。

[31] Rico Sennrich，Barry Haddow和Alexandra Birch。用子词单元对稀有词汇进行神经机器翻译。arXiv预印本arXiv：1508.07909,2015。

[32] Noam Shazeer，Azalia Mirhoseini，Krzysztof Maziarz，Andy Davis，Quoc Le，Geoffrey Hinton和Jeff Dean。《异常大的神经网络：稀疏门专家混合层》。arXiv预印本arXiv：1701.06538,2017。

[33] Nitish Srivastava，Geoffrey E Hinton，Alex Krizhevsky，Ilya Sutskever和Ruslan Salakhutdinov。Dropout：一种防止神经网络过拟合的简单方法。机器学习研究杂志，15(1)：1929-1958,2014。

[34] Sainbayar Sukhbaatar，Arthur Szlam，Jason Weston和Rob Fergus.端到端记忆网络.在C. Cortes，N. D. Lawrence，D. D. Lee，M. Sugiyama和R. Garnett编辑的《神经信息处理系统进展》第28卷中，第2440-2448页. Curran Associates，Inc.，2015.

[35] Ilya Sutskever，Oriol Vinyals和Quoc VV Le。序列到序列学习与神经网络。在神经信息处理系统进展，第3104-3112页，2014年。

[36] Christian Szegedy，Vincent Vanhoucke，Sergey Ioffe，Jonathon Shlens，Zbigniew Wojna。
重新思考计算机视觉的初始架构。CoRR，abs/1512.00567,2015年。

[37] Vinyals & Kaiser，Koo，Petrov，Sutskever，and Hinton.语法作为外语。在神经信息处理系统进展，2015年。

[38] Yonghui Wu，Mike Schuster，Zhifeng Chen，Quoc V Le，Mohammad Norouzi，Wolfgang Macherey，Maxim Krikun，Yuan Cao，Qin Gao，Klaus Macherey,谷歌的神经机器翻译系统：弥合人类和机器翻译之间的差距.arXiv预印本arXiv:1609.08144,2016.

[39] Jie Zhou，Ying Cao，Xuguang Wang，Peng Li和Wei Xu.用于神经机器翻译的具有快速连接的深度循环模型。CoRR，abs/1606.04199,2016。

[40] Muhua Zhu，Yue Zhang，Wenliang Chen，Min Zhang，and Jingbo Zhu.快速准确的移位减少成分解析.第51届ACL年会论文集（第1卷：长篇论文），第434-443页. ACL，2013年8月。

注意可视化

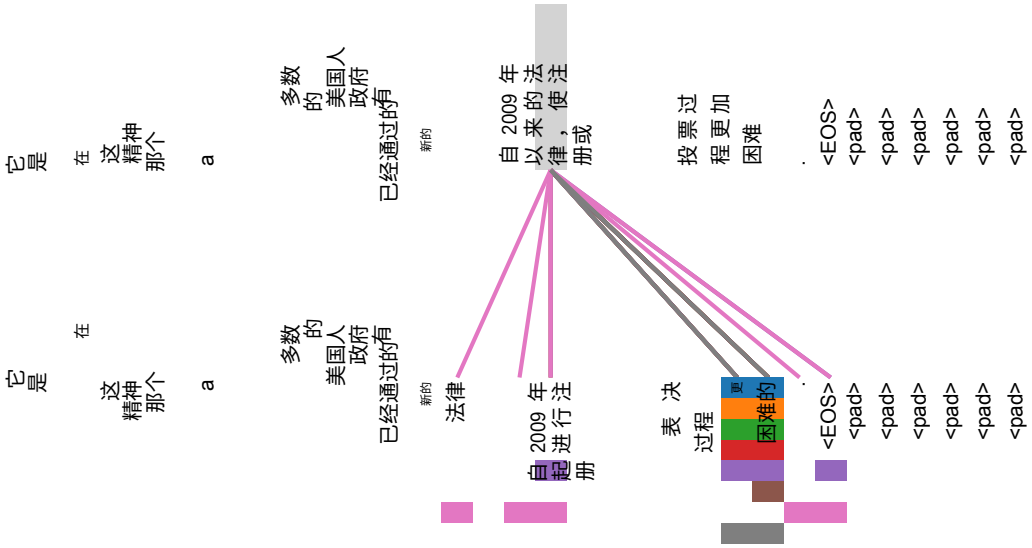

图3：编码器自注意力机制在第5层中的一个例子，遵循长距离依赖关系。许多注意力头关注动词“制造”的远距离依赖，完成短语“制造……更加困难”。这里仅显示了“制造”这个词的注意力。不同的颜色代表不同的注意力头。最佳彩色查看。

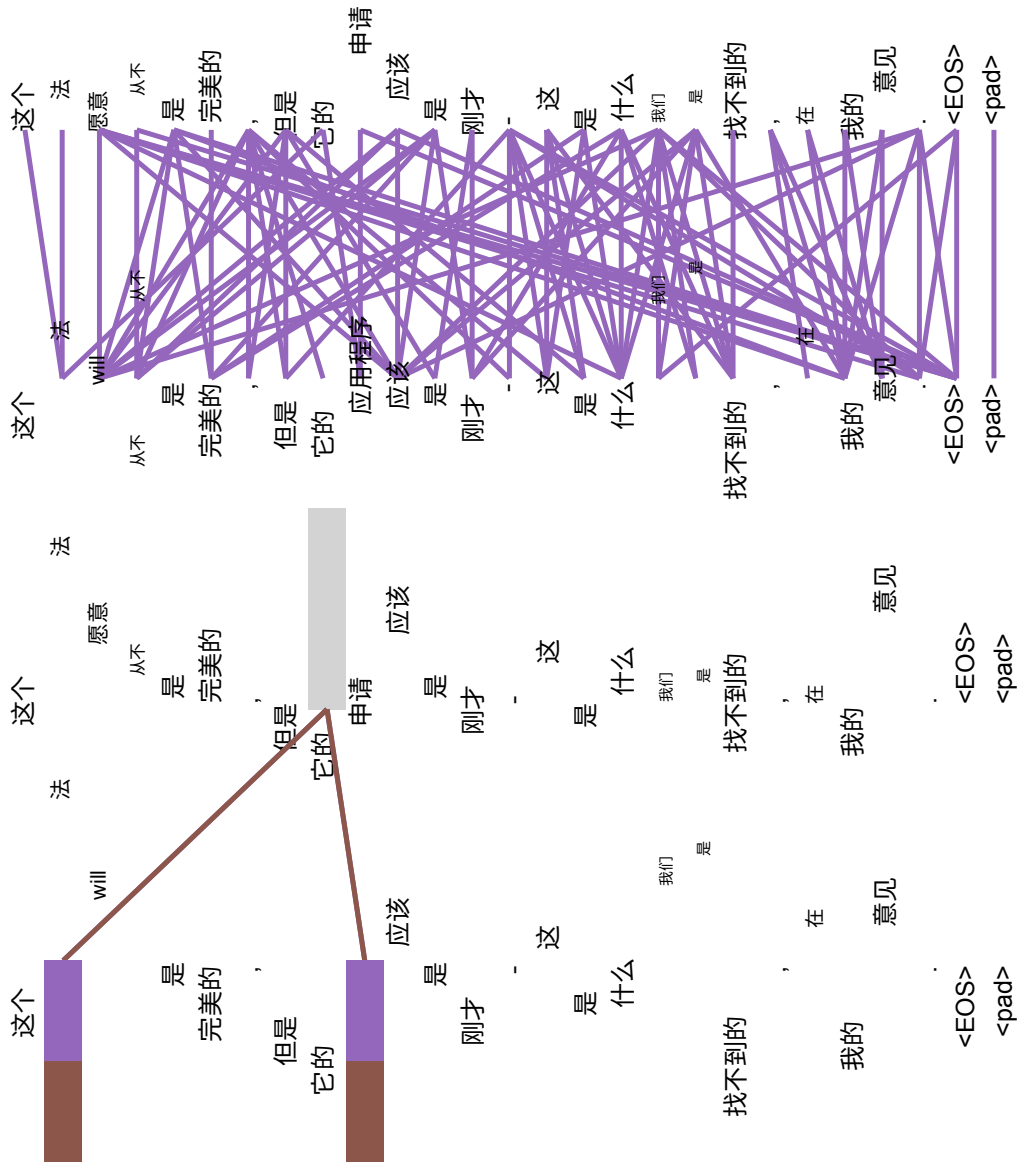

图4：两个注意力头，也位于第6层的第5层，显然参与了同位语解析。顶部：对头部5的全部注意力。底部：仅从单词“ i t ”中提取注意力，用于头部5和6。注意，对于这个单词，注意力非常尖锐。

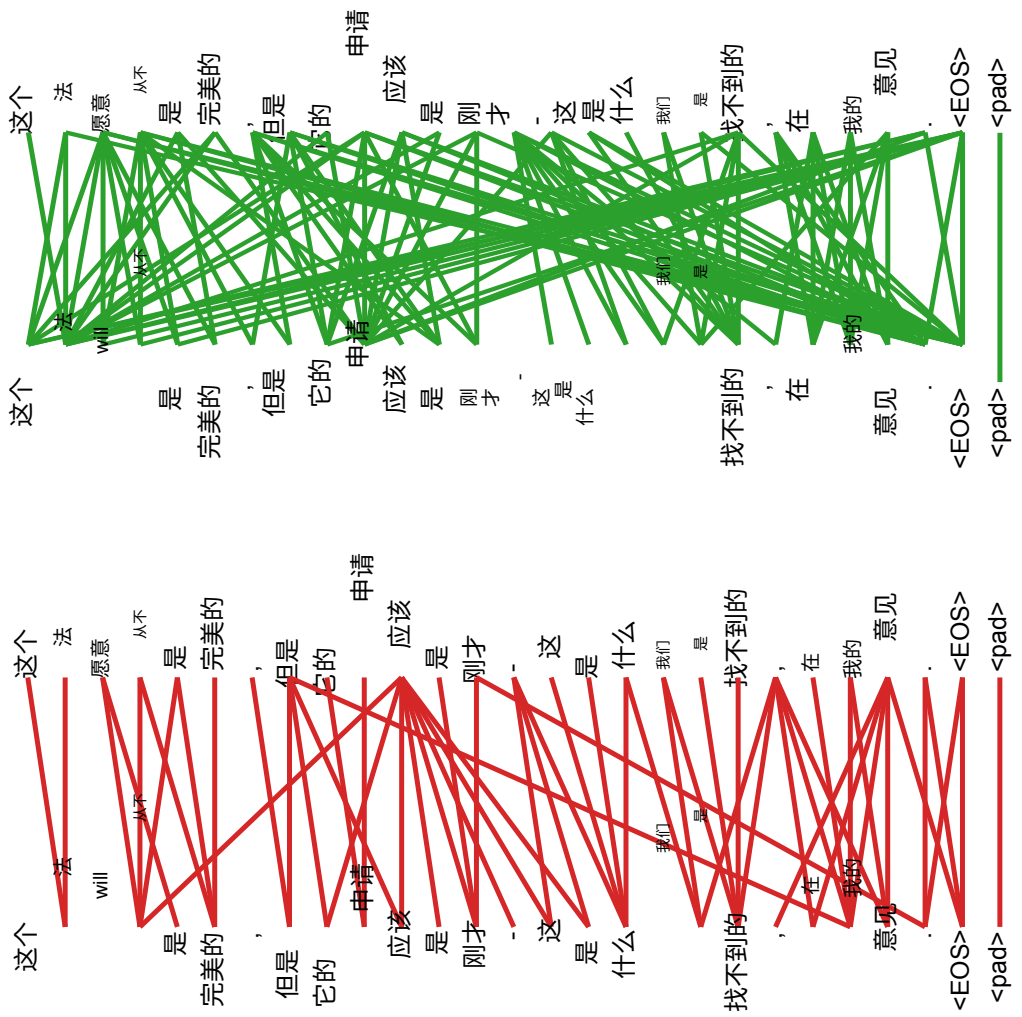

图5：许多注意力头表现出与句子结构相关的行径。我们在上面给出了两个这样的例子，它们分别来自编码器第5层的自注意力机制中的两个不同的头。这些头显然学会了执行不同的任务。