Transformer 中的 Attention 探究

摘 要

注意力模型(Attention Model)目前已成为神经网络中的一个重要概念，其被广泛地应用于计算机视觉(CV)、自然语言处理(NLP)等领域中。本文简要概述注意力模型的来源、原理结构概述，并简述注意力机制在 NLP 和 CV 的重点运用。

1 介绍

注意力模型(AM)首次出现于一篇机器翻译论文中[Bahdanau et al.2015][1].并显著地改善了 MT 的性能，在 NLP 多子任务领域(QA, Machine Translation)取得 SOTA 的 Transformer 模型的开创的一个重要组成就是引入位置编码的多头注意力机制(Multi-Head Attention)。Transformer 在 NLP 领域取得的优越性令很多人试图将其引入计算机视觉领域，由此产生了 Vision Transformer(ViT)以及后续一系列拓展模型。

我们的视觉系统可以很好地解释注意力机制。对于一句中文，我们可以很直观地捕捉到核心信息并忽略辅助信息。注意力机制就是有选择地聚焦到输入的部分相关内容，并忽略其它内容。如在机器翻译中，中文和英文同意的词语间的相似度很大，是我们需要特别关注的，下一次遇到同样中文词语时很快翻译成对应的单词。注意力机制正是这样一个机制，既可以使结果更精准，也可以解决计算复杂度过高的问题。

注意力机制帮助我们克服了 RNN 网络的一些短处，如在机器翻译中输入的语句长度增加，但编码器(encoder)的最后一个状态并不能很好地记录每一个隐藏层的输出，导致解码器(decoder)输出错误，即使采用 LSTM，当序列增加到一定程度后，BLEU(机器翻译的重要评价指标)值呈下降趋势；而使用注意力机制，使得解码器可以记住全部的信息，并能重点关注某些部分，使得一直保持一个高的翻译准确率。

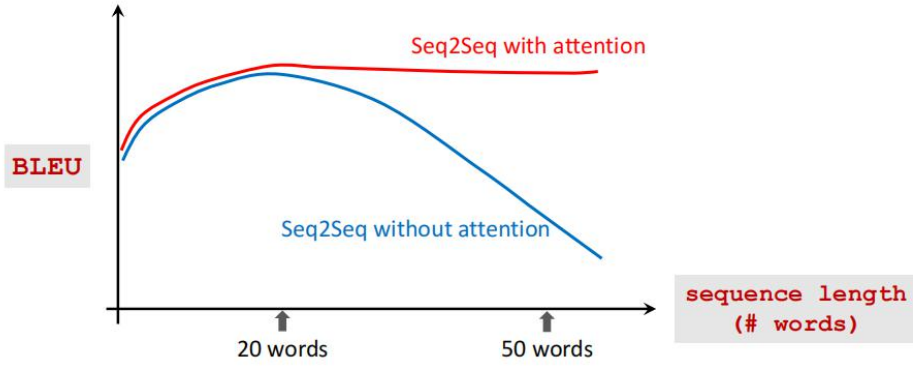


图 1 attention VS simpleRNN in Seq2Seq

本文将阐述 2015 年提出的 AM 的原理以及后续改进的部分，并引出 Transformer 模型，简要讲解其框架，并拓展计算机视觉及自然语言处理使用注意力机制取得的显著成果。

2 Attention 原理

传统的编码器-解码器(encoder-decoder)存在 2 个很大的问题。一是编码器需要将信息压缩成固定长度向量*h*, 然后传递给编码器，然而使用单个固定长度向量压缩会导致信息丢失。二是无法对输入输出序列进行有效对齐，比如英文翻译为中文单词数目减少，这需要翻译只聚焦到部分英文的有效输入，而普通的解码器无法选择部分编码器的 token.

Seq2Seq 的组成 RNN 的缺点是对于词的输入是从左往右的，第 t 个词是由前面的 t-1 个词决定的，这可以解决时序问题，但很难进行并行运算。当我们计算第 t 个词的时候，必须保证前面的h_{t-1}运算结束。时序较长时，会导致丢失信息，即使采用较大的h_t存储，也会造成内存开销大的问题。

这就引出了 AM 的核心思想，在解码器计算每一个状态向量时引入 *context vector*, 即输入的部分状态，从而改进上述短板。

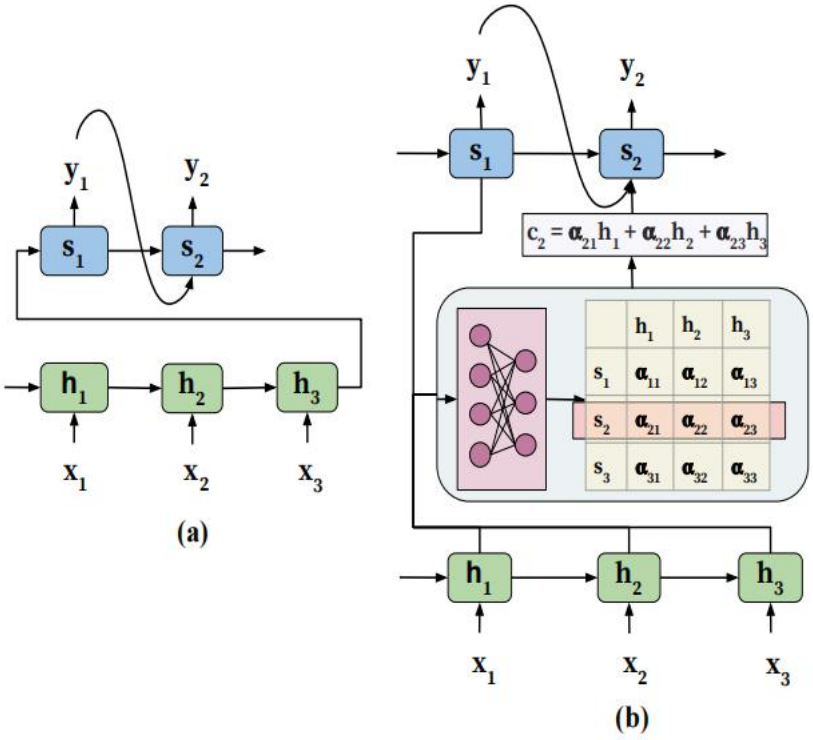


图 2[2] Encoder-decoder 架构：(a)传统的 (b)引入 AM

这里具体阐述每一步骤。

首先引入注意力权重 α 以优先考虑尊在相关信息的位置集，以生成下一个输出标记。有很多种方法计算注意力权重 α.

2015 论文中所写 $\alpha_i = align(h_i, s_j)$, s_j是每一个 decoder 状态。所有的 α_i和为 1，每一个 α_i位于 0 置 1 之间。

$\alpha_{i-normalize} = V^T \cdot tanh(w \begin{bmatrix} h_i \\ s_j \end{bmatrix})$, 其中$V$和$w$都是可以训练的参数。 。

$[\alpha_1, \ldots, \alpha_m] = Softmax([\alpha_{1-normalize}, \ldots, \alpha_{m-normalize}])$.

现在应用更广泛的是另一种权重计算方式，这也被应用在 Transformer 中。

(1) $k_i = W_K \cdot h_i, \quad for\ j = 1\ to\ m$

$q_0 = W_Q \cdot s_0$ (decoder 以第 0 个状态为例)

(2) $\alpha_{i-normalize} = k_i^T q_0, \ for\ i = 1\ to\ m$

(3) $[\alpha_1, \ldots, \alpha_m] = Softmax([\alpha_{1-normalize}, \ldots, \alpha_{m-normalize}])$.

上下文向量C_j是对编码器所有隐藏状态以及其相应注意力权重的加权和：

$Context\ cector: C_j = \alpha_1 h_1 + \ldots + \alpha_m h_m$, 其中 m 为 encoder 输入个数。

在传统的 SimpleRNN 中，计算 decoder 的状态层如下(以S_1为例)：

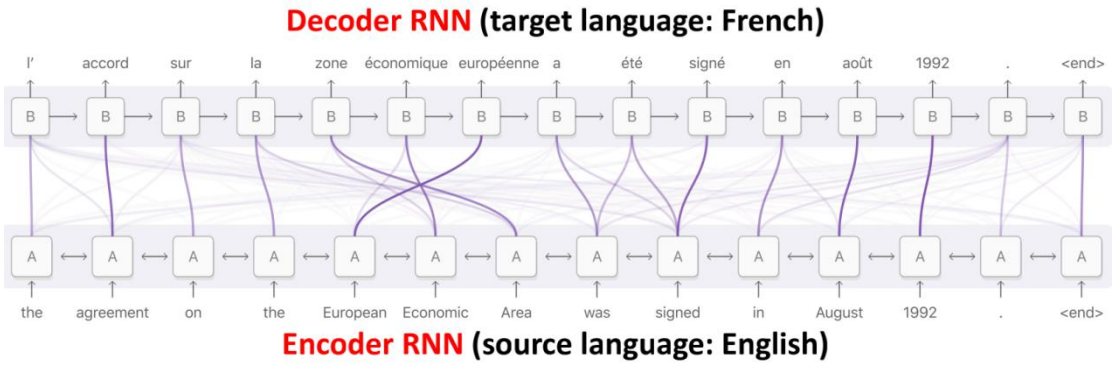
$$S_1 = tanh(A^{'} \cdot \begin{bmatrix} x_1^{'} \\ s_0 \end{bmatrix} + b)$$

引入注意力机制后，计算如下：

$$S_1 = tanh(A^{'} \cdot \begin{bmatrix} x_1^{'} \\ s_0 \\ C_0 \end{bmatrix} + b)$$

这个额外的上下文向量使解码器可以访问整个输入序列，也关注序列中的相关位置。这不仅可以提高最终任务的性能，还可以通过更好的对齐来提高输出质量。

关于时间复杂度，计算一个C_j，我们需要计算 m 个权重：$\alpha_1, \alpha_2, \ldots, \alpha_m$. 假设编码器有 t 个状态，总共有 mt 个权重，所以 AM 的时间复杂度为 mt. 当序列很长时这将会是一个很大的模型，时间复杂度很大，这样说明了 AM 需要较高的运算力。下面这张图可以直观地体现注意力机制的作用。



图片 3 来源：https://distill.pub/2016/augmented-rnns/

这张图形象地说明了 AM 的作用，连线颜色越深表明计算出来的相似度越高，可以看出法语的 zone 和英语的 area 都为区域的意思，证实了 AM 的强有力的作用。

3 Attention 和 Transformer

Transformer 是第一个只依赖于自注意力实现 encoder-decoder 的架构。在机器翻译序列模型上展示了强大的能力。逐渐成为了 NLP 领域的主要深度学习模型。最新的主导模型是一些自监督的 Transformer，它们从足够的数据集进行

预训练，然后在小型和特定的任务上进行微调。自监督和预训练作为对自然语言影响较大的两个部分，Transformer 带来了极大的影响，如预训练模型 BERT(Bidirectional Encoder Representations from Transformers).

在 Attention is all you need[3]中所提出的 Transformer 是第一个完全基于注意力的序列转导模型，运用多头自我关注(multi-headed self-attention)取代了编码器架构中最常用的循环层。传统语言模型的编码器-解码器架构采取串行 RNN，计算效率很低，为了改成并行机制，希望使用卷积神经网络替换循环神经网路哦，但使用卷积神经网路对较长序列很难建模，像素点之间距离增长，卷积层数增多，使得学习遥远位置之间的依赖关系更加困难。利用注意力机制可以消除这一问题，引入多头注意力机制可以模拟卷积中多输出通道的效果。使用多头注意力机制能够给予注意力层的输出包含有不同子空间中的编码表示信息，从而增强模型的表达能力。同时，Transformer 另一个重要组成自注意力(self-attention)，会在后文说明。

模型的输入：输入字符映射成的一组向量(x_1，…，x_n)。

模型的输出：解码器一次生成一个元素的符号的输出序列(y_1，…，y_m)，每一步都是自回归的，即上一步输出作为这一步输入。

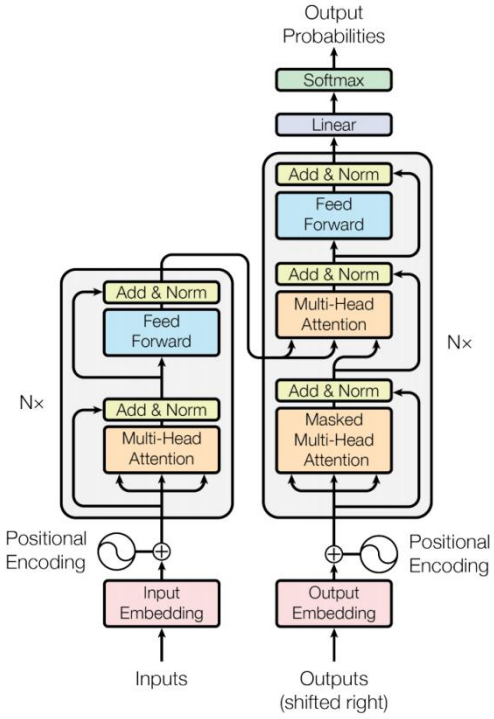


图 4[3] The Transformer - model architecture.来源

可以看出，Transformer 是典型的 encoder-decoder 结构。输入经过 embedding 层，加入了位置信息进入 encoder，encoder 由 n(N=6)个 Transformer block 构成，一个 block 大体由多头注意力机制、前馈神经网络、残差连接组成；encoder 的输出作为 decoder 的输入，解码器由 n(N=6)个 Transformer block 构成，每个 block 增加了带有掩码(masked)的多头注意力层。

这里残差处理后面需要对矩阵进行归一化来保证网络对稳定性。在 NLP 中广泛使用 Layer norm。这里使用 Layer norm 而不是 Batch norm。因为实际每个样本长度或许不一致，变化较大时，使用 Batch norm 计算单个特征对均值方差抖动性较大。Layer norm 是针对每个样本计算，更为稳定。解码器中使用掩码是为了避免预测时看到该时刻后的输入，注意力机制可以看到全部的序列，使用带有掩码的注意力机制是为了保证训练和预测测试的一致性。

这里的 attention 的输出是 value 的加权和，权重为 key 和 query 的相似性计算而来。Transformer 使用的相似性函数为缩放的点积注意力(Scaled Dot-Product Attention)，一般而言，key 和 query 是相同的。但其使用的权重矩阵不同，是因为若在同一个向量空间里进行点乘，必定是自身和自身的相似度最大，那会影响其他向量对自己的作用。

注意力机制的计算输出如下：

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

另一种常见的注意力机制为加性注意力，但点积计算更快，经实验效果分析两者的效果和d_k相关，d_k越大，加法的效果越显著。本文使用点积注意力，并且缩小 $1/\sqrt{d_k}$. 这是因为当d_k较大时，点积后值较大，相对差距变大，通过 softmax 后较大值更贴近 1，剩余值贴近 0。此时 softmax 函数的梯度会非常的小，其结果就是，softmax 函数梯度趋于 0，使得模型误差反向传播（back-propagation）经过 softmax 函数后无法继续传播到模型前面部分的参数上，造成这些参数无法得到更新，最终影响模型的训练效率。本文缩放后，$\frac{QK^T}{\sqrt{d_k}}$的方差由d_k缩小至 1，消除了梯度为 0 的影响。

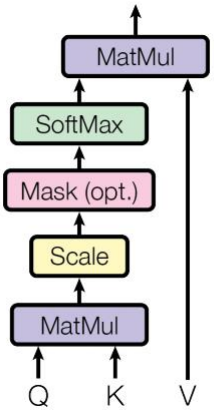


图 5[3] Scaled Dot-Product Attention

上图为点积注意力结构图，加入 mask 是为了避免 t 时刻预测遮盖 t 后面的序列。简单的做法是Q_t和K_t后的值用极大的负数替代。该值进入 SoftMax 做指数的时候会变为 0，不会对结果有影响。(第 i 个元素的 SoftMax 值为：$S_i = \frac{e^i}{\sum_j e^j}$)

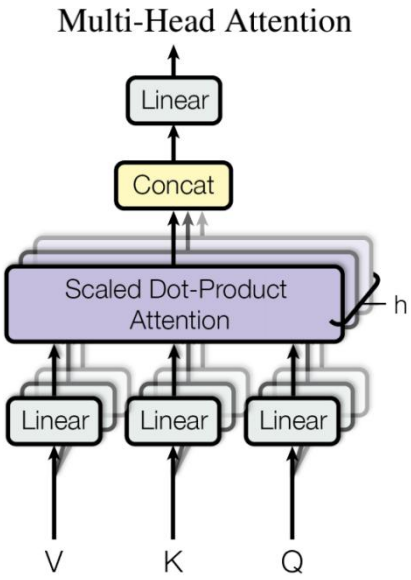


图 6[3] Multi-Head Attention consists of several attention layers running in parallel.

点积注意力并没有需要学习的参数，计算相似度方式单一，使用多头注意力机制(MHSA)可以有效缓解单头注意力建模能力粗糙，表达特征不全面的问题。该机制(MHSA)将输入线性地投射到多个特征子空间中，并通过几个独立的注意头（层）并行地处理它们。所得到的向量是连接起来的，被指定并映射到最终的输出。类似于 CNN 中多通道的表达形式。MHSA 的生成过程可以表述为[4]：

$$Q_i = XW^{Q_i}, K_i = XW^{K_i}, V_i = XW^{V_i}$$

$$Z_i = Attention(Q_i, K_i, V_i), i = 1 \ldots h,$$

$$MultiHead(Q,K,V) = Concat(Z_1, Z_2, \ldots, Z_h)W^O$$

h是头的数目，Z_i是每个头的输出向量，W^O是输出投影矩阵。

在 Transformer 中，相同的输入同时作为 key,query,value，所以称为自注意力机制。

单词在句子中的位置以及排列顺序是非常重要的，它们不仅是一个句子的语法结构的组成部分，更是表达语义的重要概念。一个单词在句子的位置或排列顺序不同，可能整个句子的意思就发生了偏差。普通的 Attentio 丢失顺序信息，模型就没有办法知道每个词在句子中的相对和绝对的位置信息。因此，有必要把词序信号加到词向量上帮助模型学习这些信息，位置编码（Positional Encoding）就是用来解决这种问题的方法。常见的解决方案是在输入中附加一个额外的位置向量，因此得到了术语“位置编码”。位置编码有很多选择。例如，一个典型的选择是差异的正弦和余弦函数。

$$PE_{(pos,i)} = \begin{cases} sin(pos \cdot \omega_k) & if \quad i = 2k \\ cos(pos \cdot \omega_k) & if \quad i = 2k+1 \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}, \quad k = 1, \ldots, d/2$$

i 是词在句子中的索引，d 是句子的长度，pos: 0,1,...,d-1.

这里使用三角函数进行位置转换，如果单一使用投影将位置 0 至 d-1 投影置一个范围，会削弱长文本中的相关次序，引入ω_k是为了根据文本长度保证映射的公平性。

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

表 1[3] 不同层类型的最大路径长度、每层复杂性和最小顺序操作数。n 为序列长度，d 为表示维数，k 为的核大小 卷积和 r 在限制自注意中邻域的大小。

可以看出 Self-Attention 运算时有较高的效率和较低的复杂度，在顺序计算时仅需要较小的步长，信息从一个点到另一个点的距离也最小，从而拥有更好的运算力。Transformer 在机器翻译领域取得了最好成果，其任务 BLEU 值最高，且花费最低，同时已被证实在其他 NLP 任务上取得了不错的结果。

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23.75			
Deep-Att + PosUnk [32]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [31]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [8]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [26]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [32]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [31]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [8]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.0**	$2.3 \cdot 10^{19}$	

表 2 [3] Transformer 在 2014 年英语到德语和英语到法语的新测试中取得了更好的英语分数。

4 Attention 和 Vision Transformer

受 NLP 中 Transformer 的突出发展的启发，一些研究人员试图将 Transformer 引入图像分类，Vision Transformer(ViT)[5]首次在主流分类基准上取得了类似甚至优越的性能。而且目前还未检测到过拟合的现象，随着模型参数增多，达到千亿级别以上，transformer 的效果也在增加，未出现性能饱和的情况。

然而，在计算机视觉方面，卷积架构仍然占主导地位。在大规模图像识别中，经典的 ResNet-like 架构仍然是最先进的技术。一些人试图用卷积和 transformer 结合，但并未完全脱离 transformer. ViT 是一种完全脱离 CNN，将 transformer 应用于图像分类的模型，仅在预处理过程的 word embedding 部分替换成了图像 patch 的处理方式，这也与 NLP 中的 token 很相似，不同的是，ViT 是有监督的图像分类任务。

ViT 适合大规模数据集，在中等数据集或小数据集上的表现弱于 ResNet，这是因为 ViT 是从 0 开始训练图片，没有丰富的先验知识，缺少归纳偏置(inductive biases)，如 translation equivariance and locality。在大规模数据集上，ViT 达到了 SOTA，最佳模型在 ImageNet 上的准确率达到 88.55%，在 ImageNet 上达到 90.72%，在 CIFAR-100 上的准确率达到 94.55%，在 VTAB 的 19 个任务套件上达到 77.63%。

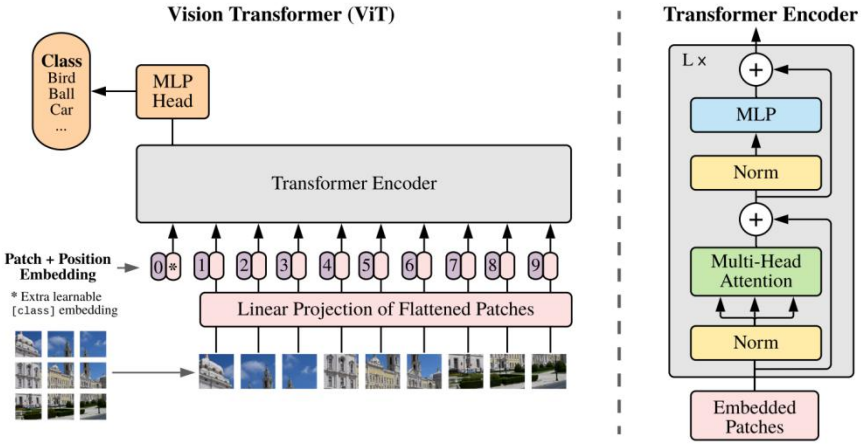


图 7 [5] ViT Model overview

ViT 将图像分割成固定大小的补丁(patch)，线性嵌入每个补丁，添加位置嵌入(Position embedding)，并将得到的向量序列提供给一个标准的 transformer. 为了进行分类，我们使用了在序列中添加一个额外的可学习的“分类标记”的标准方法(CLS)。transformer 的插图的灵感来自于 Vaswani et al. (2017).

假设原图为 224*224*3，设置每个 patch 为 16*16，那么总共有 224*224/16*16=196 个图像块，即有 196 个 token，每个 token 维度 16*16*3=768 维。这样，我们可以解决了以前的应用就是希望图像的每一个像素点都可以关注其它的每个像素导致的复杂度太高的问题。将一个 2D 图像转化为了 768 维的序列数据。经过作者的实验，patchSize 很重要，且越大越好，它有可能造成完全不同的结果。

图像信息的位置编码不仅要考虑左右，还需要考虑上下也就是四周的位置信息，但对于 ViT 而言采取和 NLP 同样的策略就可以达到很好的效果，对于 2D 的位置信息（即考虑横轴、纵轴位置信息一起嵌入）似乎并没有更好的结果，从以下图片可以清晰地看出。

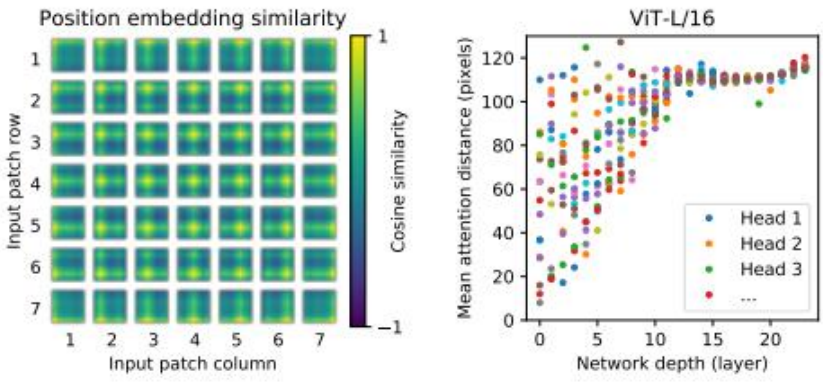


图 8[5]　Similarity of position embeddings of ViT-L/32. 图 9[5]　Size of attended area by head and network depth

图 8 可以看出距离中心点越近的地方相似度越高，说明位置层学到了相关的相似性。同行、同列的颜色更偏黄，说明虽然这是一个 1D 的位置编码，但已经学到了 2D 的位置关系，这也说明了为什么 2D 的距离 embedding 没有明显作用，也没有必要分别使用轴注意力。

图 9 可以看出加入了注意力机制后，在第一层（网络最浅层）就已经出现了可以注意到所有像素的位置点，当网络层数加深，该模型可以很稳健地使所有点拥有注意力，这与 CNN 有一个很直观的对比。

这是 ViT 的训练结果图。

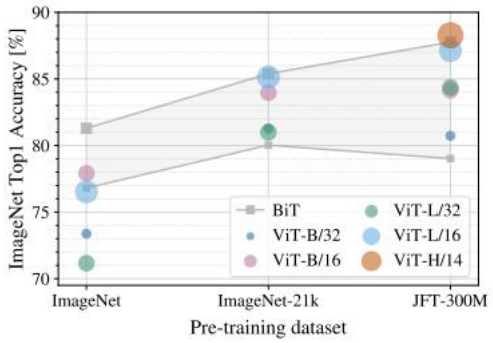


图 10[5]　Transfer to ImageNet.

虽然大型 ViT 模型在小数据集上预训练时的表现比 BiTresnet(阴影区域)更差，但在较大数据集上预训练时，它们可以全面超越 BiT(在 3 亿数据集上)。类似地，随着数据集的增长，较大的 ViT 变体也超过了较小的变体。

5 总结

本文我们讨论了注意力的基本原理，注意力的不同计算注意力权重方式，使用注意力的关键神经网络架构的应用概述。我们探讨了自然语言处理以及计算机视觉 Transformer 模型的一些应用以及注意力机制对其中的贡献。希望未来能够重点对自然语言处理上的注意力机制运用有更深的感悟。

6 参考

[1] Dzmitry Bahdanau,Kyunghyun Cho,Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate.[J]. CoRR,2014,abs/1409.0473:

[2] Sneha Chaudhari,Gungor Polatkan,Rohan Ramanath,Varun Mithal. An Attentive Survey of Attention Models.[J]. CoRR,2019,abs/1904.02874.

[3] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.

[4] Liu Y, Zhang Y, Wang Y, et al. A Survey of Visual Transformers[J]. arXiv preprint arXiv:2111.06091, 2021.

[5] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[J]. arXiv preprint arXiv:2010.11929, 2020.